

Jolita Ralyté  
Xavier Franch  
Sjaak Brinkkemper  
Stanisław Wrycza (Eds.)

LNCS 7328

# Advanced Information Systems Engineering

24th International Conference, CAiSE 2012  
Gdańsk, Poland, June 2012  
Proceedings

# CAiSE'12

 Springer

*Commenced Publication in 1973*

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

*Lancaster University, UK*

Takeo Kanade

*Carnegie Mellon University, Pittsburgh, PA, USA*

Josef Kittler

*University of Surrey, Guildford, UK*

Jon M. Kleinberg

*Cornell University, Ithaca, NY, USA*

Alfred Kobsa

*University of California, Irvine, CA, USA*

Friedemann Mattern

*ETH Zurich, Switzerland*

John C. Mitchell

*Stanford University, CA, USA*

Moni Naor

*Weizmann Institute of Science, Rehovot, Israel*

Oscar Nierstrasz

*University of Bern, Switzerland*

C. Pandu Rangan

*Indian Institute of Technology, Madras, India*

Bernhard Steffen

*TU Dortmund University, Germany*

Madhu Sudan

*Microsoft Research, Cambridge, MA, USA*

Demetri Terzopoulos

*University of California, Los Angeles, CA, USA*

Doug Tygar

*University of California, Berkeley, CA, USA*

Gerhard Weikum

*Max Planck Institute for Informatics, Saarbruecken, Germany*

Jolita Ralyté Xavier Franch  
Sjaak Brinkkemper Stanisław Wrycza (Eds.)

# Advanced Information Systems Engineering

24th International Conference, CAiSE 2012  
Gdańsk, Poland, June 25-29, 2012  
Proceedings

 Springer

## Volume Editors

Jolita Ralyté  
University of Geneva  
Institute of Services Science  
Battelle - Bâtiment A, Route de Drize 7, 1227 Carouge, Switzerland  
E-mail: jolita.ralyte@unige.ch

Xavier Franch  
Technical University of Catalonia  
Department ESSI, UPC-Campus Nord  
c/Jordi Girona 1-3, 08034 Barcelona, Spain  
E-mail: franch@essi.upc.edu

Sjaak Brinkkemper  
Utrecht University  
Department of Information and Computing Sciences  
Princetonplein 5, De Uithof, 3584 CC Utrecht, The Netherlands  
E-mail: s.brinkkemper@uu.nl

Stanisław Wrycza  
University of Gdańsk  
Faculty of Management  
Department of Business Informatics  
ul. Piaskowa 9, 81-864 Sopot, Poland  
E-mail: swrycza@ug.edu.pl

ISSN 0302-9743  
ISBN 978-3-642-31094-2  
DOI 10.1007/978-3-642-31095-9  
Springer Heidelberg Dordrecht London New York

e-ISSN 1611-3349  
e-ISBN 978-3-642-31095-9

Library of Congress Control Number: 2012939395

CR Subject Classification (1998): H.4, D.2.2, D.2.1, D.2, J.1, I.2, H.2

LNCS Sublibrary: SL 3 – Information Systems and Application, incl. Internet/Web and HCI

© Springer-Verlag Berlin Heidelberg 2012

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

*Typesetting:* Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

# Preface

We are pleased to welcome you to the proceedings of the 24th International Conference on Advanced Information Systems Engineering (CAiSE 2012). The CAiSE series of conferences started in 1989 aiming at providing a venue to researchers and practitioners in the field of information systems engineering. Twenty-three years later, CAiSE has established itself as a leading event in the information systems area for presenting and exchanging results of emerging methods and technologies that facilitate innovation and create business opportunities.

CAiSE 2012, held in Gdańsk during June 25–29, 2012, continued this tradition. The theme of CAiSE 2012 was “Information Services.” The notion of service plays a more and more extensive role in enterprise development. Indeed, most of the enterprise management and manufacture role is based on the exchange of services: services to the customers and/or citizens, services to support the inter-organizational collaboration, as well as services to accomplish intra-organizational activities. Many organizations and companies are sharing services with others, interfacing services from others, or outsourcing their ICT resources to various locations worldwide aided by the Internet. For all of them, the concept of service becomes a cornerstone of their processes of collaboration, innovation, and value creation. In this context, information systems engineering is moving toward the adoption of service-driven architectures where intra- and inter-organizational business activities are carried out with the help of information services. Information services are considered as a new means to deal with the complexity, modularity, and interoperability of the constantly growing information systems. Design and development of information services and information service-driven architectures become key to the success of organizations and their business. Consequently, the service-driven IS domain is becoming a new complex domain, which requires new interdisciplinary approaches and new transdisciplinary ways of thinking.

This year, CAiSE 2012 received 297 submissions. In a first round, each submission was reviewed by three Program Committee members. After examining the final reviews, 171 papers entered into the on-line discussion process, each paper moderated by one Program Board member. As a result, the papers were ranked and then discussed during the Program Committee/Program Board meeting held in Geneva during February 13–14, 2012. At the end of the meeting, 33 papers were definitively accepted as full research papers, and nine others went through a last quality-assurance phase in which a shepherd ensured that some reviewers’ comments were handled properly. As a result, the conference program included 42 full research papers that were organized into 14 themes, each of them scheduled as a session in the conference program.

The main conference program also included four mini-tutorials, the CAiSE Forum, an industrial track on Practice-Driven Research on Enterprise Transformation (PRET 2012), a panel and two keynote speeches: Michele Missikoff talked about “Looking at Future Research Challenges in Enterprise Information Systems,” while Krzysztof Kurowski delivered a talk on “Challenges for Future Platforms, Services and Networked Applications.” Before the conference, some other events were scheduled: ten workshops, a Doctoral Consortium and two working conferences: Business Process Modeling, Development, and Support (BPMDS 2012) and Exploring Modelling Methods for Systems Analysis and Design (EMMSAD 2012), this year arranged jointly with EuroSymposium 2012.

The organization and successful running of a large conference such as CAiSE would not be possible without the valuable help and time of a large number of people. As editors of this volume, we would like to express our gratitude to the Program Committee members, the Program Board members and additional reviewers for their valuable work in selecting the papers for the scientific program of the conference; to the authors of the papers for sending their work to CAiSE; and to the presenters of the papers and Session Chairs. We also thank the Chairs of the various CAiSE 2012 committees for their assistance in creating an exciting scientific program. We would also like to thank all members of the local Organizing Committee at the University of Gdańsk for their hospitality and the organization of the social events of the conference, and extend our gratitude to the sponsors who made the event financially feasible. Finally, we thank the participants both from academia and industry and we hope that their active participation in the CAiSE conference was inspiring for their research and a good support for industrial innovation.

June 2012

Jolita Ralyté  
Xavier Franch  
Sjaak Brinkkemper  
Stanisław Wrycza

# Organization

## Steering Committee

Barbara Pernici  
Oscar Pastor  
John Krogstie

Politecnico di Milano, Italy  
Technical University of Valencia, Spain  
Norwegian University of Science and  
Technology, Norway

## Advisory Committee

Arne Sølvberg

Norwegian University of Science and  
Technology, Norway

Janis Bubenko Jr.  
Colette Rolland

Royal Institute of Technology, Sweden  
Université Paris 1 Panthéon Sorbonne, France

## General Chair

Sjaak Brinkkemper

Utrecht University, The Netherlands

## Program Chairs

Jolita Ralyté  
Xavier Franch

University of Geneva, Switzerland  
Technical University of Catalonia, Spain

## Organizing Chair

Stanisław Wrycza

University of Gdańsk, Poland

## Workshop Chairs

Marko Bajec  
Johann Eder

University of Ljubljana, Slovenia  
Alpen Adria Universität Klagenfurt, Austria

## Tutorial Chairs

Ana Moreira  
Raimundas Matulevičius

Universidade Nova de Lisboa, Portugal  
University of Tartu, Estonia

## Forum Chairs

Marite Kirikova  
Janis Stirna

Riga Technical University, Latvia  
Stockholm University, Sweden

## Industry Chair

Erik Proper

CRP Henri Tudor, Luxembourg

## Doctoral Consortium Chairs

Isabelle Mirbel  
Barbara Pernici

University of Nice, France  
Politecnico di Milano, Italy

## Publication Chair

Claudia P. Ayala

Technical University of Catalonia, Spain

## Publicity Chairs

Rébecca Deneckère  
Carina Alves

University of Paris 1, France  
Universidade Federal de Pernambuco, Recife,  
Brazil

Marta Indulska  
Lin Liu  
Keng Siau

University of Queensland, Australia  
Tsinghua University, China  
University of Nebraska-Lincoln, USA

## Webmaster

Lukasz Malon

University of Gdańsk, Poland

## Program Committee Board

Marko Bajec, Slovenia  
Zohra Bellahsene, France  
João Falcão e Cunha, Portugal  
Giancarlo Guizzardi, Brazil  
John Krogstie, Norway  
Haris Mouratidis, UK  
Oscar Pastor López, Spain

Barbara Pernici, Italy  
Anne Persson, Sweden  
Michael Petit, Belgium  
Erik Proper, Luxembourg  
Colette Rolland, France  
Camille Salinesi, France  
Pnina Soffer, Israel



## Program Committee

Wil van der Aalst, The Netherlands  
Daniel Amyot, Canada  
Paris Avgeriou, The Netherlands  
Luciano Baresi, Italy  
Boalem Benatallah, Australia  
Giuseppe Berio, France  
Nacer Boudjlida, France  
Marco Brambilla, Italy  
Jordi Cabot, France  
Albertas Čaplinskas, Lithuania  
Silvana Castano, Italy  
Jaelson Castro, Brazil  
Corine Cauvet, France  
Isabelle Comyn-Wattiau, France  
Panos Constantopoulos, Greece  
Alfredo Cuzzocrea, Italy  
Fabiano Dalpiaz, Italy  
Valeria De Antonellis, Italy  
Rébecca Deneckère, France  
Eric Dubois, Luxembourg  
Johann Eder, Austria  
Mariagrazia Fugini, Italy  
Guido Geerts, USA  
Paolo Giorgini, Italy  
Cristina Gómez, Spain  
Stefanos Gritzalis, Greece  
Michael Grossniklaus, USA  
Irit Hadar, Israel  
Terry Halpin, Australia and Malaysia  
Markus Helfert, Ireland  
Brian Henderson-Sellers, Australia  
Willem-Jan van den Heuvel,  
The Netherlands  
Marta Indulska, Australia  
Matthias Jarke, Germany  
Manfred Jeusfeld, The Netherlands  
Paul Johannesson, Sweden  
Ivan Jureta, Belgium  
Haruhiko Kaiya, Japan  
Dimitris Karagiannis, Austria  
Panagiotis Karras, USA  
Evangelia Kavakli, Greece  
Marite Kirikova, Latvia  
Christian Kop, Austria  
Régine Laleau, France  
Alexei Lapouchnian, Canada  
Wilfried Lemahieu, Belgium  
Michel Léonard, Switzerland  
Kecheng Liu, UK  
Lin Liu, China  
Peri Loucopoulos, UK  
Kalle Lyytinen, USA  
Lech Madeyski, Poland  
Raimundas Matulevičius, Estonia  
Jan Mendling, Austria  
Isabelle Mirbel, France  
Jerzy Nawrocki, Poland  
Moirra Norrie, Switzerland  
Selmin Nurcan, France  
Andreas Oberweis, Germany  
Antoni Olivé, Spain  
Andreas Opdahl, Norway  
Michael Pantazoglou, Greece  
Mike Papazoglou, The Netherlands  
Gilles Perrouin, Belgium  
Dimitris Plexousakis, Greece  
Geert Poels, Belgium  
Klaus Pohl, Germany  
Jaroslav Pokorný, Czech Republic  
Naveen Prakash, India  
Sudha Ram, USA  
Ruth Raventós, Spain  
Manfred Reichert, Germany  
Iris Reinhartz-Berger, Israel  
Dominique Rieu, France  
Bill Robinson, USA  
Michael Rosemann, Australia  
Gustavo Rossi, Argentina  
Matti Rossi, Finland  
Antonio Ruiz Cortés, Spain  
Motoshi Saeki, Japan  
Ana Šaša, Slovenia  
Keng Siau, USA  
Guttorm Sindre, Norway  
Monique Snoeck, Belgium  
Janis Stirna, Sweden

Arnon Sturm, Israel  
David Taniar, Australia  
Ernest Teniente, Spain  
Bernhard Thalheim, Germany  
Juan-Carlos Trujillo Mondéjar, Spain  
Irene Vanderfeesten, The Netherlands  
Olegas Vasilecas, Lithuania  
Yannis Vassiliou, Greece  
Yair Wand, Canada  
Barbara Weber, Austria

Jan Weglarz, Poland  
Hans Weigand, The Netherlands  
Mathias Weske, Germany  
Jon Whittle, UK  
Roel Wieringa, The Netherlands  
Eric Yu, Canada  
Jelena Zdravkovic, Sweden  
Didar Zowghi, Australia  
Meiyun Zuo, China

## Additional Referees

Ebrahim Khalil Abbasi  
David Ameller  
Birger Andersson  
Ofer Arazy  
George Athanasopoulos  
George Baryannis  
David Bednarek  
Saeed Ahmadi Behnam  
Palash Bera  
Maria Bergholtz  
Alexander Bergmayr  
Pierre Berlioux  
Devis Bianchini  
Bojana Bislimovska  
Quentin Boucher  
Renaud Bougueng  
Hugo Bruneliere  
Robert Buchmann  
Fabian Büttner  
Cristina Cabanillas  
Javier Canovas  
Samira Si-said Cherfi  
Jan Claes  
Anthony Cleve  
Maya Daneva  
Xavier Devroey  
Thang Le Dinh  
Prokopios Drogkaris  
Daniel J. Dubois  
Hubert Dubois  
Pascal van Eck  
Hanna Farah

Hassan Fatemi  
Marie-Christine Fauvet  
Pablo Fernández  
Alfio Ferrara  
Matthias Galster  
Luciano Garcia-Banuelos  
José María García  
Guido Geerts  
Frederic Gervais  
Sepideh Ghanavati  
David Gil  
Sam Guinea  
Uwe van Heesch  
Martin Henkel  
Pieter Hens  
Nico Herzberg  
André Heuer  
Toma. Hovelja  
Vedran Hrgovcic  
Charlotte Hug  
Slinger Jansen  
Jakub Jurkiewicz  
Jaap Kabbedijk  
Christos Kalloniatis  
Maria Karyda  
Spyros Kokolakis  
Haridimos Kondylakis  
Sylwia Kopczynska  
Yannis Kotidis  
Kyriakos Kritikos  
Sylvain Kubicki  
Matthias Kunze

Alexander Lübbe  
Andreas Lanz  
Cyprian Laskowski  
Algirdas Laukaitis  
Stéphane Leblanc  
Yang Lee  
Henrik Leopold  
Weizi Li  
Zengyang Li  
Hector Llorens  
Mathias Lohrmann  
Amel Mammari  
Michal Mackowiak  
Andreas Metzger  
Andreas Meyer  
Bartosz Michalik  
Michele Melchior  
Geert Monsieur  
Stefano Montanelli  
Kafui Monu  
Christoph Moser  
Luca Mottola  
Alain Mouttham  
Miroslaw Ochodek  
Karolyne Oliveira  
Artur Opala  
Dieter Pfoser  
João Pimentel  
Jakob Pinggera  
Pierluigi Plebani  
Richard Pohl  
Artem Polyvyanyy  
Viara Popova  
Alireza Pourshahid  
Nicolas Prat  
Anna Queralt  
Evangelos Rekleitis

Alain Renault  
Manuel Resinas  
Panagiotis Rizomiliotis  
Oscar Romero  
Anne Rousseau  
Emanuel Santos  
Brahmananda Sapkota  
Farida Semmak  
Azalia Shamsaei  
Carla Silva  
Patricio Silva  
Aidas Smaizis  
Sergey Smirnov  
Jakub Starka  
Vanessa Stricker  
Yehia Taher  
Virginie Thion  
Robert Tairas  
Giordano Tamburrelli  
Chekfoung Tan  
Jasmine Tehrani  
Virginie Thion  
Massimo Tisi  
Dan Tofan  
Pablo Trinidad  
Justas Trinkunas  
Wilfrid Utz  
Carmen Vaca  
Gaia Varese  
Antonio Villegas  
Niksa Visic  
Changrui Yu  
A. Zarghami  
Chrysostomos Zeginis  
Iyad Zikra  
Srdjan Zivkovic  
Stefan Zugal

# Table of Contents

## Keynotes

The Future of Enterprise Systems in a Fully Networked Society . . . . .	1
<i>Michele Missikoff</i>	
Challenges for Future Platforms, Services and Networked Applications . . . . .	19
<i>Krzysztof Kurowski</i>	

## Business Process Model Analysis

Understanding Business Process Models: The Costs and Benefits of Structuredness . . . . .	31
<i>Marlon Dumas, Marcello La Rosa, Jan Mendling, Raul Mäesalu, Hajo A. Reijers, and Nataliia Semenenko</i>	
Aggregating Individual Models of Decision-Making Processes . . . . .	47
<i>Razvan Petrusel</i>	
Generating Natural Language Texts from Business Process Models . . . . .	64
<i>Henrik Leopold, Jan Mendling, and Artem Polyvyanyy</i>	

## Service and Component Composition

Towards Conflict-Free Composition of Non-functional Concerns . . . . .	80
<i>Benjamin Schmeling, Anis Charfi, Marko Martin, and Mira Mezini</i>	
Fuzzy Verification of Service Value Networks . . . . .	95
<i>Iván S. Razo-Zapata, Pieter De Leenheer, Jaap Gordijn, and Hans Akkermans</i>	
Cooperative Service Composition . . . . .	111
<i>Nikolay Mehandjiev, Freddy Lécué, Martin Carpenter, and Fethi A. Rabhi</i>	

## Language and Models

Abstracting Modelling Languages: A Reutilization Approach . . . . .	127
<i>Juan de Lara, Esther Guerra, and Jesús Sánchez-Cuadrado</i>	
Logical Invalidations of Semantic Annotations . . . . .	144
<i>Julius Köpke and Johann Eder</i>	

Uniform Access to Non-relational Database Systems: The SOS Platform ..... 160  
*Paolo Atzeni, Francesca Bugiotti, and Luca Rossi*

**System Variants and Configuration**

Variability as a Service: Outsourcing Variability Management in Multi-tenant SaaS Applications ..... 175  
*Ali Ghaddar, Dalila Tamzalit, Ali Assaf, and Abdalla Bitar*

Configurable Process Models for the Swedish Public Sector ..... 190  
*Carl-Mikael Lönn, Elin Uppström, Petia Wohed, and Gustaf Juell-Skielse*

Aligning Software Configuration with Business and IT Context ..... 206  
*Fabiano Dalpiaz, Raian Ali, and Paolo Giorgini*

**Process Mining**

Mining Inter-organizational Business Process Models from EDI Messages: A Case Study from the Automotive Sector ..... 222  
*Robert Engel, Wil M.P. van der Aalst, Marco Zapletal, Christian Pichler, and Hannes Werthner*

Data Transformation and Semantic Log Purging for Process Mining .... 238  
*Linh Thao Ly, Conrad Indiono, Jürgen Mangler, and Stefanie Rinderle-Ma*

Improved Artificial Negative Event Generation to Enhance Process Event Logs ..... 254  
*Sepe K.L.M. vanden Broucke, Jochen De Weerd, Bart Baesens, and Jan Vanthienen*

Efficient Discovery of Understandable Declarative Process Models from Event Logs ..... 270  
*Fabrizio M. Maggi, R.P. Jagadeesh Chandra Bose, and Wil M.P. van der Aalst*

**Ontologies**

OtO Matching System: A Multi-strategy Approach to Instance Matching ..... 286  
*Evangelia Daskalaki and Dimitris Plexousakis*

SCIMS: A Social Context Information Management System for Socially-Aware Applications ..... 301  
*Muhammad Ashad Kabir, Jun Han, Jian Yu, and Alan Colman*

Ontological Meta-properties of Derived Object Types . . . . .	318
<i>Giancarlo Guizzardi</i>	

An Automatic Approach for Mapping Product Taxonomies in E-Commerce Systems . . . . .	334
<i>Lennart J. Niderstigt, Steven S. Aanen, Damir Vandić, and Flavius Fräsincar</i>	

## Requirements and Goal Models

Requirements-Driven Root Cause Analysis Using Markov Logic Networks . . . . .	350
<i>Hamzeh Zawawy, Kostas Kontogiannis, John Mylopoulos, and Serge Mankovskii</i>	

Validation of User Intentions in Process Models . . . . .	366
<i>Gerd Gröner, Mohsen Asadi, Bardia Mohabbati, Dragan Gašević, Fernando Silva Parreiras, and Marko Bošković</i>	

Agile Requirements Evolution via Paraconsistent Reasoning . . . . .	382
<i>Neil A. Ernst, Alexander Borgida, John Mylopoulos, and Ivan J. Jureta</i>	

## Compliance

On Analyzing Process Compliance in Skin Cancer Treatment: An Experience Report from the Evidence-Based Medical Compliance Cluster (EBMC <sup>2</sup> ) . . . . .	398
<i>Michael Binder, Wolfgang Dorda, Georg Duftschmid, Reinhold Dunkl, Karl Anton Fröschl, Walter Gall, Wilfried Grossmann, Kaan Harmankaya, Milan Hronsky, Stefanie Rinderle-Ma, Christoph Rinner, and Stefanie Weber</i>	

Compliance Evaluation Featuring Heat Maps (CE-HM): A Meta-Modeling-Based Approach . . . . .	414
<i>Dimitris Karagiannis, Christoph Moser, and Arash Mostashari</i>	

A Compliance Management Ontology: Developing Shared Understanding through Models . . . . .	429
<i>Norris Syed Abdullah, Shazia Sadiq, and Marta Indulska</i>	

## Monitoring and Prediction

Patterns to Enable Mass-Customized Business Process Monitoring . . . . .	445
<i>Marco Comuzzi, Samuil Angelov, and Jochem Vonk</i>	

Predicting QoS in Scheduled Crowdsourcing . . . . .	460
<i>Roman Khazankin, Daniel Schall, and Schahram Dustdar</i>	

## Services

Towards Proactive Web Service Adaptation . . . . .	473
<i>Ahmed Moustafa and Minjie Zhang</i>	
Clouding Services for Linked Data Exploration . . . . .	486
<i>Silvana Castano, Alfio Ferrara, and Stefano Montanelli</i>	

## Case Studies

Business Intelligence Modeling in Action: A Hospital Case Study . . . . .	502
<i>Daniele Barone, Thodoros Topaloglou, and John Mylopoulos</i>	
RadioMarché: Distributed Voice- and Web-Interfaced Market Information Systems under Rural Conditions . . . . .	518
<i>Victor de Boer, Pieter De Leenheer, Anna Bon, Nana Baah Gyan, Chris van Aart, Christophe Guéret, Wendelien Tuyp, Stephane Boyera, Mary Allen, and Hans Akkermans</i>	
Publication of Geodetic Documentation Center Resources on Internet . . . . .	533
<i>Marcin Luckner and Waldemar Izdebski</i>	

## Business Process Design

Business Process Design from Virtual Organization Intentional Models . . . . .	549
<i>Luz María Priego-Roche, Lucinéia Heloisa Thom, Agnès Front, Dominique Rieu, and Jan Mendling</i>	
A Novel Approach to Modeling Context-Aware and Social Collaboration Processes . . . . .	565
<i>Vitaliy Liptchinsky, Roman Khazankin, Hong-Linh Truong, and Schahram Dustdar</i>	
Process Redesign for Liquidity Planning in Practice: An Empirical Assessment . . . . .	581
<i>Jochen Martin, Tobias Conte, and Athanasios Mazarakis</i>	

## Feature Models and Product Lines

Building Information System Variants with Tailored Database Schemas Using Features . . . . .	597
<i>Martin Schäler, Thomas Leich, Marko Rosenmüller, and Gunter Saake</i>	

Evolutionary Search-Based Test Generation for Software Product Line Feature Models . . . . .	613
<i>Faezeh Ensan, Ebrahim Bagheri, and Dragan Gašević</i>	
Feature Model Differences . . . . .	629
<i>Mathieu Acher, Patrick Heymans, Philippe Collet, Clément Quinton, Philippe Lahire, and Philippe Merle</i>	
<b>Human Factor</b>	
Wiki Refactoring as Mind Map Reshaping . . . . .	646
<i>Gorka Puente and Oscar Díaz</i>	
Purpose Driven Competency Planning for Enterprise Modeling Projects . . . . .	662
<i>Janis Stirna and Anne Persson</i>	
Work Experience in PAIS – Concepts, Measurements and Potentials . . . .	678
<i>Sonja Kabicher-Fuchs and Stefanie Rinderle-Ma</i>	
<b>Tutorials</b>	
Ontological Foundations for Conceptual Modeling with Applications . . . .	695
<i>Giancarlo Guizzardi</i>	
Designing Technical Action Research and Generalizing from Real-World Cases . . . . .	697
<i>Roel Wieringa</i>	
Improvisational Theater for Information Systems: An Agile, Experience-Based, Prototyping Technique . . . . .	699
<i>Martin Mahaux and Patrick Heymans</i>	
Full Model-Driven Practice: From Requirements to Code Generation . . .	701
<i>Óscar Pastor and Sergio España</i>	
<b>Author Index</b> . . . . .	703



# The Future of Enterprise Systems in a Fully Networked Society

Michele Missikoff

IASI-CNR, Viale Manzoni 30, 00185, Rome, Italy  
michele.missikoff@iasi.cnr.it

**Abstract.** The industrialised countries are living in an crucial phase for their economies and social systems. Computers and digital technologies have traditionally played a central role for the development of the socio-economic systems, and of enterprises in particular. But if the socio-economic scenario undergoes profound changes, also the enterprises need to change and, consequently, enterprise computing. In the next decade, enterprise software systems cannot continue to evolve along the beaten paths, there is an urgent need for new directions in the ways enterprise software is conceived, built, deployed and evolved. In this paper we illustrate the main outcome of the Future Internet Enterprise Systems (*FInES*) Research Roadmap 2025<sup>1</sup>, a study on the future research lines of enterprise systems, promoted by the *FInES* Cluster of the European Commission (DG Information Society and Media.)

**Keywords:** Future information systems, enterprise systems, new enterprise values, enabling technologies, knowledge technologies.

## 1 Introduction

This paper has the objective of tracing a synthetic picture of a possible future for the enterprise systems and, connected to them, some research objectives. The presented material is drawn upon the outcome of the *FInES* Research Roadmap Task Force. The work has started by analysing a large number of documents issued by the European Commission, such as the Digital Agenda for Europe<sup>2</sup> and the Innovation Union Europe 2020<sup>3</sup>, and, at a more specific level, sectorial studies on technology trends, business trends, etc. Furthermore, there have been several meetings of the *FInES* Cluster and specifically appointed groups and committees that have supported the elaboration of the report.

The work has been characterised by a knowledge management approach that produced an organization articulated in four main knowledge spaces: socio-economic,

---

<sup>1</sup> [http://cordis.europa.eu/fp7/ict/enet/documents/fines-research-roadmap-v20\\_en.pdf](http://cordis.europa.eu/fp7/ict/enet/documents/fines-research-roadmap-v20_en.pdf)

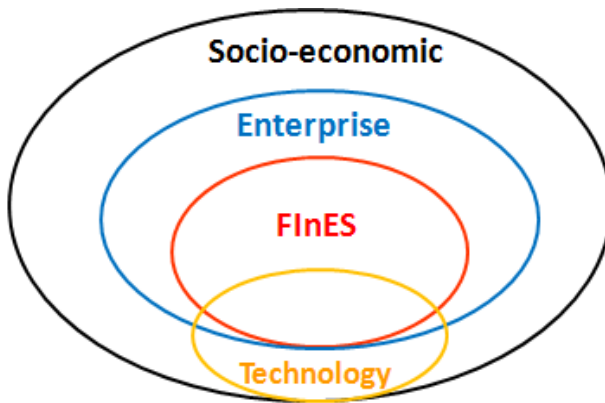
<sup>2</sup> [http://ec.europa.eu/information\\_society/digital-agenda/index\\_en.htm](http://ec.europa.eu/information_society/digital-agenda/index_en.htm)

<sup>3</sup> [http://ec.europa.eu/research/innovation-union/index\\_en.cfm](http://ec.europa.eu/research/innovation-union/index_en.cfm)

enterprise, enterprise systems, and technology. This first section provides a brief recap of such a knowledge. Note that in some cases the terminology has been drawn upon the *FInES* Research Roadmap, for a greater fidelity to the original document, although sometimes it may appear to be 'unconventional'.

### 1.1 The Reference Framework

As anticipated, the research roadmapping activity has been organised as a knowledge base creation activity. Then, we initially identified four knowledge spaces that form the *FInES* Universe; they are briefly introduced below, bearing in mind that our main focus is on the third space, where the *FInES* research challenges have been identified. However, the first two spaces are important to understand the context, while the fourth is the space that provide the technological support.



**Fig. 1.** The four knowledge spaces of *FInES* Research Roadmap

1. **Socio-economic Space** – It represents the larger context in which enterprises operate. It includes topics such as the social responsibility of enterprises, the impact on the environment and their carbon footprint, including the system of values that, in our view, goes beyond the pure financial dimension.
2. **Enterprise Space** – Here we address the key characteristics of future enterprises, the emerging business and production paradigms, new governance and organization models, new forms of cooperation: all geared towards a continuous innovation quest. This space includes the investigation on new styles in the relationships with customers, yielding new market forms and logics.
3. **Enterprise Systems, Platforms, and Applications Space** – this space is specifically concerned with *FInES* Cluster themes, i.e., with the ICT solutions and socio-technical systems aimed at supporting the emerging future enterprises, largely operating over the Future Internet. Continuous Business/IT Alignment is another key issue in this space.

4. **Enabling Technology Space** – this is the knowledge space that concerns the ICT support to *FInES*, including: Future Internet solutions, knowledge technologies, cooperation and interoperability, advanced trust and security services, etc., that will be necessary for the development of a *FInES*. We know that ICT solutions will be evolving according to their own strategies and trajectories, so it is important to understand what ICT solutions will be available ‘by default’ and what solutions will need to be ‘solicited’ for the purpose of *FInES*.

The four knowledge spaces will be elaborated in more details in the following sections.

## 2 A Vision on the Socio-economic Space

The Socio-economic space represents the larger context in which enterprises operate, interacting with the other players and the environment, aiming at increasing the value production and well-being, while satisfying customers’ needs. In this frame, it is important to analyse the trends that characterize the various drivers of societal aspects, in order to forecast what our society will be like in ten years from now (avoiding a systematic forecast that is outside the scope of the this work.) In elaborating our vision of the future, we will focus on the impact that the socio-economic space will have on the way enterprises will operate to achieve their objectives. In this analysis, we intend to consider a societal context where there are values that go beyond the pure economic dimension, such as ethical values and social responsibility, transparency, impact on the environment and carbon footprint.

### 2.1 The Need for a Socio-economic Discontinuity

Since more than a decade, the Western economies are facing a troubled phase where economic crises follow one another. Today there is large agreement that the existing socio-economic models cannot continue to exist as they used to be: we reached a point of discontinuity (a point of ‘bifurcation’, according to Complexity Theory.) There are a number of significant phenomena that anticipated such a change: from the serious economic crises of the last decade to the enormous sovereign debts accumulated by the Western countries, and to the limited expansion of their economies, opposed to the marked growth of the emerging economies (well represented by the so-called BRIC: Brazil, Russia, India, China). The marked growth of the latter have an impact on a global scale, pushing upwards the costs of raw material and natural resources while competing against Western economies with low cost goods and services (Emerging economies represent also new expanding markets that, unfortunately, are addressed with the ‘usual’ consumerism approach.) The mentioned signs have been anticipated and analysed by a number of experts, such as David C. Korten when talking about the advent of ‘The Perfect Economic Storm’ [1], and the related consequences: from the failure of the financial systems, to the

deterioration of the environment and the increase of social inequality. In essence, there are clear signs that Western countries, and Europe in particular, cannot proceed along the beaten paths, just practicing ‘business as usual’. The Western development models require a change of paradigm to guarantee that people will maintain (not to mention improve) the current standard of life.

## **2.2 Different Growth Rates for Wealth and Well-Being**

The key problem of Europe for the next decade will be to find a socio-economic model capable of achieving a growing social well-being in absence of an equivalent growth of the wealth produced by the economic system. To achieve such a double speed socio-economic model, it is necessary to proceed along different lines. Primarily addressing the economic development model, where the established mechanisms, based on consumerism, with a parallel expansion of production and consumption, needs to be revisited. With this respect, it is noticeable that there is a growing awareness in some sectors of the Society and a cultural trend that promotes a set of new socio-economic values. A number of essays and publications provide good clues towards the interpretation of the riches of a country by means of a number of indicators that go beyond pure monetary values to measure the wealth of a country (see for instance the well-known Stiglitz-Sen-Fitoussi Report [2]). Other directions propose an approach that considers the possibility of satisfying the needs of people in different ways with respect to the society of consumption (e.g., see the ‘Economics of Enough’, by Diane Coyle). Furthermore, there are studies indicating how to approach a future characterised by a limited growth (see Serge Latouche and his ‘graceful de-growth [3]) and how to cope at best with such a perspective. The idea is that, having lived a long period of (relative) abundance, we have large margins for improving the life quality by means of optimization, reuse, refurbishing, etc., in essence, using better and longer what we already have.

## **2.3 Towards a Totally Connected Society**

The above sketched scenario needs new forms of social cohesion to be achieved. Internet is changing the way people know each other, get in touch, exchange information, opinion, knowledge; we are rapidly evolving towards a totally connected society, where cultural interoperability will be at the basis of new forms of social innovation [4]. But also solidarity and new types of subsidiary economies (e.g., advanced forms of private-public partnerships and the Third Sector) need to be developed, aiming at exploring new ways of production and consumption for goods and services (i.e., producing to live better with less.) Accordingly, the current notion of ‘job market’ will progressively evolve, leaving the scene to new forms of enterprise and productive occupation (e.g., with the advent of ‘*workeprenuer*’ as a figure that synthesises a self-employed worker, consultant, flexible employee), jointly

with new solutions for social protection (e.g., evolving along the line of '*flexicurity*'). Finally, the role of Future Internet is central, to support also new forms of social and political participation. (e.g., deliberative democracy.)

## 2.4 Innovation in a Knowledge-Based Society

Since more than a decade (see the Lisbon Strategy) it is widely shared that Europe needs to evolve towards a knowledge-based economy<sup>4</sup>. But this objective resulted harder to be achieved than expected (due also to the recurring economic crises.) It appeared that the adoption of knowledge technologies is not enough, the heart of the knowledge-based Society is the people. In the next 'Decade of Discontinuity', it is necessary to foresee a socio-economic model where technological development will take place having the people (citizens, workers, entrepreneurs, etc.) at the centre. Only people are able to deploy the creativity that, supported by the necessary knowledge, is able to promote innovation and social growth.

In this socio-economic frame enterprises play a central role, since they represent the primary source of wealth production, being at the same time one of the key players of the delineated social and cultural evolution. Just think about the marketing campaigns, where advertising is based on the promotion of certain life models. Also here we can see important signs of change that will presumably continue in the future. For instance, today we see many ads where a given product (a car, a pair of glasses, etc.) is publicised connecting it to a style of life respectful of the environment: an evident sign that the marketing strategies are changing their 'mantra'. The role of enterprises is, and will continue to be, central also in the cultural development of a Society. Then, we expect enterprises to be the protagonists of such transformations in different contexts: internally, e.g., addressing their own organization, human resources, production and logistics models, and, externally, with the marketing strategies and customer relationships; then, in the socio-political arena, with their lobbying capabilities. Enterprises can be one of the central 'engines' for the coming decade of innovation and discontinuity.

## 3 The Future of Internet-Based Enterprises

The Enterprise space is where we delineate the key characteristics of future enterprises, for which we expected that their success will be based on knowledge assets, skills and competencies, creativity and innovation, trust and security, awareness of innovation opportunities and related risks (with a wise risk taking attitude) and, last but not least, the capacity of adopting ICT solutions constantly aligned with business needs.

As anticipated, the next decade is expected to see deep changes in the way enterprises operate, mainly due to both a changed socio-economic scenario and the

---

<sup>4</sup> [http://www.consilium.europa.eu/uedocs/cms\\_data/docs/pressdata/en/ec/00100-r1.en0.htm](http://www.consilium.europa.eu/uedocs/cms_data/docs/pressdata/en/ec/00100-r1.en0.htm)

availability of important enabling ICT innovations, driven by the Future Internet. In this frame, the current section addresses two main themes:

- The first theme concerns a number of key characteristics of Future Internet-based enterprises, referred to as **Qualities of Being**, that we believe will be central in a virtuous development of the socio-economic system of European enterprises. Such qualities are sufficiently general to be applied to the majority of enterprises, independently of their size, nationality, or industrial sector.
- The second theme proposes an **operational framework**, essentially a behavioural paradigm for future enterprises in organising their activities. Such an operational framework represents also a bridge towards the next section, where the systems supporting the future enterprises will be addressed, together with the related research challenges.

### 3.1 The Qualities of Being of the Future Internet-Based Enterprises

This section reports the key Qualities of Being (QB) that are considered strategic for the enterprises of the future, independently of the industrial sector, the size, the organizational model they follow. Such QBs are considered as directions towards which to proceed, requiring specific organization models and activities to be adopted, rather than as targets to be met once forever. Please note that the presented QB are not orthogonal one another, instead they are mutually interlinked and complementary. The identified qualities, that refer to different aspects of the enterprise life, are tagged with specific keywords (useful to find a better elaboration in the original *FInES* Research Roadmap document.)

The first quality emerges from the constant shift of enterprise systems focus from enterprise resource planning (see ERP platforms) to the support of continuous innovation, with the parallel need to develop creativity and skills in an open and cooperative environment (*Inventive Enterprise*) [5]. New models of cooperative work are another characteristic feature of the future enterprise, which will be able to adopt and exploit new production and organizational models, based on social media (*Community-oriented Enterprise*). The new models and organizational structures will be conceived to put the human skills and capabilities, but also the needs and quality of life of the workers, at the center of the enterprise (*Humanistic Enterprise*.) The cooperation will extend beyond the boundaries of the company, which will gradually fade away making it impossible to distinguish the 'inside' and the 'outside' [6]. It will be the advent of the *Liquid Enterprise*, where the rigidity of today's organizational models and employment forms will be replaced by new models, based on different levels of involvement and work flexibility. As anticipated, value production activities will be carried out by new professionals (such as '*workpreneur*', synthesising the employer, freelancer, and employee). New professional figures and new working relationships are needed to quickly respond to the unceasing challenges of the market and to stimulate continuous improvement required by the global competition and made possible by the adoption of new technological solutions [7]. And the answers

have to be implemented quickly, through flexible productive and organizational models (*Agile Enterprise*.)

The future enterprise will be able to interpret the needs of different markets, scattered all over the planet, understanding local specificities and constraints while maintaining an overall view of the opportunities (*Glocal Enterprise*.) Finally, a great change will be in the firm's ability to create winning strategies combining different values and strategies, beyond the profits, developing environmental awareness, social responsibility, attention for the private lives (*Sustainable Enterprise*.)

With the evolution of the *Internet of Things* information will be generated and consumed in large part by devices, equipments, and technology infrastructures with high levels of autonomy and intelligence (i.e., advanced processing capabilities), but also products and business objects will play an active role, cooperating and forming a sort of 'nervous system', perceptive, active and reactive within the enterprise (*Sensing Enterprise*). Information flows will grow with a geometric progression, at different levels, concerning both the value production operations and strategic decisions for organizational and operational models, supporting the development of strategies in medium to long-term visions. In essence, the company will be able to collect, organize, process and distribute useful knowledge in a targeted way to different agents attending at different activities. But the enterprise systems will be strongly based on the knowledge of the company and of the scenario in which the latter operates (*Cognitive Enterprise*), as described in the next section. In essence, knowledge will be the primary asset in the Knowledge Economy, it will allow the different economic and social players to operate in delivering economic value and social wellbeing, provided that such actors will be able to understand and interpret at best the continuous changing scenarios.

### 3.2 The Operational Dimension

The above enterprise profiles, the Qualities of Being of Future Internet-based Enterprises, are not easily achieved. They require for an enterprise a marked attitude towards changing and the capacity of continuous improvement and innovation.

This implies the adoption of entirely new operational approaches, with a shift in the priorities from the management of existing assets, where established systems and practices are largely available, to the management of the future, through methods and tool aimed at seizing the innovation opportunities, consistently reshaping the organization, business processes, and activities. A paradigm that encourages continuous change is represented by the fractal model [8]: global, in that it can be repeated in different areas of the company, and iterative, as the pattern is applicable at different operational levels. Having this in mind, we introduce a framework based on 6 operational stages.

**Invent.** This is the first stage of the innovation cycle that consists in the preliminary identification of new solutions to be adopted, in any possible areas of the enterprise (from production to HR, from logistics to management to marketing).

**Plan.** Planning is required in order to devise a trajectory capable of transforming a new idea into a concrete solution to be adopted. Here, techniques such as resource analysis, SWOT, and risk assessment play a central role. Simulation and *what-if* analysis can also improve the understanding on the expected cost and performance (including ROI) of the new solution.

**Build.** This is the stage where the new solutions are actually implemented. Again, this may apply to different enterprise areas and domains, e.g., building new business solutions (including new organizational models, new processes, and new capabilities), or new products, or new competencies for the employees.

**Operate.** In this stage, the new solutions and capabilities become operational, adopted as integral parts of the enterprise activities and production. The start of this phase is critical since in an Inventive Enterprise, continuous improvements and innovations need to be adopted without affecting the ongoing business (or limiting the impact, in case of radical changes.)

**Monitor and Manage.** This stage is actually overlapping the previous one, having a specific focus on assessing how the innovation is performing. But in general, the M&M activities need to be constantly operational, also to check how the business operations are performing in ‘stable’ situations.

**Dissolute.** In enterprise operations, this phase concerns the termination of a business, a project, a product, requiring stopping activities and moving people, dismissing existing products to introduce new ones. The higher is the dynamicity of business the more this operational stage will be needed.

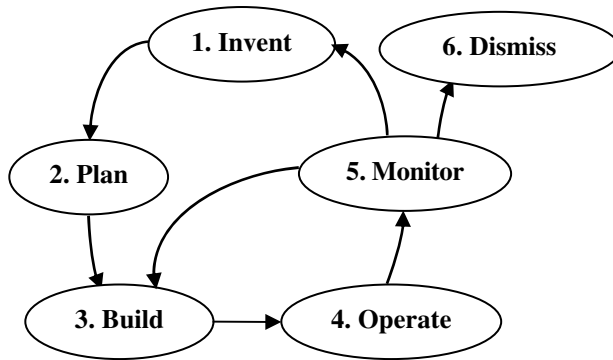


Fig. 2. The Behavioural cycle

Please note in Fig. 2 the three different circuits, where the internal loop represents the improvement cycle, the outer loop the innovation cycle and the radial arc the end of the lifecycle of a business solution.



### 3.3 Supporting the Advent of the Future Internet-Based Enterprises

An enterprise is a complex artefact, and its unique anatomy, composed of very different active (human and artificial) and passive (tangible and intangible) elements, has hindered the extensive use of Engineering disciplines. Enterprise Engineering is seen as ‘the application of knowledge, principles, and disciplines related to the analysis, design, implementation and operation of all elements associated with an enterprise’ [9]. But Enterprise Engineering cannot draw upon traditional engineering discipline that appears inadequate for the purpose. In fact, the latter stems from the Industrial Revolution and the Positivism that has the ultimate objective to keep the reality (in particular, the artificial reality) under the human control. Conversely, with the widening of the horizons and the growing of complexity, we need to accept the idea that for certain artefacts we allow for a limited control, accepting raising levels of autonomy for important business units and actors [10]. For instance, Business Process Modeling and Management will be incomplete, allowing for unplanned business operations to take place, relying on the enterprise capacity of reacting to unexpected events (see Complex Event Processing.) This vision calls for new approaches to Enterprise Engineering based, among others, on uncertainty and Complexity Theory. Furthermore, the Enterprise Engineering approach will be also intersected with social computing, leading to the so called Enterprise 2.0.

## 4 The Future Internet-Based Enterprise Systems

This section is specifically concerned with *FInES*, i.e., with the socio-technical methodologies, platforms, applications, systems, and ICT solutions aimed at supporting emerging future enterprises [11]. Here we intend to delineate the key characteristics of a *FInES* and the related research challenges to be faced in the next decade. The key objective of a *FInES* is to support enterprises in achieving the illustrated QB, while producing value for stakeholders and customers [12].

This section is organized along three basic dimensions: the **Knowledge Dimension**, since before doing it is necessary to know, the **Functional Dimension**, to see what will be the main functions of a *FInES*, and the **Engineering Dimension**, to investigate new development techniques, with a specific focus on software applications.

### 4.1 The Knowledge Dimension

The degree of penetration of the ICT in the production reality will continue to a point where all that we need to know about the enterprise will be coded in digital form, equally accessible and processable by computers and (mediated by the latter) by humans. Such an extensive digital representation will be referred to as *Unified Digital Enterprise (UDE)*. Today we are very close to this, if you consider the massive amount of documents and data that are electronically produced, acquired, and

circulated within an enterprise [13]. The knowledge dimension we consider here has basically a methodological nature: the main challenge being to identify methods and paradigms aimed at modelling the enterprise reality, including resources and objectives, to be directly used by business experts (substantially reducing the role of knowledge engineers.) To reach this goal we need to promote high quality research in several directions. Among the key ones, we have the following Research Challenges (Again, the interested reader can use **RC** tags as pointers to the more elaborated *FInES* Research Roadmap document.)

In the knowledge representation and reasoning area, research has already produced important results, based on various forms of logic and algebra, aimed at processing and consistently managing enterprise knowledge bases (e.g., inference engines, query systems, truth maintenance systems). However, not as much has been done on the application side, in particular to achieve an actual impact on the business world, for instance focusing on enterprise architectural frameworks. The first **RC1** therefore refers to advanced methods to obtain a unified view of enterprise knowledge, and the related methodologies. But different business sectors require different solutions, and rigorous knowledge requires a high level of disciplinary specialization, with diversified content and applications that contrast with the need to reach an open and integrated view of an enterprise knowledge base. *Linked Open Knowledge* (along the line of *Linked Data*) represents a promising direction of research to broaden and interlink expertise and knowledge (**RC2**), overcoming the various enterprise barriers (cultural, organizational, spatial, etc.) existing today. Finally, it is well known the impossibility of building an accurate model of reality, if it exceeds a certain level of complexity<sup>5</sup>. In essence, we need to accept the idea that we will increasingly develop artefacts (such as an enterprise, with its application systems) that we can only partially know, thus working in the absence of overall models able to predict behaviours and evolutionary trajectories. Here the studies on complex systems will help us living in a future of 'limited sovereignty' over the business reality (**RC3**).

## 4.2 The Functional Dimension of a *FInES*

Traditionally, enterprise software applications (ESA) are primarily conceived to support the day-by-day value production of an enterprise, with an optimal management and planning of the resources (ref. ERP). There are other vital functions and activities that are partially integrated, from the strategic marketing to the R&D, to financial scouting, to organizational innovation. According to the notion of a *UDE* (*Unified Digital Enterprise*), the idea is to proceed towards a totally integrated approach also from a functional point of view, where different aspects and activities are seen in a unique frame. This must be achieved in the context of a highly dynamic scenario that requires constant monitoring of internal and external events and a capacity of quickly aligning to changes, but also the capacity of generating winning discontinuities, i.e., business innovation to achieve a competitive advantage.

---

<sup>5</sup> See H. Stachowiak: Models. In Scientific thought, Some underlying concepts, methods and procedures, UNESCO 1972  
(<http://unesdoc.unesco.org/images/0000/000022/002251eo.pdf>)

In this dimension, the first research challenge (**RC4**) deals with the functional coverage of the 6 operational stages presented in the enterprise section. In particular, we consider a vision where the key challenge consisting in a unified system integrating the enterprise applications developed to support: invention, planning, building, operations, monitoring and management, and dismissing of business solutions. The second challenge of this section (**RC5**), linked to the previously introduced knowledge management issues, concerns platforms for enterprise knowledge, with advanced capabilities for collecting and aggregating knowledge coming from very diverse sources, both internal (from sensors to documents, from databases to intranet forums) and external (primarily from the Web). The (logically) integrated enterprise knowledge will be made available to people and machines in various ways, e.g., in response to queries, but also proactively, e.g., with semantic routing of the information to the info-needy actors, even if they are unconscious of the need, in the right moment, in the right form and level of detail (determined by the task currently performed.)

The third challenge of this group (**RC6**) refers to the functions of communication, community building, resource sharing, and cooperative work. Here also the goals are ambitious, reaching up to the development of new forms of collective intelligence.

### 4.3 The Engineering Dimension of a *FInES*

Having discussed the joint issues related to enterprise knowledge management and enterprise applications, we investigate now the third pillar of *FINES*: the set of methods and tools needed to develop such systems. Software Engineering, that show a long time evolution in parallel with the information systems of today, requires a profound rethinking to meet the challenges of *FInES*. Traditionally, software engineering techniques [14] have difficulties in supporting the software updates, required by the fast pace of the ever changing reality [15]. This will be even harder in a *FInES* scenario, seen its increased complexity and richness with respect to the existing enterprise systems. As anticipated, the structure and behaviour of a *FINES* will be centrally based on the enterprise knowledge collected and managed as an *UDE* repository. Here, the first challenge is to ensure a continuous alignment of the *UDE* with a business reality that continuously evolves, and that the application software is aligned with the latter (and therefore with the enterprise needs.) These objectives require major research results, as described below.

The Fig. 3 depicts the three *FInES* contexts but, being necessarily sketchy, it does not render that fact that both *UDE* and *FInES* are largely interwoven with the enterprise reality. This fact derives from the progressive spread of networks of intelligent objects. Be they physical or intangible objects, there will be a growing number of processing units capable of performing business operations, especially the more repetitive ones, with a good degree of autonomy, keeping therefore the humans out of the loop. This will be achieved pushing forward the research on autonomic systems (**RC8**).

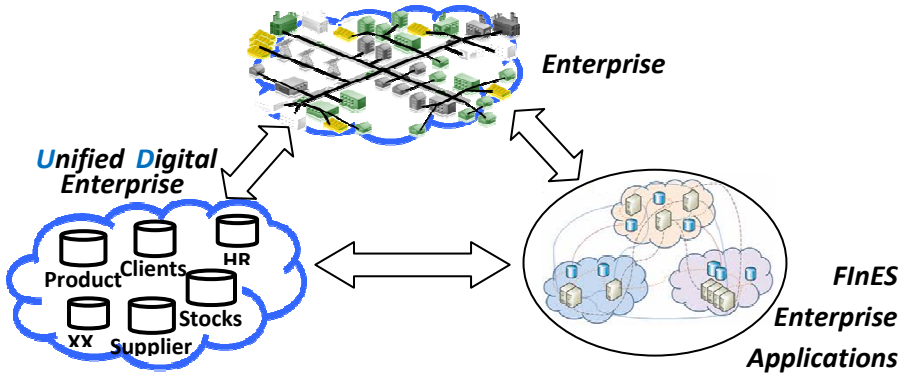


Fig. 3. The Three *FInES* contexts

In essence, when developing a *FInES*, business operations will be progressively delegated to autonomic networks, while humans will focus on high level operations and business strategies. The development of higher level functions will be based on innovative tools, conceived for end-users, supporting the analysis, modelling, and design of business solutions. Such tools will be able to generate computational business models, starting from repositories of highly reusable business objects. Such business objects will be implemented by intelligent software components, directly deployable and executable (called *FINER: Future Internet Enterprise Resource* [16]), capable of connecting and cooperating to achieve increasingly important business solutions [17]. The final architecture of a *FInES* will thus emerge from the interplay of explicit 'human-driven' (often incomplete) specification of the business solutions and the autonomic activity of cooperating smart components (we refer to this approach as *Proactive Mashup* that represents **RC7**).

A *FInES* that will be developed as indicated above, will make use of a number of advanced technologies (see next section): from intelligent agents to rule-based systems, from web services to business process management systems, from ontologies to 'traditional' legacy software, that need to effectively cooperate in performing business operations. Here also there is the need for intense research activities to devise flexible, versatile runtime platforms (**RC9**). Such platforms will be capable of hosting a rich diversity of active components, while guaranteeing continuous, on-the-fly reconfigurations, and interoperability (both of data and functions) for different applications, developed by using different technologies, without penalising the performances [18].

## 5 Future Technologies for *FInES*

Digital Technologies, from its inception, have been characterised by an impressive innovation rate, particularly relevant in the last 20+ years, after the VLSI and the telecommunication revolutions. Then, in the last decade, the diffusion of the Internet

started to exhibit a significant impact on the socio-economic sphere. ICT development will continue with a marked innovation rate, but with increasing difficulties for what concerns the socio-economic impact that will hardly maintain the same pace. One of the main problems is due to the fact that the ICT engineering methods of today will hardly scale up to tackle the enormous future challenges, in particular the size and complexity of the Future Internet and, specifically, of the *FInES* applications. Innovation and developments of the Future Internet applications will heavily challenge computer science and engineering methods and tools we use today. The massive amount of data [19] (the well-known *data deluge* issue [20]), the management and coordination of trillions of intelligent objects, with convergent and pervasive networks connecting everyone and everything: all this needs new methods and tools to developing, maintaining, and managing future large, complex, interconnected socio-technical systems [21].

An encompassing study of technology trends is outside of the scope of this document. Here we intend to focus on a subset of ICT research areas connected to the *FInES*. Furthermore, we will abstract from the basic computational and networking technologies, assuming that in the next decade they will substantially develop (according to the Moore's law that appears still valid.) The objective of this section is to provide a *FInES-oriented* point of view on a possible (and/or desirable) evolution of ICT solutions. In particular, the technological areas included in this section concern: networking, knowledge, applications, computing and storage, user interactions.

## 5.1 Future Networking Technologies

Networking (Future Internet) will be one of the key areas that will exhibit the most impressive progression, sweeping away any barrier of range (LAN, WAN, sensors networks, Zig-Bee, ...), technology (TCP/IP, Ethernet, WiFi, WiLD, ...), carrier infrastructure and mobility solutions (cable, radio signals, ...), etc. Future Internet is expected to fully and seamlessly connect nodes of a different nature, belonging to 4 categories: (1) Real entities, (2) Virtual entities, (3) Natural entities (firstly people), (4) Artificial entities, allowing them to effectively exchange data and cooperate in a secure and trusted way.

The Future Internet will be characterised by a progressive functional enrichment, fostering the commoditization of a growing number of services and facilities that will stretch the current notion of networking, supporting, e.g., advanced forms of collaboration, interoperability (ref. *ISU: Interoperability Service Utility*), trust and security, social computing [22], etc. Particular attention will be also dedicated to the correct handling of digital (multiple) identities.

## 5.2 Future Knowledge Technologies

This is another key area that will enable the development of high-performance knowledge networks aimed at managing content from any possible source or actor (natural or artificial), represented in different forms (from text to video, to structured

information) [22]. The knowledge will travel freely (safeguarding IPR) through the network to reach (on request or spontaneously, proactively) any entity that needs it (e.g., with semantic routing): don't search, the right information will find you! When you need it, where you need it. In particular, among the most interesting areas of technology, it is appropriate to mention the spreading of Fuzzy Knowledge Bases, based on the evolution of the current Linked Open Data. Another key technology is that of the Knowledge Mining, that is, methods and tools for creating 'noble' knowledge, at a conceptual level, from the analysis of large amounts of factual data, both structured and unstructured, numerical and textual. Finally, in enterprises, there will be a substantial increase of the importance of the knowledge bases for business innovation, with the contribution of open repositories of scientific findings, results of simulations, information technology and market trends, with advanced services of similarity reasoning to support 'lateral thinking.'

### 5.3 Future Application Technologies and Complex Systems Engineering

This is the key enabling technology for *FInES* in the next decade. According to MITRE, "As [enterprise] systems become increasingly large and must seamlessly interoperate with other systems in ways that were never envisioned, system engineers are bumping into the limits of the tenets, principles, and practices traditionally used in systems engineering." [23]

When trillions of intelligent entities (natural or artificial, real or virtual) will be able to connect and interoperate, the problem of developing, deploying, and maintaining software applications will be another challenge hard to be addressed with today's methods and tools. Application software engineering and enabling technologies need new developing paradigms. Specifically, it is important to base the development of future enterprise systems on the future knowledge management assets (starting with *UDE*), keeping the two areas constantly aligned (and the *UDE* aligned with the business reality).

One of the future architectural models, based on computational business entities (style *FINER* [16]), will emerge from the evolution of some existing technologies, such as: multi-agent systems and Swarm Intelligence [24]. Along this line, there are numerous challenges that the research will have to face, such as:

- How to create smart objects with marked cooperation capabilities;
- How to interconnect them in an efficient and flexible way, providing them with the ability of dynamic networking and reconfiguration (horizontal aggregation);
- How to safely grant an increasing degree of freedom and autonomy (for objects and people), self-organizing to form more complex entities, without a central authority which drives the processes (vertical aggregation).

In dealing with the construction of large interconnected systems, it is important to note that, in general, consistency will be locally ensured (in the small), but it will be hardly possible to achieve it at a global level (in the large). Furthermore, these systems will be based on heterogeneous technologies, which, as anticipated, will require new development methods and, when operational, new interoperability and governance solutions.

## 5.4 Future Computation and Storage Technologies

As anticipated, computation and storage will progressively shift away from the traditional computer centres, moving towards two different (but connected) spaces: on the clouds and on the earth. The former represents a well established and expanding technology (see market figures<sup>6</sup>), and it is plausible that the existing problems, from cloud interoperability to trust and security, to reliability [25], will be satisfactorily solved in the next years. The latter (referred to as *Swarm Computing*) will emerge from the interconnection of the trillions of smart proactive objects that will be able to locally store and process significant amounts of data, and cooperate to provide information and services at higher levels of aggregation [21]. The computation will possibly adopt a 'glocal' paradigm, going from a local dimension, with detailed and analytical computation on locally confined data, to a global dimension, with general and synthesis computation, yielding and consuming knowledge assets.

In this approach, the current vision of services and service-oriented architecture (SOA) paradigm will be absorbed and superseded by the key notion of smart objects and entities: *Smart Object-Oriented Architecture (SOOA)* that will be able to (autonomously) aggregate to provide complex services (i.e., placing in the centre the service provider rather than the specific service, with an increased semantic approach<sup>7</sup>.)

## 5.5 Future Natural Interaction

In the foreseeable future we will have two main interacting players: people and objects, with computers that will progressively disappear, behind, e.g., a car dashboard, a household appliance, a complex document representing a marketing strategy, images representing people, enterprises, etc., while we will practice less and less with interfaces to computer terminals, laptops, PCs, etc. In essence, we will be interacting with computational entities (common objects, active documents, people, etc.) mainly to perform our everyday activities, getting information and providing our feedbacks in fashions that are different from the keyboard-screen paradigms of today. We will rather have natural interactions [26] (an evolution of today Natural User Interface, for which Kinect is a good example) with the objects and the people we meet during our daily activities. Natural interactions will involve all the entities of the 4 categories indicated in the section 5.1 (object-object, human-human, object-human). Particularly relevant will be the remote, both synchronous and asynchronous, human interactions, characterized by an ever growing sophistication (avatars, acting in personalized and metaphorical ambient, holograms, and the like), yielding to new forms of participation in all the phases of the production cycle (Invent, Plan, Build, Operate, Manage, Dismiss.)

---

<sup>6</sup> [http://blogs.computerworld.com/16863/cloud\\_computing\\_by\\_the\\_numbers\\_what\\_do\\_all\\_the\\_statistics\\_mean](http://blogs.computerworld.com/16863/cloud_computing_by_the_numbers_what_do_all_the_statistics_mean)

<sup>7</sup> Beyond the functional and non functional capabilities of a service, the service provider is scrutinised for the sake of trust and security, reliability, costs, etc, but also for administrative issues, like quoting, contracting, billing, etc.

Natural interaction will extensively involve knowledge technologies and augmented reality. For instance, augmented reality will allow us to know details of a beans' can in a food store<sup>8</sup> by simply pointing a mobile device to it. Then, gesture and voice may represent other natural ways of interacting with smart objects around us.

## 6 Conclusions

The study reported in this article aimed at drawing a framework for research and innovation in the field of enterprise systems, focused on business management and innovation, which plausibly will appear in the next decade. The European task force was aware from the beginning of the difficulties inherent in an assignment of this importance, both for the breadth and complexity of the addressed subject, and for the long-term time horizon. For this reason, the base material has been carefully collected from numerous high-profile sources, while the strategic vision has been largely based on documents produced by the European Commission itself. The original contribution of this paper mainly consists in having outlined a methodological framework, based on a knowledge management approach, that guided the work of selecting and organizing the extensive material available.

A prospective study like this has a high risk that the time, and the continuous innovations that roll one after another, will quickly make obsolete important parts of the document. For this reason, we planned to complement the production of the 'paper' document with a Knowledge Wiki. Initially created from the material of this study (including the additional material collected but not presented for space reasons), it will be conceived to be periodically updated, so as to have a resource constantly aligned with the state of the art. This portal will be realized within the European project Ensemble; we aim at its success and survival, after the end of the Ensemble (August 2012), that will strongly depend on the commitment of the *FInES* Cluster and an active constituency, Wikipedia-style, that can guarantee in the future updates with high quality content.

**Acknowledgements.** This work was possible only thanks to the efforts and support of the *FInES* Cluster operating in DG Information Society and Media, Unit D4, and all European experts who actively participate in the Cluster. A special thanks to Cristina Martinez, *FInES* Cluster Chair, and Man-Sze Li, Co-chair. Moreover, an important role was that of the Scientific Advisory Group, created as part of *FInES Research Roadmap Task Force* that the author had the honour to co-ordinate (the names of those who have contributed, too numerous to be listed here, are available on the site of the Cluster: [www.fines-cluster.eu](http://www.fines-cluster.eu)). A final thanks, but no lesser, goes to the partners of the Ensemble, who provided an important support to the work.

---

<sup>8</sup> See also: 6<sup>th</sup> Sense, from MIT:

[http://www.ted.com/talks/pattie\\_maes\\_demos\\_the\\_sixth\\_sense.html](http://www.ted.com/talks/pattie_maes_demos_the_sixth_sense.html)



## References

1. Korten, D.C.: *The Great Turning* (Volume 1 of 2) (EasyRead Large Edition). ReadHowYouWant.com (2009)
2. Stiglitz, J., Sen, A., Fitoussi, J.-P.: *Commission on the Measurement of Economic Performance and Social Progress*, <http://www.stiglitz-sen-fitoussi.fr/en/index.htm>
3. Latouche, S.: *Farewell to Growth*. Polity (2009)
4. EUROPA - Press Releases - New EU report shows active labour policy can increase employment rate despite low growth, <http://europa.eu/rapid/pressReleasesAction.do?reference=IP/05/1308>
5. Chesbrough, H., Vanhaverbeke, W., West, J.: *Open Innovation: Researching a New Paradigm*. Oxford University Press, USA (2008)
6. *Beyond boundaries - The emerging work culture of independence and responsibility* (2007), <http://www.orangecoalition.com/whitepapers/download.php/8>
7. Tapscott, D., Williams, A.D.: *Wikinomics: How Mass Collaboration Changes Everything*. Portfolio Trade (2010)
8. Fingar, P.: *Fractal Enterprise Architecture and Agent-Oriented BPM: Can UML or BPMN Model a Cloud?* (2010), [http://www.bptrends.com/publicationfiles/FOUR%2009-14-10-ExtCompetition-Fractal%20Enterprise-Fingar\\_V5\\_-final.pdf](http://www.bptrends.com/publicationfiles/FOUR%2009-14-10-ExtCompetition-Fractal%20Enterprise-Fingar_V5_-final.pdf)
9. Dietz, J.L.G.: *Advances in Enterprise Engineering I: 4th International Workshop CIAO! and 4th International Workshop EOMAS, Held at CAiSE 2008, Proceedings, Montpellier, France, June 16-17*. Springer (2008)
10. Miller, P., Skidmore, P.: *Disorganization - Why future organisations must «loosen up»*, <http://www.orangecoalition.com/whitepapers/download.php/1>
11. Saenz, O.A.: *Framework for Enterprise Systems Engineering* (2005), <http://digitalcommons.fiu.edu/etd/32>
12. Melville, N., Kraemer, K., Gurbaxani, V.: *Review: information technology and organizational performance: an integrative model of IT business* (2004)
13. Maedche, A., Motik, B., Stojanovic, L., Studer, R., Volz, R.: *Ontologies for enterprise knowledge management*. *IEEE Intelligent Systems* 18, 26–33 (2003)
14. Pfleeger, S.L.: *Software engineering: Theory and practice*. Prentice Hall, Upper Saddle River (1998)
15. Ewusi-Mensah, K.: *Software Development Failures*. MIT Press (2003)
16. Angelucci, D., Missikoff, M., Taglino, F.: *Future Internet Enterprise Systems: a Flexible Architectural Approach for Innovation*. In: Domingue, J., et al. (eds.) *Future Internet Assembly*. LNCS, vol. 6656, pp. 407–418. Springer, Heidelberg (2011)
17. Hall, M.W., Gil, Y., Lucas, R.F.: *Self-Configuring Applications for Heterogeneous Systems: Program Composition and Optimization Using Cognitive Techniques*. *Proc. IEEE* 96, 849–862 (2008)
18. Sharma, S.: *Towards Holistic Performance Scorecard: A New Strategic Imperative*. *Vilakshan The XIMB Journal of Management* 5, 33–44 (2008)
19. *Future File Systems: Intelligent. Object-based Storage* (2008)
20. *Technology: The data deluge* | *The Economist*, <http://www.economist.com/node/15579717>

21. Tennenhouse, D.: Proactive computing. *Commun. ACM* 43, 43–50 (2000)
22. Helbing, D.: The FuturICT Knowledge Accelerator: Unleashing the Power of Information for a Sustainable Future. SSRN eLibrary. CCSS-10 (2010)
23. MITRE: Perspectives on Complex-System Engineering (2005)
24. Persaud, R.K.: Investigating the Fundamentals of Swarm Computing (2001), <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.9.9.222&rep=rep1&type=pdf>
25. ProblemsFacedbyCloudComputing.pdf Oggetto application/pdf, <http://dl.packetstormsecurity.net/papers/general/ProblemsFacedbyCloudComputing.pdf>
26. Valli, A.: Notes on Natural interaction (2005)

# Challenges for Future Platforms, Services and Networked Applications

Krzysztof Kurowski

Poznan Supercomputing and Networking Center,  
Noskowskiego 10, 61-704 Poznan, Poland  
krzysztof.kurowski@man.poznan.pl  
<http://www.psnk.pl>

**Abstract.** The keynote will address various ICT aspects related to future platforms, applications and services that may impact and hopefully improve the way various business and enterprise processes will be organized in the future. New development tools, easy-to-program powerful mobile devices, interactive panels, etc. enable users to prototype new IT systems with much better interfaces, to quickly setup collaborative environments and easily deploy online applications. However, many people have not realized how quickly data, computational and networking requirements of these new information systems have increased and what constraints are behind existing underlying e-Infrastructures. Therefore, the main aim of this keynote will be to share with the audience example scenarios and lessons learned. Additionally, various challenges encountered in practice for future information systems and software development methods will be briefly discussed. The keynote will be summarized by an updated report on existing research e-Infrastructures in Poland and worldwide.

**Keywords:** Future Internet, Grid Computing, Cloud Computing, Data Management, Multimedia.

## 1 Introduction

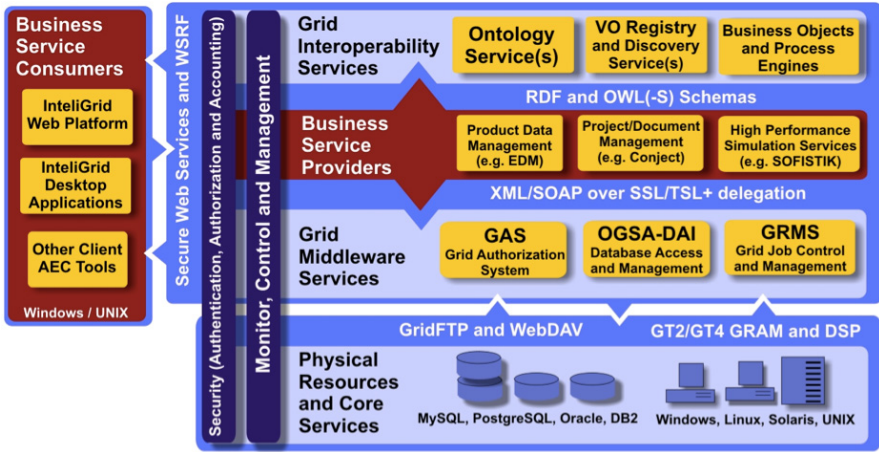
Grid computing has been promoted for more than ten years as the global computing infrastructure of the future. Many scientists and others have considered Grid computing as one of main sources of the impact that technological changes have made on the economy and society [18]. This claim is based on the observation that the usage of large data volumes has become increasingly important to many disciplines, from natural sciences, engineering to the humanities and social sciences. However, despite significant investments in the grid concept, the number of users is not increasing. Instead, new concepts (or at least new terms) like Clouds seem to be replacing the grid computing approach (or name). Whatever is said about grid computing, it is still a key element of many global e-Infrastructures. Today, the largest scientific computational collaborations have deployed and depend on grid computing infrastructures as their

production computing platforms. For instance, the high energy physics detector ATLAS at the Large Hadron Collider (LHC) is recording data which is distributed within the Worldwide LHC Computing Grid (WLCG) consisting of more than 140 computing center in 35 countries and the 4 LHC experiments [21]. Fusion research is concentrated on studying the properties of plasmas, which require a large computing effort that is provided by different grid environments [6]. Moreover, the grid concept has affected design and development processes of many innovative IT systems over the last decade, and two of them are briefly introduced in this paper together with various challenges for future applications and services delivered on future e-Infrastructures.

The rest of the paper is organized as follows. In Section 2, the IntelliGrid project is presented as a good example how grid technologies can be used to address the lack of integrated computing environment for engineering industries. The IntelliGrid case study shows challenging integration and interoperability needs with a flexible, secure, robust, ambient accessible, interoperable, pay-per-demand access to information, communication and processing provided by the e-Infrastructure. Then, Section 3 shows interesting case study aimed at the development of open-source, semantic and grid-based technologies in support of post genomic clinical trials in cancer research. In Section 4, some key computational challenges in the use of emerging hardware architectures for future applications are presented. Section 5 introduces another challenging problems related to an increasing number of users, digital devices and services delivered via the high-speed Internet and Next Generation Networks. Finally, Section 6 concludes the paper by defining some challenges for future e-Science e-Infrastructures.

## 2 The IntelliGrid Project: e-Infrastructure for the Architecture, Engineering and Construction Industry

In addition to large data volumes distributed processing, one of the key ideas introduced by Grid computing in the late nineties was the concept of virtual organization (VO). VO is perhaps best understood through the following: "... the sharing that we are concerned with is not primarily file exchange but rather direct access to computers, software, data, and other resources, as is required by a range of collaborative problem solving and resource brokering strategies emerging in industry, science, and engineering. This sharing is, necessarily, highly controlled, with resource providers and consumers defining clearly and carefully just what is shared, who is allowed to share, and the conditions under which sharing occurs. A set of individuals and/or institutions defined by such sharing rules form what we call a virtual organization" [8]. Over the last decade, project-oriented modes of operation were also becoming ubiquitous in many industrial sectors as an attempt to provide both increased flexibility and agility to operations, and one-of-a-kind products and services were becoming a norm rather than an exception. For instance, building construction or facilities management increasingly required the one-time collaboration of different organizations to consolidate



**Fig. 1.** The concept of Virtual Organization with example enterprise and business applications integrated through the distributed IntelliGrid platform

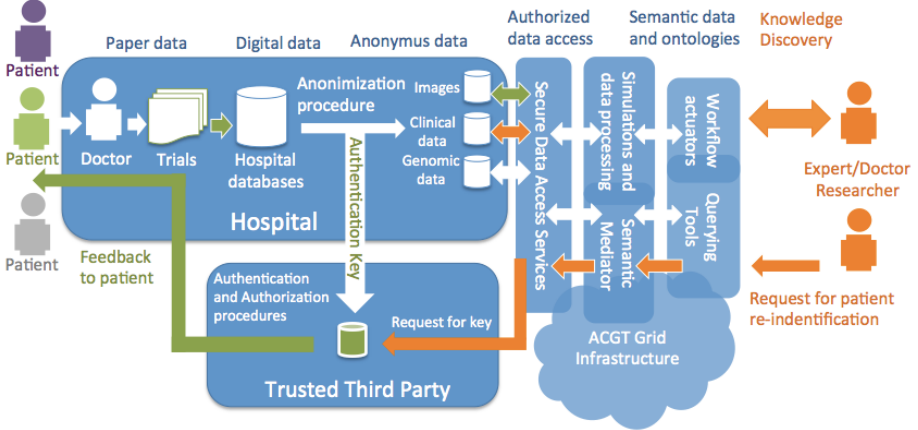
and synergize their dispersed competencies in order to deliver a desired product or service. This naturally had an implication not only on the way information (related to the to-be-delivered product or service) was exchanged and shared, but also on the way in which secure, quick to set-up, transparent (to the end-user) and nonintrusive for day to day work of an individual/organization. Thus, ICT was naturally used for this purpose. Consequently, the VO concept defined as geographically distributed, functionally and culturally diverse, dynamic and agile organizational entities linked through ICT has been a central research theme in various research initiatives [12]. The EU-funded project, IntelliGrid (2004-2007) was meant to be a bridge between differing research communities and made available Grid solutions that are usable by VOs as seen by the Architecture, Engineering and Construction (AEC) industry [10]. In a nutshell, IntelliGrid combined and extended the state-of-the-art research and technologies in the areas of semantic interoperability, virtual organizations and Grid computing in order to provide diverse engineering industries with a platform prototype ensuring flexible, secure, robust, interoperable, pay-per-demand access to information, communication and processing infrastructure, see Fig. 1. This introduction will present some of the key findings and developments related to the semantic grid architecture for virtual organizations and primarily distributed engineering for the AEC industry. Additionally, the introduction part will present key user roles that have been identified in the platform as well as crucial end user requirements. This will be followed by a description of the system architecture including conceptual, service frameworks and key developed components that the platform offers. In this case, the following main '5S' challenges have been identified in IntelliGrid:

- Security - industry eager to move only to a ground-up secure environment,
- Simplicity - must work seamlessly with current client applications and operating systems together with personalization and natural user interfaces,
- Standards - a need for stable long-term specifications,
- Service oriente architecture - scalable, adoptable, reuse, agile,
- Semantics - must support rich, domain specific semantics.

Recently, Cloud computing has been promoted as a solution to many challenging problems listed above. In fact, at both conceptual and technological levels, Cloud computing has many roots in Grid computing. Nevertheless, with much better support from various IT and independent software vendors, Cloud computing could today offer various capabilities needed by enterprises at the production rather than the prototype level as it was experienced back in InteliGrid. For instance, Cloud-based solutions provide dynamic or even event-driven configuration capabilities that can be easily reused and adopted to various changing conditions of business processes. Additionally, Cloud computing has introduced new system management technologies based on its core technology - virtualization by isolating different applications and operating systems from the underlying hardware. Automatically, virtualization technologies also solved many previously encountered security problems in Grids related to user-level access control or firewalls as the whole hosted operating system together with all user's applications and data are separated from the underlying core operating system mechanisms. Cloud computing could also help users scale their software in aggregate conditions. To conclude, it may be stated that Cloud computing wins by leveraging automation, virtualization, dynamic provision, massive scaling and multi-tenancy. Even though some new solutions have been proposed, the above-identified challenging problems are still present. Nevertheless, Cloud computing has introduced new challenges as advanced technologies development has led to power (mainly for cooling) and IT administration becoming dominant costs in data centers. Thus today, many researchers are working on resource management techniques to support energy efficiency in distributed virtualized IT systems [16] [14] [2].

### 3 The ACGT Project: e-Infrastructure for the Medical Research and e-Health Platforms

Providing people with personalized treatment, post genomic clinical trials in particular, was one of the key challenges addressed by another EU-funded project, Advancing Clinico-Genomic Trials on Cancer (ACGT) (2006-2010) [20]. In this case, based on an advanced state-of-the-art analysis and obtained results, another two key challenges have been identified. The first important challenge in carrying out medical research as well as providing better ICT-based services for an active health monitoring and early diagnosis was an efficient access to and management of sensitive patient data located in many heterogeneous sources. In most cases, patient health records have already been collected in various hospitals, clinics or laboratories without the need for further data processing, semantic



**Fig. 2.** An overall distributed IT architecture of the ACGT platform with main actors and processes involved

annotation or interoperability. Moreover, patient records were protected behind many healthcare organizational barriers and shared under legal constraints that may vary significantly between countries. The second key challenge was related to an active participation of a patient in the whole process of secure data gathering, sharing and access control, which became extremely difficult with increasing cross-border patient mobility and the advent of ubiquitous Internet connections and new services.

Within the ACGT project we have successfully demonstrated an innovative approach which addresses aforementioned challenges by linking technical expertise in semantic and grid technologies together with legal, ethical and data-protection issues surrounding patient data. The solution has been defined in the form of a framework consisting of a combination of technical, organizational and legal measures. Even if the ACGT project focused mainly on clinico-genomic trials on cancer, our databases included a wide variety of anonymous data, such as: symptoms, histology, administered treatment, response to treatment, imaging data (X-ray, MR, US, CT, etc.), genomic data, or pathology data. To achieve syntactic integration, data access services first need to provide uniform and secure data access interfaces. This includes uniformity of transport protocol, message syntax, query language, and data format as well as data protection. Through the ACGT platform, in particular syntactic access services, data can be queried over the Internet using SPARQL, thus hiding the different query mechanisms provided by the underlying databases.

From the technological perspective, a set of interdependent and reusable software components, including Grid and Web services and access tools form the foundation of the entire ACGT platform, see Fig. 2. All security related components and message/transfer-level protection mechanisms are largely based on

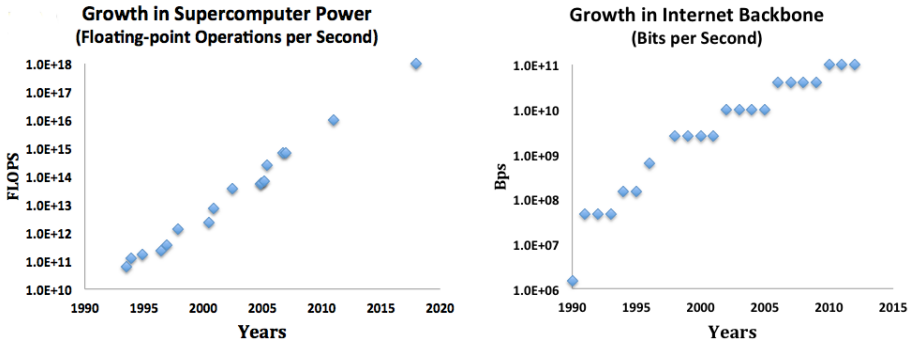
a set of proven concepts, technologies and implementations [1]. Thus, risks associated with purely technological aspects have been limited. However, due to architecture and design limitations or simply because it is technically impossible to prevent all cyber attacks and actions, technical security measures leave a number of open gaps. Therefore, we identified the need for a separate governing body such as Centers for Data Protection in international collaborative environments for coordinated technical and procedural auditing, especially in the case of sustained operational and transnational eHealth services. Owing to innovative and powerful data integration and exploitation tools as well as multi-scale modeling and large-scale simulations that have been demonstrated in the ACGT project, we may also envision innovative healthcare services concerning and integrating all levels from the molecular and basic organs to the living organism that will also be a part of the virtual health records. Thus, not only basic historical data will be collected in the trusted networks, but also more sophisticated data structures integrated with personalized computing models for a better prediction and treatment of diseases will be available. Many best practices and lessons learned from this project could pave the way for new ICT approaches for future platforms, especially in the e-Health domain, fostering bio-medical knowledge acquisition and offer better health care services at the regional, national and international scale [3].

## 4 The CaKernel Project: A Tool for Emerging Complex Computing Architectures

According to the famous Moore's law, the number of transistors on a chip will double approximately every two years. In fact, capabilities of most digital electronic devices are strongly linked to Moore's law: processing speed, networking bandwidth, memory capacity, sensors or even a resolution of multimedia displays. All of these have been improving at roughly exponential rates and dramatically enhanced the impact of digital electronics in nearly every segment of the world economy. Processors vendors always tended to implement their own benchmarks by optimizing applications and programming models for their new IT systems delivered to the market. However, there are some well-known application benchmarking suits that were commonly accepted to evaluate the overall computing power over the last twenty years, in particular LINPACK [11]. Fig. 3 shows the exponential growth in Top500 supercomputing power measured by the LINPACK application benchmark and the networking bandwidth in Internet backbone.

One should understand that the computing power has increased by a factor of one million over the last forty years reaching the level of 10 000 000 000 000 FLOPS (10 Peta FLOPS) in 2011. For comparison, a hand-held calculator performs relatively few FLOPS, whereas majority of personal computers are capable of performing at the level of hundreds of Giga FLOPS. With the available 100GB/s optical networking connections, in fact using simultaneously multiple connections, users are able to move 1 petabyte from one place to another in





**Fig. 3.** The improvement at roughly exponential rates of capabilities offered by digital devices, for instance a) growth in supercomputing power, b) growth in Internet backbone

about few hours. One petabyte of data is roughly equivalent to few billion good quality digital photos and Google processes more than twenty petabytes of data per day [9].

The Moore's law can also be of relevance in the case of yet another computing paradigm shift. Many users have not noticed a huge shift behind the transition from single-core to multi-core CPUs in 2005. From smartphones, through desktops, up to huge supercomputers, processing units have been permanently transformed into heterogeneous computing clusters. All the changes in hardware design have also an enormous impact on the software stack, from operating systems to software languages, programming tools and finally applications. A single compute intensive application will need to harness the power of heterogeneous cores, and in practice it means for software developers that majority of applications have to be reimplemented from scratch based on new parallel data structures, algorithms and synchronization routines. Metaphorically speaking, the Moore's law hit the wall in 2005 and since then has kept many software developers busy with application redesigning, reimplementation, and performance testing. Moreover, to effectively exploit new capabilities provided by multi-core processors, their modern architectures and low level APIs, software developers must be aware of many factors that will impact the overall performance of their applications. From the hardware perspective, one should take into account not only the clock rate and a number of cores, but also check the memory bandwidth, efficiency of communication channels, cache topologies, etc. On the other hand, from the software perspective, it has become extremely important to understand data dependencies, data structures and synchronization among multiple parallel tasks for a given problem as all these factors will play a critical role during the execution [13].

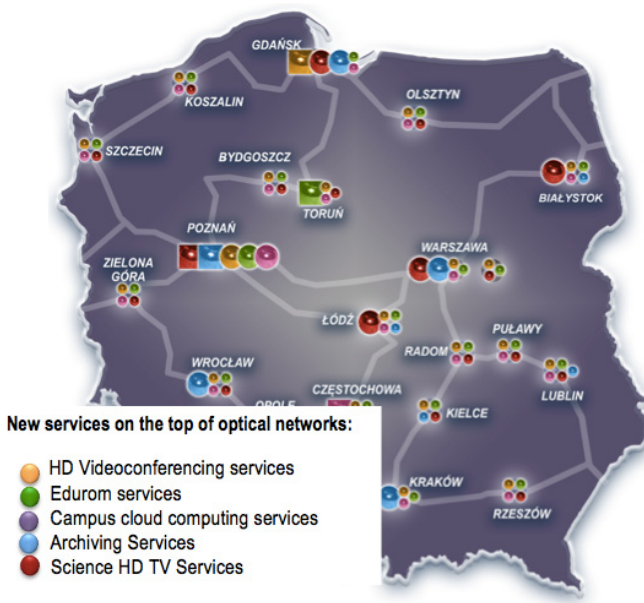
With the recent advent of new heterogeneous computing architectures, there is still a lack of parallel problem solving environments that can help scientists to use existing computing facilities easily and efficiently, in particular new hybrid

supercomputers. For instance, many scientific simulations which use structured grids to solve partial differential equations in fact rely on stencil computations. Stencil computations have become crucial in solving many challenging problems in various domains, e.g. engineering or physics. Although many parallel stencil computing approaches have been proposed, in most cases they solve only particular problems. As a result, scientists are struggling when it comes to the subject of implementing a new stencil-based simulation, especially on high performance hybrid supercomputers. Therefore, in response to the presented need, an example of parallel programming framework for different parallel programming environments will be presented, in particular recently created the CaKernel framework [5]. In a nutshell, the framework is a tool that simplifies the development of parallel applications on emerging hybrid computing architectures. CaKernel has been built on the highly scalable and portable Cactus framework. In the CaKernel framework, Cactus manages the inter-process communication via MPI while CaKernel manages the code running on GPUs and interactions between them. As a non-trivial test case, the performance and scalability of the automatically generated code from CaKernel will be presented.

## 5 The Platon Project: e-Infrastructure for the e-Science

Nowadays, an advanced analysis of complex, large-scale, multidimensional data sets is recognized as a key component in many e-Science areas, including computational fluid dynamics, medical imaging, life science, and computational engineering. Modern supercomputers, as it was briefly discussed in the previous section, especially hybrid clusters based on accelerated graphics, have recently reached the level of petaflops and are able to easily produce petabytes of data. In the near future so called Big Data will be obtained from huge scientific instruments, such as satellites, telescopes, or large-scale sensor networks, accelerators, but also from a large number of commonly used instruments as their prices have dropped significantly, e.g. high-resolution CT medical scanners or gene sequencers. Additionally, recent developments in the fields of High Definition (HD), 4K, 8K or even 16K video capture and projection, rich multimedia devices, high-speed data networks and storage hardware, and advanced digital image compression algorithms, are making ultra high digital broadcasting, on demand visualization, video streaming and digital cinema feasible and in the future easily available for everyone [7]. Moreover, there are expected to be 1 trillion new mobile devices, with high-speed networking interfaces and high resolution panels, connected to the Internet in the near future, which will help drive even more exponential growth by the year 2020. One should note, that the exponential growth in Big Data poses significant scalability and performance challenges for traditional data analytics, social media monitoring and business intelligence solutions. Deriving knowledge from large data sets presents specific scaling problems due to a large number of items, dimensions, sources, users, and disparate user communities. A human ability to process visual information can augment the analysis, especially when analytic results are presented in real-time iterative

and interactive ways. What we are witnessing is another shift in computing generating information out of distributed data sets in real-time. There will be many e-Science communities that will benefit from this advancement.



**Fig. 4.** The successful deployment of five key e-Science PLATON services on the top of PIONIER high-speed optical network with the 100Gb/s backbone inside Poland and up to 40Gb/s cross-border connections to different countries

Over the last decade, the successful development of the national high-speed optical PIONIER network allowed researchers to design and deliver innovative e-Science services to support researchers in Poland [17]. Thus, the main goal of the follow up PLATON project (2009-2012) was to deliver a set of key ICT services [4] [19] [15]: videoconference, eduroam, campus clouds, archiving and science HD TV services to foster research and scientific collaboration within and outside Poland, see Fig. 4.

- Videoconference Services: realized by building a high-quality, secure video-conference system in the PIONIER network enabling point-to-point connections, as well as simultaneous connections among multiple locations in Poland, giving the possibility of recording and replaying particular videoconferences from remote data archiving services.
- Eduroam Services: easy to manage, secure roaming and single-sign-on solution for people from the scientific and academic communities in Poland by launching model secure systems of access to the wireless network in every MAN network and HPC centre.

- Campus Services: Cloud services offered on the top of an innovative computing infrastructure consisting of a large number of small clusters deployed on campuses which deliver applications on demand thanks to virtualization technologies, capable of providing a wide range of campus users, mostly students and researchers, an elastic and scalable access to specific applications, both in MS Windows and Linux systems.
- Archiving Services: available at a national level, offer remote archiving and backup capabilities as a value added to the national, academic and research PIONIER network. Archiving services, which increase the real-time data protection, are one of the elements necessary to increase reliability of functioning of each unit, and are targeted at academic environments, including the higher education system, research and development units and clinics dependent on universities and medical universities.
- Science HD TV Services: a set of a national level services offering interactive science HD television based on high definition digital content for both education and popularization of science and telemedicine.

## 6 Conclusions

Recently published the Cyberinfrastructure Framework for 21st Century Science and Engineering (CIF21) presents a set of key complementary and overlapping components that are critical to effectively address and solve the many complex problems facing science and society [22], [23]. Core components include data, software, campus bridging and cybersecurity, learning and workforce development, grand challenge communities, computational and data-enabled science and engineering, and scientific instruments. It is clear that e-Science is increasingly based on global collaborations that need access to different resources (e.g., people, instruments, libraries, data, computations, software, algorithms, etc.) across different science communities (e.g., Research Infrastructures, Grand Science Challenges, Societal Challenges) which are at different levels of maturity and of different sizes and impact (e.g., niche high-end science, campus science, citizen science, etc.). To deliver sustainable e-Infrastructure across all of these areas, it is necessary to engage with representative stakeholders from across all of these areas. It is vital to understand their usage scenarios and resulting requirements from which a multi-year plan that identifies and delivers the operation, maintenance and development of the key common components that can be agreed across the communities. A sustainable future for the digital science community that allows for maximal science return through easy controlled sharing between domains requires at the global scale solutions to the following challenging problems in the near future:

- easy access to e-Infrastructure providers and their core services at the national and international levels,
- the dynamic provisioning of high-performance networking,
- the services and tools needed to provision and use e-Infrastructure,

- people should not use directly Big Data, but information created on-demand via appropriate tools and services,
- the data services of curation, discovery and movement are needed.

Each of these areas will continue to evolve along different timelines and costs associated with them. To ensure long-term sustainability for what will be a multi-year plan where continuity and vision is essential, engagement with policy makers and funders.

The service-driven Information System domain becomes a new complex domain, which requires new interdisciplinary approaches. If and how all the identified and briefly discussed challenges for e-Science as well as lessons learned from the past research will influence specific business and enterprise processes are still open questions. Nevertheless, to deal with the increased complexity of inter-organizational and intra-organizational processes it is worth to be aware of innovative ICT solutions successfully implemented and deployed on e-Infrastructures to be able better design, implement, deploy and integrate your own IT solution.

## References

1. Adamski, M., Kulczewski, M., Kurowski, K., Nabrzyski, J., Hume, A.: Security and performance enhancements to OGSA-DAI for Grid data virtualization. *Concurrency and Computation: Practice and Experience* 19(16), 2171–2182 (2007)
2. Bak, S., Krystek, M., Kurowski, K., Oleksiak, A., Piatek, W., Weglarz, J.: GSSIM - a Tool for Distributed Computing Experiments. *Scientific Programming* 19(4), 231–251 (2011)
3. Binczewski, A., Kurowski, K., Mazurek, C., Stroinski, M.: A Concept of a patient-centered Healthcare system based on the virtualized networking and information infrastructure. In: *Proceedings of The Third International Conference on eHealth, Telemedicine, and Social Medicine*, pp. 51–58. IARIA (2011)
4. Binczewski, A., Starzak, S., Stroinski, M.: Academic MANs and PIONIER - Polish Road e-Infrastructure for e-Science. In: Davoli, F., Meyer, N., Pugliese, R., Zappatore, S. (eds.) *Remote Instrumentation Services on the e-Infrastructure*, pp. 193–208. Springer (2011)
5. Blazewicz, M., Brandt, S.R., Kierzynka, M., Kurowski, K., Ludwiczak, B., Tao, J., Weglarz, J.: CaKernelA parallel application programming framework for heterogeneous computing architectures. *Scientific Programming* 19(4), 185–197 (2011)
6. Castejon, F., Reynolds, J., Serrano, F., Valles, R., Tarancn, A., Velasco, J.: Fusion Plasma Simulation in the Interactive Grid. *Computing and Informatics* 27, 261–270 (2008)
7. Ciznicki, M., Kurowski, K., Plaza, A.: GPU Implementation of JPEG 2000 for Hyperspectral Image Compression. In: *SPIE Remote Sensing Europe, High Performance Computing in Remote Sensing Conference, Prague, Czech Republic* (2011)
8. Czajkowski, K., Fitzgerald, S., Foster, I., Kesselman, C.: Grid Information Services for Distributed Resource Sharing. In: *10th IEEE International Symposium on High Performance Distributed Computing*, pp. 181–184. IEEE Press, New York (2001)
9. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. *Communications of the ACM - 50th anniversary issue: 1958 - 2008* 51(1) (January 2008)

10. Dolenc, M., Klinc, R., Turk, Z., Katranuschkov, P., Kurowski, K.: Semantic Grid Platform in Support of Engineering Virtual Organisations. *Informatica* 32, 39–49 (2008)
11. Dongarra, J., Luszczek, P., Petit, A.: The linpack benchmark: past, present and future. *Concurrency and Computation: Practice and Experience* 15(9), 803–820 (2003)
12. Kazi, A.S., Hannu, M., Laitnen, J.: ICT Support Requirements for Distributed Engineering in Construction. In: Stanford-Smith, B., Chiozza, E. (eds.) *E-work and E-commerce: Novel Solutions and Practices for a Global Network Economy*, pp. 909–915. IOS Press (2001)
13. Kopta, P., Kulczewski, M., Kurowski, K., Piontek, T., Gepner, P., Puchalski, M., Komasa, J.: Parallel application benchmarks and performance evaluation of the Intel Xeon 7500 family processors. *Procedia Computer Science* 4, 372–381 (2011)
14. Kurowski, K., Oleksiak, A., Piatek, W., Weglarz, J.: Hierarchical Scheduling Strategies for Parallel Tasks and Advance Reservations in Grids. *Journal of Scheduling* 11(14), 1–20 (2011), doi:10.1007/s10951-011-0254-9
15. Pekal, R., Stroinski, M.: PLATON - Advanced Service Platform for e-Science - Capabilities and Further Development Directions, Towards and Beyond Europe 2020 the significance of Future Internet for regional development Future Internet Event Report. In: *Proceedings of The Future Internet Conference, Poznan* (2011) (to appear)
16. Rozycki, R., Weglarz, J.: Power-aware scheduling of preemptable jobs on identical parallel processors to minimize makespan. *Annals of Operations Research* (2011), doi:10.1007/s10479-011-0957-5
17. Rychlewski, J., Weglarz, J., Starzak, S., Stroinski, M., Nakonieczny, M.: PIONIER: Polish Optical Internet. In: *The Proceedings of ISThmus 2000 Research and Development for the Information Society Conference, Poznan Poland*, pp. 19–28 (2000)
18. Seidel, E., Allen, G., Merzky, A., Nabrzyski, J.: GridLab - A Grid Application Toolkit and Testbed. *Future Generation Computer Systems* 18(8), 1143–1153 (2002)
19. Stroinski, M.: Innowacyjność PIONIERa szansa dla przyspieszenia realizacji celów strategii lizbońskiej w Polsce. In: *Proceedings of the 1st Conference i3: Internet - Infrastruktury - Innowacje*, pp. 7–21 (2009)
20. Tsiknakis, M., Brochhausen, M., Nabrzyski, J., Pucacki, J., Sfakianakis, S., Potamias, G., Desmedt, C., Kafetzopoulos, D.: A Semantic Grid Infrastructure Enabling Integrated Access and Analysis of Multilevel Biomedical Data in Support of Postgenomic Clinical Trials on Cancer. *IEEE Transactions on Information Technology in Biomedicine*, 205–221 (2008)
21. Worldwide LHC Computing Grid Technical Site, <http://lcg.web.cern.ch/LCG/Default.htm>
22. Cyberinfrastructure Framework for 21st Century Science and Engineering, [http://www.nsf.gov/about/budget/fy2012/pdf/40\\_fy2012.pdf](http://www.nsf.gov/about/budget/fy2012/pdf/40_fy2012.pdf)
23. Advanced Computing Infrastructure: Vision and Strategic Plan, <http://www.nsf.gov/pubs/2012/nsf12051/nsf12051.pdf>

# Understanding Business Process Models: The Costs and Benefits of Structuredness

Marlon Dumas<sup>1</sup>, Marcello La Rosa<sup>2,3</sup>, Jan Mendling<sup>4</sup>, Raul Mäesalu<sup>1</sup>,  
Hajo A. Reijers<sup>5</sup>, and Nataliia Semenenko<sup>1</sup>

<sup>1</sup> University of Tartu, Estonia

{marlon.dumas,maekas,nataliia}@ut.ee

<sup>2</sup> Queensland University of Technology, Australia  
m.larosa@qut.edu.au

<sup>3</sup> NICTA Queensland Research Lab., Australia

<sup>4</sup> Vienna University of Business and Economics, Austria  
contact@mendling.com

<sup>5</sup> Eindhoven University of Technology, The Netherlands  
h.a.reijers@tue.nl

**Abstract.** Previous research has put forward various metrics of business process models that are correlated with understandability. Two such metrics are size and degree of (block-)structuredness. What has not been sufficiently appreciated at this point is that these desirable properties may be at odds with one another. This paper presents the results of a two-pronged study aimed at exploring the trade-off between size and structuredness of process models. The first prong of the study is a comparative analysis of the complexity of a set of unstructured process models from industrial practice and of their corresponding structured versions. The second prong is an experiment wherein a cohort of students was exposed to semantically equivalent unstructured and structured process models. The key finding is that structuredness is not an absolute desideratum vis-a-vis for process model understandability. Instead, subtle trade-offs between structuredness and other model properties are at play.

**Keywords:** structured process model, process model complexity, process model understandability.

## 1 Introduction

In many contexts where information systems are developed and used, conceptual models are employed to inform stakeholders on the business processes supported by such systems. Sometimes, hundreds or thousands of process models are created and maintained in order to document large information systems. Given that such model collections are consulted, validated and updated by a wide range of stakeholders with various levels of expertise, ensuring the understandability of process models is a key concern in such settings.

In this respect, a central guideline for business process modeling is to use structured building blocks as much as possible [19]. This insight has triggered

a stream of research on transforming unstructured process models into structured ones. The approach in [25] provides a formal foundation for determining when and how such a transformation is possible. However, it turns out that in some cases structuredness can only be achieved at the expense of increased size. Specifically, there are situations where nodes must be duplicated in order to transform an unstructured model into a structured one. Importantly, this node duplication is not a limitation of the technique proposed in [25], but an unavoidable constraint that has been studied in the field of compiler theory [23].

While both structuredness and conciseness are recognized as basic design principles for process models [19], to the best of our knowledge the tradeoff between these two principles has not been studied so far. In this context, this paper examines whether or not the benefits of structuring an unstructured process model outweigh the costs of duplication from the perspective of understandability. Specifically, the following research question is addressed:

If an unstructured process model  $U$  is transformed into a structured one  $S$ , is  $S$  more understandable than  $U$ ?

To tackle this question, we adopt a two-pronged approach. First, by using a collection of industrial process models, we compare the relative complexity of unstructured process models and of their corresponding structured versions. This study is based on a collection of complexity metrics that have been shown to be negatively correlated with understandability [19], that is, higher complexity is associated with lower understandability. The results of this comparative complexity study are inconclusive with respect to the hypothesis. In this sense, they confirm the existence of a tradeoff between structured and unstructured process models. Second, we conducted an experiment with structured and unstructured models as a treatment, and measured the understanding performance of a cohort of process modeling students. The results show that in some cases it is preferable to leave a process model unstructured from the perspective of understandability.

The paper is structured as follows. Section 2 introduces background notions, including that of structuredness and some complexity metrics, and motivates our work against the state of the art. Section 3 presents the comparative complexity study, which is followed by the design of the conducted experiment (Section 4) and its results (Section 5). Section 6 concludes the paper.

## 2 Background

This section provides the background knowledge for understanding the rest of this paper. It introduces the notions of structuredness, complexity metrics and understandability of process models. It also summarizes previous research on structuring process models and empirical studies on the importance of structuredness for model understanding and correctness.



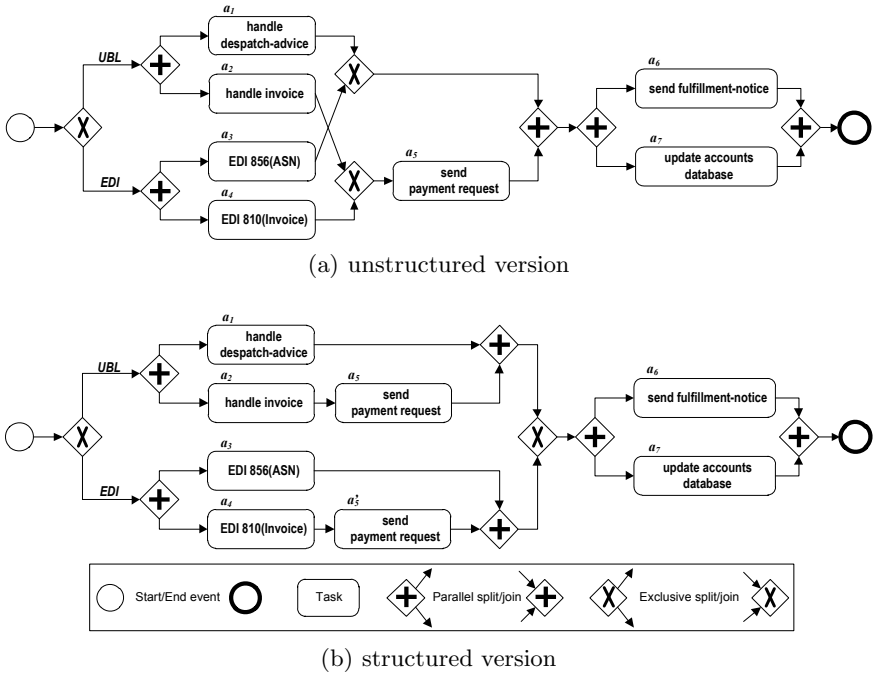


Fig. 1. Unstructured process model and its equivalent structured version

## 2.1 The Notion of Structuredness

Although process models captured in graph-oriented languages such as BPMN or EPCs may have almost any topology, it is often preferable that they adhere to some structural rules. In this respect, a well-known property of process models is that of *block-structuredness*, meaning that for every node with multiple outgoing arcs (a *split*) there is a corresponding node with multiple incoming arcs (a *join*) such that the subgraph between the split and the join forms a single-entry-single-exit (SESE) region. For example, the BPMN process model shown in Fig. 1(a) is unstructured because the parallel split gateways do not satisfy the above condition. Fig. 1(b) shows a semantically equivalent yet structured model.

Previous research on structured process modeling has demonstrated that not all unstructured process models can be transformed into equivalent structured models [11]. Such models are hereby called *inherently unstructured*. There are two root causes for process models being inherently unstructured. The first one arises from the structure shown in Fig. 2; it has been shown to be inherently unstructured in [11]. The second cause is a cycle that has more than one exit point, in which case it can only be structured by introducing boolean variables to

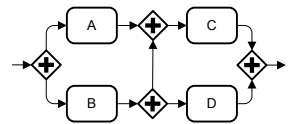
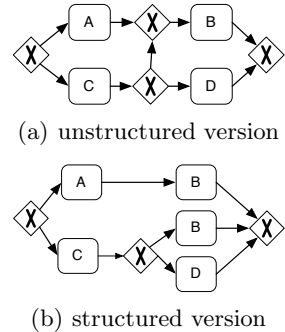


Fig. 2. Inherently unstructured BPMN model

capture part of the control-flow [23] – arguably an undesirable solution in the context of process modeling.

It is also known that transforming an unstructured process model into a structured one may lead to the introduction of duplicate nodes [23]. This duplication arises when there is an XOR-join that is not the boundary of a SESE region of the model. In some cases, to structure these models, the paths that follow the XOR-join need to be duplicated until the exit point of the smallest enclosing SESE region is reached. For example, in the structured model in Fig. 1(b), the “send payment-request” task is duplicated. In cases where loops have multiple entries, the paths between the entry of the loop and the XOR-join need to be duplicated [23]. A direct effect of this duplication is that when transforming an unstructured process model into a structured one, the size of the resulting model is typically higher. In a study based on the SAP R/3 reference model collection, converting unstructured process models into structured ones led to an average size increase of 22% and a maximum size increase of 63% [25]. This size increase is attributable to an increase in the number of task/event nodes. The number of gateways increases marginally and, in some cases, even decreases due to consolidation. This is illustrated in Figure 3.



**Fig. 3.** Unstructured model with less gateways than structured version

## 2.2 Structuring Process Models

The problem of structuring process models has been extensively studied since the early 2000s. It has its roots in compiler theory [23] and code refactoring [35]. Kiepuszewski et al. [11] made a first attempt at classifying unstructured process models that can be transformed to structured equivalents. They showed that unstructured process models cannot always be replaced with structured models that are behavior-equivalent. Later, Liu and Kumar [15] presented an alternative taxonomy of unstructured process models and sketched a method to transform some types of unstructured models into structured versions. Different transformation strategies were also illustrated in [18] and an alternative classification of (unstructured) process models was proposed in [9]. A method specifically tailored to untangling unstructured cyclic models and transforming them into structured BPEL models is presented in [10,12]. Finally, Polyvyanyy et al. [25] provide a complete characterization of unstructured acyclic process models that are inherently unstructured, based on the RPST decomposition developed by Vanhatalo et al. [33].

Previous research on structuring process models has been consolidated in a tool called BPStruct<sup>1</sup>. BPStruct tries to structure a model to the maximum possible extent. The resulting model is then said to be *maximally-structured*.

<sup>1</sup> <http://code.google.com/p/bpstruct/>

### 2.3 Complexity Metrics and Understandability

In addition to formal work on structuring process models, there is a growing body of research on how the understandability of a process model can be evaluated from an empirical perspective. Different factors of model understanding have been studied based on experiments and survey research, including modeling notation [1], model complexity [28], modeling expertise [24,28], secondary notation [7,27], and labeling conventions [21]. Throughout this paper, we will focus on model complexity and its relationship to understandability.

The importance of complexity for model understanding has been emphasized by various authors. Inspired by works in software measurement [5,17], early work focused on the definition of metrics for process models [14,22]. Further metrics were proposed since then in [2,3,4,31,32]. These measures can be roughly categorized into six groups [20]: measures of size, density, modularity, connector interplay, cyclicity, and concurrency. *Size measures* count different types of elements that appear in a process model, such as the number of nodes or number of arcs. *Density measures* capture the relationship between nodes and arcs. For instance, density is the ratio of arcs divided by the maximum possible number of arcs for a given set of nodes. Another option is to use the average connector degree (ACD), which yields the average number of arcs a node is associated with. *Modularity* covers aspects such as structuredness. The degree of structuredness defines the ratio of nodes in structured components to all components. *Connector interplay* quantifies a potential mismatch of split and join nodes. For instance, the cross connectivity (CC) metric defines how many routing elements can be expected on a path between two nodes [31]. *Cyclicity* refers to the extent to which models include cycles. Finally, *concurrency measures* describe how much concurrency can be encountered in a process model. However, while in software engineering duplication is generally considered harmful for maintenance and comprehension [29], the issue of duplication is completely missing from the complexity metrics of process models up until now.

Empirical research in the area of process model understanding has focused on controlled experiments in order to find out how strong understanding is affected by a varying complexity. Several studies demonstrate the effect of size (see [28]). As a result of a study on predicting error probability for a set of 2,000 process models from practice [20], structuredness appears to be the best determinant to distinguish low-error-probability models from ones with high error probability. Another study confirms the significance of structuredness, albeit that different definitions are used [13]. These and other experiments are summarized in the seven process modeling guidelines [19]. Specifically, one of these guidelines is to model processes as structured as possible, which ranked as the guideline with the highest relative potential for improving process model understandability. Recently, this work has been complemented with methodological guidance for determining statistically significant threshold values for the complexity metrics [30].

Importantly, all of the above studies on the link between complexity metrics and understandability take as input samples consisting of distinct sets of

structured and unstructured process models, by which we mean that when the sample includes an unstructured process model, it does not include an equivalent structured version of this model. Thus, the above studies do not directly address the question formulated in Section 1. Also, these studies altogether do not permit one to draw any conclusions on the trade-off between structuredness and duplication. A process model with small size and high structuredness appears to be preferable from a perspective of model comprehension. Yet, it is not clear whether a structured model should be preferred in case it implies duplicating a specific amount of nodes and gateways. While an increase in size is considered as harmful, duplication might even be worse. The negative effect of duplication when structuring a process model has been briefly considered in [8]. Here the authors point out that when structuring leads to unwanted duplications, it might be preferable to leave a process model unstructured. However, they do not validate these intuitions empirically.

### 3 Complexity Comparison

In this section, we present a comparative complexity study aimed at testing the hypothesis formulated in Section 1. For this study, we relied on the IBM Business Integration Technology (BIT) library: a publicly-available collection of process models gathered from IBM’s consultancy practice [6]. The BIT library is divided into three collections, namely A, B3 and C. [2] After removing semantically incorrect models, these collections contain 269, 247 and 17 models respectively.

In order to test the hypothesis, we are interested in comparing the complexity of unstructured models against the complexity of the corresponding structured models (or the maximally-structured models in case of inherently-unstructured models). Accordingly, we extracted all 59 correct and unstructured models from the BIT library (41 in collection A, 15 in B3 and 3 in C). Each of these models was structured using BPStruct. The output was manually checked to ensure that no errors had been introduced by the tool. 11 models with cycles were found to be inherently unstructured (cycles with multiple exit points), in which case we took the maximally-structured model produced by BPStruct as being the “structured version”. None of the correct, acyclic models in the library was found to be inherently unstructured.

For each of the 59 resulting pairs of models (one structured, one unstructured), we computed seven metrics: number of arcs, number of gateways, number of tasks, size (number of nodes), Average Connector Degree (ACD), Cross-Connectivity (CC), and Density as introduced in Section 2. The average values of these metrics for the unstructured models and for the corresponding structured models are shown in Table 1.

As expected, the average size of the structured models is higher. This increase in size is entirely due to node duplication introduced when structuring a model, as discussed in Section 2. On average, the increase in size is in the order of

<sup>2</sup> There are also two other collections (B1 and B2) containing earlier versions of the models in B3, but for the purposes of this study these collections are redundant.

**Table 1.** Complexity of unstructured models vs. corresponding structured models

Metric	Avg. unstructured	Avg. structured	Increase/decrease (%)
# arcs	30.02	44.76	49.12%
# gateways	8.41	12.31	46.37%
# tasks	16.81	25.76	53.23%
Size	25.22	38.07	46.53%
ACD	3.41590	3.29957	-3.04%
CC	0.05118	0.04455	-11.44%
Density	0.16096	0.12376	-22.5%

50%, which is consistent with previous findings reported in [25]. On the other hand, we observe a notable decrease in CC and density when the models become structured. The decrease in density reflects the intuition that structured process models are less “cluttered” while the decrease in CC supports the intuition that the control-flow relations between pairs of tasks in structured models are cognitively simpler. We also note a marginal decrease in ACD because of structuring, attributable to gateway reshuffling.

The differential in size, which is entirely attributable to duplication, entails that the structured process models are likely to be less understandable than the unstructured ones [19]. On the other hand, lower ACD, CC and density suggest that structured models are less complex and, based on the same empirical study, we expect that the structured process models are more understandable than the unstructured ones. Therefore, the comparative complexity study shows that there is a tradeoff between different complexity indices when the unstructured models in the dataset are transformed into structured ones. In other words, the comparative complexity analysis neither supports nor refutes the hypothesis that structuring an unstructured process model leads to a higher understandability.

## 4 Controlled Experiment

Considering the ambivalent outcomes produced by the comparative complexity study, we decided to approach the research question with a questionnaire-based, controlled experiment. In this section, we describe its design.

**Subjects:** The experiment was run at Queensland University of Technology in early October 2011. The population consisted of 110 students from two business process modeling courses. The students were in their tenth week of training, such that they had acquired the requisite background to read BPMN process models. Participation in the study was voluntary.

**Objects:** We took 8 models from the IBM dataset (cf. Section 3) as the objects of our experimental investigation. The models were selected on the basis of their size and whether they had cycles or not. Based on these two factors, four categories of models were created:

1. Acyclic unstructured models whose equivalent structured models are at least 25% larger than the unstructured versions.
2. Cyclic unstructured models whose equivalent structured models are at least 25% larger than the unstructured versions.
3. Acyclic unstructured models whose equivalent structured models are less than 10% larger than the unstructured versions.
4. Cyclic unstructured models whose equivalent structured models are less than 10% larger than the unstructured versions.

For each of these four categories, two models were selected. When selecting models in a given category, we sought models with heterogeneous values for size, CC, ACD and density. The sizes of each original unstructured model and the differential in size and other complexity metrics between the unstructured and structured versions are given in Section 5 (Table 3).

**Factor and Factor Levels:** The main factor manipulated in our experiment is the *structuredness* of each process model, as presented to the respondents. Given the binomial character of this factor, it has *structured* and *unstructured* as its levels for each of the used models.

**Response Variables:** As response variable we used a composite measure to reflect a respondent’s understanding of a process model – in a structured or unstructured form – as determined by the sum of correct answers to a set of six questions asked for each model.

**Hypothesis Formulation:** The goal of the experiment was to investigate whether the level of structuredness of process model influences a respondent’s understanding of a process model. While we would expect a beneficial effect of structuring a process model, we appreciate the potential drawback of using duplicate nodes. Accordingly, we postulate the following hypotheses:

- **Hypothesis:** Structured models yield a greater amount of correct answers than unstructured models.
- **Alternative Hypothesis:** Unstructured models yield a greater amount of correct answers than structured models.

**Instrumentation:** The participants conducted the experiment by observing the selected process models on paper, as part of a work-book. On a single page, one process model along with a set of six questions to test the respondent’s comprehension of that model were presented. All models were presented in the BPMN notation. The questions used were designed in a similar fashion to the questions of earlier experiments into process model understanding [28]. An example question from the questionnaire is: “If J is executed for a case, can F be executed for the same case?” Additionally, subjects were asked to subjectively rate the complexity of each model on a 5-point Likert scale. Besides these understanding questions, the questionnaire recorded personal characteristics of the participants, including their theoretical knowledge on process modeling as well as the amount of process modeling training they had received in the past.

**Experimental Design:** The experimental setup was based on literature providing guidelines for designing experiments [34]. Following these guidelines a *randomized balanced single factor* experiment was conducted with a *single measurement*. The experiment is called *randomized* because subjects are randomly assigned to two different groups. We denote the experiment as *balanced* as each factor level is used by each subject, i.e., each student is confronted to both structured and unstructured process models. As only a single factor is manipulated (i.e., the level of structuredness of each process model), the design is called *single factor*. Due to the balanced nature of the experiment, each subject generates data for both factor levels. To avoid learning effects, each subject sees each process model only once: either in its structured or unstructured form. This is achieved by having two versions of the questionnaire: one per group. In the version of the questionnaire for the first group, all odd-numbered models were structured while in the version for the second group all even-numbered models were structured. Thus, each subject evaluates four structured and four unstructured models and none of the models a subject evaluates is equivalent to any other model evaluated by the same subject. The questionnaires are available at <http://d1.dropbox.com/u/15565756/questionnaires.zip>.

In separate work [16], we reported on a similar experiment with two groups of students: one group only seeing structured models in their questionnaire and the other group only seeing unstructured models. The results of this experiment were similar to the ones reported below.

Another alternative experimental design would be to include unstructured and structured versions of the same model in the same questionnaire (with relabelling in order to minimize learning effects). In this case, it would be possible to ask each subject to compare the models relative to one another in terms of understandability. The design of such experiment would however be rather different from the one reported in this paper, and thus deserves a separate study.

## 5 Results

In this section, we present the results of the experiment described in the previous section. First, we explain the data cleansing steps we conducted. The resulting sample serves us for testing the research hypothesis.

### 5.1 Data Cleansing and Demographics

To avoid bias in terms of modeling expertise, we decided to filter out the responses of those respondents who were highly familiar with process modeling and who had experience with process modeling in practice. Furthermore, we filtered out some potentially meaningless answers (e.g. training of more than 20 work days on process modeling in the last year). The resulting analysis sample includes the responses from 55 students who can all be classified as educated novices in process modeling. On average, these participants got 6.07 out of twelve theory questions right (std. deviation of 2.209) and 4.35 of the six understanding questions for each of the eight models (std. deviation 1.701). By having 55

students evaluating eight models each, we received 440 model understanding observations. 27 and 28 participants account for each version of the questionnaire.

## 5.2 Hypothesis Testing

By splitting the 440 observations into structured versus unstructured models, we did not find an aggregated effect. Therefore, we run a univariate ANOVA for each of the eight models by taking theory and structuredness as independent variables into account. In this way, we can analyze the hypothesis that participants might understand structured process models better than unstructured ones.

**Table 2.** Results of ANOVA analysis for all eight models

Model		Mean (Std.D.)	Estim.	Intercept	Struct.	Struct.	Theory	Theory	$R^2$	better
		Correct	Mean	Sig.	Eff.Size	Sig.	Eff.Size	Sig.		
1	s	4.29 (1.21)	4.213							
	u	4.26 (1.63)	4.211	.001	.002	.776	.303	.115	.375	s
2	s	4.85 (1.13)	4.823							
	u	4.64 (1.31)	4.611	.001	.001	.853	.165	.628	.257	s
3	s	3.32 (1.28)	3.264							
	u	3.85 (1.46)	4.263	.001	.122*	.032	.245	.271	.342	u
4	s	5.07 (1.64)	4.955							
	u	4.46 (1.35)	4.278	.001	.176*	.009	.396*	.019	.495	s
5	s	4.43 (1.89)	3.866							
	u	4.70 (1.71)	4.600	.001	.136*	.023	.420*	.011	.494	u
6	s	4.89 (1.79)	4.786							
	u	4.32 (2.16)	3.694	.001	.168*	.011	.355*	.045	.446	s
7	s	4.18 (2.28)	3.676							
	u	4.56 (1.78)	4.140	.001	.066	.121	.454*	.005	.518	u
8	s	4.30 (1.71)	3.711							
	u	3.57 (1.89)	3.273	.001	.065	.122	.395*	.020	.485	s

The results of the ANOVA analysis are summarized in Table 2. The table lists the mean sum of correct answers along with standard deviation and estimated marginal means for both the structured and the unstructured version of each model. From these mean values, it can be seen that the effect of structuring has a different orientation for the models. Models 3, 5, and 7 yield better understanding in their unstructured version while for the other models the structured version is better. The columns following the estimated mean shows the significance of the ANOVA model coefficients. The intercept is significant in all cases. The effect of structure is only significant for models 3 to 6. Notably, the direction for these four models is leaning towards the structured version in two cases, and also two cases towards the unstructured model. The effect size is between 0.122 and 0.176. For models 4 to 8, there is a significant effect of theory ranging between 0.355 to 0.454. The  $R^2$  values range from 0.257 to 0.518. These statistical results demonstrate that a considerable share of variance can be explained



by structuredness and theoretical knowledge. They also point to the fact that structuredness and theory alone cannot explain the direction of the structuring effect on understanding. In the following, we will investigate which further characteristics might help to explain these diverging effects.

### 5.3 Qualitative Analysis

In order to analyze the factors that may determine whether structured models are preferred over unstructured ones, we summarize in Table 3 the characteristics of each model, the aggregate scores for perceived complexity, and the number of correct answers broken down by model. Columns 1-2 indicate the model number and a label indicating whether the model is the structured (s) version or the unstructured (u) one. For example, the first row corresponds to model 1, structured version. Columns 3-9 provide the following properties for each model: number of nodes, number of gateways, whether or not the model contains cycles, and the increase in size, CC, ACD and Density between the structured version of the model and the unstructured one. Column 10 provides the mean score for perceived complexity. This score was obtained by asking the following question for each model: “Please rate the difficulty of the model on a 5-point scale” with 1 mapping to “very simple” and 5 mapping to “very difficult”. Finally, column 12 provides the mean number of correct answers – copied here for the sake of convenience from Table 2.

From this table, we observe the following:

- Expectedly, an increase in perceived complexity is accompanied in all cases by a decrease in the number of correct answers and vice-versa (except for Model 2 for which the perceived complexity is identical for both versions).

**Table 3.** Model characteristics versus dependent variables

Model		# nodes	# gateways	Cycles?	Size Increase	CC Increase	ACD Increase	Density Increase	Perceived complexity	Mean correct
1	s	30	5	no	1.43	2.07	14.55	-17.44	2.22	4.29
	u	21	7						2.84	4.26
2	s	24	5	no	1.50	-7.63	8.00	-21.43	2.66	4.85
	u	16	6						2.66	4.64
3	s	21	8	yes	1.31	-21.60	0.00	-11.11	3.21	3.32
	u	16	6						3.04	3.85
4	s	33	14	yes	1.65	-41.10	-12.50	-44.75	3.52	5.07
	u	20	7						3.75	4.46
5	s	32	10	no	1.23	2.62	-9.33	-26.00	3.60	4.43
	u	26	8						3.13	4.70
6	s	27	8	no	1.13	11.37	5.00	0.00	3.04	4.89
	u	24	9						3.49	3.32
7	s	21	8	yes	1.17	-9.97	-4.55	-18.75	3.51	4.18
	u	18	7						3.18	4.56
8	s	25	7	yes	1.04	30.60	4.76	6.06	2.85	4.30
	u	24	8						3.60	3.57

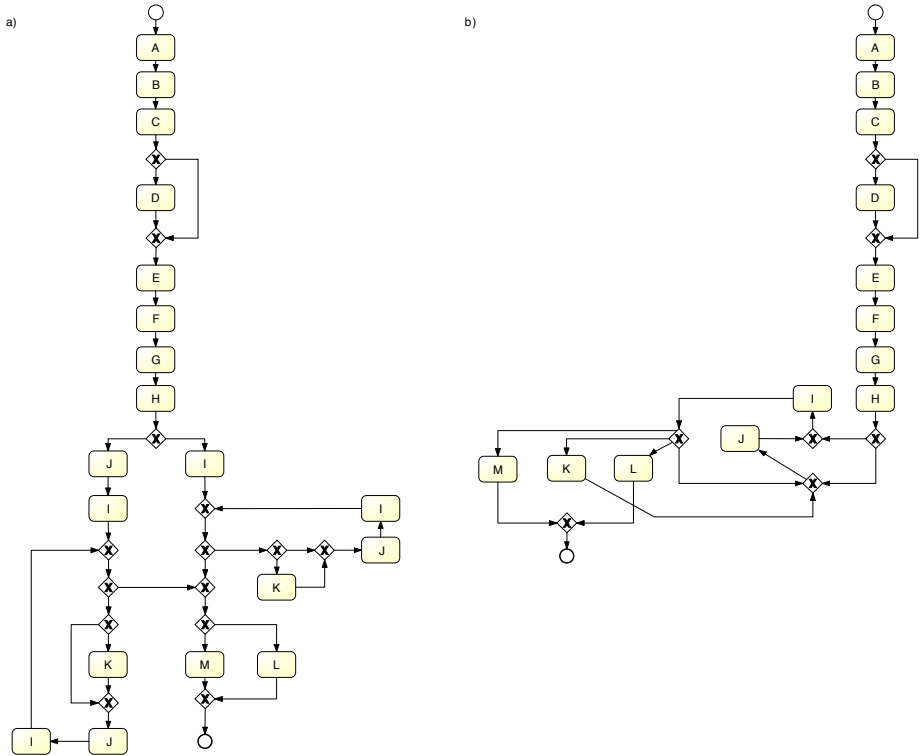
- Increases in the four complexity metrics (size, CC, ACD and Density) do not explain why some unstructured models are preferred over the structured counterparts or vice-versa. For example, for Model 3 there is a decrease in CC and yet the unstructured version is preferred. Meanwhile, for Model 1 and Model 8 there is an increase in CC and yet the structured version is preferred. The same can be said for the density metrics. More striking is the fact that for the models with the highest increase in size (Models 1, 2 and 4), the structured version was preferred. Thus, subjects sometimes prefer the structured model even when duplication is high, while other times they prefer the unstructured model despite low duplication in the structured model.
- In all three cases where the unstructured version is preferred, the number of gateways is *higher* in the structured version than in the unstructured one. Similarly, in 4 of the 5 instances where the structured model is preferred, the number of gateways is *lower* in the structured versions. The sole exception is Model 4.

With reference to the last two observations, Fig. 4 shows the unstructured and structured versions of Model 4. This model has the highest increase in size after restructuring. It also has a higher number of gateways in the structured version. Still, this version is perceived to be less complex and the number of correct answers is higher in the structured version. Interestingly, this is the model with the strongest decrease in CC and in density. Thus, whilst these two complexity metrics cannot predict in the general case whether subjects would prefer the structured or the unstructured version of a model, it appears that the predictions made by these metrics are accurate in cases where the differentials in complexity are clear-cut. This observation suggests that there might be thresholds beyond which a decrease in CC or density when structuring an unstructured model do imply an increase in understandability. Previous research has identified thresholds beyond which complexity metrics (including density) are significant [30]. Unfortunately, this previous study does not consider the case where an unstructured process model is pitched against its corresponding structured version.

Although the number of models in the experiment is too small to draw a statistically significant conclusion, it appears that structuring leads to more understandable models if it does not increase the number of gateways. This is not a general rule as we have found one instance where a clear increase in understandability is achieved despite an increase in the number of gateways.

## 5.4 Threats to Validity

The controlled experiment was conducted with a cohort of students. It might be argued that the results may be biased by the fact that students would tend to do more mistakes than experienced analysts. While this indeed is a limitation of the study, it is worth noting that previous work has shown that students can be treated as proxies for early-career analysts in similar experiments [26].



**Fig. 4.** Structured (a) and unstructured (b) versions of Model 4 used in the experiment

The students in the cohort had heterogeneous backgrounds. Answers given by subjects with higher expertise level were filtered out for analysis since dealing with multiple expertise levels would require a much larger group or, better still, multiple groups with different backgrounds. A possible extension of this work is to conduct similar experiments with cohorts of different expertise levels.

The number of models (8) is insufficient to make definite conclusions regarding the factors that affect whether or not the structured model is preferred over the unstructured one. This is why the discussion in Section 5.3 is qualitative in nature. A separate study would be needed to zoom into this question.

The fact that all models in the study came from the same source, i.e. the IBM BIT library, can also be seen as a limitation. While this library contains process models from industrial practice and encompasses three collections of models (from different origins), it is possible that taking process models from other sources might impact our insights. Also, in this library, the labels in the process model were anonymized into labels of the form “s00000982” or “s00001088” prior to the public release of the library [6]. For our controlled experiment, we replaced these anonymized labels with single symbols (“A”, “B”, “C”, ...) in order to more easily refer to them in the questionnaire. It is possible that the

lack of node labels referring to a real-world application domain might have an effect on the results of the study. The latter limitation is shared with several other studies on complexity and understandability [13,28,30].

## 6 Conclusion

This study has exposed the subtlety of tradeoffs involved when attempting to improve the understandability of process models by applying guidelines. Specifically, we have shown that while structuring an unstructured process model is in line with existing guidelines for ensuring process model understandability, the expected benefits of this guideline may be thwarted by its side-effects.

The comparative complexity study provided no conclusive insights into the relative tradeoff between structuredness and various complexity measures. The controlled experiment involving a cohort of process modeling novices confirmed the equivocal nature of the research question at hand by showing that in certain cases a unstructured version of a process model is more easily interpreted than its corresponding structured equivalent, while in other cases the opposite holds.

The results of the study suggest that if structuring a process model does not increase the number of gateways, the structured model would be preferred over presenting the unstructured version of that model. While we are the first to report on this specific relation and accompanying suggestion, it is congruent with earlier work that has pointed at specific properties of gateways in a process model as a source of mental strain [28].

We believe that this work contributes to the further development and fine-tuning of guidelines that will help to optimize the use of business process models. Considering the interest from research and praxis into such guidelines, it makes sense to push further towards a more complete understanding of the conditions under which refactoring an unstructured process model into a structure one is beneficial. In future work, we aim to analyze the observed mediating effect of gateways identified in this work. To this end, we need to conduct experiments with a larger set of process models. A specific challenge will be to find a research design that circumvents fatigue effects with participants. Showing eight models can still be considered acceptable, but this is not so for a set of 30 or 40 models.

Another avenue for future work is to investigate in more depth the tradeoffs between understandability, structuredness and other complexity metrics. The comparative complexity analysis showed that structuring an unstructured process model has different effects on different complexity metrics. No complexity metric alone seems to be able to predict whether or not the structured model will be preferred over the unstructured one. However, it is possible that if we aggregated the effects of structuring a process model on multiple complexity metrics, we could obtain a combined measure that could serve as a predictor of whether structuring an unstructured process model would improve its understandability.

**Acknowledgments.** This work is partly funded by the Estonian Science Foundation (Dumas), the ERDF via the Estonian Centre of Excellence in Computer Science (Dumas), and by the ARC Linkage project LP110100252 (La Rosa).

## References

1. Agarwal, R., De, P., Sinha, A.P.: Comprehending object and process models: An empirical study. *IEEE Transactions on Software Engineering* 25(4), 541–556 (1999)
2. Rolón Aguilar, E., García, F., Ruiz, F., Piattini, M.: An exploratory experiment to validate measures for business process models. In: *First International Conference on Research Challenges in Information Science, RCIS (2007)*
3. Canfora, G., García, F., Piattini, M., Ruiz, F., Visaggio, C.A.: A family of experiments to validate metrics for software process models. *Journal of Systems and Software* 77(2), 113–129 (2005)
4. Cardoso, J.: Evaluating workflows and web process complexity. In: Fischer, L. (ed.) *Workflow Handbook 2005*, pp. 284–290. Future Strategies, Inc., Lighthouse Point (2005)
5. Chidamber, S.R., Kemerer, C.F.: A metrics suite for object oriented design. *IEEE Transaction on Software Engineering* 20(6), 476–493 (1994)
6. Fahland, D., Favre, C., Jobstmann, B., Koehler, J., Lohmann, N., Völzer, H., Wolf, K.: Instantaneous Soundness Checking of Industrial Business Process Models. In: Dayal, U., Eder, J., Koehler, J., Reijers, H.A. (eds.) *BPM 2009. LNCS*, vol. 5701, pp. 278–293. Springer, Heidelberg (2009)
7. Green, T.R.G., Petre, M.: Usability analysis of visual programming environments: A 'cognitive dimensions' framework. *J. Vis. Lang. Comput.* 7(2), 131–174 (1996)
8. Gruhn, V., Laue, R.: Good and bad excuses for unstructured business process models. In: *Proc. of EuroPLoP*, pp. 279–292. UVK - Universitaetsverlag Konstanz (2008)
9. Hauser, R., Friess, M., Kuster, J.M., Vanhatalo, J.: An Incremental Approach to the Analysis and Transformation of Workflows Using Region Trees. *IEEE Transactions on Systems, Man, and Cybernetics, Part C* 38(3), 347–359 (2008)
10. Hauser, R., Koehler, J.: Compiling Process Graphs into Executable Code. In: Karsai, G., Visser, E. (eds.) *GPCE 2004. LNCS*, vol. 3286, pp. 317–336. Springer, Heidelberg (2004)
11. Kiepuszewski, B., ter Hofstede, A.H.M., Bussler, C.J.: On Structured Workflow Modelling. In: Wangler, B., Bergman, L.D. (eds.) *CAiSE 2000. LNCS*, vol. 1789, pp. 431–445. Springer, Heidelberg (2000)
12. Koehler, J., Hauser, R.: Untangling Unstructured Cyclic Flows – A Solution Based on Continuations. In: Meersman, R. (ed.) *OTM 2004. LNCS*, vol. 3290, pp. 121–138. Springer, Heidelberg (2004)
13. Laue, R., Mendling, J.: Structuredness and its significance for correctness of process models. *Inf. Syst. E-Business Management* 8(3), 287–307 (2010)
14. Lee, G.S., Yoon, J.-M.: An empirical study on the complexity metrics of petri nets. *Microelectronics and Reliability* 32(3), 323–329 (1992)
15. Liu, R., Kumar, A.: An Analysis and Taxonomy of Unstructured Workflows. In: van der Aalst, W.M.P., Benatallah, B., Casati, F., Curbera, F. (eds.) *BPM 2005. LNCS*, vol. 3649, pp. 268–284. Springer, Heidelberg (2005)
16. Mäesalu, R.: Complexity and Understandability Comparison between Unstructured and Structured Business Process Models. Master's thesis, University of Tartu (June 2011), <http://tinyurl.com/75gfnuz>
17. McCabe, T.J.: A complexity measure. *IEEE Transaction on Software Engineering* 2(4), 308–320 (1976)
18. Mendling, J., Lassen, K.B., Zdun, U.: On the transformation of control flow between block-oriented and graph-oriented process modelling languages. *International Journal of Business Process Integration and Management* 3(2), 96–108 (2008)

19. Mendling, J., Reijers, H.A., van der Aalst, W.M.P.: Seven Process Modeling Guidelines (7PMG). *Information and Software Technology* 52(2), 127–136 (2010)
20. Mendling, J.: Metrics for Process Models: Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness. LNBP, vol. 6. Springer, Heidelberg (2008)
21. Mendling, J., Reijers, H.A., Recker, J.: Activity labeling in process modeling: Empirical insights and recommendations. *Inf. Syst.* 35(4), 467–482 (2010)
22. Nissen, M.E.: Redesigning reengineering through measurement-driven inference. *MIS Quarterly* 22(4), 509–534 (1998)
23. Oulsnam, G.: Unravelling unstructured programs. *Comput. J.* 25(3), 379–387 (1982)
24. Petre, M.: Why looking isn't always seeing: Readership skills and graphical programming. *Commun. ACM* 38(6), 33–44 (1995)
25. Polyvyanyy, A., García-Bañuelos, L., Dumas, M.: Structuring Acyclic Process Models. *Information Systems* (to appear, 2012)
26. Reijers, H.A., Mendling, J.: A Study into the Factors that Influence the Understandability of Business Process Models. *IEEE Transactions on Systems Man and Cybernetics, Part A* (2010)
27. Reijers, H.A., Freytag, T., Mendling, J., Eckleder, A.: Syntax highlighting in business process models. *Decision Support Systems* 51(3), 339–349 (2011)
28. Reijers, H.A., Mendling, J.: A study into the factors that influence the understandability of business process models. *IEEE Transactions on Systems, Man, and Cybernetics, Part A* 41(3), 449–462 (2011)
29. Rieger, M., Ducasse, S., Lanza, M.: Insights into system-wide code duplication. In: *Proceedings of 11th Working Conference on Reverse Engineering*, pp. 100–109. IEEE (2004)
30. Sánchez-González, L., García, F., Mendling, J., Ruiz, F.: Quality Assessment of Business Process Models Based on Thresholds. In: Meersman, R., Dillon, T.S., Herrero, P. (eds.) *OTM 2010, Part I. LNCS*, vol. 6426, pp. 78–95. Springer, Heidelberg (2010)
31. Vanderfeesten, I., Reijers, H.A., Mendling, J., van der Aalst, W.M.P., Cardoso, J.: On a Quest for Good Process Models: The Cross-Connectivity Metric. In: Bellahsene, Z., Léonard, M. (eds.) *CAiSE 2008. LNCS*, vol. 5074, pp. 480–494. Springer, Heidelberg (2008)
32. Vanhatalo, J., Völzer, H., Leymann, F.: Faster and More Focused Control-Flow Analysis for Business Process Models Through SESE Decomposition. In: Krämer, B.J., Lin, K.-J., Narasimhan, P. (eds.) *ICSOC 2007. LNCS*, vol. 4749, pp. 43–55. Springer, Heidelberg (2007)
33. Vanhatalo, J., Volzer, J., Kohler, J.: The Refined Process Structure Tree. *DKE* 68(9), 793–818 (2009)
34. Wohlin, C.: *Experimentation in software engineering: an introduction*, vol. 6. Springer (2000)
35. Zhang, F., D'Hollander, E.H.: Using Hammock Graphs to Structure Programs. *IEEE Transactions on Software Engineering* 30(4), 231–245 (2004)

# Aggregating Individual Models of Decision-Making Processes

Razvan Petrusel

Faculty of Economics and Business Administration, Babes-Bolyai University,  
Teodor Mihali str. 58-60, 400591 Cluj-Napoca, Romania  
razvan.petrusel@econ.ubbcluj.ro

**Abstract.** When faced with a difficult decision, it would be nice to have access to a model that shows the essence of what others did in the same situation. Such a model should show what, and in which sequence, needs to be done so that alternatives can be correctly determined and criterions can be carefully considered. To make it trustworthy, the model should be mined from a large number of previous instances of similar decisions. Our decision-process mining framework aims to capture, in logs, the processes of large numbers of individuals and extract meaningful models from those logs. This paper shows how individual decision data models can be aggregated into a single model and how less frequent behavior can be removed from the aggregated model. We also argue that main process mining algorithms perform poorly on decision logs.

**Keywords:** decision data model, decision process mining, product based workflow design.

## 1 Introduction

Using the collective intelligence (e.g. opinions, preferences expressed by many individuals) in various ways is a hot topic both for research and practice. The basic idea behind a lot of approaches in this area is that ‘if a lot of people prefer, recommend or simply do something, it must be valuable’. This is the approach which works very well in various contexts (e.g. Google, Wikipedia, Amazon, etc.). But what if we apply this simple, yet powerful, principle in decision making? For example, what if somebody wants to buy a car, or a laptop, or to make an investment? There are a lot of different decision alternatives to choose from and there are even a larger number of criterions to be considered. So, wouldn’t it be nice to have, as a guide, some sort of a model that depicts the essence of what a large number of other individuals did, when faced with the same decision? Of course, there are a lot of recommender systems that do just that. Fundamentally, our approach is different. We aim to extract a workflow model that depicts the sequence of actions that were performed in order to choose a decision alternative. We do not aim to indicate which alternative should be chosen, but to show the user and lead him through the ‘right’ path of actions, so that a fully informed decision is made. ‘Right’ is the essence of

what many others have done in a similar situation. For example, we can look at the decision problem of buying shares on the stock exchange. The model can indicate that the decision makers first need to check the quotations of some stock, then compare it with some other stocks, then calculate some indicators, then check the upcoming changes of laws, and so on. From such a model can benefit decision makers with different background and knowledge regarding the issue at hand. For example, individuals who never used the stock exchange can find out what to do and when to do it; while experts can compare their approach against what others have done. It is outside our goal to recommend which choice to make (e.g. that  $x$  shares from  $Y$  company should be purchased).

The goal introduced above is quite ambitious, and raises a lot of problems. Therefore, some limitation and formalization is needed. We define the overall goal of our research as: ‘create a model of the decision making process by extracting it from a large number of individuals’. But, the decision making process is unstructured and is performed (mostly) mentally. So, we need to focus on two sub-problems. The first one is the problem of extracting the flow of mental activities from decision makers and, the second one, finding the right approach to convert what was extracted into a meaningful model. This paper focuses on providing answers to the second sub-problem, while the first one was already addressed in [1].

Our approach relies on using any software that provides data needed for a decision and is enhanced with the feature of logging what the user does while making the decision. Then, we can mine the log and create a model. Extracting process models from logged data is the basic approach in process mining. Therefore, our effort can be placed in this research area. We are interested, at this point, in the data perspective of the decision making process (i.e. we are focusing on the data items used by the decision maker and on the sequence of data derivations).

The paper is organized as follows. In the next section we provide the formal approach for the aggregation of decision making models. In the third section we give an overview of the framework we propose. In the fourth section we aim to validate our approach by showing a running example, a case study, and an experiment. In the last two sections we go through the related work and the conclusions.

## 2 The Formal Approach

In this section we define the notions that are essential for the aggregation of decision making processes.

**Definition 1 (Data Element):** Let  $D$  be a set of data elements  $d$ ,  $N$  be a set of data element names  $n$  and  $V$  be a set of data elements values  $v$ . By definition,  $d \in D$  is a pair  $d = (n, v)$ . As shorthand we use  $d.n$  to refer to the name of a data element and  $d.v$  to refer to its value. For example,  $d = (\text{income}, 2000) \in D$  is a data element where  $d.n = \text{“income”}$  and  $d.v = 2000$ .

**Property 1 (Value of a Data Element):** The value of a data element is assigned when that element is created and it doesn’t change over time. Given  $d_i, d_j \in D$  if  $d_i.n = d_j.n \implies d_i.v = d_j.v$



**Definition 2 (Basic Data Element):** A basic data element,  $bd \in BD$  where  $BD \subset D$  is a piece of data readily available, which can be used without any additional processing. In the case of decision making, this is data known/found out explicitly by the decision maker. In the case of software a  $bd$  is data that is displayed to the user.

**Definition 3 (Inputted Data Element):** An inputted data element,  $id \in ID$  where  $ID \subset D$  is a piece of data which is not available explicitly but is known (e.g. the number of months in a year may not be explicitly available). In the case of decision making, this is data that the decision maker knows implicitly.

**Definition 4 (Derived Data Element):** A derived data element,  $dd \in DD$  where  $DD \subset D$  is a piece of data that is created by the decision maker in the decision making process. The logic behind such a data element is that there are some input data elements,  $i_j, i_n \in D$ , which, through some derivation, are converted to an output element  $o_j \in DD$ . The inputs are data elements ( $bd, id$  or  $dd$ ) that must be available at the time the derivation takes place. This needs to be seen as a cause-effect situation. All causes must be enabled before the effect is produced.

**Property 3 (Data Derivation):** If a derivation is observable, the inputs and the outputs will always be visible, even if the derivation itself is a black-box. For example, in stock exchange the share quotations are the input data which, through some derivations, is converted to the Stock Exchange Index (e.g. Dow Jones). For decision-aware software, if a mental activity can be observed, the inputs and outputs should be visible in the logs.

**Definition 5 (Decision Activity):** The notion of decision activity is used to define the conscious mental action of a decision maker to transform some inputs into some outputs. Since we look only at data elements at this time, a decision activity is a data transformation (e.g. using mathematical operations) of some input data elements ( $bd, id$  or  $dd$ ) into some derived data elements  $dd$ .

In process mining, a basic trace is a finite sequence ( $\sigma$ ) of activities ( $a$ ) denoted as  $\sigma = (a_1, a_2, \dots, a_n)$  [2]. Such a trace encodes the sequence of activities but it does not show any explicit dependency between the activities. In process mining [2], it is assumed that, given the sequence  $a_i, a_{i+1} \Rightarrow a_i$  is the cause of  $a_j$  because it is a predecessor in the observed behavior. Since the decision making process is less structured than a business process, such an assumption does not hold (i.e. a decision maker might perform a set of activities in a random sequence). Therefore, we need to define a new trace format which formalizes the dependency in an explicit manner.

**Definition 6 (Decision Trace):** A decision log (DL) is defined as a multiset of decision traces. A decision trace (DT) is a tuple ( $D, DPF, T$ ) where:

- $D$  the set of data elements  $d$ ,  $D = BD \cup ID \cup DD$ ;
- $DPF$  is a set of  $dpf$ , where  $dpf$  is a function that describes the dependency of each  $DD$  as  $dpf: D \rightarrow DD$ ;
- $T$  is a set of  $t$ , where  $t$  is a timestamp.

Since a DT is defined as a tuple, a sequence of data elements is implied. Note that a derived data element depends on other data elements (i.e. is a consequence of other

data elements). Therefore, in the tuple, the ‘source’ data elements will show up before the ‘consequence’ derived data element. Basic or inputted data elements need no transformation; therefore the input element is the empty set.

To give the reader a better understanding of the definition introduced above, we will show a running example of three partial traces from the case study in the paper.

- DT1 =  $\langle\langle(\text{income}, 2000), \emptyset, \text{time1}\rangle, \langle\langle(\text{savings}, 50000), \emptyset, \text{time2}\rangle, \langle\langle(\text{property\_price}, 100000), \emptyset, \text{time3}\rangle, \langle\langle(\text{dd1}, -50000), (\text{savings}, \text{property\_price}), \text{time4}\rangle, \langle\langle(\text{period}, 240), \emptyset, \text{time5}\rangle, \langle\langle(\text{dd2}, 208.33), (\text{dd1}, \text{period}), \text{time6}\rangle, \langle\langle(\text{dd3}, 1791.67), (\text{income}, \text{dd2}), \text{time7}\rangle, \langle\langle(\text{dd4}, 25), (\text{dd1}, \text{income}), \text{time8}\rangle\rangle\rangle$
- DT2 =  $\langle\langle(\text{property\_price}, 100000), \emptyset, \text{time9}\rangle, \langle\langle(\text{savings}, 50000), \emptyset, \text{time10}\rangle, \langle\langle(\text{period}, 240), \emptyset, \text{time11}\rangle, \langle\langle(\text{dd1}, -50000), (\text{savings}, \text{property\_price}), \text{time12}\rangle, \langle\langle(\text{dd2}, 208.33), (\text{dd1}, \text{period}), \text{time13}\rangle, \langle\langle(\text{income}, 2000), \emptyset, \text{time14}\rangle, \langle\langle(\text{dd3}, 25), (\text{dd1}, \text{income}), \text{time15}\rangle, \langle\langle(\text{dd4}, 1791.67), (\text{income}, \text{dd2}), \text{time16}\rangle\rangle\rangle$
- DT3 =  $\langle\langle(\text{savings}, 50000), \emptyset, \text{time17}\rangle, \langle\langle(\text{property\_price}, 100000), \emptyset, \text{time18}\rangle, \langle\langle(\text{dd1}, -50000), (\text{savings}, \text{property\_price}), \text{time19}\rangle, \langle\langle(\text{income}, 2000), \emptyset, \text{time20}\rangle, \langle\langle(\text{period}, 240), \emptyset, \text{time21}\rangle, \langle\langle(\text{dd2}, 1791.67), (\text{income}, \text{dd1}, \text{period}), \text{time22}\rangle, \langle\langle(\text{dd3}, 25), (\text{dd1}, \text{income}), \text{time23}\rangle\rangle\rangle$

**Definition 7 (Equivalence of Basic/Inputted Data Elements):** Let  $bd_i$  and  $bd_j$  be two elements of two Decision Traces  $DT_i$  and  $DT_j$ .  $bd_i = bd_j$  if and only if  $bd_i.n = bd_j.n$  and  $bd_i.v = bd_j.v$  (i.e. both data items have the same names and the same absolute values).

**Definition 8 (Equivalence of derived data elements):** Let  $dd_i$  and  $dd_j$  be two elements of two Decision traces  $DT_i$  and  $DT_j$ .  $dd_i = dd_j$  if and only if  $(DPF_i, V_i) = (DPF_j, V_j)$ , i.e. both data items depend on the same data elements and their absolute values are equal. The basic equivalence looks only at direct predecessors (e.g. for the example decision traces  $DT2.dd1 = DT3.dd1$  but  $DT2.dd4 \neq DT3.dd2$  because, even if the values are equal, the direct inputs are different). The extended equivalence looks at all the predecessors down to the basic/inputted data items (e.g. in this case  $DT2.dd4 = DT3.dd2$  because  $DT2.dd4.dpf = \{\text{savings}, \text{property\_price}, \text{period}, \text{income}\}$  and  $DT3.dd2.dpf = \{\text{income}, \text{property\_price}, \text{savings}, \text{period}\}$  and their value is equal).

**Definition 9 (Decision Data Model):** A Decision Data Model (DDM) is a tuple  $(D, O)$  with:

- $D$ : the set of data elements  $d$ ,  $D = BD \cup ID \cup DD$
- $O$ : the set of operations on the data elements. Each operation,  $o$  is a tuple  $(d, v, DS, t)$ , where:
  - $d \in DD$ ,  $d$  is the name of the output element of the operation;
  - $v$  is the value outputted by the operation. Can be numeric or Boolean;
  - $DS$  is a list of  $ds$ ,  $ds = (ao \ X \ d)$ , where:
    - $ao \in AO$ ,  $AO = \{+, -, *, /, <\>\}$  is a set of arithmetic operations specifying how to produce the output element  $d$  based on the input elements  $ie$ ;
    - $d \in D$ ,  $d$  is an input data element.
  - $t \in T$  is the set of timestamps at which an operation from  $O$  occurs (i.e. the time when the element  $d$  is created using  $o$ ).
- $D$  and  $O$  form a hyper-graph  $H = (D, O)$ , connected and acyclic.

We use as shorthand the notation  $o_i.ds$  to refer to the set of input elements for operation  $o_i$  and  $d.v$  as the value of the data element produced by the operation.

The graphical representation of the DDM created based on DT3, is shown in Fig. 1. The model is annotated with details regarding the operations. We argue that the DDM is easy to understand even without previous knowledge of its semantics. If we look at the model in Fig. 1., it depicts clearly, for example, that the savings must be aggregated with the property price. If we look at the detail of operation 1 we can see that the savings must, actually, be deducted from the property price. The time of the operations is there to provide some sense of sequence, even if it is not strictly enforced by the semantics of the model. We argue that this loose approach is best fitted for decision processes. It balances between the strict process imposed by a workflow model and the free approach of declarative or rule-based models. The DDMs for all the decision traces introduced earlier in this sub-section are shown in Fig. 4.

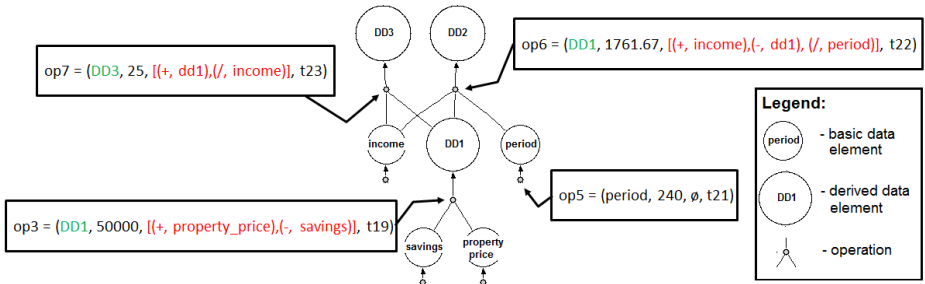


Fig. 1. DDM model of DT3, annotated with the details of the operations

**Definition 10:** An aggregated DDM is a DDM with the following annotations:

- Frequency of data elements. It indicates how many times a data element is observed in the log;
- Frequency of an operation. It too indicates how many times an operation shows up in a log. There are frequent cases in which the same data element is produced using alternative operations (e.g. the same derived data is created by a single operation with many inputs or by a structured sequence of operations). In

Fig. 4 we provide such an example.

*Algorithm 1 (creating an aggregated DDM from  $n$  DDMs):*

Let  $DDM_1, \dots, DDM_n$  be  $n$  decision data models.  $DDM_{agg}$  is created as follows:

- $D_{agg} = D_1 \cup \dots \cup D_n$
- Discard duplicates in  $D_{agg}$
- $O_{agg} = O_1 \cup \dots \cup O_n$
- Do Find\_Equivalent\_operation()
  - o evaluate every pair  $(o_i, o_j)$  where  $o_i, o_j \in O_{agg}$
  - o if  $o_i = o_j$  (by extending Definition 8 to DDM operations  $o_i = o_j$  if and only if  $o_i.ds = o_j.ds$  and  $o_i.v = o_j.v$ . The user may choose if the basic or extended definition is applied)
    - increase frequency. $(o_i)$  by 1
    - discard  $o_j$

A running example for this algorithm is introduced in sub-section 4.1.

*Algorithm 2 (abstracting from an aggregated DDM):* The goal of such an algorithm is to provide a ‘zoom-in/zoom-out’ feature for an aggregated DDM. Zooming-in only shows very frequent behavior while zooming-out also shows less frequent behavior. Let  $t$  be the frequency threshold so that the abstracted DDM will show only data elements and operations which are observed more frequently than the threshold. The abstracted DDM is created by:

- a) Do Extended\_Find\_Equivalent\_operation()
  - evaluate  $(o_i, o_j)$  where  $o_i, o_j \in O_{agg}$
  - if  $o_i = o_j$  (using Definition 8 Extended)
    - increase frequency. $(o_i)$  by 1
    - discard  $o_j$
- b) If frequency. $oi < t$ 
  - Remove  $oi$
- c) If frequency. $di < t$ 
  - Remove  $di$
- d) Do Find\_unconnected\_derived\_data
  - If found() create artificial  $oj$  with only basic or inputted data elements as inputs and  $ddj$  as output.

The last step is needed because there might be derived data items above the threshold, but produced by several alternative operations with a frequency below the threshold. For example, the threshold may be set to 3 and a derived data element may show up 4 times. But it might be produced by 2 different operations, each with a frequency of 2. So the operations would be removed while the data element would still be in the model. In this case, the artificial operation would link the derived data element directly to the basic data items involved in the derivation. Any intermediary operations below the threshold would be removed. This problem shows only when setting low thresholds (e.g. 2, 3 or 4). For an example please refer to section 4.1.

In order to validate that there is a good match between the recorded events and the DDM, we can use the notion of conformance checking. It is trivial to check the fitness for an instance DDM. Checking it for an aggregated DDM is an issue that will be approached in a separate paper. As a brief preview, we argue that, in a DDM an operation can be looked at as a transition. For replaying the sequence of the actions in the log, we require that all input elements are available when an operation is executed. However, there are differences from a Petri Net due to semantics (i.e. a transition consumes tokens from the input places) which cannot be applied to DDMs. If we strictly apply the fitness measures to an aggregated DDM it is under-fitting (i.e. there are a lot of ‘tokens’ left behind). But it is not the best validation measure.

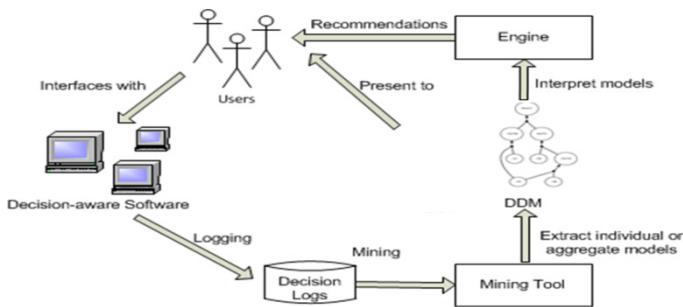
### 3 The Framework of Decision-Making Process Mining

If we move past the main idea stated in the introduction to a more practical approach, we can see that there are two main issues that need to be solved. The first one is to define a knowledge extraction method for the decision making process. This is not a

trivial matter mostly because of: the fuzzy mental processes; the heterogeneity of decision making styles; and the number of individuals that need to be involved in the studies. We proposed the use of simulation software that logs the actions of the users [1]. Such a log needs to clearly capture what are the elements used by the decision maker in the decision making process and the correlation between them.

While keeping in mind that the goal of this research is to automatically extract a model of the data perspective of the decision-making process, as performed by multiple decision makers, we show in Fig. 2, the overview of our approach.

Everything starts with a large number of decision makers that interact with software in order to make some decision. It can be any software, even a simple spreadsheet, which simply provides the users with a set of values that are relevant to the decision to be made. There should also be the possibility to log the values that are used by the decision maker in order to reach the decision. Since we are interested in business decision making, we can use as examples of such software the on-line stock trading platforms, management simulation games, on-line shops, etc. Any software can be upgraded to do the logging by various means (e.g. following the mouse moves and clicks, eye-tracking, etc.). To ease our empirical research we created our own decision simulation software. It is designed to be flexible so that any data centric decision can be easily implemented. It shows to the decision maker the values of various data items related to the decision that needs to be made. It also logs the ones actually considered by the user. The decision maker is not provided with any guidance or help. There is only a ‘calculator’ tool that assists with the data derivations (such actions are also logged). The goal of the simulation is to pick one of the decision alternatives (displayed in a list within the software) based on the actual values shown in each scenario. An example of the interface is shown in Fig. 5. We call this kind of software ‘decision-aware’. More details on the software and on how the logging is performed are available in [1].



**Fig. 2.** The framework of decision making process mining

The logs are processed by a mining application that outputs individual or aggregated Decision Data Models. More details, on how the data is mined and the individual DDM is created, are available also in [1]. The aggregated model is the main focus of this paper. The aggregation can be done for all the traces in the log or just for a selection. Once an aggregated model is created, the process of one individual cannot be distinguished from the one of another individual. Also, in an

aggregated model, one cannot look at the most frequent path in the sense of a workflow model. But it is annotated with the frequency of each data item, so the most frequent behavior of the subjects is easily visible (see Fig. 7).

Clustering traces can be interesting because it provides useful insights into decision processes. Clustering can be done based on pre-determined particularities of decision makers or based on the similarity of the mined models. The similarity metrics and the clustering based on DDMs are not the focus of the current paper and will be available in a separate paper. But once a cluster is created, the aggregated DDM can be created to depict the behavior of the decision makers in it.

Once the DDM (individual or aggregated) is created it can be introduced, as it is, to the users. In the experiment introduced in section 4 we try to prove that a DDM is easier to read and understand than other workflow notations. It can be used to gain a better understanding of the actions of a particular decision maker or to look at the aggregated behavior of many individuals.

It is also possible to provide recommendations for actions to be performed during the decision process, based on a previously mined DDM. The goal is to guide a person through the steps that need to be performed in order to make an informed decision, up to the point where the choice of one alternative needs to be made. How the recommendation can be made, based on the DDM, is also not the focus of this paper and will be approached in a separate paper.

There are several limitations to our approach:

- our entire approach relies on the ability to log the activities (some of them mental) performed by the users of the software, during the decision process. The quality of the logs relies on the logging methods. We use direct and indirect methods. Direct methods rely on the direct observation of the decision maker (e.g. eye tracking). The indirect methods rely on the possibility to rebuild the mental activities of the decision maker from his actions. The key is to find and implement the tools that will force and enable the user to enact his every thought. The formal approach is created to support both types of logging. However, the case studies and experiments rely on decision-aware software that uses only indirect methods for logging;
- it is very important not to guide the decision maker in any way. This means, basically, that all the data should be provided in a single form and in a random manner. Of course, this is unfeasible. For example, in the implementation used for the case study, we show to the user two tabs, one with data related to buying a house and one related to renting a house, with some common data items available in both (e.g. the monthly income). Hopefully, it has a minimal influence on the decision process.

There are also several assumptions we are making:

- the decision scenario provided to the user contains all the data needed for the decision at hand. The user doesn't need any other essential information;
- since the user is not guided in any way, the actions performed, and their sequence, are a direct reflection of his mental process;
- the data provided in the scenario allows the user to explore and evaluate all the possible decision alternatives;
- A particular user might overlook certain aspects of a decision. But, all the aspects of a particular decision should be discoverable, if a large number of users are observed.

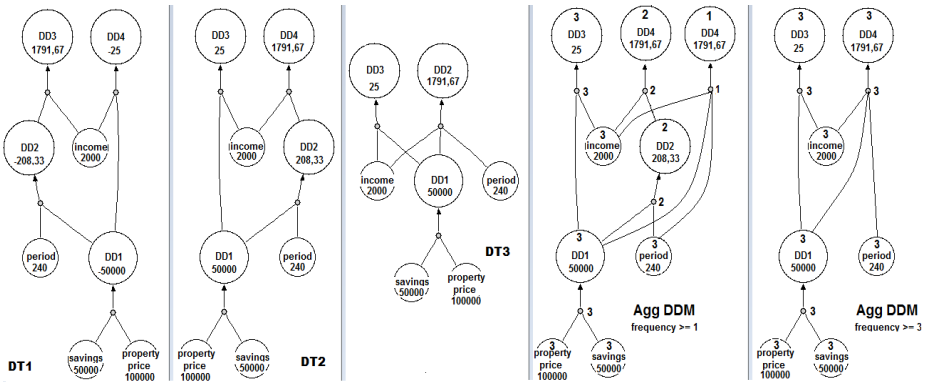
## 4 Case Study and Evaluation

This section aims to prove that our approach is feasible, and that new insight into the behavior of large numbers of decision makers is possible using a mined, aggregated DDM. The first two sub-sections also argue that the main process mining algorithms are not appropriate for extracting decision models based on logged user software interaction. The last sub-section introduces one of the experiments we performed in order to validate our claims.

### 4.1 Running Example and Comparison with Process Mining Algorithms

This subsection aims to provide a walk through our framework using a small running example. The user is presented with data regarding his financial position. He needs to decide if he can afford to buy a house. We assume a log of actions of three users, therefore obtaining three decision traces (DT1, DT2 and DT3 introduced in section 2).

The individual models for the three traces are shown in Fig. 3. We argue that the DDM clearly depicts how the data was used by the decision maker in order to reach a decision. For example, in DT1 the model shows that one action of the user was to aggregate the price of the house with the savings, producing the first derived data item (DD1). Then, given a possible loan period of 240 months, the minimum monthly installment is calculated (DD2) which is checked against the monthly income (DD3) to determine if it is affordable. Another check is performed to determine the minimum number of months needed to repay the debt (i.e. if all income was available).



**Fig. 3.** The individual models of the three decisional traces introduced in section 2 (DT1, DT2 and DT3), the aggregated DDM showing all data elements and the abstracted aggregated DDM showing only elements with frequency of at least 3

The first aggregated DDM is created using the basic notion of derived element equivalence and Algorithm 1 (defined in section 2). For example, it is obvious that  $DT1.DD1 = DT2.DD1 = DT3.DD1$  or that  $DT1.DD2 = DT2.DD2$ . But  $DT1.DD3 = DT2.DD4 \neq DT3.DD2$ , therefore DD4 in the first aggregated DDM can be produced using an operation observed twice and DD5 one observed once. The aggregated DDM

showing data items with frequency greater than 3 uses extended element equivalence and Algorithm 2. As shown in Definition 8, if we use the extended equivalence notion, the input sets of  $DT1.DD3 = DT2.DD4 = DT3.DD2 = \{DD1, income, period\}$ . Therefore, the operation  $op_3 = (DD4, 1761.67, [(+, income),(-, dd1), (/ , period)], t_3)$  is created and it is labeled as being observed 3 times.

The model in Fig. 4. is mined using Heuristics algorithm, one of the most popular process mining (PM) algorithms. There are various flaws in the logic of the process. For example, after the start activity both savings and property\_price are enabled. A possible sequence is: savings, (property\_price - savings), property\_price. But this lacks logic because one cannot calculate if he can afford a house without knowing first the price of that house. This logic is used and captured in all the traces and is not shown by the model in Fig. 4. But it is enforced by the semantics of the DDMs in Fig. 3. And there are many such examples that prove, even on such small example traces, that the PM algorithms are not suitable for mining decision logs. This is because PM looks only at the names (labels) of the activities. In a mental decision process, the decision maker rarely assigns a name to a calculated value. It is close to impossible to automatically assign a name based only on logged data. Another cause is that, basically, PM looks at the sequence and the frequency of items in a log. The sequence is essential in a business process (e.g. one cannot ‘process’ a claim if it is not ‘registered’ first) while in decision making it is rather loose. For example, it is the same process the one in which a decision maker first finds out all the basic data items then performs calculations; with the one in which he finds some of the basic data, performs some calculations, realizes new basic data is needed and finds it out, then performs further calculations. The only condition required for similar decision processes is that the choice should be based on the same basic and derived data items.

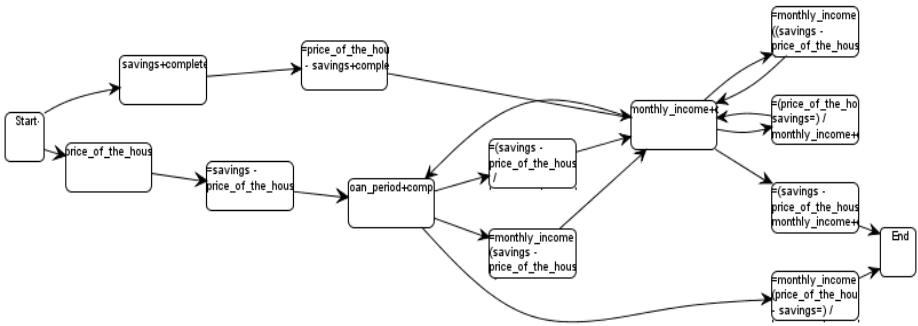


Fig. 4. The model mined with Heuristics Miner from DT1, DT2 and DT3

## 4.2 Case Study

In order to be able to experiment with our framework, we created a decision-aware system with several implementations. One implementation is simpler and introduces the user to a decision related to buying or renting a house. It only has 10 basic data items and two decision alternatives (to buy or rent the house). It is available at [http://www.edirector.ro/v3\\_1/](http://www.edirector.ro/v3_1/) (username “test” and password “test”). The second



implementation introduces the decision maker to the financial data of a company (balance sheet, cash flow, revenues and expenses, trial balance), the loan market (various data for six different loans) and one investment evaluation. The user needs to make decisions for the value of the loan to be contracted, the period, the type of the loan or the down payment to be made. This implementation is available at: [http://edirector.ro/processmining\\_v2\\_en/](http://edirector.ro/processmining_v2_en/) (also with username and password “test”).

The interface of the implementation is shown in Fig. 5. The goal of Fig. 5 is to show how we mapped the formalism introduced in Section 2 to a software. It shows some basic data items related to the decision at hand (A), allows the user to input new data elements (B) and to derive data (C). The user is not guided in any way during the decision process. All the actions (clicks) performed by a user (which basic data items are used, how a derivation is performed, what new data is inputted, etc.) are logged.

There is one limitation of the current decision-aware software which is that all data items need to be numeric in order to be involved in data derivations. The software does not support data derivations based on qualitative data items.

The screenshot shows a web browser window with the URL [www.edirector.ro/vs\\_1/pagina/cashflow](http://www.edirector.ro/vs_1/pagina/cashflow). The page has a blue header with navigation links: BUYING, RENTING, DECISION, and LOG OUT. The main content area is a financial calculator interface. It is divided into three sections marked with dashed boxes and letters A, B, and C. Section A, labeled 'BUYING', contains several input fields: 'Price of the house' (100000), 'Expenses for purchasing house' (empty), 'Monthly income' (2000), 'Savings' (50000), 'Maximum sum that can be borrowed' (empty), 'Interest per year loan' (empty), and 'Loan period' (240). Each field has an 'Add' button. Section B, located above A, contains two input fields: 'monthly\_income - (price\_of\_the\_house - savings) / loan\_period' (1791.66666667) and '(price\_of\_the\_house - savings) / monthly\_income' (25). Both have 'Add' buttons. Section C, at the top, shows a 'Fields' section with a 'Result' field containing the calculation  $(50000)/2000=25$ . Below the result are several small buttons: '+', '-', '\*', '/', and '='.

**Fig. 5.** Interface of the implemented decision-aware software, highlighting the *basic data* (A), *inputted data* (B) and *derived data* (C) areas

In Fig. 6 we show a small portion of a log used in this case study. The entire log is available at [http://edirector.ro/v3\\_1/export/pm.xml](http://edirector.ro/v3_1/export/pm.xml). The important records that are used by the mining algorithm are highlighted. The main purpose of this figure is to show that a derived data element (e.g. last highlighted record) is named in the log by the formula used to calculate it. Therefore if the same formula is used by two users, the labels of the two activities are similar in the two traces. But if the same derived data item is calculated differently (e.g. instead of A+B it is calculated as B+A) the labels will be different. More details on logging and logs are available in [1].

PI-ID	Value	Timestamp	ATE-ID	WFMEIt	Name	Value
114	user10	11/20/2011 10:05:48:449	2574	click menu item	buying   buying	
114	user10	11/20/2011 10:05:58:750	2575	click textbox	price_of_the_house   100000	
114	user10	11/20/2011 10:05:59:924	2576	click button	Add_price_of_the_house   price_of_the_house	
114	user10	11/20/2011 10:06:01:667	2577	click button	minus   -	
114	user10	11/20/2011 10:06:03:318	2578	click textbox	savings   50000	
114	user10	11/20/2011 10:06:04:205	2579	click button	Add_savings   savings	
114	user10	11/20/2011 10:06:05:554	2580	click button	=price_of_the_house - savings   50000	
114	user10	11/20/2011 10:06:31:296	2581	click textbox	monthly_income   2000	
114	user10	11/20/2011 10:06:32:7	2582	click button	Add_monthly_income   monthly_income	
114	user10	11/20/2011 10:06:33:483	2583	click button	minus   -	
114	user10	11/20/2011 10:06:34:956	2584	click button	Add_(price_of_the_house - savings)   (price_of_the_house - savings=)	
114	user10	11/20/2011 10:06:36:955	2585	click button	impartire   /	
114	user10	11/20/2011 10:06:39:88	2586	click textbox	loan_period   240	
114	user10	11/20/2011 10:06:39:874	2587	click button	Add_loan_period   loan_period	
114	user10	11/20/2011 10:06:45:421	2588	click button	=monthly_income - (price_of_the_house - savings) / loan_period   1791.66666667	
114	user10	11/20/2011 10:07:19:286	2589	click button	Add_(price_of_the_house - savings)   (price_of_the_house - savings=)	
114	user10	11/20/2011 10:07:20:538	2590	click button	impartire   /	
114	user10	11/20/2011 10:07:22:121	2591	click textbox	monthly_income   2000	
114	user10	11/20/2011 10:07:23:216	2592	click button	Add_monthly_income   monthly_income	
114	user10	11/20/2011 10:07:24:381	2593	click button	=(price_of_the_house - savings) / monthly_income   25	

Fig. 6. Partial trace from the logged user actions

The DDM created using both Algorithm 1 and Algorithm 2 is shown in Fig. 7. It is an abstracted model that shows only data elements with frequency greater than 2. This greatly reduces the size of the model because there are many items that have a frequency of 1. We believe that if a data element shows up once, it is exceptional behavior which can be safely abstracted from. Items with frequency of 2 are mostly outliers that show up in the same trace. The frequency of a data element is the number shown above its name. Because there are no derived data elements produced by two different operations, the frequency of an operation is the same as the frequency of the produced derived data element (therefore it is hidden in this particular model). To encourage the reader to follow the main behavior, the most frequent data elements are highlighted.

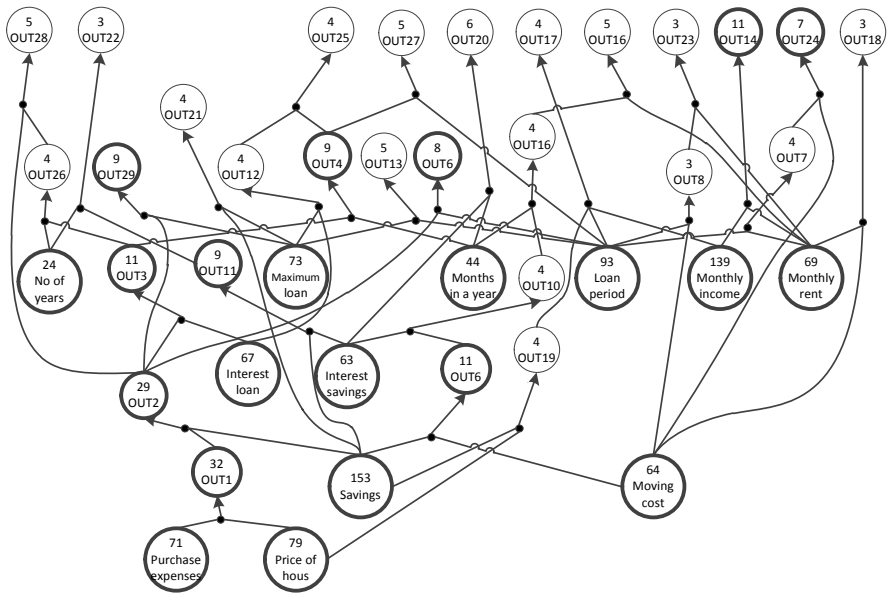


Fig. 7. Aggregated DDM (frequency of data elements greater than 2)

As one can notice, the size of the model is quite small at 40 elements (we use as a reference the conclusions presented in [2]). It is also interesting to mention that the size of the model remained relatively stable after 25 of the traces were aggregated (it already had 35 elements). This allows us to argue that, even if another 50 traces were aggregated, there is a below linear increase in the size of the model if a frequency threshold of at least 2 is used. This cannot be said about the process mining algorithms we used as a comparison. The Alpha, Heuristics and Genetic Algorithm models are spaghetti like, therefore unusable (due to the lack of space we cannot show them in this paper). Also, the size increase from 25 to 50 traces is noticeable. This is mainly due to the new behavior that is added and because of the naming of the derived data elements. The Fuzzy model is readable due to the zoom in/out feature. However, the model (for this example) does not show any new insights mostly because activities in the clusters are not logically connected.

Based on this aggregated model, it is possible to extrapolate the main behavior of the decision makers. They first find out if the house can be purchased from savings (element OUT2 with frequency 29). Since this is not possible, they find out if the purchase can be afforded by using a loan (OUT29). Since this is possible they calculate the monthly installment of capital (OUT4) and interest (OUT5). Some of the users compare the monthly rent with the monthly income (OUT14) while others also take into account that the savings (in case the money is not used for the purchase) will fetch a monthly interest (OUT11).

If we take a closer look at some of the infrequent elements we can see that about 10% of the users were also interested in how much rent they would pay for the entire period (OUT7). This is a part of an alternative path to reaching the final decision.

Also, it is noticeable that some of the basic data elements are more important (more frequently used) than others. It is possible to say, for example, that the most important piece of data, for this decision, is the monthly income.

Because of the abstraction one can notice that some of the input derived elements are less frequent than the derived elements (e.g. look at OUT7 and OUT24). This is due to the fact that the alternative operations were less frequent than the threshold).

There is also no root (final decision). If we place a root in the model and want to keep the semantics of DDM, there should be alternate operations producing the root that show which are the data items used to make the actual decision (and use the frequency to show if the same operation is used by multiple decision makers). But it is a difficult task to log the actual criterions that are used to make the final decision, without asking the users to explicitly point them out.

### 4.3 Experiment

To evaluate if a DDM is understandable, we designed an experiment that asks the subjects to follow the steps prescribed by the model. The point we try to prove is that: "A DDM is easy to read and understand and provides better insights into the decision process than other knowledge representations (YAWL or C-nets)".

To provide the answer to the statement, we created a balanced single factor experiment with repeated measurements. We investigated the effects of a single factor (a particular model type) on a common response variable to prove the degree to which the subjects understand the model. The measured response variable is two output

values that should be calculated by strictly following the model. The correctness of the output values prove if the model was correctly interpreted by the subjects.

During the experiment, the members of the focus group had to calculate a value by following a printed process model. There were three different types of models, equivalent to each other. The experiment had 3 rounds so that each subject had to use each of the three models.

Some of the concerns that limit the validity of the results are:

- the subjects performed the same process three times, using different depictions of it. Of course that, by the third use, the subjects got to know the process. Therefore, how much of the model was used in the last round is questionable.
- subject's knowledge of each model may be different. The subjects must know and have similar experience with each model. To mitigate this risk we used as subjects master students that were given a-priori lectures on workflow models. Even more, the focus was on the YAWL and C-net models and less on the DDMs.
- the domain knowledge may influence the degree to which the subjects rely on the process models. To mitigate this risk, the scenario we used requires above average knowledge of accounting and the subjects had only basic knowledge on this subject.

We are aware that the results are not statistically valid because of the limited number of subjects involved in this first experiment. We used 12 master students at our Faculty, so that in each round there were 3 groups of four students. We believe that, even so, the findings are interesting and worth mentioning. The main result is that none of the students were able to derive the expected output values. The main conclusion that arises from this experiment is that, in order to make the DDM available for non-experts in a field, the exact operations need to be made available, somehow, to the decision makers. There was also a questionnaire that, among other questions, required the subjects to indicate their favorite model. Interestingly, 9 out of 12 indicated they prefer the DDM.

## 5 Related Work

As stated in the introduction, this paper starts with the assumption that the logs are available. Therefore, we focus on the models that can be created by mining those logs. We draw inspiration from the process mining field, which is concerned with extracting a model from logged activities [3]. The most important process mining algorithms are: Alpha [2], Heuristics [4], Genetic [5] and Fuzzy [6]. Some of those algorithms apply various solutions for dealing with less structured or noisy logs. But, as we showed in section 4, the specifics of human decision making activities don't fit the basic assumptions used by process mining algorithms (e.g. that there is a 'mainstream' process in the logs). There are also limitations given by the underlying semantics of the extracted models. This is why a new model, the C-net is emerging [3]. The declarative approach to process mining [7] also didn't work on decision logs.

The model we create uses the approach in Product-based workflow support, especially the semantics of Product Data Model (PDM) [2]. There are differences due to the main purpose of the model (a PDM is a general, design model while a DDM is

focused on showing the decision process as it is captured in the logs). There are also differences in the definition of a DDM (e.g. there is no root; each data element has a value, etc.).

The classical data mining approach that looks at extracting association rules (as first introduced in [8] and then improved by many papers) or the more actual web mining [9], don't fit this particular research because our goal is to extract a workflow perspective of the decision process.

The DDM is often regarded as a decision tree. However, the semantics of the two models are quite different (e.g. a DDM may have the same leaf node connected to many other and a derived data item can be produced by several alternative operations). Before arriving to the DDM, we explored the knowledge representation area (more specifically the ontologies). Even if we were happy with the expressiveness and the tools available, we identified two major problems. It is difficult to extract, build and maintain an ontology for each decision problem and it is a challenge to adapt it to the fuzzy processes performed by many individuals [10].

In decision making theory field could not find any approach over decision making as a workflow or some bottom-up approach that starts from the instance decision processes and seeks to aggregate them. From this area, one of the approaches that influenced us is [11] because it places human decision making at the intersection of decision theory, artificial intelligence and information systems and behavioral sciences.

## 6 Conclusions

The user interaction logs with software or web pages are a great source of knowledge. In this paper we introduced our approach aimed at creating an aggregated model of the data perspective of business decision processes. The framework requires a software to log the actions performed by users in the decision making process. Then, the logs are mined for the data view of individual decision making process (producing a Decision Data Model). This paper shows how individual models can be aggregated into a single model. The approach can be exploited in connection to business decisions since most of them are data-centric.

We formalized the essential notions that are used. A decision log needs to explicitly show a causal relation (dependency, connection) between some inputs and some outputs. The DDM graphically depicts those relationships using notions as basic/derived data elements and operations. An aggregated model merges instance DDMs using the notions of basic or extended data element equivalence for mapping identical nodes to each other. It is also possible to create an aggregated model where the frequency of elements (nodes and operations) is above a certain threshold.

We showed how our approach works on a running example. We also demonstrated that there are particular features of decision processes that render existing process mining algorithms useless. For experimenting and validating our approach we used simulation software (i.e. we created an artificial decision problem and we supplied data, ranging from critical to trivial, for solving it). We randomly picked 50 traces from the log to show that our approach produces a readable and usable model that provides insights into how the users made the decision of buying or renting a house.

Logging user actions can easily be replicated to any system that provides data for decision making (e.g. ERP systems, on-line stock trading, etc.) either by following mouse movements or by employing eye-tracking. The logging issue limits our output only when new information is provided to the decision maker outside the boundaries of the software. This limitation can be trivial for real life situations that are computer-centric (e.g. stock trading) while it can render it useless for others (e.g. negotiations).

Applying our approach to real life situations is straight-forward. The only check that needs to be done before using it right away is whether the available real-life basic data is also included in the DDM. If there are more data items in real life, the user needs to evaluate the relevance of the extra data.

We do not claim to have found a solution for any kind of decision processes. It is our goal to create a domain-independent approach that can be applied with minor adjustments for various data-centric decisions. Still, every human being is unique and our method also tries to abstract from the personal features (e.g. patience in solving problems, IQ, etc.). We feel that there is a strong connection between the decision maker's human model and the individual DDM but it is all lost when aggregating tens or hundreds of individual models.

The paper is focused on the aggregation of models. Our decision-process mining framework also allows other kind of analysis such as clustering decision makers; comparisons and emphasis of the differences between two individual models, or comparisons between an individual model and an aggregated one.

We showed one of the validation experiments but there is more work to be done on this issue. We need to extend this particular experiment by involving larger numbers of subjects, with various backgrounds, so we can achieve statistically relevant results. We will also use more process notations and knowledge representations (e.g. BPMN, rule-based descriptions).

**Acknowledgments.** This work was supported by CNCS-UEFISCSU, project number PN II-RU 292/2010; contract no. 52/2010.

The author would like to thank Wil van der Aalst and Irene Vanderfeesten for sharing their ideas and providing valuable feedback which steered our entire research, not just the part presented in this paper.

## References

1. Petrusel, R., Vanderfeesten, I., Dolean, C.C., Mican, D.: Making Decision Process Knowledge Explicit Using the Decision Data Model. In: Abramowicz, W. (ed.) BIS 2011. LNBI, vol. 87, pp. 172–184. Springer, Heidelberg (2011)
2. Vanderfeesten, I., Reijers, H.A., van der Aalst, W.M.P.: Product-based workflow support. *J. Information Systems* 36, 517–535 (2011)
3. van der Aalst, W.M.P.: *Discovery, Conformance and Enhancement of Business Processes*. Springer, Heidelberg (2011)
4. Weijters, A.J.M.M., Ribeiro, J.T.S.: Flexible Heuristics Miner (FHM). In: IEEE Symposium on Computational Intelligence and Data Mining (CIDM), pp. 310–317. IEEE Press, New York (2011)

5. de Medeiros, A.K.A., Weijters, A.J.M.M., van der Aalst, W.M.P.: Genetic Process Mining: An Experimental Evaluation. *Data Mining and Knowledge Discovery* 14(2), 245–304 (2007)
6. Günther, C.W., van der Aalst, W.M.P.: Fuzzy Mining – Adaptive Process Simplification Based on Multi-perspective Metrics. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) *BPM 2007*. LNCS, vol. 4714, pp. 328–343. Springer, Heidelberg (2007)
7. Pesic, M., Schonenberg, H., van der Aalst, W.M.P.: DECLARE: Full Support for Loosely-Structured Processes. In: Eleventh IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007), pp. 287–298. IEEE Computer Society (2007)
8. Agrawal, R., Imielinski, T., Swami, A.: Mining Association Rules Between Sets of Items in Large Databases. In: *ACM International Conference on Management of Data (SIGMOD 1993)*, pp. 207–216. ACM Press, New York (1993)
9. Liu, B.: *Web Data Mining Exploring Hyperlinks, Contents, and Usage Data*. Springer, Heidelberg (2011)
10. Fensel, D.: *Ontologies: a Silver Bullet for Knowledge Management and Electronic Commerce*. Springer, Heidelberg (2001)
11. French, S., Maule, J., Papamichail, N.: *Decision behavior, analysis and support*. Cambridge University Press, Cambridge (2009)

# Generating Natural Language Texts from Business Process Models

Henrik Leopold<sup>1</sup>, Jan Mendling<sup>2</sup>, and Artem Polyvyanyy<sup>3</sup>

<sup>1</sup> Humboldt-Universität zu Berlin, Unter den Linden 6, 10099 Berlin, Germany

[henrik.leopold@wiwi.hu-berlin.de](mailto:henrik.leopold@wiwi.hu-berlin.de)

<sup>2</sup> WU Vienna, Augasse 2-6, A-1090 Vienna, Austria

[jan.mendling@wu.ac.at](mailto:jan.mendling@wu.ac.at)

<sup>3</sup> Hasso Plattner Institute, Prof.-Dr.-Helmert-Str. 2-3, 14482 Potsdam, Germany

[artem.polyvyanyy@hpi.uni-potsdam.de](mailto:artem.polyvyanyy@hpi.uni-potsdam.de)

**Abstract.** Process Modeling is a widely used concept for understanding, documenting and also redesigning the operations of organizations. The validation and usage of process models is however affected by the fact that only business analysts fully understand them in detail. This is in particular a problem because they are typically not domain experts. In this paper, we investigate in how far the concept of verbalization can be adapted from object-role modeling to process models. To this end, we define an approach which automatically transforms BPMN process models into natural language texts and combines different techniques from linguistics and graph decomposition in a flexible and accurate manner. The evaluation of the technique is based on a prototypical implementation and involves a test set of 53 BPMN process models showing that natural language texts can be generated in a reliable fashion.

**Keywords:** Natural Language Generation, Verbalization, Business Process Models.

## 1 Introduction

Business process modeling is nowadays an integral part of information systems engineering and of organizational design. Many organizations document their operations in an extensive way, often involving several dozen modelers and resulting in thousands of business process models [1]. The audience of these models is even bigger, ranging from well-trained system analysts and developers to casual staff members who are unexperienced in terms of modeling. Most of the latter lack confidence to interpret visual process diagrams. Clearly, there is a divide between the persons with modeling expertise and those without.

This problem of diverging skills is reflected by modeling methods which explicitly distinguish between modeling experts and domain experts. In this setting, *system analysts* have to formalize facts about a specific domain with which they are often not familiar. *Domain experts* have to provide details about this domain although they do not fully understand the models the analysts create.



Therefore, the validation of the models has to rely on a discourse in natural language. For data modeling, this concept is supported by Object-Role Modeling (ORM) and its predecessor NIAM [2,3]. A key feature of ORM is a direct mapping from models to natural language text called *verbalization*. This verbalization capability has been emphasized as a crucial advantage for the validation of models in a discourse between system analyst and domain expert [4].

The reason why a verbalization technique for process models is missing might be a result of several facts. First, the validation of process models in general is difficult. Domain experts are typically not familiar with fundamental process concepts such as concurrency, decision points or synchronization. In the same vein, process models are also more difficult to translate into natural language text. The non-sequential structure of a process model has to be serialized into sequential, yet execution-order preserving text. Furthermore, activity labels have to be parsed into fragments of verb-phrases, which have to be used for constructing sentences. Such a procedure has to take optionality of information (e.g. using passive voice if no actor is mentioned) explicitly into account. Finally, the overall text has to be structured in such a way that the reader can understand the process effectively.

In this paper, we address the challenge of automatic verbalization for process models. Our contribution is a technique that is able to generate natural language text from a process model, taking into account the issues of parsing text labels, sequentializing the structure of the process, as much as flexible sentence and text planning. We deem this technique to be beneficial not only for process model validation, but also for various scenarios where diagrams have to be translated into a corresponding piece of text. These include the generation of work instructions from process models, creating process handbooks, or publishing processes on the intranet to an audience that might have little experience with process modeling.

The paper is structured as follows. Section 2 introduces the background of our work. We illustrate the text generation problem by the help of an example and identify a list of challenges. Section 3 defines our generation technique. It addresses the problems of sequentializing the process model, parsing the labels and flexibly planning sentences and text structure. Section 4 provides an evaluation of this technique using a collection of process models from practice. Section 5 discusses related work before Section 6 concludes the paper.

## 2 Background

Our solution for generating natural language texts from process models builds on general process modeling concepts and natural language architectures. In this section, we introduce BPMN as the process modeling language upon which we define our approach. Furthermore, we provide an overview of natural language generation architectures and the challenges associated with generating text from process models.

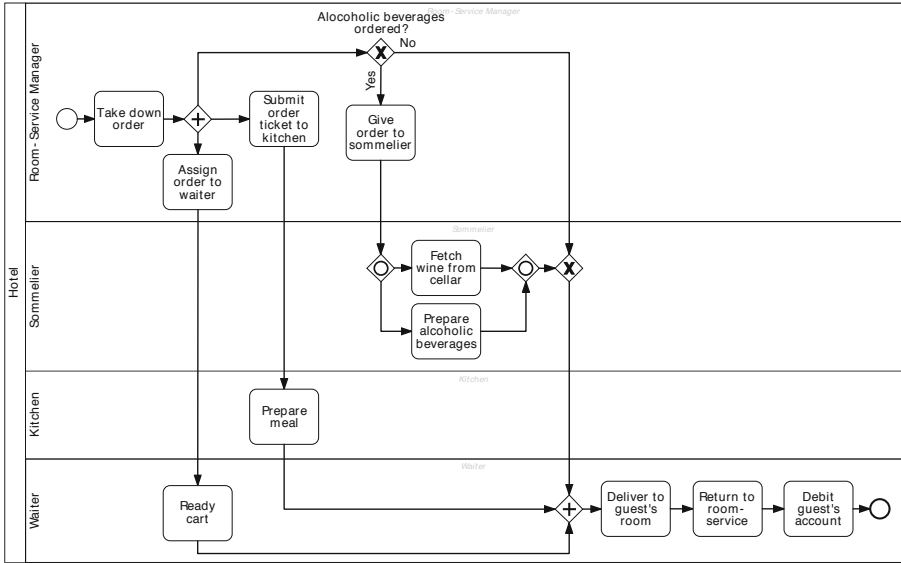


Fig. 1. Exemplary BPMN Process

## 2.1 Business Process Model and Notation

The Business Process Model and Notation (BPMN) is a process modeling standard providing a graphical notation for the specification of business processes. Recently, it was published in its current version 2.0 [5]. The symbol set of BPMN covers four types of elements: flow objects (activities, events, gateways), connecting objects (sequence and message flow, associations), swim lanes (pools and lanes) and artifacts (e.g. data objects and annotations).

Figure 1 shows an example of a business process in a hotel represented as a BPMN model. It describes how an order is handled. The process includes four roles and is hence subdivided into four lanes. The activities performed by the different actors are depicted as boxes with round corners, the diamond shaped gateways define the routing behavior. The plus in a gateway specifies a parallel execution, the circle an inclusive choice (one or more branches can be executed) and the cross an exclusive choice (only one branch can be executed). The process starts with the room-service manager taking an order. This triggers three streams of action: a meal is prepared in the kitchen, beverages may be prepared if required, and the delivery is arranged by the waiter.

## 2.2 Architectures of Natural Language Generation Systems

In general, there are different techniques available in order to translate process models into natural language text. Simple, also often called non-linguistic, approaches are based on canned text or templates-based systems. In the first case

some input data is directly mapped to a predefined string. For instance, a system translating weather data into natural language text could use the sentence *The weather will be nice today* for expressing a warm and sunny day. Slightly more advanced is the use of templates where at least some information is added to the predefined string. In this case, the template *Today there is a X% probability of rain* could be filled with the according rain probability derived from a data source. However, such approaches are not considered to be truly linguistic techniques as the manipulation is done at the character string level [6].

Linguistic, or *real* natural language generation approaches, use intermediate structures to obtain a deeper representation of the text. Such an intermediate structure usually specifies the main lexemes (abstract representations of words encapsulating inflectional aspects) for each sentence. Moreover, it carries additional information, defining for instance the tense or the modus of the verb. As pointed out by Reiter, many natural language generation systems take a three-step pipeline approach including the following stages [7]:

1. **Text Planning:** First, the information is determined which is communicated in the text. Furthermore, it is specified in which order this information will be conveyed.
2. **Sentence Planning:** Afterwards, specific words are chosen to express the information determined in the preceding phase. If applicable, messages are aggregated and pronouns are introduced in order to obtain variety.
3. **Surface Realization:** Finally, the messages are transformed into grammatically correct sentences.

Natural language generation systems have also been defined in a functional way [8]. Nevertheless, core of all these architectures is the usage of an intermediate structure for storing messages before they are transformed into natural language sentences. The advantage of this procedure is the significant gain in maintainability and flexibility. In a template-based system, each template must be manually modified if a change in the output text is required. In a linguistic-based approach, the output of the generation system can be altered by changing a parameter of the intermediate structure. For instance, the sentence *The weather will be nice today* can be easily transformed into *The weather is nice today* by adapting the tense feature of the main verb in the intermediate representation. Although templates and canned text have been critically discussed, they also have advantages [9]. Therefore, Reiter and Mellish propose a cost benefit analysis [10]. As a result, many natural language generation systems use hybrid approaches where linguistic techniques are combined with canned text and templates [11][12].

### 2.3 Challenges in Generating Text from Process Models

There are a number of challenges for the automatic generation of text from process models. We identified a list of challenges by analyzing the required generation steps and by investigating the respective literature on natural language

**Table 1.** Challenges in Generating Text from Process Models

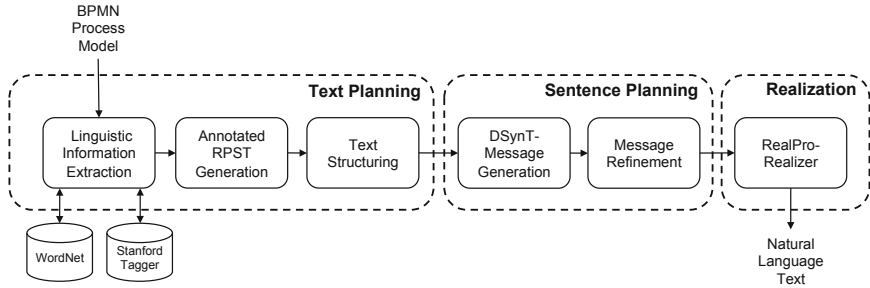
Challenge	References
1 Text Planning	
1.1 Linguistic Information Extraction	<a href="#">13</a> , <a href="#">14</a>
1.2 Model Linearization	<a href="#">15</a> , <a href="#">16</a>
1.3 Text Structuring	<a href="#">17</a>
2 Sentence Planning	
2.1 Lexicalization	<a href="#">18</a>
2.2 Message Refinement	<a href="#">19</a> , <a href="#">20</a>
3 Surface Realization	<a href="#">21</a> , <a href="#">22</a>
4 Flexibility	<a href="#">7</a> , <a href="#">23</a>

generation systems. The challenges can be assigned to one of four different categories including text planning, sentence planning, surface realization and flexibility. Table 1 provides an overview of these challenges and according references.

In the *text planning* phase we face three main challenges. First, we have to adequately infer given linguistic information from process model elements. For instance, the activity *Take down order* must be automatically split up into the action *take down* and the business object *order*. Without this separation, it would be unclear which of the two words defines the verb. Label analysis is further complicated by the shortness of process model labels and the ambiguity of the English language [24]. The second challenge is the linearization of the process model to a sequence of sentences. Process models rarely consist of a plain sequence of tasks, but also include concurrent branches and decision points. In addition to these tasks, it must be decided where techniques of text structuring and formatting such as paragraphs and bullet points should be applied.

The *sentence planning* phase entails the tasks of lexicalization and message refinement. The aspect of lexicalization refers to the mapping from BPMN constructs to specific words. It requires the integration of linguistic information extracted from the process model elements and of control structures as splits and joins in such a way that the process is described in an understandable manner. The aspect of message refinement refers to the construction of text. It includes the aggregation of messages, the introduction of referring expressions as pronouns and also the insertion of discourse markers such as *afterwards* and *subsequently*. In order to suitably consolidate sentences, the option of aggregation must first be identified and then decided where it can be applied to improve the text quality. The introduction of referring expressions requires the automatic recognition of entity types. For instance, the role *kitchen* must be referenced with *it* while the role *waiter* must be referenced with *he* or *she*. The insertion of discourse markers should further increase the readability and variability of the text. Hence, varying markers must be inserted at suitable positions.

In the context of the *surface realization*, the actual generation of a grammatically correct sentences is performed. This requires the determination of a suitable word order, the inflection of words, introduction of function words (e.g. articles) and also tasks such as punctuation and capitalization.



**Fig. 2.** Architecture of our NLG System

Besides the core natural language generation tasks, we consider *flexibility* to be an important feature. As we do not expect the input models to adhere to certain conventions, we have to deal with considerably differing characteristics of the input models. Different scenarios have to be covered, with varying implications for the output text. For instance, if a model uses lanes and thus provides a role description, the sentence can be presented in active voice (e.g. The clerk checks the application). If it is unknown who performs the considered task, the description must be presented in passive voice (The application is checked).

### 3 Text Generation Approach

This section defines our approach to text generation. Figure 2 gives an overview of the six major components building a pipeline architecture. The following subsections will introduce each component in detail.

#### 3.1 Linguistic Information Extraction

The goal of this component is the adequate inference of linguistic information from all labeled process model elements. Thereby, it is important that we are able to deal with different types of linguistic representations, so-called label styles. Therefore, we extended prior work which extracted action and business object from activity labels and events [25,13,14]. We included gateways and arc labels in the extraction algorithm to cover all relevant process model elements. The Stanford Parser [26] and the lexical database WordNet [27] are used to recognize different patterns in gateway and arc labels which we identified in the context of a comprehensive analysis of industry process models. Based on the structural insights of process model labels, we can extract action, business object and possible modifying attributes from any process model label, independently from the actual linguistic presentation.

Consider the gateway *Alcoholic Beverages Ordered?* from Figure 1. In this case the application of the Stanford Parser is impeded by the shortness of the label. It returns the tag result *Alcoholic/NNP Beverages/NNP Ordered/NNP ?/.* indicating that all words are nouns. Hence, it neither recognizes the adjective

nor the participle at the end. Being aware of regular labeling structures, we can use Wordnet to determine that the first word is an adjective and that the last word is a verb with the suffix *ed*. As a result we obtain an annotation record for this gateway containing the extracted information. Once this has been done for all labeled process model elements, the annotation records are handed over to the next module.

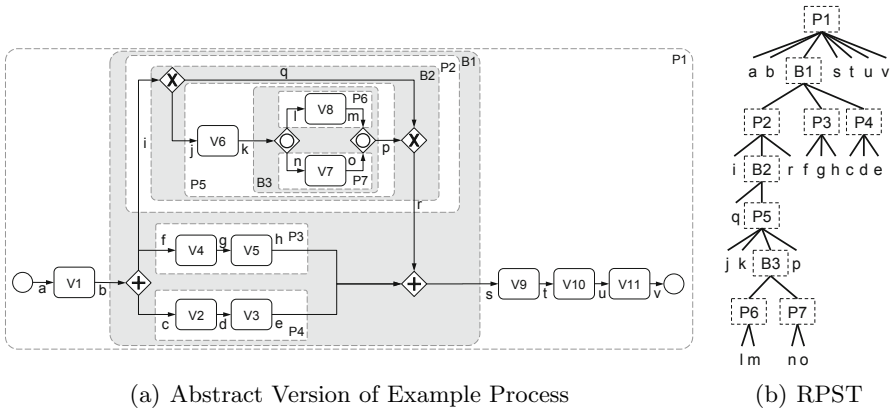
### 3.2 Annotated RPST Generation

The RPST Generation module derives a tree representation of the input model in order to provide a basis for describing the process step by step. In particular, we compute a Refined Process Structure Tree (RPST) which is a parse tree containing a hierarchy of subgraphs derived from the original model [15,16]. The RPST is based on the observation that every workflow graph can be decomposed into a hierarchy of logically independent subgraphs having a single entry and single exit. Such subgraphs with a single entry and a single exit are referred to as fragments. In a RPST any two of these fragments are either nested or disjoint. The resulting hierarchy can be shown as a tree where the root is the entire tree and the leaves are fragments with a single arc.

In total we may encounter four different fragment classes: trivial fragments (T), bonds (B), polygons (P) and rigids (R). Trivial fragments consist of two nodes connected with a single arc. A bond represents a set of fragments sharing two common nodes. In BPMN process models this generally applies for split and join gateways, including more complex split and join structures such as loops. Polygons capture sequences of other fragments. Hence, any sequence in a process model is reflected by an according polygon fragment. If a fragment cannot be assigned to one of the latter classes, it is categorized as a rigid. Although the original version of the RPST was based on graphs having only a single entry and exit point, the technique can be easily extended to compute a RPST for arbitrary process models. Figure 3 illustrates the concepts using an abstracted version of the hotel process and its corresponding RPST.

The RPST generation algorithm by [16] does not order the fragments with respect to the control flow. However, this can be easily accomplished. For each level in the RPST the order can be determined by arranging the fragments according to their appearance in the process model. Hence, the first level starts with the trivial fragment *a*, connecting the start event and vertex *V1*. Accordingly, the trivial fragment *b*, the bond *B1* and the trivial fragments *s*, *t*, *u* and *v* are following. If the order is not defined, for instance in case of parallel branches as within the bond *B1*, an objective criteria as the length of each path can be imposed which is conducive for text generation purposes.

In addition to the introduction of a suitable order, we annotate the RPST with the linguistic information from the extraction phase and with additional meta information. For instance, the vertex *V1* from the trivial fragment *a* is annotated with the action *take down*, the business object *order* and the role *room-service manager*. The bond *B1* is annotated with the action *order*, the business object



**Fig. 3.** Abstract Version of Figure 1 and its RPST

*beverages* and the adjective *alcoholic*. Further, the bond as tagged as an XOR-gateway with *Yes/No*-arcs of the type *skip*. The latter aspect is derived from the fact that one branch is directly flowing into the join gateway and hence provides a possibility to skip the activities on the alternative branch.

### 3.3 Text Structuring

To obtain a manageable and well readable text we use paragraphs and bullet points to structure the messages. We include editable parameters for defining the size of paragraphs. Once such a threshold is reached, the change of the performing role or an intermediate event is used to introduce a new paragraph. Similarly, we make use of bullet points. Every sequence longer than a definable parameter performed by the same role is presented as a bullet list. Further, the branches of any split having more than two outgoing arcs are presented as bullet points. In case of nested splits, the bullet points are indented accordingly in order to enable the reader to easily keep track of nested structures.

### 3.4 DSynt-Message Generation

The message generation component transforms the annotated RPST into a list of intermediate messages. This means that the linguistic information from the model is not directly mapped to the output text, but to a conceptual representation which still allows for diverse modifications. In particular, we store each sentence in a deep-syntactic tree (DSyntT), which is a dependency representation introduced in the context of the Meaning Text Theory [28]. In a deep-syntactic tree each node is labeled with a semantically full lexeme, meaning that lexemes such as conjunctions or auxiliary verbs are excluded. Further, each lexeme carries grammatical meta information, so-called grammemes. Grammmemes include voice and tense of verbs or number and definiteness of nouns. The advantages of deep-syntactic trees are the rich but still manageable representation of sentences and

the existence of off-the-shelf surface realizers which take deep-syntactic representations as input and directly transform it into a grammatically correct sentence.

Based on this intermediate representation we developed an algorithm which recursively traverses the RPST and generates DSynT-based messages for trivial fragments and bonds. The following paragraphs introduce the main concepts how this mapping is conducted in our generation system.

As already pointed out, *trivial fragments* always consist of two activities with a simple connection. For the message generation we only consider the source activity, as the target activity is also included in the subsequent fragment as a source. Using the annotation from the RPST the considered activity can be directly mapped to a DSynT. For illustrating this procedure consider the first activity from the example process (Figure 4). Using the action *take down* as the main verb, the role *room-service manager* as subject and the business object *order* as object, we can derive the DSynT depicted in Figure 4(a) representing the sentence *The room-service manager takes down the order*. In case a considered activity is a sub process, we add a footnote to the sentence stating that this step is further explained in an extra model.

The transformation of bonds is more complex. For demonstrating the approach we examine the bond *B1* from our example process. This bond starts with an XOR-gateway labeled with *Alcoholic Beverages Ordered*. Based on the annotation we derive the condition clause *If alcoholic beverages are ordered*. This clause is then passed to the first activity of the *yes*-arc, where the condition and the main clause are combined. As a result, we obtain a DSynT representing the sentence *If alcoholic beverages are ordered the room-service manager gives the order to the sommelier*. The according DSynT is depicted in Figure 4(b). Similarly, we can accomplish the transformation of the join-gateway. The join-condition clause is then passed to the first activity after the bond (*Deliver to Guest's Room*) and incorporated accordingly. The procedure is used to handle bonds of different size and type. In case a process model does not provide any information about the condition of a split, we use predefined DSynT-templates for obtaining a suitable description. Depending on the requirements of the target group, these templates can be easily modified or extended. Within the bond, the recursive transformation algorithm is executed accordingly.

The text generation process builds on the creation of DSynTs with grammemes capturing all required meta information. This includes simple attributes like the word class, but also more sophisticated features as the tense or voice of verbs, the definiteness of nouns or the position of a conditional phrase (see attribute *starting\_point* in Figure 4(b)). In order to obtain a high degree of flexibility, we implemented a rule system covering an extensive set of modeling scenarios. As a result, our generation algorithm can automatically decide on these features based on the given circumstances. Accordingly, the absence of a role description automatically leads to a passive sentence. In such a case, the first activity of the example process would lead to a DSynT representing the sentence *The order is taken down*.



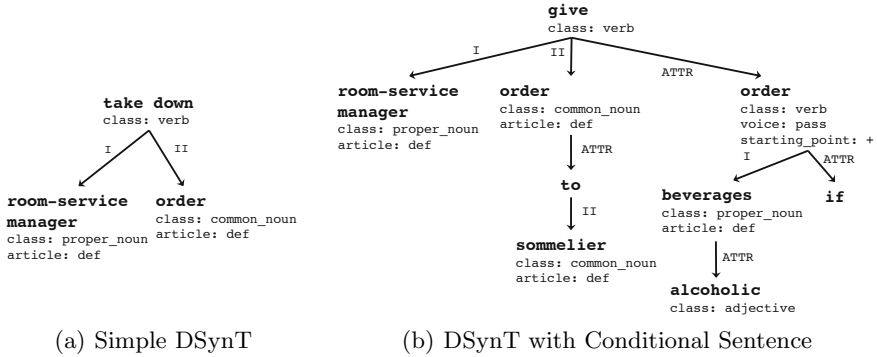


Fig. 4. Deep-Syntactic Tree of Two Messages from Figure 11

### 3.5 Message Refinement

Within the message refinement component, we take care of message aggregation, referring expression generation and discourse marker insertion.

The need for *message aggregation* usually arises when the considered process contains long sequences. In such cases we make use of three aggregation techniques: role aggregation, business object aggregation and action aggregation. For instance, if two successive activities are performed by the same role, the messages are merged to a single sentence. Similarly, activities with equal actions or business objects are merged to one sentence, if they appear in two sequential activities.

If there are still adjacent messages with the same role after the aggregation, the role description in the second message is replaced with a *referring expression*. We use WordNet for replacing a role with a suitable personal pronoun. More specifically, we infer all hypernyms of the term associated with the considered role. As a result we obtain a set of more abstract words which semantically include the role description. If we e.g. look up the role *waiter* we will find the hypernym *person* indicating that this role can be replaced with *he* or *she*. By contrast, the set of hypernyms of *kitchen* only contains words like *artifact* or *physical entity* and no link to a human being. Hence, the role *kitchen* is reference with *it*.

For the discourse marker introduction, we identify messages appearing in a strict sequence. Using an extendible set of connectors, we randomly insert a suitable word. In this way, we obtain a well readable text with sufficient variety.

### 3.6 Surface Realization

As already pointed out earlier, the complexity of the surface realization task led to the development of publicly available realizers such as TG/2 [22] or Real-Pro [21]. Considering aspects as the manageability of the intermediate structure,

license costs, generation speed and Java compatibility, we decided to utilize the DSynT-based realizer RealPro from CoGenTex. RealPro requires an XML-based DSynT as input and returns a grammatically correct sentence.

## 4 Evaluation

In this section, we demonstrate the capability of our approach to transform process models to texts. We implemented a prototype and tested it on a set of 53 BPMN models by evaluating the generated texts.

### 4.1 Prototypical Implementation

We used the approach as defined in the previous section for implementing a Java prototype. To this end, we confine the system with regard to two dimensions.

First, we reduced the amount of symbols covered by our system due to the extensiveness of the BPMN 2.0 symbol set. This decision was based on the observation that only a small set of elements is used in practice [29]. As a result, our prototype supports the transformation of the following elements: pools, lanes, activities, standard start and end events, message events, all gateway types, sequence and message flows and sub processes. However, as the capabilities of our technique are not associated with the coverage of individual symbols but with the conceptual approach, our technique can be easily extended.

Secondly, the current implementation supports structured processes and is not yet able to transform rigids. Nevertheless, there are strategies for obtaining structured processes from rigids [30]. These transformation algorithms can be utilized as a preprocessing step.

### 4.2 Evaluation Setup

To assess the capability of our technique we conducted an experiment. The overall goal of the experiment was to answer the following four questions:

1. Does the text contain an adequate sentence for each process model element?
2. Can models be adequately transformed?
3. Are the generated sentences grammatically correct?
4. Are the generated sentences stylistically favourable?

We designed a test collection of 53 BPMN process models, which cover diverse domains and mainly stem from academic training. Table 2 summarizes the characteristics of the test set. We asked three BPM domain experts with on average 4.3 years of process modeling experience to independently assess the 53 generated natural language texts with regard to the introduced questions. Any remark from one of the three evaluators was included in the final assessment. In addition to this qualitative dimension, we included characteristics such as the number of words per sentence to further evaluate the texts from a quantitative perspective.

**Table 2.** Test Set Characteristics

Property	Min	Max	Average	Total
Number of Activities	3	16	8.1	428
Number of Events	2	4	2.4	128
Number of Gateways	1	10	3.9	204

### 4.3 Results

From the evaluation experiment we learned that the first three of our assessment criteria are considered to be fulfilled. Our technique generated grammatically correct texts, which appropriately and completely describe the according process models. As an example consider the following text which was generated by our prototype and represents the process model from Figure 1. It illustrates the handling of labeled and unlabeled gateways, nested structures and how referring expressions are introduced in sequences.

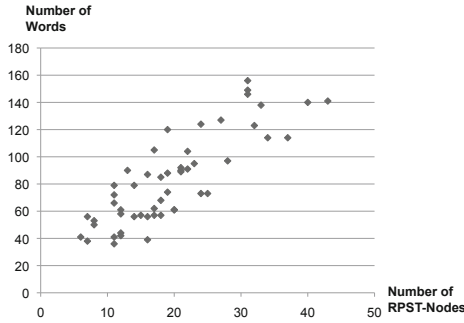
*The process begins, when the Room-Service Manager takes down an order. Then, the process is split into 3 parallel branches:*

- In case alcoholic beverages are ordered, the Room-Service Manager gives the order to the sommelier. Afterwards, one or more of the following paths is executed:*
  - The Sommelier fetches the wine from the cellar.*
  - The Sommelier prepares the alcoholic beverages.*
- The Room-Service Manager assigns the order to the waiter. Subsequently, the Waiter readies the cart.*
- The Room-Service Manager submits the order ticket to the kitchen. Then, the Kitchen prepares the meal.*

*Once all 3 branches were executed, the Waiter delivers to the guest's room. Afterwards, he returns to the room-service. Finally, the Waiter debits the guest's account.*

One remark of the evaluators was that some models describe the processes in a very general manner. As an example, consider the transformation of the OR-join in the generated text. However, this results from the fact that a process model does not provide further information on the split condition and the performed activities. Accordingly, the lack of information in the model always translates to a lack of information in the text.

With respect to the fourth criterion, in total five sentences were considered to be stylistically flawed. In particular, the evaluators criticized sentences where definite articles were unnecessarily inserted. Consider the activity labels *Assign negative points* and *Ship motorcycles to customer*. The first label is transformed into the sentence *The teacher assigns the negative points*. Obviously, the use of the definite article *the* before *negative points* is not appropriate. Although it is not a grammatical mistake, humans would most likely leave out this article. In the second case, the activity label is transformed into *The vendor ships the*



**Fig. 5.** Correlation between Model and Text Size

*motorcycles to the customer.* Here the usage of the definite article before *customer* is necessary in order to obtain a naturally appealing text. These examples highlight that the introduction of articles is a context-sensitive task and has to be further investigated in the future. However, due to the small amount of these cases, we still consider our technique to produce text of suitable quality.

The quality of the generated texts is also supported by different quantitative characteristics. As originally pointed out in [31], the sentence length is an important factor for the overall text comprehension. With an average sentence length of 9.4 words, the generated text can be considered to be easily understandable. Further factors for text comprehension are the size of the text and its paragraphs. Figure 5 illustrates the correlation between text and model size, where the latter aspect is represented by the number of RPST nodes resulting from the model. Although some variation is caused by deeper explanations for bonds, the correlation is approximately linear. As a result, also more complex models are converted into texts of manageable size. In addition, the average size of the generated paragraphs of 3.1 sentences indicates a proper organization of the presented text.

## 5 Related Work

Techniques for Natural Language Generation have been around for many years and were applied in many different scenarios. Some examples are the generation of weather forecasts [23] or the creation of reports on computer systems [32]. Research related to text generation for conceptual models can be divided into three categories: text generation systems, usage of natural language for formal specifications and comprehension of text versus models.

There are a few applications of text generation for conceptual modelling. The ModelExplainer generates natural language descriptions of object models [33] and the GeNLangUML system transforms UML class diagrams into text specifications [34]. None of these approaches tackles the specifics of process models. Our approach complements these systems by introducing a flexible technique which is able to handle different aspects of control flow.

Natural language is also used for formally specifying business semantics. For instance, the OMG standard Semantics of Business Vocabulary and Business Rules (SBVR) uses natural language for specifying complex business rules [42]. Our technique contrasts this approach by providing a rather informal complementation to a model while the natural language in SBVR represents a formal specification itself.

The merits of verbalization for model comprehension and validation are discussed from different perspectives. The need for natural language feedback in the validation of conceptual models is addressed in [35,36] by proposing natural language generation as an appropriate solution. This is in line with the approach of ORM and NIAM in terms of verbalization [2,3]. The advantages of text and diagram have been intensively debated in the area of visual programming languages (see [37] for an overview). Today, the consensus is rather that text plus diagram provides better comprehension than any of the two in isolation [38]. The most important reference for this insight is provided by the Cognitive Theory of Multi Media Learning introduced by Mayer [39]. In a series of experiments Mayer demonstrated the value of the combination of text and a graphical representation.

## 6 Conclusion

In this paper, we presented an automatic approach for generating natural language texts from BPMN process models. This approach combines natural language analysis, graph decomposition techniques and a linguistic framework for flexible text generation. The approach has been implemented as a prototype. An evaluation of 53 BPMN process models shows that our technique is capable of reliably generating grammatically and stylistically appropriate texts, which adequately describe the considered process models.

Although the current results are very promising, our technique still requires further empirical tests. Most importantly, we plan to demonstrate that the generated text really helps users to understand process models. We assume that especially non-frequently used model elements such as attached events or complex gateways might be unclear to model readers. Further, we think that particularly people who are not very familiar with BPMN can benefit from an additional text. We aim to address these assumptions in a series of experiments. Moreover, we also plan to consider additional use cases for our technique such as the generation of test scenarios or personalized work instructions.

In addition to these points, the approach should be further generalized. We intend to include the whole BPMN symbol set and to cover text generation for rigids that cannot be structured. This could be done, for instance, by determining preconditions for specific unstructured sequences of activities as proposed in [40].

Beyond that, we plan to extend the current prototype towards a comprehensive technique for process model validation. By consulting the generated texts, also users who are not confident with interpreting process models can decide

about the validity of a model. They should then have an interface to edit the generated text. A reverse generation procedure as proposed in [41] could then be used to apply the changes in the model.

## References

1. Rosemann, M.: Potential Pitfalls of Process Modeling: Part A. *Business Process Management Journal* 12(2), 249–254 (2006)
2. Verheijen, G., van Bekkum, J.: NIAM, an information analysis method. In: IFIP WG8.1 Conf. on Comparative Review of Inf. System Method, pp. 537–590 (1982)
3. Nijssen, G., Halpin, T.: *Conceptual Schema and Relational Database Design: a fact oriented approach*. Prentice-Hall, Inc. (1989)
4. Frederiks, P., Weide, T.: Information modeling: The process and the required competencies of its participants. *Data & Knowledge Engineering* 58(1), 4–20 (2006)
5. OMG: *Business process model and notation (bpmn) version 2.0*. (2011)
6. Reiter, E.: Nlg vs. templates. In: *Proceedings of the Fifth European Workshop on Natural Language Generation*, pp. 95–106 (1995)
7. Reiter, E., Dale, R.: *Building applied natural language generation systems*. *Nat. Lang. Eng.* 3, 57–87 (1997)
8. Cahill, I., et al.: In search of a reference architecture for nlg systems, 77–85 (1999)
9. van Deemter, K., Krahmer, E., Theune, M.: Real versus Template-Based Natural Language Generation: A False Opposition? *Comput. Linguistics* 31(1), 15–24 (2005)
10. Reiter, E., Mellish, C.: Optimizing the costs and benefits of natural language generation. In: *IJCAI*, pp. 1164–1171 (1993)
11. Galley, M., Fosler-Lussier, E., Potamianos, A.: Hybrid natural language generation for spoken dialogue systems (2001)
12. Reiter, E., Mellish, C., Levine, J., Bridge, S.: Automatic generation of on-line documentation in the idas project. In: *ANLP*, pp. 64–71 (1992)
13. Leopold, H., Smirnov, S., Mendling, J.: Recognising Activity Labeling Styles in Business Process Models. *EMISA* 6(1), 16–29 (2011)
14. Leopold, H., Mendling, J., Reijers, H.: On the Automatic Labeling of Process Models. In: Mouratidis, H., Rolland, C. (eds.) *CAiSE 2011*. LNCS, vol. 6741, pp. 512–520. Springer, Heidelberg (2011)
15. Vanhatalo, J., Völzer, H., Koehler, J.: The refined process structure tree. *Data & Knowledge Engineering* 68(9), 793–818 (2009)
16. Polyvyanyy, A., Vanhatalo, J., Völzer, H.: Simplified Computation and Generalization of the Refined Process Structure Tree. In: Bravetti, M. (ed.) *WS-FM 2010*. LNCS, vol. 6551, pp. 25–41. Springer, Heidelberg (2011)
17. Meteer, M.W.: *Expressibility and the Problem of Efficient Text Planning*. St. Martin's Press, Inc., New York (1992)
18. Stede, M.: Lexicalization in natural language generation: A survey. *Artificial Intelligence Review* 8, 309–336 (1994)
19. Dalianis, H.: Aggregation in natural language generation. *Computational Intelligence* 15(4), 384–414 (1999)
20. Kibble, R., Power, R.: An integrated framework for text planning and pronominalisation. In: *Natural Language Generation*, pp. 77–84. *ACL* (2000)
21. Lavoie, B., Rambow, O.: A fast and portable realizer for text generation systems. In: *Applied Natural Language Processing*, pp. 265–268. *ACL* (1997)

22. Busemann, S.: Best-first surface realization. *Interface* 10 (1996)
23. Goldberg, E., Driedger, N., Kittredge, R.: Using natural-language processing to produce weather forecasts. *IEEE Expert* 9(2), 45–53 (1994)
24. Dixon, R.: Deriving Verbs in English. *Language Sciences* 30(1), 31–52 (2008)
25. Leopold, H., Smirnov, S., Mendling, J.: Refactoring of Process Model Activity Labels. In: Hopfe, C.J., Rezgui, Y., Métais, E., Preece, A., Li, H. (eds.) *NLDB 2010*. LNCS, vol. 6177, pp. 268–276. Springer, Heidelberg (2010)
26. Klein, D., Manning, C.D.: Fast Exact Inference with a Factored Model for Natural Language Parsing. In: *NIPS 2003*, vol. 15. MIT Press (2003)
27. Miller, G.: WordNet: a lexical database for English. *Comm. ACM* 38(11), 39–41 (1995)
28. Mel'cuk, I., Polguère, A.: A formal lexicon in the meaning-text theory (or how to do lexica with words). *Computational Linguistics* 13(3-4), 261–275 (1987)
29. zur Muehlen, M., Recker, J.: How Much Language Is Enough? Theoretical and Practical Use of the Business Process Modeling Notation. In: Bellahsène, Z., Léonard, M. (eds.) *CAiSE 2008*. LNCS, vol. 5074, pp. 465–479. Springer, Heidelberg (2008)
30. Polyvyanyy, A., García-Bañuelos, L., Dumas, M.: Structuring acyclic process models. *Information Systems* (to appear, 2012)
31. Flesch, R.: How to test readability (1951)
32. Iordanskaja, L., Kittredge, R., Polguère, A.: Lexical selection and paraphrase in a meaning-text generation model. In: *Natural Language Generation in Artificial Intelligence and Computational Linguistics*, pp. 293–312 (1991)
33. Lavoie, B., Rambow, O., Reiter, E.: The modelexplainer. In: *Proceedings of the 8th International Workshop on Natural Language Generation*, pp. 9–12 (1996)
34. Meziane, F., Athanasakis, N., Ananiadou, S.: Generating natural language specifications from uml class diagrams. *Requirements Engineering* 13, 1–18 (2008)
35. Dalianis, H.: A Method for Validating a Conceptual Model by Natural Language Discourse Generation. In: Loucopoulos, P. (ed.) *CAiSE 1992*. LNCS, vol. 593, pp. 425–444. Springer, Heidelberg (1992)
36. Rolland, C., Proix, C.: A Natural Language Approach for Requirements Engineering. In: Loucopoulos, P. (ed.) *CAiSE 1992*. LNCS, vol. 593, pp. 257–277. Springer, Heidelberg (1992)
37. Whitley, K.N.: Visual programming languages and the empirical evidence for and against. *J. Vis. Lang. Comput.* 8(1), 109–142 (1997)
38. Ottensooser, A., Fekete, A., Reijers, H.A., Mendling, J., Menictas, C.: Making sense of business process descriptions: An experimental comparison of graphical and textual notations. *Journal of Systems and Software* (to appear, 2012)
39. Mayer, R.: *Multimedia Learning*, 2nd edn. Cambridge Univ. Press (2009)
40. Ouyang, C., Dumas, M., van der Aalst, W., ter Hofstede, A., Mendling, J.: From business process models to process-oriented software systems. *ACM Trans. Softw. Eng. Methodol.* 19(1) (2009)
41. Friedrich, F., Mendling, J., Puhmann, F.: Process Model Generation from Natural Language Text. In: Mouratidis, H., Rolland, C. (eds.) *CAiSE 2011*. LNCS, vol. 6741, pp. 482–496. Springer, Heidelberg (2011)
42. OMG: *Semantics of Business Vocabulary and Business Rules (SBVR)* (2008)

# Towards Conflict-Free Composition of Non-functional Concerns

Benjamin Schmeling<sup>1,2</sup>, Anis Charfi<sup>1</sup>, Marko Martin<sup>1</sup>, and Mira Mezini<sup>2</sup>

<sup>1</sup> Software Engineering & Tools, SAP Research Darmstadt, Germany  
`firstname.lastname@sap.com`

<sup>2</sup> Technische Universität Darmstadt, Darmstadt, Germany  
`mezini@informatik.tu-darmstadt.de`

**Abstract.** In component-based software development, applications are decomposed, e.g., into functional and non-functional components which have to be composed to a working system. The composition of non-functional behavior from different non-functional domains such as security, reliability, and performance is particularly complex. Finding a valid composition is challenging because there are different types of interdependencies between concerns, e.g. mutual exclusion, conflicts, and ordering restrictions, which should not be violated.

In this paper we formalize a set of interdependency types between non-functional actions realizing non-functional behavior. These interdependencies can either be specified explicitly or implicitly by taking action properties into account. This rich set of interdependencies can then be used to ease the task of action composition by validating compositions against interdependency constraints, proposing conflict resolution strategies, and by applying our guided composition procedure. This procedure proposes next valid modeling steps leading to conflict-free compositions.

**Keywords:** Feature Interaction, NFC Composition, Model-driven development, Web Services.

## 1 Introduction

Non-functional behavior — in contrast to functional behavior — does not provide consumable business functionality. The execution of non-functional behavior rather improves the quality of a software system by satisfying certain non-functional attributes. For example, an encryption algorithm is a behavior which can be executed in order to support confidentiality. Functional behavior is often encapsulated by the main language constructs of the underlying programming platform such as operations or classes in case of object-oriented languages whereas non-functional behavior is hard to modularize by these constructs. Hence, language extensions have been introduced such as aspect-oriented programming which offers aspects to complement the deficiency of modularizing crosscutting concerns.



In [15] and [16], we introduced a model-driven approach for composing non-functional concerns in web services. This approach supports web services (black box view) and composite web services (gray box view) as well as the specification and enforcement of non-functional concerns. It is based on a well-defined process with different phases: requirements specification, action definition, action composition, action-to-service mapping, action-to-middleware-service mapping, and generation of NFC enforcement code. In our approach, the non-functional behavior is represented by non-functional actions (NFAs), defined in the action definition phase, which are supporting certain non-functional attributes defined in the requirements specification phase. Non-functional behavior is often non-orthogonal, i.e., when there are multiple actions activated simultaneously, a certain execution order has to be respected (cf. Beauvois [1]). More generally, there are different types of interdependencies (aka. interactions as motivated by [14], [11], [4]) between NFAs, e.g., mutual exclusion, data dependencies, and ordering restrictions. These interdependencies are constraints upon the execution of behavior. Consequently, a set of NFAs should constitute a well-defined execution order which can be defined in the action composition phase. The action compositions can then be mapped to web services in the mapping phase and related to concrete middleware services implementing the runtime behavior of NFAs. This allows to generate code for enforcing the composition at runtime.

In our previous works, we have already defined the modeling framework for the black and gray box view and provided a runtime environment based on proxies to enforce the models at runtime. In this paper, we complement this work by adding tool support to the complex task of action composition. This composition is modeled by domain experts; however, knowledge from different non-functional domains is required in order to understand the impact of different NFAs. This knowledge should be captured by a reusable model and provided to the domain experts. The contributions of this paper are thus more specifically:

- Defining a formal interdependency model that helps to identify invalid composition definitions at design time.
- Enriching this model by discovery of cross-domain interdependencies through analysis of the data impact of NFAs.
- Using the interdependency model to provide support for composing NFAs by:
  - visualizing constraint violations in the composition,
  - suggesting conflict resolution strategies for violated constraints,
  - introducing a guided modeling procedure.

The structure of the paper is as follows. In Section 2, our interdependency model is introduced and additional properties are described which can be used to discover implicit interdependencies. Section 3 describes our solution for modeling conflict-free compositions of non-functional behavior. In Section 4, the implementation of the modeling tool is described. Furthermore, our approach is evaluated by instantiating it in the web services context. Section 5 analyzes related work and Section 6 concludes this paper.

## 2 Formalizing the Interdependency Model

### 2.1 Tasks and Actions

Let  $\mathcal{A}$  be a set of NFAs, and let  $\mathcal{T}$  be tasks each executing an NFA. We define  $executes \subseteq \mathcal{T} \times \mathcal{A}$  as the relation defining which task executes which action. In contrast to actions, tasks are part of a specific execution context (i.e., a process) and therefore have a well-defined execution order. This relation can be compared to the relation between BPMN [12] service tasks and the services called by these tasks. More details of the composition model can be found in Section 3.1.

### 2.2 Interdependencies

We define  $\mathcal{I} := mutex \cup requires \cup precedes$  as the set of interdependencies between actions and tasks. More specifically it is:

- $requires = \{(x_1, x_2) \in (\mathcal{A} \times \mathcal{A}) \cup (\mathcal{T} \times \mathcal{T}) \mid \text{Execution of } x_1 \text{ requires the execution of } x_2\}$
- $precedes = \{(x_1, x_2) \in (\mathcal{A} \times \mathcal{A}) \cup (\mathcal{T} \times \mathcal{T}) \mid \text{If } x_1 \text{ and } x_2 \text{ are both executed, } x_1 \text{ has to be executed before } x_2\}$
- $mutex = \{(x_1, x_2) \in (\mathcal{A} \times \mathcal{A}) \cup (\mathcal{T} \times \mathcal{T}) \mid \text{Execution of } x_1 \text{ excludes the execution of } x_2\}$

From a given set of interdependencies, further interdependencies can be inferred by symmetry and transitivity. *Mutex* is symmetric, i.e.,  $mutex(x_1, x_2) \rightarrow mutex(x_2, x_1)$ . *Requires* and *precedes* are both transitive, i.e.,  $precedes(x_1, x_2) \wedge precedes(x_2, x_3) \rightarrow precedes(x_1, x_3)$  (where  $x_1, x_2, x_3 \in \mathcal{A} \cup \mathcal{T}$ ). In addition to these interdependencies, there are also action properties that play an important role for the composition. These properties can be categorized into data-related properties and control-flow-related properties (not in the scope of this paper).

### 2.3 Data Dependencies

Let  $\mathcal{A}$  be a set of actions and  $\mathcal{D}$  be a set of data items (which can be of complex type) and  $a \in \mathcal{A}$  and  $d \in \mathcal{D}$ . Then,  $\mathcal{P} := \{read, add, remove, modify\}$  is the set of binary relations between actions and data which we call impact types (because they define the impact on data) with the following semantics:

- $read = \{(a, d) \in \mathcal{A} \times \mathcal{D} \mid a \text{ reads data item } d\}$
- $add = \{(a, d) \in \mathcal{A} \times \mathcal{D} \mid a \text{ adds data to data item } d\}$
- $remove = \{(a, d) \in \mathcal{A} \times \mathcal{D} \mid a \text{ removes data item } d\}$
- $modify = \{(a, d) \in \mathcal{A} \times \mathcal{D} \mid a \text{ modifies (and reads) data item } d\}$

Let  $a$  and  $b$  be actions accessing data item  $d$  and let  $a$  be executed directly before  $b$ , i.e., there is no other action  $c \neq a, b$  accessing  $d$  executed between  $a$  and  $b$ . Then, there are 16 possible combinations of impact types to be analyzed. We found 10 combinations that cause or might cause conflicts w.r.t. their impact on data as shown in Table 1.

**Table 1.** Conflicting combinations of impact types:  $-$  conflict,  $(-)$  potential conflict,  $+$  no conflict,  $(+)$  warning, R = reverse order also in conflict. Subscripted numbers indicate the number of the enumeration item which explains the respective conflict.

	read(b,d)	add(b,d)	remove(b,d)	modify(b,d)
read(a,d)	+	- <b>2</b>	+	+
add(a,d)	+	- <b>6</b> <sub>R</sub>	(+ <b>5</b> )	+
remove(a,d)	- <b>11</b>	+	- <b>7</b> <sub>R</sub>	- <b>9</b> <sub>R</sub>
modify(a,d)	(- <b>4</b> )	- <b>3</b>	(+ <b>8</b> <sub>R</sub> )	(- <b>10</b> <sub>R</sub> )

1.  $remove(a, d) \wedge read(b, d)$  Data  $d$  is removed by  $a$  before  $b$  is able to read it.
2.  $read(a, d) \wedge add(b, d)$  Data  $d$  is read by  $a$  before  $b$  adds it. Either  $d$  exists before execution of  $a$  which would lead to duplicated data by the execution of  $b$ , or  $d$  has to be added by  $b$  because it does not exist, so  $a$  would read non-existing data.
3.  $modify(a, d) \wedge add(b, d)$  Data  $d$  is modified by  $a$  before  $b$  adds it. This combination is similar to **2**, because  $\forall a \in \mathcal{A}. modify(a, d) \rightarrow read(a, d)$ .
4.  $modify(a, d) \wedge read(b, d)$  Data  $d$  is modified by  $a$  before  $b$  can read it. This might be a potential conflict because  $a$  modifies the data which causes a state change from  $d_1$  to  $d_2$ . It depends which state  $b$  expects. If  $b$  expects  $d$  to be in state  $d_1$ , this combination is a conflict.
5.  $add(a, d) \wedge remove(b, d)$  Data  $d$  is added by  $a$  and then removed by  $b$ . Removing data directly after adding it makes no sense, but does not cause problems at runtime.
6.  $add(a, d) \wedge add(b, d)$  Data  $d$  is added by action  $a$  and  $b$  and hence duplicated.
7.  $remove(a, d) \wedge remove(b, d)$  Data  $d$  is removed by action  $a$  and  $b$ . After execution of  $a$ , data  $d$  does not exist anymore; hence  $b$  cannot remove it.
8.  $modify(a, d) \wedge remove(b, d)$  Data  $d$  is modified by  $a$  and then removed by  $b$ . Removing data directly after modifying it makes no sense, but does not cause problems at runtime.
9.  $remove(a, d) \wedge modify(b, d)$  Data  $d$  is removed by action  $a$  and then modified by  $b$ . This is similar to **11** because  $\forall a \in \mathcal{A}. modify(a, d) \rightarrow read(a, d)$ .
10.  $modify(a, d) \wedge modify(b, d)$  Data  $d$  is modified by  $a$  and then modified again by  $b$ . As modify also reads data, this is similar to **4**.

We will discuss strategies for resolving these conflicts in the following. Since the strategies can only be applied when the execution order of actions is already known, we assume tasks  $x$  and  $y$  executing conflicting actions  $a$  and  $b$  which access the same data item  $d$  such that  $x$  is executed before  $y$  and there is no task  $z$  between them which also accesses  $d$ . The first five conflict situations are resolvable by inverting the execution order of tasks  $x$  and  $y$  because the reverse order causes no data conflicts as can be seen in Table **1**. For those combinations, we add the *precedes* interdependency to the set of interdependencies, i.e., *precedes*( $y, x$ ) in this case. All other combinations cannot be resolved by reordering because the inverse order might also cause conflicts. Those conflicts can be resolved either by removing one of the tasks or by executing them exclusively, i.e., by executing

either  $x$  or  $y$  depending on some condition. For all those combinations, we add *mutex* to the set of the existing interdependencies, i.e.,  $mutex(x, y)$  in this case.

### 3 Towards Conflict-Free Action Compositions

#### 3.1 The Composition Model

For the composition of actions, we use a subset of BPMN2 [12]. A model in BPMN2 is a set of nodes and transitions between them. We define our simplified process model as follows. A non-functional activity is a directed graph containing all process elements. We define *activity*  $:= (\mathcal{N}, \mathcal{E})$  with the following semantics:

$$\begin{aligned}
 \mathcal{N} &:= \{x \mid node(x)\} \equiv \{x \mid x \text{ is process node}\} \\
 \mathcal{E} &:= \{(x, y) \mid transition(x, y)\} \\
 &\equiv \{(x, y) \in \mathcal{N} \times \mathcal{N} \mid \text{there is a transition from } x \text{ to } y\} \\
 start &:= \{x \mid x \text{ is the start node of the process}\} \\
 end &:= \{x \mid x \text{ is an end node of the process}\} \\
 \mathcal{T} &:= \{x \mid task(x)\} \equiv \{x \mid x \text{ is task node}\} \subset \mathcal{N} \\
 \mathcal{G} &:= \{x \mid gateway(x)\} \equiv \{x \mid x \text{ is gateway node}\} \subset \mathcal{N} \\
 \mathcal{XOR} &:= \{x \mid gw\_xor(x)\} \equiv \{x \mid x \text{ is xor gateway}\} \subseteq \mathcal{G} \\
 \mathcal{OR} &:= \{x \mid gw\_or(x)\} \equiv \{x \mid x \text{ is or gateway}\} \subseteq \mathcal{G} \\
 \mathcal{AND} &:= \{x \mid gw\_and(x)\} \equiv \{x \mid x \text{ is and gateway}\} \subseteq \mathcal{G} \\
 \mathcal{M} &:= (\mathcal{T}, \mathcal{XOR}, \mathcal{OR}, \mathcal{AND}, start, end)
 \end{aligned}$$

$\mathcal{M}$  is a tuple of sets  $M_0, M_1, \dots$  and each node  $n$  is exactly in one of its set elements:  $(\forall i, j < |\mathcal{M}|) M_i \cap M_j = \emptyset$  for  $i \neq j$ , and  $(\forall n \in \mathcal{N})(\exists i) n \in M_i$ . Moreover, there is exactly one start node:  $|start| = 1$ .

#### 3.2 Identifying Constraint Violations

To identify violations of the given interdependency constraints, we have to check a non-functional activity against each individual interdependency. For each interdependency  $i = (a, b) \in \mathcal{I}$ , the occurrence and order of actions (or tasks)  $a$  and  $b$  in the same execution path can lead to constraint violations depending on the given type. Hence, all possible execution paths through the process graph have to be analyzed. The number of those paths depends on the control flow of the process, more specifically on the number of OR and XOR gateways and the number of outgoing sequence flows per gateway. Presuming all gateways are used in sequence and  $out : \mathcal{G} \rightarrow \mathbb{N}$  is a function that calculates the number of outgoing sequence flows for each gateway, the number of possible paths for a given number of XOR and OR gateways, respectively, in the worst case is:

$$path_{s_{xor}} = \prod_{x \in \mathcal{XOR}} (out(x)) \qquad path_{s_{or}} = \prod_{x \in \mathcal{OR}} (2^{out(x)})$$

Obviously, the number of possible paths for *OR* is much higher than that of *XOR*. Let  $k$  be the number of all outgoing sequence flows from *OR* gateways and *traverse* be a function which traverses all possible paths, then the complexity of this function can be estimated using the  $\mathcal{O}$ -Notation:  $traverse \in \mathcal{O}(2^k)$  resulting in exponential runtime complexity. Since there are already existing methods for searching defined spaces for possible solutions, we decided to leverage those solutions. We make use of the declarative Prolog language because it provides very efficient ways to do depth-first search with backtracking.

### 3.3 Using Prolog to Find Violations and Counter Examples

In order to use Prolog for constraint checking, we have to import our BPMN models into Prolog. This can be achieved by transforming activities into a fact data base which contains a set of predicates. Hence, the following predicates have been defined:  $start(x)$ ,  $end(x)$ ,  $task(x)$ ,  $gw\_xor(x)$ ,  $gw\_or(x)$ ,  $gw\_and(x)$ ,  $node(x)$ ,  $transition(x,y)$ , and  $executes(x, a)$ . The last predicate defines that the task node  $x$  of the process executes action  $a$ . We distinguish between tasks and actions in order to cope with processes in which different task nodes execute the same action. The constraints given by the interdependency model form the rules that should apply to the fact base. However, the challenge for defining these rules was that the violation of some rule should not result in a simple boolean true or false decision, but should also provide some counter examples to give the modeler of non-functional activities constructive feedback.

With its backtracking concepts, Prolog allows for obtaining all values for which a certain predicate evaluates to true. Therefore, we defined a predicate with parameters  $A$ ,  $B$ ,  $X$ ,  $Y$ , and  $P$  for each interdependency type so that the predicates are true if and only if  $P$  is a counter example for the respective interdependency regarding the actions  $A$  and  $B$  which are executed in nodes  $X$  and  $Y$ , respectively. Within a query, Prolog distinguishes between constants and variables: If a variable is used for a certain parameter of a predicate, Prolog will search for values of this variable fulfilling the predicate whereas constant parameters restrict the search space. We may assume to have a constant list of interdependencies between actions and tasks. In case of an action interdependency, we can just apply the appropriate predicate for the respective interdependency type to constants for  $A$  and  $B$  and variables  $X$ ,  $Y$ , and  $P$  for Prolog to yield possible counter examples as solutions for  $X$ ,  $Y$ , and  $P$ . In case of a task interdependency, the procedure is analogous, but then we use the task constants for  $X$  and  $Y$ .

Regarding the structural representation of counter examples, we introduced the following concepts of paths in BPMN processes: A **Plain Graph Path** (PGP) from  $X$  to  $Y$  is a simple path from  $X$  to  $Y$  in the BPMN process graph as known from graph theory of directed graphs. It is represented as the list of nodes contained in the path. A **Block Path** (BP) is a PGP where nodes between opposite gateways are left out. BPs can only exist between two nodes if they have the same parent node. The term *parent node* refers to a tree representation of the BPMN process nodes in which the parent node of each node is the gateway

in which it is contained, or, if it is not contained in any gateway, an imaginary root node. For each pair of nodes  $X$ ,  $Y$  with the same parent node, there is exactly one BP between these nodes if a PGP from  $X$  to  $Y$  exists. A **Block Execution Path** (BEP) is a BP where each gateway node is replaced with a pair  $(X, P)$ .  $X$  is the replaced gateway node itself, and  $P$  is a list of BEPs that are executed in parallel starting from the gateway  $X$ . BEPs respect gateway semantics, e.g., the number of paths starting from an XOR gateway is always 1. A BEP is therefore an appropriate representation of a concrete execution of the BPMN process. Particularly, BEPs are used to represent counter examples in the aforementioned rules. A BEP is called complete if it begins with a start node and ends with an end node. Specifically, we defined the following rules for the interdependency types, each starting with *ce* as an abbreviation for *counter example*. Lets assume that  $P$  is a complete BEP, then it is:

- $ce\_conflicts(A, B, X, Y, P)$  is true if  $P$  contains both  $X$  and  $Y$  which in turn execute the actions  $A$  and  $B$ , respectively.
- $ce\_precedes(A, B, X, Y, P)$  is true if both action  $A$  and action  $B$  are executed in  $P$ , but task  $Y$  which executes  $B$  is in that path not guaranteed to be preceded by another task which executes  $A$ .  $X$  is just any task executing  $A$  in this path. Intuitively, this predicate is true if action  $A$  is not guaranteed to be executed before  $B$ . This is the case if  $B$  appears sequentially before the first  $A$ , or if  $A$  and  $B$  are executed in parallel paths of the same gateway.
- $ce\_requires(A, B, X, Y, P)$  is true if  $A$  is executed by  $X$  in  $P$ , but there is no task executing  $B$ . By convention, we set  $Y$  to 0.

Each of the predicates can be used to obtain counter examples by defining constants for  $A$  and  $B$  and using variables for  $X$ ,  $Y$ , and  $P$  for which Prolog will try to find instances which make the predicate true. For that purpose, Prolog iterates over all complete BEPs and tasks  $X$ ,  $Y$  executing  $A$ ,  $B$  and returns the first combination fulfilling the respective predicate. If no counter example exists, no solution will be found.

### 3.4 Conflict Resolution

After identifying interdependency violations in a non-functional activity, strategies for solving these conflicts should be defined. Let us assume that  $(a, b) \in \mathcal{I}$  is a violated task or action interdependency. Table 2 shows the different strategy classes. The difference between *rearrange* and *move* is that with *rearrange* an action will not move from one execution branch to another. As can be seen in the table, all resolution strategies might impose new conflicts. To avoid these undesired side effects we analyzed under which conditions a certain strategy can be applied safely. Due to space limitations we only give a short example for the *remove action* strategy. Before *remove action* can be applied safely to action  $a$ , for instance, we just have to check if there is a *requires* interdependency from any other action to  $a$ . In general, we distinguish both safe and potentially unsafe strategies.

**Table 2.** Resolution strategies: Interdependency conflicts solved and introduced

Resolution Strategy	Solves	Might Introduce
Remove Action	Mutex, Prec	Req
Insert Action	Req, Prec	Prec, Mutex
Rearrange Action	Prec	Prec
Move Action	All	All
Transform Gateway	All	All

### 3.5 Conflict-Free Composition Procedure

As we have seen in the previous subsection, it is complex to provide a validation mechanism with automatic conflict resolution. Mostly, some kind of human intervention is required at some point. It is even harder to propose a complete conflict-free activity because of the possibly small sets of given interdependencies. In order to combine the power of our validation approach with the ability of human non-functional domain experts to compose activities, we propose to use a guided modeling procedure. The idea is that a composition tool can be used to model a start event and the tool then proposes the next valid steps leading always to correct processes w.r.t. interdependency constraints. For that purpose, we had to extend our Prolog implementation in the following way: It should take a predefined set of candidate actions and the BPMN node from where to insert the next action as input. The output should be a list of valid actions which, when inserted at this point, would not cause any interdependency violation.

The concrete process for obtaining a list of valid actions  $A$  to be proposed for insertion at a certain position consists of the following steps: (1) Virtually extend the current (incomplete) BPMN process by adding a placeholder task  $x$  at the position where the user wants to insert a new element. Also, for each node of the process without an outgoing edge, add an edge to the end event which is newly created if necessary. This allows to have a complete BPMN process enclosed by a start and an end event which our Prolog program is able to process. (2) Send a query to Prolog to obtain all actions  $I$  which would violate a constraint if they were executed by  $x$ . This query is based on the Prolog model of the BPMN process such as used during validation and, additionally, on the Prolog model of all interdependencies relevant for the BPMN process. These are expressed in terms of a list of Prolog facts based on Prolog predicates *precedes*, *requires*, and *conflicts*, each having two parameters defining the actions or tasks between which the respective interdependency exists. The query also contains the list of candidate actions  $C$  to be tested at the position of  $x$ . (3) Our Prolog program then consecutively assumes  $x$  to execute each of the candidate actions and returns the list  $I$  of them for which at least one of the defined interdependencies is violated. (4) The actions  $A := C \setminus I$  are proposed to the user for insertion at the specified position. As, by definition of the proposed actions  $A$ , the obtained process after insertion of one of them would never result in a new constraint violation, we state that this composition technique considerably facilitates the definition of conflict-free compositions.

## 4 Running Example and Implementation

In [15] and [16], we introduced a new methodology for composing non-functional concerns in web services. This methodology proposes an engineering process with different phases. These phases are in general (with slight modifications) applicable to all kinds of software systems since they do not assume any concrete technology. However, the mapping phase is web-service-specific because actions and activities are mapped to concrete web services. In the following, we will use web services as a concrete technology for applying our abstract approach in order to show the instantiation of the interdependency model and to prove the applicability of the conflict detection and our solution strategy. Thereby, we will adhere to a part of the modeling procedure from our previous works as presented in the introduction, namely requirements definition, action definition, and action composition. For each of these phases — and also for the action-to-service mapping which is not in the focus of this paper — a separate editor has been implemented within the Eclipse IDE using the Graphiti framework<sup>1</sup>.

Let us assume an enterprise which has transformed its IT assets into a set of commercial web services. We further assume these web services have been implemented in different programming languages, e.g., due to constraints introduced by some legacy systems. During the development of the web service, non-functional concerns have been ignored on purpose: they should be strictly separated from the business functionality of the services. Hence, depending on the respective features the service provides, different non-functional requirements have been identified. For example, since the services are commercial, authentication and authorization are required to restrict the access to the services only to registered customers. To bill the customers based on the service usage, an accounting mechanism is required as well as support for non-repudiation and integrity of the messages. Furthermore, the company decided to log messages in the early introduction phase and to monitor the response time of their services. Another requirement is that the response time should be as low as possible.

The security, billing/accounting, performance, monitoring/logging, and general web service experts of the company transform the requirements into non-functional actions capable of fulfilling the requirements. The security expert knows how to support the security requirements and defines the following actions using our action editor: *Authenticate* (for authenticity), *Authorize*, and *VerifySignature* (for non-repudiation and integrity). Having defined those actions, the security expert identifies possible interdependencies between them. He defines that *Authenticate* has to precede *Authorize* and that *Authorize* requires *Authenticate*. Furthermore, he uses XPath [5] expressions to describe the data items of the SOAP message which the actions have an impact on, e.g., *Authenticate* will read the *UsernameToken* which is part of the *Security* XML tag of the SOAP message header. A summary of all actions which have been defined by all experts can be found in Table 3. Table 4 (Column 1) summarizes all interdependencies that have been discovered by the experts. In the example, one

<sup>1</sup> <http://eclipse.org/>, <http://www.eclipse.org/graphiti/>



**Table 3.** Actions and their impact

Action	Expert	Impact (XPath)
<b>Authenticate</b>	Security	Read(/Header/Security/UsernameToken)
<b>Authorize</b>	Security	Read(/Header/Security/UsernameToken)
<b>VerifySignature</b>	Security	Read(/Header/Security/BinarySecurityToken, /Security/Signature)
<b>RemSecHeaders</b>	Security	Remove(/Header/Security)
<b>Log</b>	Log/Mon.	Read(/Message/**)
<b>StartTimer</b>	Log/Mon.	None
<b>StopTimer</b>	Log/Mon.	None
<b>ReadFromCache</b>	Perform.	Read(/Body/**)
<b>RemoteAccounting</b>	Acc./Bill.	Read(/Body/**)
<b>LocalAccounting</b>	Acc./Bill.	Read(/Body/**)
<b>RemAllHeaders</b>	General	Remove(/Header/**)

can see that most of the interdependencies are only discovered between actions defined by the same expert. Cross-concern interdependencies are only defined by the accounting expert who knows that due to legal issues he has to prove that a service invocation has really been caused by a certain customer. Cross-concern interdependencies are generally hard to identify because the experts have to understand and analyze all actions of all non-functional domains.

**Table 4.** Explicitly defined & implicit interdependencies

Explicit Interdependencies	Implicit Interdependencies
precedes(Authenticate, Authorize)	precedes(Authenticate, RemAllHeaders)
requires(Authorize, Authenticate)	precedes(Authorize, RemAllHeaders)
precedes(StartTimer, StopTimer)	precedes(VerifySignature, RemAllHeaders)
requires(StopTimer, StartTimer)	precedes(Log, RemAllHeaders)
requires(LocalAccounting, VerifySignature)	precedes(Log, RemSecHeaders)
requires(RemoteAccounting, VerifySignature)	mutex(RemSecHeaders, RemAllHeaders)
mutex(RemoteAccounting, LocalAccounting)	
precedes(Authenticate, RemSecHeaders)	
precedes(Authorize, RemSecHeaders)	
precedes(VerifySignature, RemSecHeaders)	

In the next modeling phase, the non-functional activity is created with the composition editor (shown in Figure 11) by importing the action definition and dragging the available actions from the palette into the activity. An action is executed by a special BPMN task (the arrow symbol) and additional gateways and sequence flow elements can be used to define the control flow.

When a concrete execution order at the task level is given, the previously defined interdependencies can be enriched by additional task interdependencies derived from the data dependencies: Possible data conflicts are identified by an

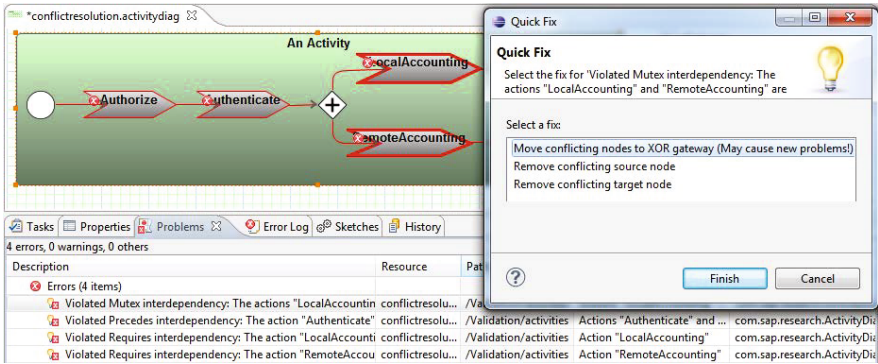


Fig. 1. Composition Editor: Conflict detection and resolution

intersection of the XPath expressions defining the data items that are affected by an action (shown in Table 3). If there is at least one node that both expressions have in common, the impact types are compared to each other. If, for instance, in the given process a task executing the *RemAllHeaders* action precedes another task executing an action accessing parts of the message header, there is a remove-read conflict between the two tasks. This data conflict can be resolved by introducing a precedes constraint upon these tasks. Another data conflict can be found when looking at the *RemAllHeaders* and the *RemSecurityHeaders* actions. The latter removes a subset of the data that *RemAllHeaders* removes. This is a remove-remove conflict which can be solved by introducing a mutex interdependency between the tasks executing those actions. The inferable interdependencies have been collected in Table 4 (Column 2).

Validation for a completely modeled action composition can be started by pushing the *Validate* button in the composition editor. Internally, the process and interdependency data, which is saved as an Ecore model, is transformed into Prolog facts and processed by our Prolog program. A list of all problems is shown in the problems view: a violation of *precedes* between *Authorize* and *Authenticate*, a violation of *mutex* between the accounting actions, and two *requires* violations due to the lack of *VerifySignature*. The selected problem is highlighted (see Figure 1). Moreover, so-called quick fixes are available via the context menu of each problem. In our example, the modeler can for example remove one of the tasks that are executing mutual exclusive actions or introduce an XOR gateway.

In the approach presented above the modeler gets feedback only when he triggers the validation. However, it is usually better to avoid those mistakes already during the modeling process. This is supported by our guided modeling procedure. Using this procedure, the user starts modeling and a context pad shows all available actions he can add next as shown in Figure 2. In the Graphiti-based context pad, the next valid actions are shown, e.g., after choosing the *LocalAccounting* action, the *RemoteAccounting* action is not available anymore except in another branch of an XOR gateway.

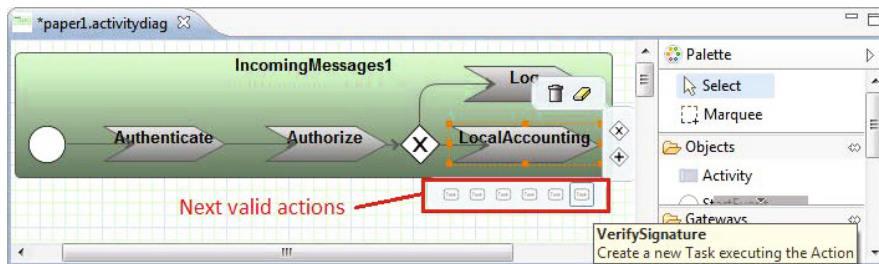


Fig. 2. Composition Editor: Guided composition proposes next valid actions

## 5 Related Work

### 5.1 General Approaches

The feature interaction problem, originally from the telecommunications domain, can also be found in object-oriented and component-oriented systems. For example, Pulvermüller et al. [13] define features as observable behaviors that can be of functional or non-functional nature. They distinguish unintended and intended feature interactions and interactions with positive and with negative effect. The feature interaction problem is similar to that of interdependencies between NFAs and also defines similar types of interactions.

Beauvois [1] defines composition constraints for *interweaving execution sequences* when composing non-orthogonal concerns. The author generates a scheduler from these constraints which is responsible for capturing all valid execution sequences in a dynamic way. The functional behavior as well as the composition constraints are described using an extension of hierarchical finite state machines. Beauvois describes constraints in form of state machines by describing the valid execution sequences explicitly whereas we use purely declarative constraints which fosters reuse and flexibility.

Sanen et al. [14] provide a conceptual model for concern interactions. They define a set of interaction types which we extended for our approach. Furthermore, they developed a prototype that automatically generates rules out of this model in order to improve concern interaction understanding. Those rules are used to build an expert system to support software engineers during development. The input to that expert system is a component composition specification and the output is a list of interactions and solution tactics. The authors do not further specify how this interaction list is then processed by a tool or which solution tactics are generated. They also do not provide interaction types to specify execution ordering (e.g., there is no precedes interaction).

In the context of data dependencies, various works exist in the area of concurrent data management. In one of the earliest, Bernstein [2] gives hints on when the order of instructions modifying or reading data matters: Three conditions are identified, namely flow dependence, anti-dependence (mirrored flow dependence),

and output dependence, which have all influenced our data dependency considerations of Table 1. Bernstein et al. [3] have shaped terms related to database management systems such as serializability and recoverability. However, this and related works are mainly about interleaving multiple concurrent transactions in a serializable manner at runtime whereas in our approach, we aim at ordering data-accessing actions at design time.

## 5.2 AOP Approaches

Shaker and Peters [17] use UML class and statechart diagrams for modeling core concerns (functional) and aspects (non-functional). In addition, they provide a statechart weaving language for weaving the functional and non-functional behavior. The authors present a static analysis of their design model which produces a list of potential core-aspect and aspect-aspect interactions. The verification is done on the woven model. The authors focus more on the static analysis and verification of their design model whereas in our approach we also provide resolution strategies and the guided-modeling procedure.

Nagy et al. [11] analyze problems and requirements for the composition of multiple aspects that match at the same join point. Furthermore, they propose a general declarative model for defining constraints upon possible aspect compositions and show how this model can be applied to AspectJ [10] and Compose\* [6]. The main requirements they identified are that it should be possible to specify the execution order of aspects and to define conditional dependencies. There are three types of constraints, namely the *pre* (an aspect has to be executed before another), the *cond* (an aspect is executed depending on the outcome of another aspect), and the *skip* constraint (an aspect execution might be skipped). The constraints are then used to generate a set of concrete valid execution orders. The problem with this strategy is that the set of valid orders strongly depends on the quality and quantity of interdependencies. The lower the quantity of interdependency information, the higher the number of possible execution orders being generated. To overcome this complexity, our approach is rather supportive and still involves human interaction. This is comparable to business process modeling which is also done by human experts. However, in our approach the interdependencies can also be enriched by the use of data impact properties.

Katz [9] describes different categories (impact types) of aspects in terms of semantic transformations of state graphs of the base systems: spectative (read-only), regulative, and invasive (modify) ones. He defines syntactical identification procedures to determine the category of an aspect. The main goal of introducing categories is to simplify proofs of correctness, e.g., w.r.t. liveness and safety properties. Our approach also aims for the correctness, but mainly on the correctness of the composition of aspects (NFAs) and not on how aspects influence the correctness of the base system. Furthermore, in our approach, the category is determined manually as all aspects are considered black boxes whereas Katz uses static analyses to automatically determine the category of an aspect.

Durr et al. [7,8] abstract the behavior of advices into a resource operation model which presents common or shared interactions. This model is complemented by a conflict model including data and control flow conflicts. The impact of advices on data is classified as read, nondestructive write (add), destructive write (modify), and unknown. They also identify conflicting situations with pairs of those impact types. In their analyzing process, a message flow graph is generated which represents all possible paths through the filter set (their approach is based on Composition Filters). This graph is then used to find the conflicting paths. In our approach, we also use data or control conflicts (impact types), but we use this information to enrich our interdependency model which is on a higher level than the concrete execution paths. Moreover, the constraints on the composition implied by the interdependencies are not only used for validation purposes but also for conflict resolution and guided composition. Another difference is that Durr et al. do static code analysis to determine impact types of advices whereas in our approach this is done manually.

## 6 Conclusions

In this paper, we presented our interdependency model which can be used to discover conflicts in compositions and non-functional behavior already at design time in order to avoid conflicts at runtime. Additionally, action properties have been introduced. The benefit of these properties is that they allow — in contrast to interdependencies — to regard one action in isolation. This helps to discover additional interdependencies even across different non-functional domains. Our rich interdependency model also enables conflict resolution and supports a guided, conflict-free composition procedure.

The presented concepts have been evaluated in the context of web services. A realistic web service example has been introduced in order to show the applicability and feasibility of our approach. Moreover, a set of graphical Eclipse-based editors has been implemented in order to support modelers during the complex task of action composition. Finally, related work has been identified showing that most approaches rely on completely automated processes, do not involve human actors, and thus strongly depend on the quality of the available interdependency information. These drawbacks are addressed in our approach by firstly obtaining a rich interdependency model with help of data dependencies and secondly by introducing our guided composition procedure.

However, our approach also has a few limitations. First, loops in the functional or non-functional composition logic are not yet supported. Second, interdependencies between functional and non-functional actions are currently out of scope. Third, conditions for process branching are not processed by our validation algorithm. These issues need to be investigated in the future.

**Acknowledgements.** This work has been partially supported by the German Federal Ministry of Education and Research (BMBF) in the project InDiNet (01IC10S04A).

## References

1. Beauvois, M.: Brenda: Towards a Composition Framework for Non-orthogonal Non-functional Properties. In: Distributed Applications and Interoperable Systems (DAIS 2003), Paris, France, pp. 29–40 (November 2003)
2. Bernstein, A.J.: Analysis of programs for parallel processing. *IEEE Transactions on Electronic Computers* EC-15(5), 757–763 (1966)
3. Bernstein, P.A., Hadzilacos, V., Goodman, N.: Concurrency control and recovery in database systems. Addison-Wesley Longman Publishing Co., Inc., Boston (1987)
4. Bowen, T.F., Dworack, F.S., Chow, C.H., Griffeth, N., Herman, G.E., Lin, Y.-J.: The feature interaction problem in telecommunications systems. In: Seventh International Conference on Software Engineering for Telecommunication Switching Systems, SETSS 1989, pp. 59–62 (July 1989)
5. Clark, J., DeRose, S.: XML Path Language (XPath) Version 1.0 (November 1999)
6. de Roo, A.J., Hendriks, M.F.H., Havinga, W.K., Durr, P.E.A., Bergmans, L.M.J.: Compose\*: a language- and platform-independent aspect compiler for composition filters. In: First International Workshop on Advanced Software Development Tools and Techniques, WASDeTT 2008, Paphos, Cyprus (July 2008)
7. Durr, P., Bergmans, L., Aksit, M.: Reasoning about semantic conflicts between aspects. In *EIWAS 2005: The 2nd European Interactive Workshop on Aspects in Software*, Brussels, Belgium, pp. 10–18 (September 2006)
8. Durr, P., Bergmans, L., Aksit, M.: Static and Dynamic Detection of Behavioral Conflicts Between Aspects. In: Sokolsky, O., Taşıran, S. (eds.) *RV 2007*. LNCS, vol. 4839, pp. 38–50. Springer, Heidelberg (2007)
9. Katz, S.: Aspect Categories and Classes of Temporal Properties. In: Rashid, A., Aksit, M. (eds.) *Transactions on Aspect-Oriented Software Development I*. LNCS, vol. 3880, pp. 106–134. Springer, Heidelberg (2006)
10. Kiczales, G., Lamping, J., Mendhekar, A., et al.: Aspect-oriented Programming. In: Aksit, M., Auletta, V. (eds.) *ECOOP 1997*. LNCS, vol. 1241, pp. 220–242. Springer, Heidelberg (1997)
11. Nagy, I., Bergmans, L., Aksit, M.: Composing Aspects at Shared Join Points. In: Hirschfeld, R., Kowalczyk, R., Polze, A., Weske, M. (eds.) *NetObjectDays (NODe/GSEM)*, Erfurt, Germany. LNI, vol. 69, pp. 19–38. GI (September 2005)
12. OMG. Business Process Model and Notation (BPMN) 2.0 (January 2011)
13. Pulvermüller, E., Speck, A., Coplien, J.O., D’Hondt, M., De Meuter, W.: Feature Interaction in Composed Systems. In: Frohner, A. (ed.) *ECOOP 2001*. LNCS, vol. 2323, pp. 86–97. Springer, Heidelberg (2002)
14. Sanen, F., Truyen, E., Joosen, W.: Managing Concern Interactions in Middleware. In: Indulska, J., Raymond, K. (eds.) *DAIS 2007*. LNCS, vol. 4531, pp. 267–283. Springer, Heidelberg (2007)
15. Schmeling, B., Charfi, A., Mezini, M.: Composing Non-Functional Concerns in Composite Web Services. In: *IEEE International Conference on Web Services (ICWS 2011)*, Washington DC, USA, pp. 331–338 (July 2011)
16. Schmeling, B., Charfi, A., Thome, R., Mezini, M.: Composing Non-Functional Concerns in Web Services. In: *The 9th European Conference on Web Services (ECOWS 2011)*, Lugano, Switzerland, pp. 73–80 (September 2011)
17. Shaker, P., Peters, D.K.: Design-level detection of interactions in aspect-oriented systems. In: *Proceedings of the 20th European Conference on Object-Oriented Programming*, Nantes, France, pp. 23–32 (July 2006)

# Fuzzy Verification of Service Value Networks

Iván S. Razo-Zapata<sup>1</sup>, Pieter De Leenheer<sup>1,2</sup>,  
Jaap Gordijn<sup>1</sup>, and Hans Akkermans<sup>1</sup>

<sup>1</sup> VU University, Amsterdam, The Netherlands

{i.s.razozapata,p.g.m.de.leenheer,j.gordijn,elly.lammers}@vu.nl

<sup>2</sup> Collibra nv/sa, Brussels, Belgium

**Abstract.** Service Value Networks (*SVNs*) represent a flexible design for service suppliers to offer attractive value propositions to final customers. Furthermore, networked services can satisfy more complex customer needs than single services acting on their own. Although, *SVNs* can cover complex needs, there is usually a mismatch between what the *SVNs* offer and what the customer needs. We present a framework to achieve *SVN* composition by means of the propose-critique-modify (PCM) problem-solving method and a Fuzzy Inference System (FIS). Whereas the PCM method composes alternative *SVNs* given some customer need, the FIS verifies the fitness of the composed *SVNs* for the given need. Our framework offers not only an interactive dialogue in which the customer can refine the composed *SVNs* but also visualizes the final composition, by making use of  $e^3$ -value models. Finally, the applicability of our approach is shown by means of a case study in the educational service sector.

**Keywords:** Service Value Networks, PCM, Composition, Verification, Fuzzy Logic.

## 1 Introduction

Commercial services do not only cover an important segment of countries' economies but also are thought to play a big role in the Future Internet [18,17]. Since networked services can cover more complex customer needs, to offer service-based solutions for customers in a changing environment like the Web, service providers must be able to interact with other business entities to create *service networks (SNs)* [18]. In general, since commercial services are economic activities offering tangible and intangible value resources, value aspects within these networks must be also analyzed [2]. More specifically, it is important to understand how customers and suppliers can co-create *service value networks (SVNs)*.

We have previously implemented part of our framework describing how *SVNs* can be (semi)automatically composed to meet specific (complex) customer requirements by means of networking offerings coming from different service suppliers [14,19,21]. In this paper, our contribution is twofold. First, we present a problem-solving method to allow composition of *SVNs* based on customers' and suppliers' perspectives. Customers try to find answers to their needs whereas

suppliers can be organized into a *SVN* to satisfy the customer needs. Moreover, in case none of the composed *SVN* offers a good answer to the customer, the framework allows the customer to give feedback for the composed *SVNs* and restart the composition process until (s)he finds a *SVN* that fits her/his need.

Second, we also describe a Fuzzy Inference System (FIS) to automatically verify how *good* the composed *SVNs* fit the given customer need. In this way, we can determine to what extent the *SVNs* satisfy the intended customer need. To this aim, for each *SVN*, we analyze the amount of provided, missing and non-required functionalities. Finally, based on their fitness, the *SVNs* are ranked so the customer can select the one better fitting her/his requirements.

This paper is organized as follows. A discussion about basic concepts, problem definition and related work is given in Sect. 2. Sect. 3 describes our composition framework. Finally, we present our conclusions and future work in Sect. 4.

## 2 Background

### 2.1 Service Value Networks

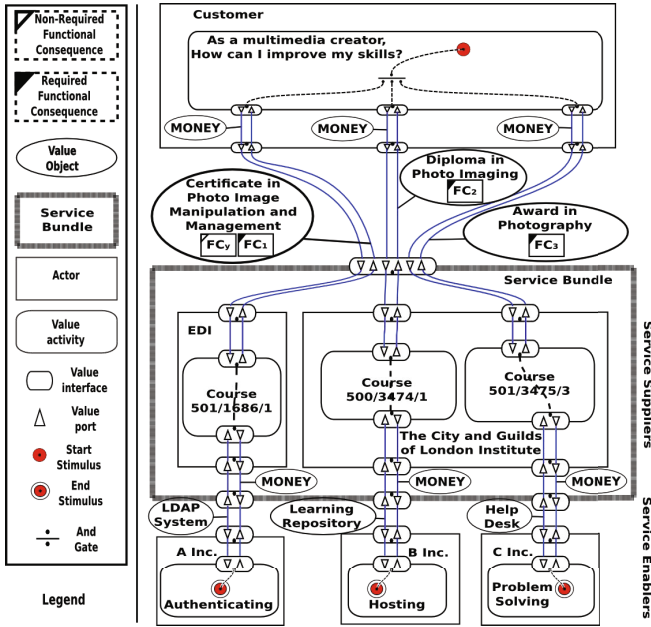
We consider a service as an economic activity that offers and requests valuable outcomes to and from its environment [12]. *E.g.* a certain colleague can offer a specific course or certificate in exchange of some tuition fee. *SVNs* are created when different customers and suppliers agree on exchanging valuable outcomes among each other. Based on Hamilton [15], Verna Allee [2], Lovelock and Witz [18], we define a *SVN* as follows:

**Definition 1.** *A Service Value Network (SVN) is a flexible and dynamic web of homogeneous enterprises and final customers who reciprocally establish relationships with other peers for delivering an added-value service to a final customer.*

The flexibility and the dynamicity within the *SVN* obey to the fact that service suppliers can freely (re)establish relationships with other peers, *i.e.* modifying the structure of the network depending on what a customer requires. By homogeneity we mean that service suppliers have a common business objective, *i.e.* offer a service solution to the final customer. Fig. 1 depicts the basic structure of a *SVN* [5]. Customers exchange valuable outcomes with a pool of suppliers that are arranged into a service bundle bringing about Business to Customer (B2C) relationships. The services within a service bundle can also exchange valuable outcomes with service enablers that support them by means of Business to Business (B2B) relationships [5,21].

The *SVN* in Fig. 1 has been (semi)automatically composed to deal with a specific customer need in the educational service sector [21]. The idea here is to compose *SVNs* for customers that need to improve their job profile by acquiring new skills through some qualifications, certificates or awards offered by educational services. The main motivation is that people looking for a job might have more success by having a better job profile. Furthermore, the skills can be considered as properties, functionalities or according to our concepts *Functional*





**Fig. 1.**  $e^3$ -value model of a SVN for a specific customer need requiring  $FC_1$  (Digital Image Manipulation),  $FC_2$  (Photo Image Capture) and  $FC_3$  (Studio Photography).  $FC_y$  (Presenting Photo Images) is a non-required skill (functionality) [5,21].

*Consequences (FCs)* that are offered by an educational service (course) [19,21]. For this case study we harvested the National Database of Accredited Qualifications<sup>1</sup> (NDAQ), which contains details of Recognised Awarding Organisations and Regulated Qualifications in England, Wales and Northern Ireland. The final result was a catalogue containing 58 services offered by four service suppliers.

At the top of Fig. 1 there is a customer that needs to improve her/his job profile by getting new skills related to multimedia creation. In this specific case the customer needs to acquire the following *FCs*: Digital Image Manipulation ( $FC_1$ ), Photo Image Capture ( $FC_2$ ) and Studio Photography ( $FC_3$ ). Furthermore, these *FCs* are packed into the indivisible valuable service outcomes (also known as value objects within the  $e^3$ -value theory and our framework [12,19,21]): *Certificate in Photo Image Manipulation and Management*, *Diploma in Photo Imaging* and *Award in Photography*. E.g. a course offered by a university is composed of a program containing different units that can only be acquired by following the complete course. i.e. a valuable service outcome is the smallest unit of economic exchange either its is money or a course.

Since the functional consequences  $FC_1$ ,  $FC_2$  and  $FC_3$  are not individually offered but contained into indivisible valuable service outcomes that are offered by different suppliers, the only way to obtain these *FCs* is by bundling the

<sup>1</sup> <http://www.accreditedqualifications.org.uk>

services from different suppliers [19]. The advantages of bundling services can be perceived by customers as well as by suppliers. First, from the customer point of view, service bundles are generally cheaper and more suitable for their needs. Second, from the point of view of suppliers, when they work together offering their outcomes, the chances of covering complex customer needs are higher. Third, they can also outsource some of their functionalities to other suppliers/enablers, which not only saves costs of (re)implementation but also allows to focus on and improve their own service outcomes, *i.e.* offering better services [19].

Finally, because service orientation also aims at outsourcing the non-core-business activities from a company, the services inside the bundle may rely on service enablers that perform those activities [18]. *E.g.* As depicted in Fig. 1, since the core business of EDI (Education Development International plc) is providing vocational qualifications, they can outsource services to deal with communication aspects such as a LDAP (Lightweight Directory Access Protocol) system. In the case of The City and Guilds of London Institute, to offer its services it requires the Learning Repository and the Help Desk services. Our framework solves these kind of B2B dependencies by looking for a service that can offer what the service supplier requires [14,21].

## 2.2 Problem Definition

Contrary to common trends in Service Science that address the service composition problem as a planning problem [10], (semi)automatic composition of *SVNs* can also be addressed from a design perspective. This is mostly due to the fact that business modelling is centred around the notion of value which requires describing *what* the participants exchange with each other rather than *how*, *i.e.* there is no need to deal with the ordering of exchanges. [11].

Nonetheless, to achieve (semi)automatic composition of *SVNs*, three issues must be addressed: 1) *Customer-Supplier Interaction (CSI)*: To facilitate the co-creation of *SVNs*, customers and suppliers must have the opportunity to completely express their needs and offerings respectively. Moreover, in case *SVNs* do not fit the customer needs, customers must be able of giving feedback to allow the composition of new *SVNs* that can better fulfill their needs. 2) *Tailored Composition (TC)*: The *SVNs* must be composed based on the functionalities required by a single customer or group of customers which requires alternative *SVNs* that must meet the customer needs. 3) *Automatic Verification (AV)*: Once *SVNs* have been composed, we must analyze the individual properties of each *SVN* to determine how well each *SVN* fits the customer need.

We have previously described a framework to (semi)automatically compose *SVNs* [14,19,21]. Since in such a framework we have presented answers for the *customer supplier interaction* and *tailored composition* issues, this paper offers an answer to the third issue, *i.e. automatic verification*. Consider, for instance, the *SVN* depicted in Fig. 1, that offers  $FC_1$ ,  $FC_2$  and  $FC_3$ . Although in this case the *SVN* offers the required *FCs*, sometimes the *SVNs* do not meet the set of required *FCs*. This problem arises because *FCs* are packed within the indivisible

valuable outcomes so the customers cannot get individual *FCs* but a full packet, consequently in some cases more *FCs* are offered than requested (resulting in non-required *FCs*), in other cases less *FCs* are offered than requested (resulting in missing *FCs*). *E.g.*, in Fig. 1 the *Certificate in Photo Image Manipulation and Management* offers  $FC_1$  and  $FC_y$  that is not required by the customer.

Furthermore, customers may also have different preferences for each *FC*. Consequently, what is required is: an automatic way to 1) *verify* how well the composed *SVNs* fit the customer needs, *i.e.* how good is the match between the requested functional consequences and the functional consequences offered by the composed *SVNs* and 2) *rank* the *SVNs* so the customer does not get lost with many alternatives. *E.g.*, just in UK, Ofqual has registered around 180 institutions with more than 10,000 services offering different types of qualifications<sup>2</sup>. This generates a very large solution space from which many alternative *SVNs* can be composed.

### 2.3 Related Work

An in-depth discussion about different service network approaches dealing with *customer supplier interaction* and *tailored composition* can be found at [20], nevertheless, we present a brief overview. There are two main trends about composition within Service Science: 1) Process orientation, and 2) Business orientation. Whereas process-oriented approaches focus on work-flow properties, *i.e.* how the activities must be performed (*e.g.* in which order), business-oriented approaches are centred around the notion of *value*, therefore it is relevant to determine who is offering what of *value* to whom and what expects of value in return, *i.e.* economic reciprocity [11].

Process-oriented approaches are covered by efforts in the field of web services, in which the composition problem is usually modelled as a planning problem and the final solution is given by a sequence of activities to be performed, *i.e.* an execution plan [10]. Some of them make use of meta models or templates [1], others offer more dynamic frameworks [24]. Nonetheless, there are also approaches such as SNN [8] that describes manual composition techniques. Contrary, business-oriented approaches usually rely either on goal or value modelling and the solution is given by a *value proposition* for the final customer [3,6,9]. Recently, Becker *et al.* [6] and De Kinderen [9] describe dialogued-based frameworks for service bundling based on predefined bundles. VNA [2] is the only approach that composes a complete *SVN* however it is also a manual-based approach. Consequently, although process-oriented approaches already offer dynamic frameworks to compose service networks, they lack the value aspects.

*Verification* in its purest form can be addressed from different perspectives. The scope covers domain-specific calculations or simulations, qualitative simulation and visual simulations [7]. De Kinderen [9] proposes a method called Rank-Order Centroid (ROC) to verify and rank service bundles, nevertheless this

<sup>2</sup> <http://www.ofqual.gov.uk/>

method only focuses on the *FCs* offered by a service bundle without estimating the impact of missing and non-required *FCs* which is required when *SVN* cannot fully cover a customer need.

### 3 The *e*<sup>3</sup>service Framework

Since the composition task must produce a clear specification about a network of services exchanging different valuable outcomes, methods for designing artifacts can be applied to solve this task [7]. More specifically, the so-called problem-solving methods address design issues from different perspectives such as constraint satisfaction, decomposition of problems, numerical optimization among others [7]. Nonetheless, since *customer supplier interaction* is one of the core issues to achieve *SVN* composition, we use the propose-critique-modify (PCM) method [7]. As described by Chandrasekaran, a PCM method combines four subtasks (for which different methods can also apply): propose, verify, critique and modify [7].

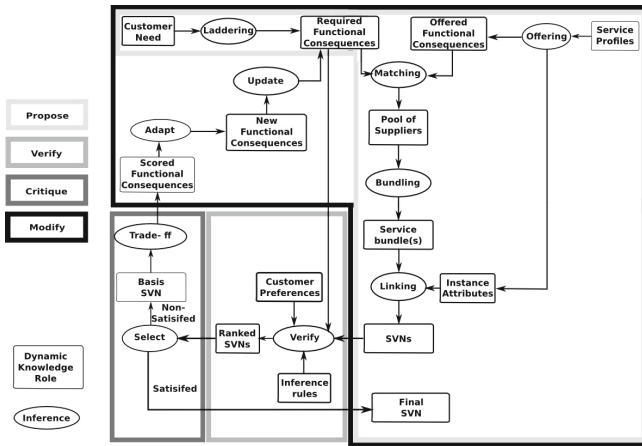


Fig. 2. The *e*<sup>3</sup>service Framework

Figure 2 depicts the main idea behind our PCM method. We use a CommonKADS inference model to describe not only the required knowledge but also the inferences that are needed to produce new knowledge as well as the interactions among them [22]. Inferences carry out reasoning processes whereas dynamic knowledge roles are the run-time inputs and outputs of inferences [22].

In previous work we have implemented part of this framework (mostly the propose task) where supply and demand of services are modelled by means of two ontologies through which suppliers express their offerings and customer express their needs [14,19,21]. Since supply and demand are always evolving, they are described as dynamic knowledge roles. As can be observed in Figure 2, due to the application of inferences, the rest of the knowledge components are also dynamically produced.

### 3.1 Propose

According to Chandrasekaran, given a design goal, the propose subtask generates a solution [7]. In our case, the goal is to compose a *SVN* to cover a given customer need. We have extensively explored and developed this subtask in previous work [14,19,21,3,9]. Therefore, we do not present a full description of each inference since this is not the core of the paper. Nevertheless, we briefly explain them as follows:

- **Laddering.** This concept has been widely used in marketing to represent how customers link specific product attributes to high-level values [9]. In our case, by making use of the customer ontology, customer needs as stated by the ontology are refined into so-called functional consequences *FCs*. For instance, a customer need such as “As a multimedia creator, How can I improve my skills?” can be refined into the following *FCs*: Digital Image Manipulation, Photo Image Capture, Studio Photography, Digital Animation and Audio Production among other functionalities that better describe a customer need in terms of specific requirements [9,19,21].
- **Offering.** By making use of the supplier ontology service providers can describe their offerings in terms of *FCs*, *i.e.* what functionalities they can offer to the customers [19,21]. In this way, service offerings can be retrieved to initiate the composition of *SVNs*.
- **Matching.** Because laddering maps customer needs onto *FCs* and service offerings are also described in terms of *FCs*, we can retrieve all the services that completely or partially offer the *FCs* as required by the customer [19,21].
- **Bundling.** Since the output of the matching steps is a pool of service suppliers, at this step we find meaningful combinations of services that can jointly offer an answer to the the final customer [19]. The output is a pool of service bundles that describe B2C relationships in terms of value exchanges.
- **Linking.** At this step we solve the B2B dependencies that service bundles might have [14,21]. For instance, within a bundle, an educational service offering a given course to the customer side might rely on a service such as a digital library that allows students to access reading material. Although the final customer only cares about the B2C with the service bundle, the educational service needs to solve its dependencies with other service enablers to allow the service bundle being sustainable.

### 3.2 Verify

Also described as *verification* by Chandrasekaran, verify refers to the process of checking whether the design satisfies functional and non-functional specifications [7]. In our framework, however, we only deal with functional requirements. Once the *SVNs* have been composed, the verification subtask determines not only if the generated *SVNs* offers the required *FCs* but also how they fit the customer preferences. For each *SVN*, three aspects must be analyzed: the *provided*, the *missing* and the *non-required FCs*.

In this way, heuristic reasoning will suggest that a good *SVN* must have *many* provided, *few* missing and *few* non-required *FCs*. Moreover, since customers may have different preferences for each *FC*, we must consider not only whether the *SVNs* offer the required *FCs* but also the importance of each *FCs*. To achieve this task we propose a Fuzzy Inference System (FIS) that can deal with two kind of issues 1) uncertainty: situations in which statements cannot be expressed as false or true but partially true or false, *e.g.* whether a *SVN* offers the *FCs* as requested by the customer, 2) computing with words: working with concepts that can be easily understood by human beings, *e.g.* by making use of simple rules we can indicate how the *SVNs* must be verified [26].

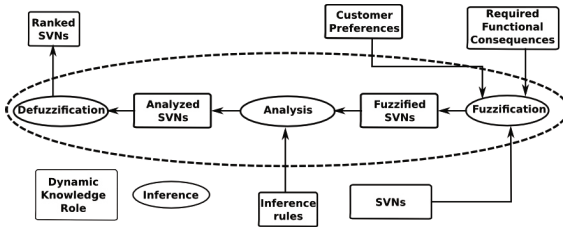


Fig. 3. Fuzzy Inference System (FIS)

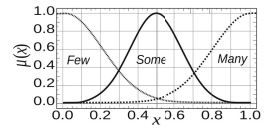


Fig. 4. Fuzzification Functions

As depicted in Fig. 3, our FIS takes as input the required *FCs*, customer preferences for each *FC*, the *FCs* offered by each *SVN* and a set of inference rules that allow to determine how good the *SVNs* are. The output is a pool of *SVNs* that are ranked based on their fitness. The next paragraphs describe each one of the inferences, *i.e.* Fuzzification, Analysis and Defuzzification.

**Fuzzification.** To determine the amount of provided, missing and non-required *FCs* for each *SVN*, we perform three steps: 1) Computing weights, 2) Computing Fractions, and 3) Fuzzification.

1) Computing weights: During the propose phase, customer needs are expressed in terms of *functional consequences (FCs)* that can be denoted as  $FC_{Customer}$ . The composed *SVNs* can also be described in terms of *FCs* highlighting three aspects: 1)  $FC_P$  the functional consequences that are *provided* as required by the customer, 2)  $FC_M$  the functional consequences that are not offered as required by the customer, *i.e.* *missing FCs*, and 3)  $FC_N$  the functional consequences that are offered by the *SVN* but are *not required* by the customer. Therefore, in terms of *FCs*, a *SVN* can be represented as follows:

$$FC_{SVN} = FC_P \cup FC_M \cup FC_N \tag{1}$$

There are two ways to express preferences for a given  $FC$ :  $w_c$  and  $w_a$  which are defined by:

$$w_c : fc \rightarrow [0, 1], \quad w_a : fc \rightarrow \{0.5\} \quad (2)$$

where  $w_c$  is the weight that a customer assigns to a  $FC$  and  $w_a$  is a predefined weight assigned to a given  $FC$ . We have chosen a 0.5 value assuming that non-required  $FC$ s have a moderate influence in the fitness of a  $SVN$ . Nonetheless, this value can be adapted to the composition context or even gathered through crowd-sourcing. Later on, we can compute the total weights (preferences) for  $FC_{Customer}$  and  $FC_{SVN}$  in the following way:

$$TW_C = \sum_{fc_i \in FC_{Customer}} w_c(fc_i), \quad TW_{SVN} = W_P + W_M + W_N \quad (3)$$

where  $TW_C$  is the total weight for the  $FC$ s as requested by the customer and  $TW_{SVN}$  is the total weight of all  $FC$ s in a  $SVN$ . The values for  $W_P$ ,  $W_M$  and  $W_N$  are computed as follows:

$$W_P = \sum_{fc_i \in FC_P} w_c(fc_i), \quad W_M = \sum_{fc_i \in FC_M} w_c(fc_i), \quad W_N = \sum_{fc_i \in FC_N} w_a(fc_i) \quad (4)$$

where  $W_P$  is the sum of the weights  $w_c(fc_i)$  for the functional consequences  $FC$ s that are *provided* as required by the customer in a  $SVN$ . Similarly,  $W_M$  is the sum of the weights for  $FC$ s that are *missing* in a  $SVN$ , and  $W_N$  is the sum of the weights for the  $FC$ s that are *non-required* in a  $SVN$ .

2) Computing Fractions: For each  $SVN$  we compute three fractions: 1)  $F_P$  the fraction of the *provided*  $FC$ s to the total  $FC$ s as requested by the customer. 2)  $F_M$  the fraction of the *missing*  $FC$ s to the total  $FC$ s as requested by the customer. 3)  $F_N$  the fraction of the *non-required*  $FC$ s to the total  $FC$ s in a  $SVN$ . These fractions are computed by Eq. 5:

$$F_P = \frac{W_P}{TW_C}, \quad F_M = \frac{W_M}{TW_C}, \quad F_N = \frac{W_N}{TW_{SVN}} \quad (5)$$

3) Fuzzification: At this step we determine the amount of *provided* ( $P$ ), *missing* ( $M$ ) and *non-required* ( $N$ ) functional consequences for each  $SVN$  in terms of three linguistic labels: *few*, *some* and *many*. To this aim, we compute the membership degree of the fractions  $F_P$ ,  $F_M$  and  $F_N$  to the functions in Fig. 4 (also known as Fuzzy Sets) [26]. The membership degree (also known as degree of truth) of a fraction in a fuzzy set allows to determine how *true* is the fact that the analyzed fraction is *few*, *some* or *many*. The degree of truth of a fraction is determined by the following distribution function [26]:

$$Fuzzification(x) = \mu(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\alpha \frac{(x-c)^2}{\sigma^2}} \quad (6)$$

where  $x = \{F_P, F_M, F_N\}$ ,  $\sigma^2$  is the variance,  $\alpha$  is a parameter that determines how width the fuzzy set is and  $c$  is the location of the fuzzy set, *i.e.* the highest

point of the fuzzy set. For all the fuzzy sets  $\alpha = 2.0$ ,  $\sigma^2 = 0.16$ . The values for  $c$  are 0.0, 0.5 and 1.0 for the fuzzy sets *few*, *some* and *many* respectively. The values for  $\alpha$ ,  $\sigma^2$ ,  $c$  were chosen to cover the fraction values  $F_P$ ,  $F_M$  and  $F_N$  within the range  $[0, 1]$  [26]. At the end, for each *SVN* we have different degrees of truth for  $P$ ,  $M$  and  $N$ . *E.g.* a *SVN* with a fraction value  $F_P = 0.8$  has the following degrees of truth for  $P$ : *few* (0.0), *some* (0.32) and *many* (0.60), which means that  $P$  simultaneously belongs to the fuzzy sets *some* and *many* but with different degrees of truth. *i.e.* it is not only true (with a 0.32 value) that the *SVN* provides *some FCs* but also true (with a 0.60 value) that the *SVN* provides *many FCs* as required by the customer.

**Analysis.** Once we have computed the degrees of truth of  $P$ ,  $M$  and  $M$  for each *SVN*, we can then analyze how good they are by making use of a set of inference rules that we designed following common sense criteria and depicted in Fig. 5. *E.g.*, a *SVN* with  $P = \textit{some}$ ,  $M = \textit{few}$  and  $N = \textit{few}$  is an *Average SVN*. Since in the fuzzification step  $P$ ,  $M$  and  $N$  may have different degrees of truth for each linguistic label (membership function), more than one rule can apply when analyzing a given *SVN*. *E.g.*, as explained before,  $P$  can simultaneously be *some* and *many*, consequently the rules in which  $P$  is not only *some* but also *many* have to be applied, the same holds for  $M$  and  $N$ .

1:	IF	P	is	many	AND	M	is	few	AND	NR	is	few	THEN	Perfect
2:	IF	P	is	many	AND	M	is	few	AND	NR	is	some	THEN	Good
3:	IF	P	is	many	AND	M	is	some	AND	NR	is	few	THEN	Good
4:	IF	P	is	many	AND	M	is	some	AND	NR	is	some	THEN	Good
5:	IF	P	is	some	AND	M	is	few	AND	NR	is	few	THEN	Average
6:	IF	P	is	some	AND	M	is	few	AND	NR	is	some	THEN	Average
7:	IF	P	is	some	AND	M	is	some	AND	NR	is	few	THEN	Poor
8:	IF	P	is	some	AND	M	is	some	AND	NR	is	some	THEN	Poor
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
N:	IF	P	is	few	AND	M	is	many	AND	NR	is	many	THEN	Bad

Fig. 5. Inference rules

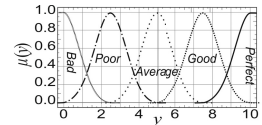


Fig. 6. Defuzzification Functions

To apply the inference rules another observation is important. Since the degrees of truth are expressed in values within the range  $[0, 1]$ , the traditional boolean AND operation does not work here, *i.e.* after applying the rule (1) as depicted in Fig. 5, IF  $P$  is *many* AND  $M$  is *few* AND  $N$  is *few* the value cannot be 0 or 1 but a value that reflects the degrees of truth of  $P$ ,  $M$  and  $N$ . The AND operator in fuzzy logic is usually replaced by the *min* function [26], *i.e.* the output of the rule is the *minimum* degree of truth among  $P$ ,  $M$  and  $N$ . *E.g.* for the rule (1), if  $P = 0.6$ ,  $M = 0.4$  and  $N = 0.6$ , the output is *Perfect* with a degree of truth 0.4. At the end of this step, the outputs of all the applied rules are aggregated into a new fuzzy set. Several aggregation methods can be used at this point, in our case we sum up the outcomes for each applied rule and combine them into a fuzzy set given by the membership functions in Fig. 6. The process is analogous to fill a set of silos based on the outcomes of the applied rules (the shape of the silos is given by the functions in Fig. 6). *E.g.* If the rules 2, 4 and 6 (as depicted in Fig. 5) are applied during the analysis and



their value outcomes are respectively  $Good = 0.2$ ,  $Good = 0.4$ ,  $Average = 0.3$ , at the end we have a fuzzy set which is the combination of the fuzzy sets  $Good$  and  $Average$  but with maximum degrees of truth of 0.6 and 0.3 respectively. The set of membership functions in Fig. 6 is also defined by means of Eq. 6, nonetheless this time the values are  $\alpha = 1/8$ ,  $\sigma^2 = 0.16$  and  $c = \{0.0, 2.5, 5.0, 7.5, 10.0\}$ .

**Defuzzification.** Based on the fuzzy set generated during the analysis, for each  $SVN$  we compute a value that is the final score of the  $SVN$ . This defuzzification process can be performed in many different ways. Due to its simplicity, we have used a discrete version of the so-called *center of gravity (COG)* method that easily computes a score and uses the following equation [25]:

$$Defuzzification(A) = D(A) = \frac{\sum_{y_{min}}^{y_{max}} y * A(y)}{\sum_{y_{min}}^{y_{max}} A(y)} \tag{7}$$

where  $A$  is the fuzzy set computed during the analysis and  $Y$  is the set of elements for which we want to determine a degree of truth with respect to  $A$ . Since we want to give scores within the range  $[0, 10]$ ,  $Y = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ . To clarify the three steps previously described we present an hypothetical example generated based on the information gathered from the NDAQ database.

**Example.** Fig. 7 depicts a setting in which three  $SVNs$  offer different  $FCs$  8.  $SVN_1$ , that can fully offer the required  $FCs$ , is a reduced version of the  $SVN$  in Fig. 1 whereas  $SVN_2$  and  $SVN_3$  are networks that can partially cover the customer need. For instance,  $SVN_3$  not only offers  $FC_2$  and  $FC_3$  which are required  $FCs$  but also misses  $FC_1$  and offers  $FC_w$  that is not required by the customer. Assuming the customer weights in Fig. 7 and applying Eqs. 2, 3, and 4 we obtain the following fractions for  $SVN_3$ :

$$F_P = \frac{0.8 + 1.0}{0.6 + 0.8 + 1.0} = \frac{1.8}{2.4} = 0.75, \quad F_M = \frac{0.6}{0.6 + 0.8 + 1.0} = \frac{0.6}{2.4} = 0.25 \tag{8}$$

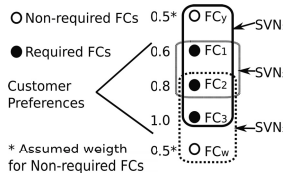
$$F_N = \frac{0.5}{0.6 + 0.8 + 1.0 + 0.5} = \frac{0.5}{2.9} = 0.17 \tag{9}$$

Afterwards, by making use of the membership functions in Fig. 4, we can compute the values for  $P$ ,  $M$  and  $N$ . Since,  $P$  is *some* (0.45) and *many* (0.45),  $M$  is *few* (0.45) and *some* (0.45) and  $N$  is *few* (0.70) and *some* (0.25), during the analysis step the rules 1 to 8 are applied as depicted in Fig. 9. As can be observed, the AND operator is replaced by the *min* function and the aggregation stage simply sums up the rules' outcomes.

Once the rules' outcomes have been aggregated, the defuzzification step computes the *COG* to produce the final score for  $SVN_3$  by means of Eq. 7 with  $A$  as depicted in Fig. 9c,  $Y = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ . In Eq. 10 we depict how the computation was carried out which can also be visually verified in Fig. 9c.

---

<sup>3</sup> For simplicity we only depict the  $FCs$  offered by each  $SVN$ .



**Fig. 7.** *SVNs* offering different *FCs*. Customer requirements are given by  $FC_1, FC_2$  and  $FC_3$ . Although  $SVN_1$  offers the requested *FCs*, it also offers a non-required *FC*. Moreover,  $SVN_2$  and  $SVN_3$  both only offer two of the requested *FCs*.

$$D(A) = \frac{1 * 0.17 + 2 * 0.70 + 3 * 0.70 + 4 * 0.45 + 5 * 0.70}{0.17 + 0.70 + 0.70 + 0.45 + 0.70 + 0.45 + 0.80 + 0.80 + 0.45 + 0.45} + \frac{6 * 0.45 + 7 * 0.80 + 8 * 0.80 + 9 * 0.45 + 10 * 0.45}{0.17 + 0.70 + 0.70 + 0.45 + 0.70 + 0.45 + 0.80 + 0.80 + 0.45 + 0.45} = 5.68 \quad (10)$$

**Table 1.** Important measures in the verification process for each *SVN*

	<i>SVN</i> <sub>1</sub>	<i>SVN</i> <sub>2</sub>	<i>SVN</i> <sub>3</sub>
<i>P</i>	Many (1.0)	Some (0.90), Many (0.10)	Some (0.45), Many (0.45)
<i>M</i>	Few (1.0)	Few (0.10), Some (0.90)	Few (0.45), Some (0.45)
<i>N</i>	Few (0.70), Some (0.25)	Few (1.0)	Few (0.70), Some (0.25)
Applied rules 1,2		1,3,5,7	1-8
Aggregation	Perfect(0.7),Good(0.25)	Perfect (0.1), Good(0.1), Average(0.1), Poor(0.9)	Perfect(0.45), Good(0.95) Average(0.7), Poor(0.7)
<b>Score</b>	8.69	3.66	5.68

Fig. 9 illustrates how the defuzzification is performed for  $SVN_1, SVN_2$  and  $SVN_3$ . Finally, Table 1 summarizes some of the important results that allow computing the final scores for  $SVN_1, SVN_2$  and  $SVN_3$ . As can be observed, for  $SVN_1$  only the rules 1 and 2 are applied whereas for  $SVN_2$  the rules 1,3,5 and 7 are applied.

### 3.3 Critique

According to Chandrasekaran, if the design task has been unsuccessful, this step identify the source of failure within a design [7]. Furthermore, the main goal at this stage is mostly about finding ways to improve the design [22]. In our case, if a customer is not satisfied by any composed *SVN*, (s)he will identify the sources of failure for a selected *SVN*, otherwise (s)he will select a *SVN* to satisfy her/his need (Fig. 2). More specifically, since *SVNs* might miss *FCs* and/or offer non-required *FCs*, the customers trade off the *FCs* within the selected *SVN*. On the one hand, the customers identify the *FCs* that are not interesting and cause failures to the *SVN*. On the other hand, they can also identify *FCs* that were not required but might be interesting to them. Consequently, in order to achieve this task, two subtasks are required: Select and Trade off.

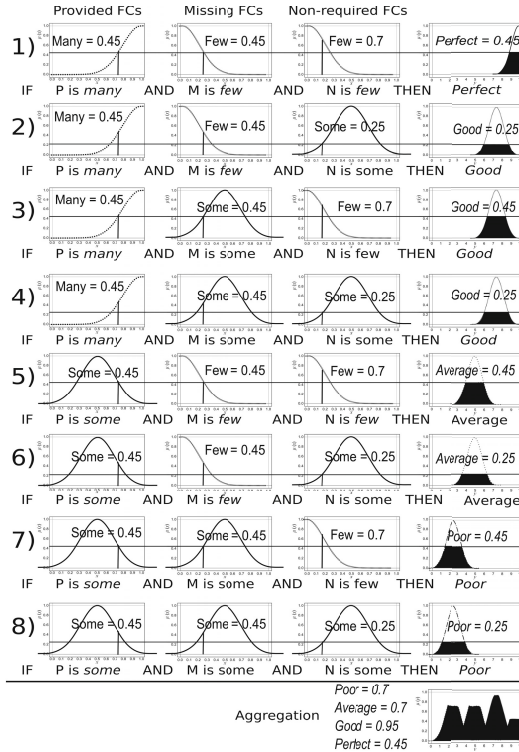


Fig. 8. Applying the first eight inference rules to  $SVN_3$

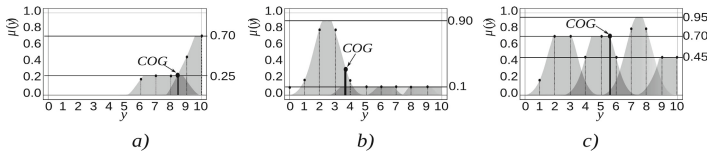


Fig. 9. Defuzzification process for: a)  $SVN_1$ , b)  $SVN_2$ , and c)  $SVN_3$

- **Select.** Based on the ranking computed by the verification step, the customer can select a  $SVN$  either to acquire its services or critique its  $FCs$ . If the customer decides to acquire a composed  $SVN$ , the goal of the PCM method has been reached. Otherwise, the customer must perform a trade off for the  $FCs$  offered by the  $SVNs$ .
- **Trade off.** Once the customer has selected a  $SVN$  to be critiqued, (s)he must give scores for the  $FCs$  offered by the  $SVN$ . The customer is only required to give scores for the  $FCs$  that were not required, however, (s)he also has the opportunity to change her/his preferences for the previously required  $FCs$ . With this information, it is possible now to modify the customer requirements and propose new  $SVNs$  to the customer.

### 3.4 Modify.

Chandrasekaran defines this step as the process of taking as input information about the failures of a solution design and then changing the design to get closer to the specifications, which involves proposing a new solution design and going again through the verify and critique steps [7]. In our framework, we take as input the customer's feedback to update her/his preferences and propose a new pool of *SVNs* that possibly fit better the customer requirements. Before proposing new *SVNs* two steps must be performed: Adaptation and Update.

- **Adapt.** Based on the scores given by the customer, this subtask can automatically recalculate a new set of *FCs*. The *FCs* with scores greater than zero (0) are part of the new set of required *FCs*, otherwise they are ignored.
- **Update.** Finally, this subtask replaces the old set of required *FCs* by the new set computed in the previous subtask (Adapt). In this way, the composition of new *SVNs* is triggered again, *i.e.* we go back through all the PCM cycle.

### 3.5 On $e^3$ service Computational Evaluation

The proposed framework has been implemented making use of 1) RDF<sup>4</sup> files to represent customer needs, service suppliers and service enablers, and 2) the Java framework Jena<sup>5</sup> that allows building semantic web applications. To (semi) automatically compose *SVNs*, we have used 114 services from the NDAQ database. Although the asymptotic complexity of the framework is  $O(2^m)$ , where  $m$  is the number of required *FCs*, after carrying on experimentation, the performance of the framework remains within reasonable time boundaries ( $10^4$  ms.) for values of  $m = [3, 15]$ , which is also acceptable since customers rarely request high number of *FCs* as already observed in previous case studies in the health care, telecommunications and energy industries [16,13,4].

## 4 Conclusions and Future Work

We have presented a novel framework to compose *SVNs* that is based on the well-known problem-solving method *Propose - Critique - Modify* and provides answers to the *customer supplier interaction (CSI)*, *tailored composition* and *automatic verification* issues. The  $e^3$  service framework enhances the co-creation of *SVNs*, by means of *CSI*, which leads to *SVNs* that better fit customer needs. Moreover the composed *SVNs* are not just tailored based on customer requirements but also automatically verified to assess fitness. The main contributions in this work are: 1) a PCM-based framework that contains the needed knowledge to achieve (semi)automatic composition of *SVNs*, and 2) an automatic verification method that determines to what extent the composed *SVNs* fit to the customer requirements.

<sup>4</sup> [www.w3.org/RDF/](http://www.w3.org/RDF/)

<sup>5</sup> <http://incubator.apache.org/jena>

For the future work we want to include non-functional requirements as well as time and location constraints for the composed *SVNs* so the networks not only offer an accurate answer in terms of functional requirements but also are aware of the quality of the services and the customers' availability and proximity. We strongly consider that the fuzzy-logic-based verification method can easily be extended to face these issues. Finally, we have also to validate more our framework making use of more case studies. Currently, we are exploring composition ideas in the health care sector [9] and global software development [23].

**Acknowledgment.** This paper has been partially funded by the NWO/Jacquard project VALUE-IT no 630.001.205

## References

1. Agarwal, S., Handschuh, S., Staab, S.: Annotation, composition and invocation of semantic web services. *Journal of Web Semantics* 33, 1–24 (2004)
2. Allee, V.: A value network approach for modeling and measuring intangibles. In: *Transparent Enterprise Conference* (2002)
3. Baida, Z.: Software-aided service bundling. PhD thesis, Free University Amsterdam (2006)
4. Baida, Z., Gordijn, J., Sæle, H., Morch, A.Z., Akkermans, H.: Energy Services: A Case Study in Real-World Service Configuration. In: Persson, A., Stirna, J. (eds.) *CAiSE 2004*. LNCS, vol. 3084, pp. 36–50. Springer, Heidelberg (2004)
5. Basole, R.C., Rouse, W.B.: Complexity of service value networks: conceptualization and empirical investigation. *IBM Syst. J.* 47, 53–70 (2008)
6. Becker, J., Beverungen, D., Knackstedt, R., Müller, O.: Model-based decision support for the customer-specific configuration of value bundles. *Enterprise Modelling and Information Systems Architectures* 4(1), 26–38 (2009)
7. Chandrasekaran, B.: Design problem solving: A task analysis. *AI Magazine* 11, 59–71 (1990)
8. Danylevych, O., Karastoyanova, D., Leymann, F.: Service networks modelling: An soa & bpm standpoint. *J. UCS*, 1668–1693 (2010)
9. de Kinderen, S.: Needs-driven service bundling in a multi-supplier setting: The computational e3service approach. PhD thesis, Vrije Universiteit Amsterdam (2010)
10. Dustdar, S., Schreiner, W.: A survey on web services composition. *Int. J. Web Grid Serv.* 1(1), 1–30 (2005)
11. Gordijn, J., Akkermans, H., van Vliet, H.: Business Modelling Is Not Process Modelling. In: Mayr, H.C., Liddle, S.W., Thalheim, B. (eds.) *ER Workshops 2000*. LNCS, vol. 1921, pp. 40–51. Springer, Heidelberg (2000)
12. Gordijn, J., Akkermans, J.M.: e3-value: Design and evaluation of e-business models. *IEEE Intelligent Systems*, 11–17 (2001)
13. Gordijn, J., de Haan, F., de Kinderen, S., Akkermans, H.: Needs-Driven Bundling of Hosted ICT Services. In: van Bommel, P., Hoppenbrouwers, S., Overbeek, S., Proper, E., Barjis, J. (eds.) *PoEM 2010*. LNBIP, vol. 68, pp. 16–30. Springer, Heidelberg (2010)
14. Gordijn, J., De Leenheer, P., Razo-Zapata, I.S.: Generating service valuewebs by hierarchical configuration: An ipr case. In: *Proceedings of HICSS 44* (2011)

15. Hamilton, J.: Service value networks: Value, performance and strategy for the services industry. *Journal of Systems Science and Systems Engineering* 13(4), 469–489 (2004)
16. de Kinderen, S., Gordijn, J., Droes, R.-M., Meiland, F.: A computational approach towards eliciting needs-driven bundles of healthcare services. In: *BLED 2009 Proceedings* (2009)
17. Li, M.S.: The Economics of Utility Services in the Future Internet. In: *Towards the Future Internet - Emerging Trends from European Research*, pp. 31–40. IOS Press (2010)
18. Lovelock, C.H., Wirtz, J.: *Services Marketing: People, Technology, Strategy*, 7th edn. Pearson Higher Education (2010)
19. Razo-Zapata, I.S., Gordijn, J., De Leenheer, P., Akkermans, H.: Dynamic cluster-based service bundling: A value-oriented framework. In: *IEEE 13th Conference on Commerce and Enterprise Computing* (2011)
20. Razo-Zapata, I.S., De Leenheer, P., Gordijn, J., Akkermans, H.: Service Network Approaches. In: *Handbook of Service Description: USDL and its Methods*, pp. 45–74. Springer (2011)
21. Razo-Zapata, I.S., De Leenheer, P., Gordijn, J., Akkermans, H.: Service value networks for competency-driven educational services: A case study. In: *6th International BUSITAL Workshop* (2011)
22. Schreiber, G., Akkermans, H., Anjewierden, A., de Hoog, R., Shadbolt, N., van de Velde, W., Wielinga, B.: *Knowledge Engineering and Management: The CommonKADS Methodology*. The MIT Press (2000)
23. Tamburri, D.A., Razo-Zapata, I.S., Fernández, H., Tedeschi, C.: Simulating awareness in global software engineering: a comparative analysis of scrum and agile service networks. In: Lewis, G.A., Lago, P., Metzger, A., Tasic, V. (eds.) *International Workshop on Principles of Engineering Service-Oriented Systems*. IEEE (2012)
24. Traverso, P., Pistore, M.: Automated composition of semantic web services into executable processes, pp. 380–394. Springer (2004)
25. Van Leekwijck, W., Kerre, E.E.: Defuzzification: criteria and classification. *Fuzzy Sets Syst* 108, 159–178 (1999)
26. Zadeh, L.A.: Fuzzy logic = computing with words. *IEEE Transactions on Fuzzy Systems* 4(2), 103–111 (1996)

# Cooperative Service Composition

Nikolay Mehandjiev<sup>1</sup>, Freddy Lécué<sup>2</sup>, Martin Carpenter<sup>1</sup>, and Fethi A. Rabhi<sup>3</sup>

<sup>1</sup> The University of Manchester Centre for Service Research, Manchester, UK  
initial.lastname@manchester.ac.uk

<sup>2</sup> IBM Research, Smarter Cities Technology Centre, Dublin, Ireland  
freddy.lecue@ie.ibm.com

<sup>3</sup> The University of New South Wales, Sydney, Australia  
f.rabhi@unsw.edu.au

**Abstract.** Traditional service composition approaches are top-down(using domain knowledge to break-down the desired functionality), or bottom-up (using planning techniques). The former rely on available problem decomposition knowledge, whilst the latter rely on the availability of a known set of services, otherwise automatic composition has been considered impossible. We address this by proposing a third approach: Cooperative Service Composition (CSC),inspired by the way organisations come together in consortia to deliver services. CSC considers each service provider as proactive in service composition, and provides a semantics-based mechanism allowing innovative service compositions to emerge as result of providers’ interactions. The key challenges we resolve are how to determine if a contribution brings the composition closer to its goal, and how to limit the number of possible solutions. In this paper we describe the approach and the solutions to the two key challenges, and demonstrate their application to the composition of financial web services.

**Keywords:** service composition, semantic services, software agents.

## 1 Introduction

Service-oriented software development often focuses on composing services to fulfil a set of user requirements. An intuitive view to service composition would perceive it as an activity which aims to satisfy the need for a (non-existing) service by bringing together existing ones. This integration activity can be done manually, yet automating it makes it more in tune with the vision of composing services at the point of need [1], and takes service-oriented computing beyond component-based software engineering.

There is a wide variety of approaches and methods for service composition [2], yet most of them fit into one of two “camps”:

- top-down approaches, which break-down the desired functionality into smaller units using domain knowledge and templates, or
- bottom-up approaches which use planning to construct a composition without a template, only by using knowledge of available services and their interfaces.

Top-down approaches [3–6] use formalised problem decomposition knowledge, breaking down desired functionality into simpler units, and then seeking services for

each such unit. For example, [6] shows how we can select a set of services which fit in terms of input and output data types. These approaches are efficient and preferred where the decomposition knowledge exists. However, they only find solutions prescribed by the decomposition knowledge, often missing the chance for innovation.

The bottom-up approaches rely on the availability of a fixed set of well-specified services. In principle these are quite inefficient yet many improvements in terms of search heuristics have been developed, and also many hybrid approaches which attempt to optimise the split between top-down and bottom-up approaches (e.g. HTN [7]).

There are however situations, where we do not have a fixed set of known services nor decomposition knowledge about the desired functionality, and innovative compositions are desired. In these circumstances neither of the two mainstream approaches is suitable, and here we propose a third, novel approach called *cooperative service composition*.

Our approach, detailed in Section 4, is inspired by the manner in which human organisations form consortia to respond to market demands and provide innovative services. It implies pro-active roles for the service composer and for the service providers, represented by autonomous software entities, called agents. These are pre-programmed to behave according to the business interests of their organisations, and are able to reason over the semantic specifications of requirements and candidate services.

In our approach, innovative service compositions emerge as a result of cooperation between service providers, responding to requests by service composers. There are two key challenges we had to resolve in achieving this idea:

1. how to determine if a contribution brings the composition closer to its goal, and
2. how to limit the number of possible solutions, thus avoiding combinatorial explosion of complexity.

In this paper we describe the approach and the solutions we developed for the two key challenges enabling the approach, in Sections 4 and 5. This is preceded by the introduction of a motivating example in Section 2, and by the description of semantic annotations of services, and techniques for calculating semantic distances we use to guide our approach in Section 3. We then proceed to apply the approach to our detailed example in Section 6, comparing with top-down approaches.

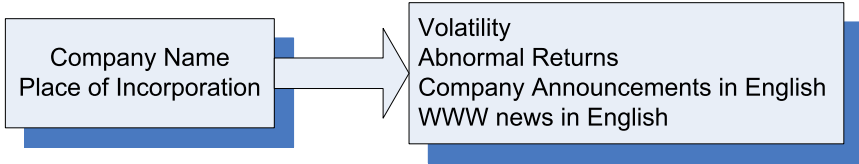
## 2 Motivating Example

This scenario illustrates the value of cooperative service composition: Carl is a fund manager, preparing a new equity (stock) fund focusing on the Far East. He needs to conduct a number of visits to companies of interest in China and Hong Kong, and he should filter target companies from a large number of candidates. For this he needs a software service which, given the details of a company, will produce a portfolio of financial and legal information about the company, including the following:

- volatility of company share price over the last 3 years;
- “abnormal returns” from the company stock over the last 3 years;
- company announcements, translated into English if necessary;
- news in English regarding the company over the last three months.



The desired functionality is illustrated in Figure 1. To formalise the concepts involved in Carl's request and thus allow semantic annotation of our services, we have customised the SUMO Financial Ontology<sup>1</sup>. Extracts from the ontology are included in the T-box presented in Figure 2.



**Fig. 1.** Desired functionality in terms of inputs and outputs

Given a formalised ontology of domain concepts, the desired functionality of the service Carl is seeking can be specified within a service broker or a service search engine, attempting to find an appropriate service to directly deliver this functionality. If this fails, we can proceed to service composition, attempting to find services which deliver parts of the desired functionality, and to bring them together [8]. The service broker may have a description of how this problem can be decomposed into sub-problems, in which case it will use it to decompose the functionality into tasks providing (a) volatility and abnormal returns information based on stock price movements; (b) the company announcements in English; and (c) the news about the company in English), and attempt to find at least one candidate service for each task. This is the conventional top-down approach [3], which allows efficient search, yet it misses innovative solutions to the problem, for example one service providing all the information we need in Chinese, and then we translate into English with another service.

This is avoided by the alternative approach which uses program synthesis and AI planning techniques [9]. Given a start state  $St_{start}$  and an end state  $St_{end}$ , it employs reasoning over the pre- and post-conditions of available services, trying to create a plan of putting them together to satisfy the required functionality. It will use the Chinese company information service since its inputs fit the available inputs of company name and place of incorporation, and will proceed to look for a service which can bridge the two states  $St_c$  (all information in Chinese) and  $St_e$  (all information in English). This approach is inefficient, although many heuristic-based improvements exist, and also hybrids between the top-down and bottom-up approaches [7]. The bigger problem is relying on the broker knowing all available services, with their functionality fully defined and fixed for the duration of the composition.

The approach we propose here also works bottom-up, yet it relies on the composition being created by the service providers in a cooperative fashion. In our example, the service broker will announce the need for the service including descriptions of start and end states on a specialist noticeboard for software services in the financial sector. Software service providers will see this opportunity, and bid to contribute to a solution as a part of a bigger consortium. Provider A, for example, will bid to provide the company

<sup>1</sup> <http://www.ontologyportal.org/>

<p> <i>Corporation</i> <math>\sqsubseteq</math> <i>Organization</i> <math>\sqsubseteq</math> <i>CognitiveAgent</i> <math>\sqsubseteq</math> <i>SentientAgent</i>  <i>SentientAgent</i> <math>\sqsubseteq</math> <i>Agent</i> <math>\sqsubseteq</math> <i>Entity</i>  <i>StockMarket</i> <math>\sqsubseteq</math> <i>Organization</i>  <i>StockMarketTransaction</i> <math>\equiv</math> <i>FinancialTransaction</i>  <math>\sqcap \forall \text{hasPatient}. \text{Stock}</math>  <math>\sqcap \forall \text{islocated}. \text{StockMarket}</math>  <i>Stock</i> <math>\equiv</math> <i>FinancialAsset</i> <math>\sqcap \exists \text{issuesBy}. \text{Corporation}</math>  <math>\sqcap \forall \text{issuesBy}. \text{Corporation}</math>  <math>\sqcap \exists \text{listedOn}. \text{Organization} \sqcap \forall \text{listedOn}. \text{Organization}</math>  <math>\sqcap \exists \text{stockSymbol}. \text{SymbolString} \sqcap \forall \text{stockSymbol}. \text{SymbolString}</math>  <i>StockSymbol</i> <math>\equiv \forall \text{hasStock}. \text{Stock} \sqcap \exists \text{stockMarket}. \text{StockMarket}</math>  <math>\sqcap \forall \text{stockMarket}. \text{StockMarket}</math>  <i>CompanyName</i> <math>\equiv \forall \text{names}. \text{Corporation} \sqcap \exists \text{names}. \text{Corporation}</math>  <i>PartialInformation</i> <math>\equiv \text{FactualText} \sqcap \forall \text{hasPart}. \text{Corporation}</math>  <i>CompanyAnnouncement</i> <math>\equiv \text{FactualText} \sqcap \forall \text{authors}. \text{Corporation}</math>  <i>CompanyNews</i> <math>\equiv \text{FactualText}</math>  <math>\sqcap \forall \text{containsInformation}. \text{PartialInformation}</math>  <i>Corporation</i> <math>\equiv \exists \text{hasAnnouncement} \text{Company}. \text{CompanyAnnouncement}</math>  <math>\sqcap \exists \text{hasNews} \text{Company}. \text{CompanyNews}</math>  <i>TimeSeries</i> <math>\equiv \text{TimeDependentQuality} \sqcap \exists \text{hasTime}. \text{Time}</math>  <math>\sqcap \exists \text{hasStockSymbol}. \text{StockSymbol}</math>  <math>\sqcap \exists \text{hasTimeSeriesPerStock}. \text{Value}</math>  <i>EODPrices</i> <math>\equiv \text{TimeSerie} \sqcap \forall \text{hasRange}. \text{ClosingPrice}</math>  <i>StockTimeSeries</i> <math>\equiv \text{TimeSeries} \sqcap \exists \text{hasStockSymbol}. \text{StockSymbol}</math>  <math>\sqcap \forall \text{hasStockSymbol}. \text{StockSymbol}</math> </p>
---

**Fig. 2.** Part of an  $\mathcal{AL}\mathcal{E}$  Terminology in the Finance Domain of SUMO

information in Chinese, whilst Provider B will bid to translate from Chinese to English. This allows for circumstances where services have just appeared on the marketplace and are still unknown to the service broker, or where the service providers are willing to change their service to address a lucrative business opportunity. In our case Provider A's original service may not provide news relating to the company, and they may be willing to sub-contract such a service and modify their offering to suit the opportunity.

The key to successful operation of this approach, in common with the planning approach, is the availability of a guiding metric indicating if a service is usefully contributing to the eventual solution or not. This is the first main issue that we address. The other main issue is how to trim the number of possible partial solutions, avoiding the unnecessary creation of millions of consortia with only marginal difference between their offerings. These two issues will be explored further in the remainder of this paper.

### 3 Preliminaries

To support the cooperative assembly of composite services by software agents, we need formal knowledge (semantic annotation) about the inputs and outputs of these services as well as about their functionality.

### 3.1 Semantically Annotating Services, Links and Compositions

To allow automated reasoning, we need to annotate software services with formal semantic descriptions of their functionality (or goals), inputs, outputs, pre-conditions, and post-conditions. These services are then known as *Semantic web services* [10].

In principle, the formal semantic specifications of semantic web services are based on Description Logics (DL) [11]. Within the domain of information processing, we can focus on three parameters - functionality, inputs and outputs, since within this domain, a pre-condition can usually be simplified to the availability of all the needed inputs, and the post-condition is usually the availability of all or some of the declared outputs.

The functionality, input and output parameters of services are described according to a common ontology or Terminology  $\mathcal{T}$  (e.g., Fig 2), where the OWL-S profile [12] or SA-WSDL [13] can be used to describe them through semantic annotations [9].

*Semantic links* [6] between input and output parameters of services can be then defined, based on semantic similarities between an output of a service  $s_1(Out_{s_1})$  and the input of a subsequent service  $s_2(In_{s_2})$ , where both are DL concept descriptions (e.g., Fig 2). Its objective is to measure how services could fit together in a semantic context. However, reaching a composition model where services are *semantically* linked is far from trivial, with more services required to fill the missing gap. We use “feeder”  $F$  and “sink”  $S$  to refer to pairs of concepts describing a desired transformation by a “yet-to-be determined” set of services, “feeder” serves as an input to this transformation which produces “sink” as a result. In this work, the gap between  $F$  and  $S$  is called *semantic gap*, referring to both semantic concept descriptions and services which are missing to properly compose services and “plug” that gap.

In this paper we use semantic gaps  $g_{i,j}$  between concepts  $F$  and  $S$  to guide the cooperative composition process. In the extreme initial case the semantic gaps covers the whole of the desired composite service ( $F = In_{start}$  and  $S = Out_{end}$ ). In the general case, though, the semantic gap will be between the output of a service  $Out_{s_i}$  and the input of another service  $In_{s_j}$ .

$$g_{i,j} \doteq \langle s_i, Sim_{\mathcal{T}}(F, S), s_j \rangle \quad (1)$$

where the *feeder*  $F = Out_{s_i}$  and the *sink*  $S = In_{s_j}$  are DL-based concept descriptions.

Given a terminology  $\mathcal{T}$ , [15] and [16] value the range of  $Sim_{\mathcal{T}}$  along five matching types: i) *Exact* i.e.,  $F \equiv S$ , ii) *PlugIn* i.e.,  $F \sqsubseteq S$ , iii) *Subsume* i.e.,  $S \sqsubseteq F$ , iv) *Intersection* i.e.,  $\neg(F \sqcap S \sqsubseteq \perp)$  and v) *Disjoint* i.e.,  $F \sqcap S \sqsubseteq \perp$  (i.e., inconsistent gap).

The valuation of semantic gaps is important to judge how two services could be close or far in term of descriptions they manipulate. To this end, the matching function  $Sim_{\mathcal{T}}$  of [1] enables, at design time, finding semantic compatibilities (i.e., Exact, PlugIn, Subsume, Intersection), incompatibilities (i.e., Disjoint) among independently defined service descriptions, and thus can be used to define the size (“quality”) of gaps between services. The process model of web service composition and its semantic gaps is specified by a directed graph which has web service specifications  $s_i$  or semantic gaps  $g_{i,j}$  (description dissimilarities, serving as placeholders for new composite

<sup>2</sup> Distributed ontologies [14] for achieving semantic annotation are not considered here but are largely independent of the problem addressed in this work.

services) as its edges, and the input and output concepts of services as its nodes. The start node(s) of the graph will be the input concept(s)  $In_{start}$ , whereas the end nodes of the graph will be the output concept(s)  $Out_{end}$ .

Semantics gaps in composition should be filled with appropriate services to bridge different data descriptions, thus we suggest to compute the information contained in  $S$  but not in  $F$ , using Concept Abduction (Definition 3 in [17]). We adapt it from [18] as follows.

**Definition 1. (Concept Abduction)**

Let  $\mathcal{L}$  be a DL,  $F, S$  be two concepts in  $\mathcal{L}$ , and  $\mathcal{T}$  be a set of axioms in  $\mathcal{L}$ . A Concept Abduction Problem, denoted as  $S \setminus F$  consists in finding a concept  $H \in \mathcal{L}$  such that  $\mathcal{T} \not\models F \sqcap H \equiv \perp$ , and  $\mathcal{T} \models F \sqcap H \sqsubseteq S$ .

In case a semantic gap [11] is not valued by a Disjoint or Exact matching, Definition 1 is applied to obtain  $S \setminus F$ . This information refers to the *Missing Description* required but not provided by  $F$  to resolve a gap with  $S$ .

### 3.2 Quality of Semantic Gaps

In this section we present the quality model used to evaluate different consortia and to also guide contributions by individual service providers. The model, inspired from [6], evaluates compositions based on both non functional (QoS attributes such as price, response time, etc. [19]) and semantic quality, based on the quality of semantic gaps. In more detail two generic quality criteria for a semantic gap [11] are considered: its i) *Common Description* rate (Definition 2), and ii) *Matching Quality*.

We compute *Common Description* between *feeder*  $F$  and *sink*  $S$  concepts using Concept Abduction as follows:

**Definition 2. (Common Description rate of a Semantic Gap)**

Given a semantic gap  $g_{i,j}$  between  $F$  and  $S$ , the *Common Description* rate  $q_{cd} \in (0, 1]$  provides one possible measure for the degree of similarity between  $F$  and  $S$  and the quality of the semantic gap  $g_{i,j}$ . This rate is computed using the following expression:

$$q_{cd}(g_{i,j}) = q_{cd}(F, S) = \frac{|lcs(F, S)|}{|S \setminus F| + |lcs(F, S)|} \quad (2)$$

This criterion estimates the proportion of descriptions which is well specified by “feeder”  $F$  in “sink”  $S$ .

The expressions in between “|” refer to the size of concept descriptions ([20] p.17) i.e.,  $|\top|, |\perp|, |A|, |\neg A|$  and  $|\exists r|$  is 1;  $|C \sqcap D| \doteq |C| + |D|$ ;  $|\forall r.C|$  and  $|\exists r.C|$  is  $1 + |C|$ .

**Definition 3. (Matching Quality of a Semantic Gap)**

The *Matching Quality*  $q_m$  of a semantic gap  $g_{i,j}$  is a value in  $(0, 1]$  defined by  $Sim_{\mathcal{T}}(i, j)$  i.e., either 1 (*Exact*),  $\frac{3}{4}$  (*PlugIn*),  $\frac{1}{2}$  (*Subsume*) or  $\frac{1}{4}$  (*Intersection*).

At composition level, Common Description rate measures the average similarity between all corresponding pairs  $(F, S)$  which create semantic gaps in the composition.

The matching quality of a composition  $c$  estimates the overall matching quality of its semantic gaps. Contrary to the common description rate, this criterion aims at easily distinguishing and identifying between very good and very bad matching quality.

### 3.3 Quality of Service Compositions

We present definitions for comparing and ranking different compositions along the common description rate and matching quality dimension. The rules for aggregating quality values (Table 1) for any concrete composition  $c$  are driven by them. In more details the approach for computing semantic quality of  $c$  is adapted from the application-driven heuristics of [6], while the computation of its non functional QoS is similar to [21].

#### Definition 4. (Common Description rate of a Composition)

*The Common Description rate of a composition  $c$  measures the average degree of similarity between all corresponding pairs  $(F, S)$  which create semantic gaps in  $c$ .*

The Common Description rate  $Q_{cd}$  of both a sequential and AND-Branching composition is defined as the average of its semantic gaps' common description rate  $q_{cd}(g_{i,j})$ . The common description rate of an OR-Branching composition is a sum of  $q_{cd}(g_{i,j})$  weighted by  $p_{g_{i,j}}$  i.e., the probability that semantic gap  $g_{i,j}$  be chosen at run time. Such probabilities are initialized by the composition designer, and then eventually updated considering the information obtained by monitoring the workflow executions.

#### Definition 5. (Matching Quality of a Composition)

*The matching quality of a composition estimates the overall matching quality of its semantic gaps. Contrary to the common description rate, this criterion aims at easily distinguishing and identifying between very good and very bad matching quality.*

The matching quality  $Q_m$  of a sequential and AND-Branching composition is defined as a product of  $q_m(g_{i,j})$ . All different (non empty) matching qualities involved in such compositions require to be considered together in such a (non-linear) aggregation function to make sure that compositions that contains semantic gaps with low or high matching quality will be more easily identified, and then pruned for the set of potential solutions. The matching quality of an OR-Branching composition is defined as its common description rate by replacing  $q_{cd}(g_{i,j})$  with  $q_m(g_{i,j})$ .

Details for computing **Execution Price**  $Q_{pr}$  and **Response Time**  $Q_t$  can be found in Table 1 and further explained in [21].

Using Table 1 the quality vector of any concrete composition can be defined by:

$$Q(c) \doteq (Q_{cd}(c), Q_m(c), Q_t(c), Q_{pr}(c)) \quad (3)$$

The adopted quality model has a limited number of criteria (for the sake of illustration), yet (3) is extensible, allowing new functional and non-functional criteria such as reputation, availability, reliability, etc.

**Table 1.** Quality Aggregation Rules for Semantic Web Service Composition

Composition Construct	Quality Criterion			
	Functional		Non Functional	
	$Q_{cd}$	$Q_m$	$Q_t$	$Q_{pr}$
Sequential/ AND- Branching	$\frac{1}{ g_{i,j} } \sum_{g_{i,j}} q_{cd}(g_{i,j})$	$\prod_{g_{i,j}} q_m(g_{i,j})$	$\frac{\sum_{s_i} q_t(s_i)}{\max_s q_t(s)}$	$\sum_{s_i} q_{pr}(s_i)$
OR-Branching	$\sum_{g_{i,j}} q_{cd}(g_{i,j}) \cdot p_{g_{i,j}}$	$\sum_{g_{i,j}} q_m(g_{i,j}) \cdot p_{g_{i,j}}$	$\sum_{s_i} q_t(s_i) \cdot p_{s_i}$	$\sum_{s_i} q_{pr}(s_i) \cdot p_{s_i}$

## 4 Our Approach to Emergent Service Composition

### 4.1 Outline of the Overall Approach

Software agents representing service providers observe a number of noticeboards of interest. When a description of a requested composite service appears, each agent will check if they can meaningfully contribute to providing a partial solution, and record this partial solution onto the noticeboard. Using our example from Section 2, AgentB representing a translator service  $s_b$  from Chinese to English will record a partial solution  $St_c \xrightarrow{s_b} St_{end}$  onto the noticeboard.

Another agent (AgentA) will notice that they can now extend the new partial solution into a full solution by providing the service  $s_a$  to link  $St_{start} \xrightarrow{s_a} St_c$ , and apply to join the consortium behind the partial solution (in our case AgentB only).

Multiple partial solutions are allowed addressing the same business opportunity. Some of them will be elaborated into complete solutions, which will be internally consistent and deliver the requested functionality. The evaluation of all complete solutions is conducted by the service composer agent, using the quality vector from (3) in Section 3. A consortium will also apply quality of service composition metrics to decide between a number of potential applicants, this is described in detail in Section 5.

Initially, we focus on how we specify each of the states within the system, and how, using this state representation, each agent can determine if one of their services can meaningfully contribute to a partial solution on the noticeboard.

### 4.2 State Description

The system we propose targets the construction of composite information processing services, where a set of service components are “wired” together to deliver a set of information items as outputs for each set of information items provided as inputs.

Each component service within such a system, and the system itself, can be specified as transforming an input set of information items  $In \equiv \{i_1, i_2, \dots, i_n\}$  into an output set of information items  $Out \equiv \{o_1, o_2, \dots, o_n\}$ . Each information item  $i_i$  and  $o_i$  is defined by a concept from a terminology  $\mathcal{T}$ , shared by all agents.

We consider a partial solution  $ps$  to be defined by two states - its input state  $St_{in}$  and its output state  $St_{out}$ :  $ps = \langle St_{in}, St_{out} \rangle$

Because of the information processing nature of the domain we are investigating, we consider a state to be defined by the set of information items available at this state. Therefore  $St_{in} \equiv In_{ps}$  whilst  $St_{out} \equiv Out_{ps}$ .

The required composite service is also defined by its two states  $St_{start}$  and  $St_{end}$ . When  $St_{in} = St_{start}$  and  $St_{out} = St_{end}$ , we have a full solution.

### 4.3 Concept Similarity Metric

Each bid to join a partial solution changes one of the states defining a partial solution, either providing further processing to bring  $St_{out}$  closer to the target  $St_{end}$ , or some pre-processing to bring  $St_{in}$  closer to the in information available at  $St_{start}$ .

In the general case, each agent will apply their knowledge of the problem domain and the overall declared goal of the target composition to determine when their contribution is meaningful. However, within the information processing domain, we can formulate a more specific heuristic metric to help both the proposers and the accepting consortia judge if a contribution is meaningful (or “constructive”). To derive this heuristic, we use the semantic distance between the concepts defining two information items (semantic gap) using the concept of Common Description Rate as defined in (2). Recall that in this interpretation, the “feeder”  $F$  and “sink”  $S$  describe a desired transformation by a “yet to be determined” set of services.

### 4.4 Heuristics for Joining a Partial Solution

The heuristic described here is used both by the service provider agents when they decide which service from several they should propose as an extension to a partial solution, and by the consortium behind a partial solution when they decide which extension bid to accept from several alternatives.

For given partial solution  $ps = \langle St_{in}, St_{out} \rangle$  and a target solution  $G = \langle St_{start}, St_{end} \rangle$ , an agent will apply to join the consortium and extend the partial solution if their service  $s$  can reduce one of the existing concept gaps  $q_{cd}(g_{i,j})$ .

In formal terms we have two cases of reducing a concept similarity value:

#### Case 1: Reducing distance from the “input” side:

$$(F \in St_{start}) \wedge (S \in St_{in}) \wedge (Out_S = S) \wedge (q_{cd}(F, In_s) < q_{cd}(F, S)) \quad (4)$$

#### Case 2: Reducing the distance from the “output” side::

$$(F \in St_{out}) \wedge (S \in St_{end}) \wedge (In_s = F) \wedge (q_{cd}(Out_s, S) < q_{cd}(F, S)) \quad (5)$$

For two alternative services provided by the agent, it will choose to put forward the service which minimises the respective  $q_{cd}$ . We presume the services are alternative, *i.e.* they extend the partial solution from the same point along the same direction.

The same heuristic will be used by the consortium behind a partial solution when they have to evaluate several alternative bids by providers who offer their services to extend the existing partial solution.

#### 4.5 Semantic Quality of Partial Solutions

Based on the composition quality metric we defined in Section 3.2, any compositions (partial or complete) can be compared along two dimensions i.e., non functional and semantic quality. Therefore the quality of compositions in Table 11 can be compared by analysing their  $Q_{cd}$ ,  $Q_m$ ,  $Q_{pr}$  and  $Q_t$  elements. For instance  $Q(c_1) > Q(c_2)$  iff the quality of composition  $c_1$  is better than quality of  $c_2$  along each and every quality criteria. In case of conflicts e.g., the value of  $Q_{cd}$  and  $Q_m$  is better for  $c_1$  but worse for values of  $Q_{pr}$  and  $Q_t$ , we compare a weighted average (with a weight of  $\frac{1}{4}$ ) of their normalised components.

### 5 Reducing the Number of Partial Solutions Considered

Without any measures to alleviate the problem, the self-interested agents representing service providers will attempt to propose their services every time when there is an input or an output matching. Since alternative partial solutions are in principle allowed by the approach, this may quickly lead to a combinatorial explosion regarding the number of partial solutions held at a given noticeboard.

The following simple heuristics are designed to ensure this is controlled as far as possible. They represent a suggested procedural framework within which the composition may proceed in a controlled manner.

#### 5.1 Service Providers Appraise Their Bids

A service provider will consider it possible to place a bid for extending a partial solution  $ps = \langle St_{in}, St_{out} \rangle$  if the inputs or the outputs of one of its services can extend the partial solution (i.e.  $In_s \in St_{out} \vee Out_s \in St_{in}$ ). The only services considered for this will be those with matching functional classification, i.e. those which fit to the declared goal for the service composition on the noticeboard.

If a provider recognises more than one service as “fitting” the conditions, they will conduct an internal evaluation using the same rule which the consortium will apply to evaluate competing offers. This means they will select the service which achieves the smallest Common Description Rate  $q_{cd}$ .

A service provider will have a limited visibility of all the parameters of the partial solution on the noticeboard, so they would not be able to calculate and use the full quality model for the partial composition (3).

#### 5.2 Consortia behind Partial Solutions Evaluate Competitive Bids

A consortium will choose a single provider from a number of alternative providers offering to extend a solution in the same direction (i.e. providing the same output concept), using two metrics:

1. Minimising  $q_{cd}$ . This will ensure the consortium picks the “most effective offer” between the different service providers.
2. The overall service quality for the partial solution. This criteria will also be used by the service broker when evaluating from different complete solutions, so the consortium should extend reasonable effort to maximise this metric.



## 6 Example of Applying the Approach to Financial Services

In the example below we demonstrate how our approach can be used to support the scenario from Section 2

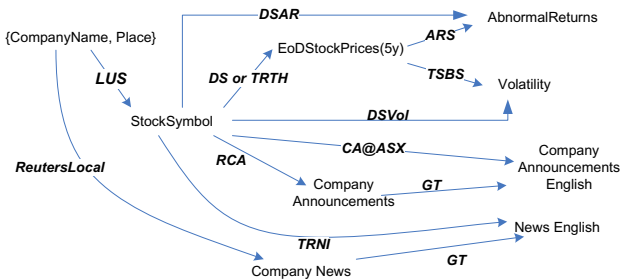
As a first step we calculate the Common Description Rates  $q_{cd}(F, S)$  for the example from the  $\mathcal{AL}\mathcal{E}$  Terminology in Figure 2. The results are as follows:

**Table 2.** Computation of Common Description (The lower the better; “-” means no compatible concept under  $\sqsubseteq_{\mathcal{T}}$ )

	CompanyName	StockSymbol	EODPRices	AR	Volatility	CompanyAn.	CompanyNews	CompanyAnEn
CompanyName	0	0.23	0.41	0.69	0.95	0.64	0.71	0.72
StockSymbol	0.12	0	0.35	0.55	0.90	0.65	0.79	0.66
EODPRices	0.33	0.31	0	0.41	0.67	0.85	0.81	0.89
AR	0.85	0.41	0.62	0	0.71	-	-	-
Volatility	0.75	0.55	0.55	0.88	0	0.95	0.91	0.97
CompanyAn.	0.55	0.67	-0.88	-	0.91	0	0.12	0.11
CompanyNews	0.76	0.63	0.76	-	0.92	0.21	0	0.09
CompanyAnEn	0.66	0.60	0.89	-	0.88	0.09	0.11	0

Please note that rows are the Sink ( $S$ ) concepts, whilst the columns are the Feeder ( $F$ ) concepts, so  $q_{cd}(CompanyName, StockSymbol) = 0.12$ .

*Services in the System.* In addition we have a number of services aligned with the domain of financial information provision. The service owners are monitoring the relevant financial service requests noticeboards. Figure 3 represents these services (in bold and italics) as a network organised around the concepts they process. Please note this is for visualisation only, and the network does not exist, since the knowledge about it is distributed amongst service providers.



**Fig. 3.** A visualisation of all financial services participating in our example, this knowledge is not available to any single agent in the system but is distributed amongst service provider agents

*Sequence of Contributions.* Carl's service requestor agent will formulate the following noticeboard request:

$$\begin{aligned} St_{start} &\equiv \{companyName, StockMarket\} \\ St_{end} &\equiv \{AbnormalReturns, Volatility, \\ &\quad CompanyAnEn, CompanyNews\} \end{aligned}$$

AgentA, representing a stock market provider, will consider the provision of one of two services for calculating Abnormal Returns (AR):

1. *ARS*, where  $In_{ARS} = EODPrices$  and  $Out_{ARS} = AR$ , or
2. *DSAR*, where  $In_{DSAR} = StockSymbol$  and  $Out_{DSAR} = AR$

AgentA will then calculate  $q_{cd}^{ARS}(CompanyName, EODPrices) = 0.33$  whilst  $q_{cd}^{DSAR}(CompanyName, StockSymbol) = 0.12$ . Therefore AgentA will propose *DSAR* to the noticeboard, and will form the first partial solution  $StockSymbol \xrightarrow{DSAR} AR$ . Note that for brevity we have not enumerated the information items which form part of the start and end states yet have not been used in this step.

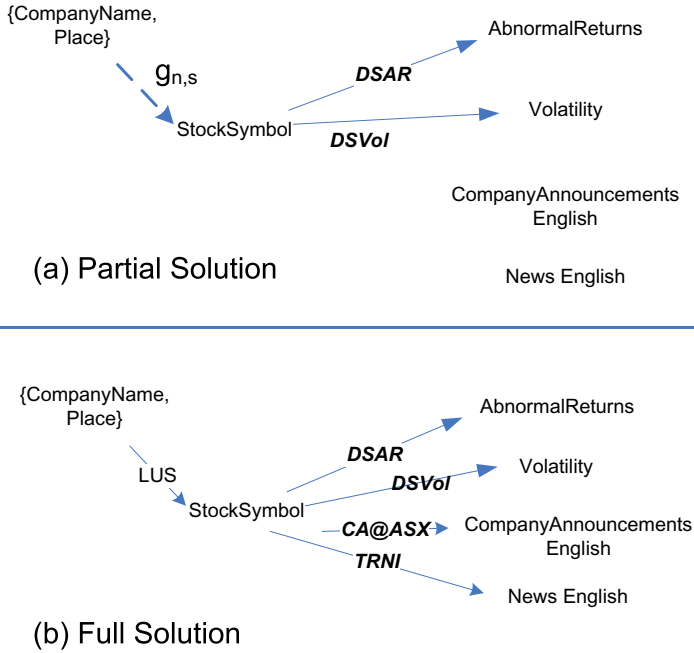
AgentA also has a service *DSVol* which calculates *Volatility* using *StockSymbol* as an input ( $In_{DSVol} = StockSymbol$  and  $Out_{DSVol} = Volatility$ ). This service is appropriate (matching inputs and outputs), and also it has no "internal competition" within AgentA. It will thus be submitted to the noticeboard as a bid for extending the current partial solution. However, the noticeboard has also received another overlapping bid from AgentB, where its service *TSBS* can take *EODPrices* and calculate *Volatility* ( $In_{TSBS} = EODPrices$  and  $Out_{TSBS} = Volatility$ ).

The consortium behind the existing partial solution will compare

1.  $q_{cd}^{DSVol}(CompanyName, StockSymbol) = 0.12$  and
2.  $q_{cd}^{TSBS}(CompanyName, EODPrices) = 0.33$

The choice to extend the partial solution will therefore be Option 1, ie. *DSVol*. We now have a new partial solution illustrated in Fig. 4(a).

AgentC will now see this partial solution where  $StockSymbol \in St_{input}$ , and since it has a stock lookup service *LUS*, where  $In_{LUS} = CompanyName$  and  $Out_{LUS} = StockSymbol$ , it offers it to the consortium behind the partial solution. This service is in a competition with FT Lookup Service *FTLUS* and Reuters LookUP Service *RLUS*, all three taking exactly the same parameters. The consortium will make the choice between these three, based on the overall quality of the partial solutions formed by each of these candidate services. Considering Sections 3.2 and 4.5, we have  $Q_{cd}$  and  $Q_m$  equal amongst all three candidate services, so the comparison will be based on differences in price and speed ( $Q_{pr}$  and  $Q_t$ ). The new service will be added to the existing partial solution in a sequence, therefore from Table 1 we have the formulae for  $Q_{pr} = \sum_{s_i} q_{pr}(s_i)$  and  $Q_t = \sum_{s_i} q_t(s_i)$ . The three candidate services can therefore be compared directly in terms of quality and price, and any trade-offs between them considered in terms of weighting factors of these two parameters.



**Fig. 4.** Two stages of the solution formation

For the purpose of this example we can presume that *LUS* has the best combination of quality and price, and that it will be added to the partial solution on the notice-board. The solution is still not complete, though, because we have no service providing *CompanyAnnouncementsEnglish* and *NewsEnglish*. *TRNI* uses *StockSymbol* to deliver *NewsEnglish*, so the consortium will prefer it to ReutersLocal where the news are not necessarily in English. Similarly, *CA@ASX* takes *StockSymbol* again, and delivers *CompanyAnnouncementsEnglish*, so it will be preferred to the combination of *RCA* and *GT* (GoogleTranslate) services.

We therefore obtain the full solution depicted in Figure 4(b). This solution will be proposed to the service requestor agent, any competing solutions will be evaluated using the composition quality metrics from Table 1.

A top-down solution to the same problem, driven by a centralised decomposition knowledge, would depend on the way in which this knowledge is structured, rather than on the ways in which the available services could fit best together as in the example above. For example, if the knowledge decomposes the problem along functional lines into a sub-solution for stock market information, a sub-solution for company announcements and a sub-solution for company news. If the centralised decomposition knowledge does not have a further breakdown indicating the use of a stock symbol lookup service for each of these sub-solutions, it will fail to find a solution within the currently available set of services. But even if such knowledge is available, the top-down approach will not be able to identify the synergy of having only one stock symbol lookup service, which feeds its output into all other information providing services.

## 7 Related Research

The role of service providers is normally not in the focus of researchers in service composition, apart from the work on quality assurances and monitoring. However, representing providers through software agents allows innovative solutions to the process of assembling robust and efficient composite services.

For example, having software agents allows us to use agent negotiation, an effective way of addressing the complex issues associated with automated service composition [22]. Negotiation processes range from simple one-shot interactions to handling counter proposals across different processes and facilitating agent-based coalition formation. The subject of coalition-formation is explored in [23] and agent-based coalition formation for service composition is discussed in [24].

There are previously proposed systems where agent-based service providers play an active role in service composition to achieve robust composition on a semantic level [25, 26]. Both papers consider fixed centralised knowledge about the composition structure, and employ agents to negotiate how to turn all semantic links into robust ones.

An application of agents to create emergent compositions of software services is considered in [27], a PhD thesis focused on using agents for emergent team formation in the domain of automotive manufacturing. The work points out some major issues in applying such ideas in the domain of software services, results which have inspired our work in addressing these issues using semantic models and composition mechanisms.

## 8 Discussion and Conclusion

Our decentralised approach to *cooperative service composition* is driven by service providers. It avoids the need for centrally available problem decomposition knowledge, instead relying on providers' knowledge of their services and on semantic measures of similarity to guide the composition process. It creates potential for innovative compositions, and is suitable for operation in an open domain, where the list of available services and service providers can change dynamically.

The approach is inherently less efficient than centralised top-down decomposition, and its computational complexity is potentially as high as brute-force planning approaches. However, we use semantic similarity metrics to limit the number of partial solutions considered and to reduce the time necessary for the approach.

At present, the approach is tuned to the domain of information processing, which lends itself to semantic reasoning with inputs and outputs of services. In the future we will endeavour to extend the approach to include reasoning with state information which is not reduced to availability of inputs and outputs.

The other directions of expanding our work are:

- We will explore reasoning with semantic information about domain structure rather than simply specialisation-based abduction as in the current paper.
- At present agents are only allowed to bid if the semantic links they introduce are exact match, in our future work we will relax this limitations to increase the variety of services eligible for inclusion in our partial solutions, and to exploit the concept of functional quality of semantic links(*c.f.* [6]) and their robustness.

- We will introduce a more general distance model which will allow contributions to gaps “in the middle” of a partial solution. The quality model is capable of covering this, yet at present we are only extending partial solution “at one end”.

In terms of strategies for reducing the numbers of partial solutions, the current restriction to only choosing the best candidate for a given extension may be sub-optimal in terms of the complete solutions thus generated. We will implement a simulator based on the approach so that we can explore the tradeoff between number of candidate solutions allowed, the overall quality of the solutions obtained and the time taken to obtain a solution. We will also explore the effectiveness of alternative control strategies for reducing the set of partial solutions.

## References

1. Bennett, K., Munro, M., Xu, J., Gold, N., Layzell, P., Mehandjiev, N., Budgen, D., Brereton, P.: Prototype implementations of an architectural model for service-based flexible software. In: Hawaii International Conference on System Sciences, vol. 3, p. 76b (2002)
2. Papazoglou, M.P., Traverso, P., Dustdar, S., Leymann, F.: Service-oriented computing: A research roadmap. *International Journal of Cooperative Information Systems* 17(2), 223–255 (2008)
3. Wu, D., Parsia, B., Sirin, E., Hendler, J., Nau, D.S.: Automating DAML-S Web Services Composition Using SHOP2. In: Fensel, D., Sycara, K., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, pp. 195–210. Springer, Heidelberg (2003)
4. Wielinga, B., Schreiber, G.: Configuration-design problem solving. *IEEE Expert: Intelligent Systems and Their Applications* 12(2), 49–56 (1997)
5. Motta, E.: Parametric Design Problem Solving - Reusable Components For Knowledge Modelling Case Studies. IOS Press (1999)
6. Lécué, F., Mehandjiev, N.: Seeking quality of web service composition in a semantic dimension. *IEEE Trans. Knowl. Data Eng.* 23(6), 942–959 (2011)
7. Erol, K., Hendler, J., Nau, D.S.: Htn planning: complexity and expressivity. In: Proceedings of the Twelfth National Conference on Artificial intelligence, AAAI 1994, vol. 2, pp. 1123–1128. American Association for Artificial Intelligence, Menlo Park (1994)
8. ten Teije, A., van Harmelen, F., Wielinga, B.: Configuration of Web Services as Parametric Design. In: Motta, E., Shadbolt, N.R., Stutt, A., Gibbins, N. (eds.) EKAW 2004. LNCS (LNAI), vol. 3257, pp. 321–336. Springer, Heidelberg (2004)
9. McIlraith, S.A., Son, T.C.: Adapting golog for composition of semantic web services. In: KR, pp. 482–496 (2002)
10. Sycara, K.P., Paolucci, M., Ankolekar, A., Srinivasan, N.: Automated discovery, interaction and composition of semantic web services. *J. Web Sem.* 1(1), 27–46 (2003)
11. Baader, F., Nutt, W.: *The Description Logic Handbook: Theory, Implementation, and Applications* (2003)
12. Ankolekar, A., Paolucci, M., Srinivasan, N., Sycara, K.: The OWL-S coalition, OWL-S 1.1. Technical report (2004)
13. Kopecký, J., Vitvar, T., Bournez, C., Farrell, J.: Sawsdl: Semantic annotations for WSDL and XML schema. *IEEE Internet Computing* 11(6), 60–67 (2007)
14. Euzenat, J.: Semantic precision and recall for ontology alignment evaluation. In: IJCAI, pp. 348–353 (2007)

15. Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.: Semantic Matching of Web Services Capabilities. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, pp. 333–347. Springer, Heidelberg (2002)
16. Li, L., Horrocks, I.: A software framework for matchmaking based on semantic web technology. In: WWW, pp. 331–339 (2003)
17. Noia, T.D., Sciascio, E.D., Donini, F.M., Mongiello, M.: Abductive matchmaking using description logics. In: IJCAI, pp. 337–342 (2003)
18. Lécué, F., Delteil, A., Léger, A.: Applying abduction in semantic web service composition. In: ICWS, pp. 94–101 (2007)
19. O’Sullivan, J., Edmond, D., ter Hofstede, A.H.M.: What’s in a service? Distributed and Parallel Databases 12(2/3), 117–133 (2002)
20. Küsters, R.: Non-Standard Inferences in Description Logics. LNCS (LNAI), vol. 2100. Springer, Heidelberg (2001)
21. Cardoso, J., Sheth, A.P., Miller, J.A., Arnold, J., Kochut, K.: Quality of service for workflows and web service processes. *J. Web Sem.* 1(3), 281–308 (2004)
22. Ben Hassine, A., Matsubara, S., Ishida, T.: A Constraint-Based Approach to Horizontal Web Service Composition. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 130–143. Springer, Heidelberg (2006)
23. Shehory, O., Kraus, S.: Methods for task allocation via agent coalition formation. *Artif. Intell.* 101(1-2), 165–200 (1998)
24. Muller, I., Kowalczyk, R., Braun, P.: Towards agent-based coalition formation for service composition. In: IAT 2006: Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology, pp. 73–80. IEEE Computer Society, Washington, DC (2006)
25. Lecue, F., Wajid, U., Mehandjiev, N.: Negotiating robustness in semantic web service composition. In: Seventh IEEE European Conference on Web Services, ECOWS 2009, pp. 75–84 (November 2009)
26. Mehandjiev, N., Lécué, F., Wajid, U.: Provider-Composer Negotiations for Semantic Robustness in Service Compositions. In: Baresi, L., Chi, C.-H., Suzuki, J. (eds.) ICSSOC-ServiceWave 2009. LNCS, vol. 5900, pp. 205–220. Springer, Heidelberg (2009)
27. Carpenter, M.: Cooperative team formation using distributed decomposition knowledge. PhD thesis, Manchester Business School (2010)

# Abstracting Modelling Languages: A Reutilization Approach

Juan de Lara, Esther Guerra, and Jesús Sánchez-Cuadrado

Universidad Autónoma de Madrid, Spain

{Juan.deLara, Esther.Guerra, Jesus.Sanchez.Cuadrado}@uam.es

**Abstract.** Model-Driven Engineering automates the development of information systems. This approach is based on the use of Domain-Specific Modelling Languages (DSMLs) for the description of the relevant aspects of the systems to be built. The increasing complexity of the target systems has raised the need for abstraction techniques able to produce simpler versions of the models, but retaining certain properties of interest. However, developing such abstractions for each DSML from scratch is a time and resource consuming activity.

Our solution to this situation is a number of techniques to build reusable abstractions that are defined once and can be reused over families of modelling languages sharing certain requirements. As a proof of concept, we present a catalogue of reusable abstractions, together with an implementation in the METADEPTH multi-level meta-modelling tool.

**Keywords:** Model-Driven Engineering, Domain-Specific Modelling Languages, Abstraction, Genericity.

## 1 Introduction

In Model-Driven Engineering (MDE), models are actively used to conduct the different phases of the project, being employed to specify, simulate, optimize, test and produce code for the final systems. Models are sometimes specified using general purpose modelling languages, like the UML, but in order to unfold the full potential of MDE, Domain-Specific Modelling Languages (DSMLs) are frequently used, specially tailored to specific areas of concern.

As the application domain becomes complex, models tend to become complex as well. This fact hinders the development, understanding and analysis of models. To tackle this issue, researchers have developed abstraction techniques to reduce model complexity for particular notations and purposes [1-4]. In our setting, an abstraction is a model operation that produces a simpler model that retains certain properties of interest from the original model.

Frequently, different notations can be abstracted following similar patterns. For example, a common abstraction consists on aggregating a linear sequence of connected elements, which can be applied to a class diagram to flatten an inheritance hierarchy, or to a process model to aggregate a sequence of activities.

Additionally, some abstractions are specific to particular domains. For example, the abstractions for Petri nets [1] produce nets easier to analyse, but which preserve liveness, safeness and boundedness. Similarly, there are abstractions specific to process modelling [2, 3] which produce more comprehensible models or facilitate their verification [4]. These abstractions are applicable (in theory) to a family of notations sharing semantics, like BPMN and Activity Diagrams.

In MDE, the abstract syntax of DSMLs is defined through a meta-model and the needed abstraction operations are defined over its elements. This enables the application of the abstractions to the instances of a specific meta-model, but not to the instances of other meta-models even if they share characteristics with the former one. Therefore, even though one can develop abstractions for the meta-model of a particular DSML, like the OMG's BPMN 2.0, these cannot be used for other related DSMLs, like YAWL [4], Activity Diagrams or other BPMN variants. A catalog of abstractions and mechanisms for their reuse across meta-models would save significant effort in defining advanced DSML environments.

In previous works [5], we developed a technique for the reutilization of model management operations. The basic idea is to introduce an extra level of indirection, so that operations are not defined over concrete meta-models, but over so-called *concepts* which gather the requirements that a meta-model should fulfil so that the operation becomes applicable. In our approach *concepts* have the form of meta-models as well, and operations defined over them become reusable, as concepts can be bound to families of meta-models satisfying their requirements.

In this work we present reusable abstraction techniques for modelling languages in the context of MDE. For this purpose, we first introduce a classification of abstractions for modelling languages, and then present a catalogue of generic, reusable abstractions, applicable to sets of meta-models. We consider four abstraction types: aggregation, merge, deletion and view generation. Orthogonally, each abstraction can be either *horizontal* if it is applicable to meta-models of different domains, or *domain-specific* if it is applicable to families of meta-models sharing semantics (e.g. languages similar to Petri nets or workflow languages). Our abstractions are: (a) reusable, as they can be applied to several modelling languages; (b) customizable, as some aspects can be configured, like similarity criteria or attribute aggregation operations; and (c) adaptable, as they provide extension mechanisms for meta-models when a language lacks support for encapsulation or aggregation. We have validated these ideas by an implementation in the METADEPTH tool [5], and several case studies (BPMN, Statecharts, DSMLs).

**Paper Organization.** Section 2 introduces a categorization of abstractions for modelling languages. Section 3 presents some techniques to define generic abstractions, which we use to define a catalogue of reusable abstractions in Section 4. Section 5 shows an implementation using METADEPTH. Section 6 compares with related research and Section 7 concludes.



## 2 Classifying Abstractions for Modelling Languages

In this section we present a categorization of abstractions for modelling languages. This classification has been built after a thorough analysis of the abstractions provided by or developed for languages of extended use like BPMN [3], as well as from our own experience on the construction of DSMLs.

In our setting, a model abstraction is an operation that reduces the complexity of some aspect of a model. In this way, the purpose of an abstraction can be to increase the comprehensibility of a large model, or to reduce the size of a model to ease its verification while retaining certain properties of interest, among others. An abstraction may imply the deletion of existing model elements, as well as the addition of new ones – like aggregate objects or hierarchical constructs – that encapsulate existing elements which share certain features. The addition of new elements may influence the way in which a model is visualized, e.g. enabling to zoom-into or to hide the aggregated objects, but the focus of this work is not on model visualization, which we touch only briefly in Section 5.

We distinguish two types of abstractions according to their applicability:

- *Horizontal abstractions*, applicable to modelling languages of (perhaps very) different domains. For example, an abstraction that encapsulates a sequence of model elements can be used to flatten a linear inheritance hierarchy in a class diagram, or to simplify a business process model by creating a subprocess that groups a sequence of consecutive activities.
- *Domain-specific abstractions*, specific to a family of DSMLs sharing semantics. Examples of this kind of abstractions include the reduction techniques for Petri nets [1] and the ones for workflows [4]. Being domain-specific, they can take into account the semantics of the languages in the domain and ensure the preservation of properties, which permits their use for verification purposes. For example, the reduction techniques in [1] result in simpler models that preserve liveness, safeness and boundedness.

Orthogonally, we identify four abstraction types according to their behaviour:

- *Merge*. In this kind of abstraction, a set of model elements considered similar is replaced by one element of the same type which collects the properties of the merged elements. A frequent use of this abstraction is for verification. E.g. the reduction rules for Petri nets [1] merge groups of places or transitions into a unique place or transition collecting the arcs of the merged elements. In some cases, the element replacing the group is assigned property values that result from a calculation using the properties of the merged elements.
- *Aggregation*. In this case, a set of similar model elements is grouped hierarchically under a higher-level element, normally of a different type, which serves as an aggregate. An example is the encapsulation of a linear sequence of activities in a process model into a subprocess, obtaining a more comprehensible, hierarchical model. Properties of the aggregate object may be calculated using properties of the aggregated objects.

- *Deletion*. This kind of abstraction deletes elements that are not considered relevant or that do not modify some observed property of a model. An example is the elimination of self-loops in Petri nets [1] and workflow nets [4].
- *Views*. In this case, the abstraction produces a new model (called *view*) which may be expressed using the same language as the original model or a different one. The view discards those features of the original model which are irrelevant for the aim of the abstraction. This is the most general type of abstraction as it includes the previous ones, whenever the languages of the original model and the view are the same. There are many examples of this type of abstraction for verification, where the view uses a language with a rich body of theoretical results enabling the analysis of the source model. Transforming BPMN models into Petri nets is an example of this abstraction

Once we have seen the different model abstraction types, next section discusses a possible approach to make abstractions reusable for several DSMLs.

### 3 Making Abstractions Generic

MDE is a type-centric approach because model manipulations use the types of a concrete meta-model, becoming hard to reuse for other meta-models. In [5] we introduced some mechanisms to define reusable model management operations, which we review and apply to define generic model abstractions in the following.

Assume we need an operation to simplify a business process model by creating an aggregate object abstracting the flow elements between two complementary gateways (e.g. a parallel fork and join). Currently, there is a plethora of different business process modelling notations like Activity diagrams, YAWL, the OMG's BPMN 2.0, and different variations of BPMN, like the Intalio's BPM modeller<sup>1</sup>. Hence, one needs to select a particular meta-model and implement the operation for it, so that the operation is not applicable for the other meta-models anymore.

To overcome this limitation, we propose building a *generic* model abstraction operation that can be applied to a family of meta-models. For this purpose, we do not build the operation over the specific types of a meta-model, but over the variable types defined in a so-called *structural concept* which gathers the requirements that a meta-model should fulfil to qualify for the operation. Concepts have the form of a meta-model, but their elements (classes, attributes and references) are interpreted as variables. A generic operation is used by *binding* its associated concept to a meta-model, as shown in Fig. 1. The binding defines a 1-to-1 mapping from each class, attribute and reference in the concept to a class, attribute and reference in the meta-model, respectively. It needs to follow some well-formedness rules (see [5]) to ensure a correct application of the generic operation. For example, if a class  $c$  in the concept is bound to a class  $c'$  in the meta-model, the attributes and references in  $c$  must be bound to features defined in  $c'$  or in some superclass of  $c'$ . We can map two classes  $c$  and  $d$  in the concept to a unique class in the meta-model provided it contains all features demanded by

<sup>1</sup> <http://www.intalio.com/bpms/designer>

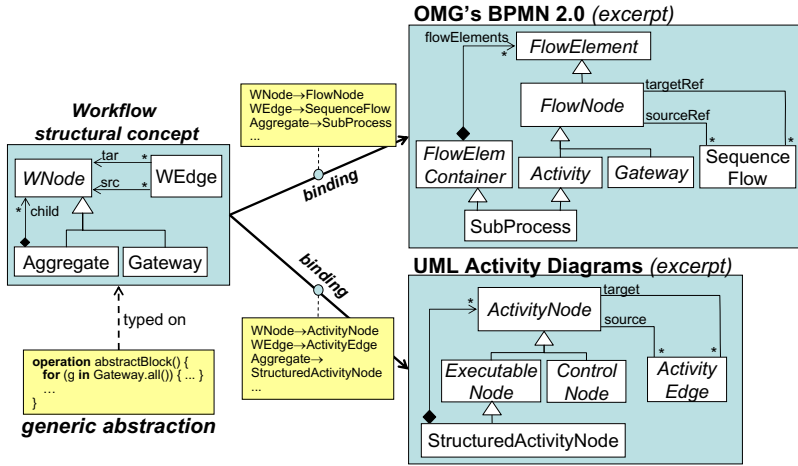


Fig. 1. Generic model abstraction operation through a structural concept

*c* and *d*. Once defined, the binding induces a re-typing of the generic operation, which becomes applicable to the instances of the bound meta-models, obtaining reusability (the *same* operation becomes applicable to *several* meta-models).

As an example, Fig. 1 defines the generic abstraction `abstractBlock` using the types of the concept *Workflow*. The operation creates an aggregate object abstracting the flow elements between two complementary gateways. The concept gathers the variable types used by the operation, like *Gateway* and *Aggregate*, but note that it does not include a class representing a task (even though workflow languages usually include such a class) because the operation does not need to distinguish any flow node other than gateways. In this way, the concept and its bindings are kept as simple as possible. Then, we can bind the concept to different meta-models, like those of the OMG’s BPMN 2.0 and the UML 2.0 Activity Diagrams, enabling the application of the abstraction operation to the model instances of these notations. The figure shows an excerpt of the binding for both meta-models. For BPMN, *WNode* is bound to *FlowNode* and *WEdge* to *SequenceFlow*. The bindings permit certain heterogeneity in the subtyping, as *Aggregate* is a direct subtype of *WNode* in the concept, but *SubProcess* is an indirect subtype of *FlowNode* in the BPMN meta-model.

### 3.1 Configuration and Adaptation to the Modelling Language

A *structural concept* has the form of a meta-model and reflects a design decision that some meta-models could implement differently. For example, the Intalio’s BPMN meta-model represents all kinds of flow nodes through a unique class *Activity* with an attribute to discriminate the type. As a consequence, we cannot bind the previous structural concept *Workflow* to this meta-model. Our solution to widen the range of boundable meta-models for a generic operation is

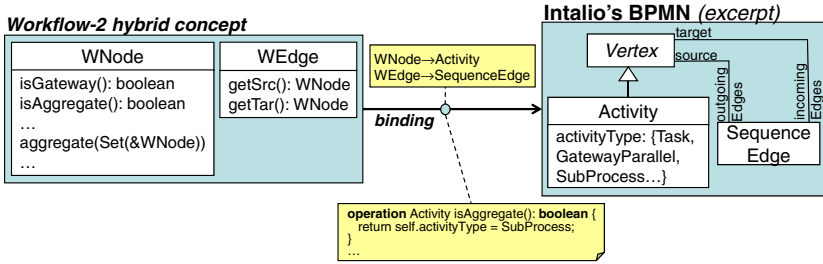


Fig. 2. Binding to Intalio’s BPMN meta-model with a hybrid concept

to use so-called *hybrid concepts* [5], which abstract away the accidental details introduced by the specific choice of structure in the concept behind a suitable interface. Thus, *hybrid concepts* are like structural ones but require a number of operations from the classes they define. The binding is then obtained by mapping the elements and implementing the operations declared in the concept.

Fig. 2 shows the definition of the hybrid concept *Workflow-2*, which is a more flexible version of the structural concept presented in Fig. 1, as it imposes less structural requirements. This concept can be bound to the Intalio’s BPMN meta-model, which requires implementing the operations required by the concept. For instance, the figure shows the implementation of operation `isAggregate`, which in this case returns true whenever the attribute `activityType` of an activity takes the value `SubProcess`.

Finally, sometimes an abstraction needs to be configured with similarity criteria or with a function over attribute values. These configuration points can be expressed through *hook* operations in a hybrid concept as well. Hook operations provide a default implementation, and only need to be implemented during the binding if a special behaviour is required. For instance, in Fig. 2 operation `aggregate` in class `WNode` is a hook to customize an aggregation operation on attributes (e.g. adding up a time attribute in all aggregated nodes).

### 3.2 Extending the Modelling Language

Some abstraction operations may create an aggregate object grouping similar model elements. However, some notations were not designed with hierarchical or encapsulation elements, and therefore we cannot apply these abstractions to them. To overcome this limitation, our solution is to define a so-called *mixin* layer, which is a parameterized meta-model that contains those auxiliary elements needed by an operation [5]. Then, we use concepts to express the requirements that a meta-model should fulfil to become extendible by the mixin.

As an example, the mixin in Fig. 3 adds an aggregate class to any meta-model boundable to the *Workflow-3* concept, which in this case does not demand the existence of such a class. The generic operation is defined over the gluing of the mixin and the concept (label 2 in the figure) through the *extension points* or parameters of the mixin (label 1). In the figure, class `WNode` is the only parameter

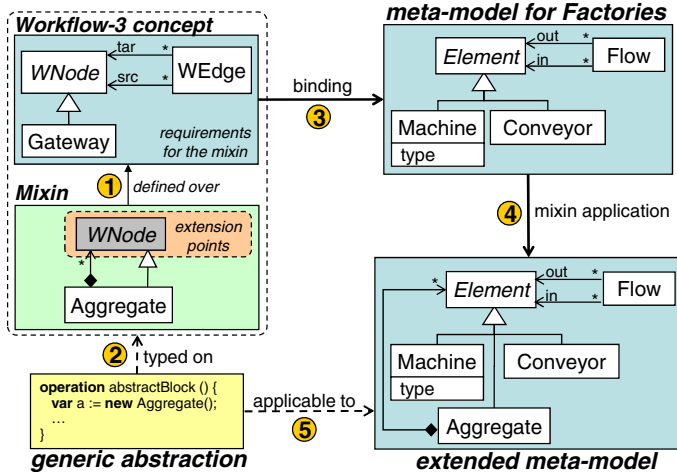


Fig. 3. Generic model abstraction operations through a mixin

of the mixin. The rest of elements of the mixin will be added to any meta-model to which we can bind the concept. In this way, we can bind the concept to meta-models like Petri nets, or to DSMLs to represent plant factories like the one shown in the figure (label 3). Applying the mixin to the DSML (label 4) adds the aggregate class to its meta-model, and hence the abstraction becomes applicable to their instances. Moreover, this kind of mixin preserves the compatibility of models conformant to the old meta-model with the extended one.

Now that we have seen the different ways to make abstractions generic, customizable and adaptable, we provide a catalogue of reusable abstractions, defined using the presented techniques.

## 4 A Catalogue of Generic Abstractions

One of the aims of the present work is to make easier the incorporation of abstraction capabilities and operations to existing or new modelling languages, so that these operations do not need to be developed from scratch. For this purpose, based on the techniques presented in the previous section, we have built a catalogue of generic abstractions that a designer can select and customise for his particular modelling language. Our abstractions are generalizations of the domain-specific ones we have found in the literature, and can easily be customised for a particular modelling language by identifying the element types participating in the abstraction. Additionally, some of them permit configuring additional details through the use of hook methods (e.g. the similarity criteria used to discriminate which elements of a model should be abstracted), or present variations in their semantics (e.g. whether a set of elements should be either encapsulated in an aggregate object or merged).

Technically, our generic abstractions are operations defined over suitable concepts to be bound to the meta-models of specific modelling languages. To increase the reuse opportunities of each abstraction type, we provide different binding possibilities for them: (i) from a structural concept, which is the simplest approach when the structures of the concept and the meta-model are similar, (ii) from a hybrid concept, which gives freedom regarding the particular structure of the bound meta-model, and (iii) from a concept associated to a mixin in case the bound meta-model has no support for abstractions, e.g. it lacks a class to represent object aggregations, in which case the mixin extends the meta-model with such a class, enabling the abstraction application. Internally, for each abstraction type we have defined a binding from its hybrid to its structural concept, which permits encoding the abstraction operation just once over the hybrid concept and reusing it for the structural one. In any case, this is transparent to the reuser of the abstraction. Finally, we provide two implementations for each abstraction operation: (i) one performing the abstraction on the input model in-place, and (ii) another one generating a view (i.e. a different model) with the result.

In the remaining of this section we present our catalogue of abstractions, classified depending on their applicability: horizontal (i.e. general) and domain-specific. We do not claim that this catalogue is complete, as we expect to add new abstractions in the future. Nonetheless, we will show some examples illustrating that our current catalogue can be used to customize meaningful abstractions for well-known notations.

## 4.1 Horizontal Abstractions

**Target parallel.** It abstracts a maximal set of objects that refer to the same target objects. There are two variants of this abstraction: *aggregate* and *merge*. The *aggregate* variant creates an aggregate that encapsulates all objects with same target, whereas the *merge* variant replaces the parallel objects by another one of the same type. In both cases, all original references are substituted by non-duplicate references from the created abstraction.

To use this abstraction, a binding has to be provided from the associated concept (either structural, hybrid or mixin-based if the target meta-model does not define an aggregate class) to a meta-model. Fig. 4 shows to the left the structural concept for the *aggregate* variant of this abstraction. Hence, the abstraction can be reused for a particular notation by specifying the type of the objects to abstract (class `Item`), the referenced target (class `Context` and reference `target`), and the aggregation elements (`Aggregate` and `child`). The concept includes a hook method `canAggregate` to configure extra conditions that the abstracted elements need to fulfill. The right of the same figure shows the working scheme of this abstraction, where three objects are aggregated as all refer to the same target objects (which in this case is only one) and their references to the target object are substituted by a unique reference from the created aggregate. Later, a particular tool may decide whether showing or not the objects inside the aggregate.

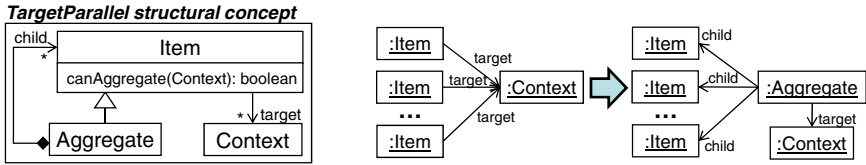


Fig. 4. Target parallel aggregation abstraction: concept (left) and behaviour (right)

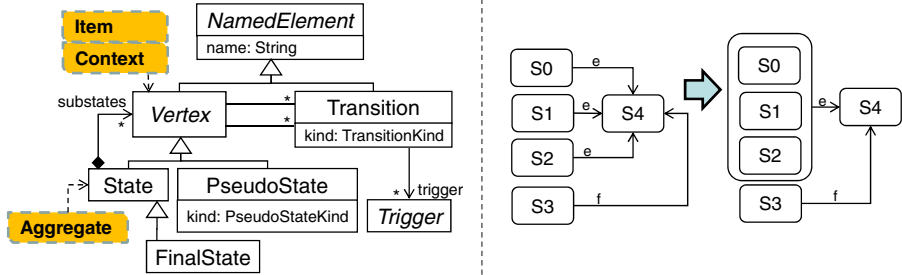
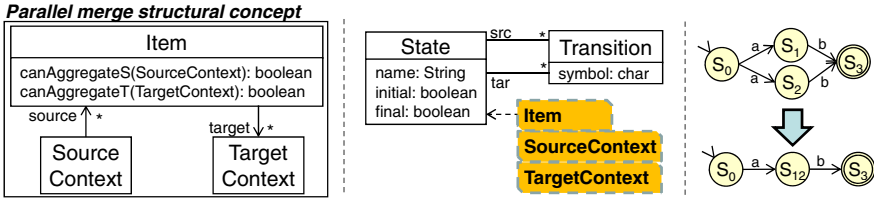


Fig. 5. Target parallel aggregation abstraction: binding to Statecharts (left) and application (right)

Fig. 5 shows an application of this abstraction over UML 2.0 Statecharts, to encapsulate sets of states that reach a same state through a same trigger (i.e. it performs an unflattening of statecharts). The left of the figure shows an excerpt of the UML Statecharts meta-model (simplified as it does not include *Regions*). We have depicted the binding through annotations. Hence, *Item* and *Context* in the concept are both bound to *Vertex* in the meta-model as we want to detect states connected to the same states. We have parameterized the abstraction by overriding operation *canAggregate* to include only target states with the same *Trigger*. In fact, we have used the hybrid version of the concept for this abstraction, as states are not connected to states through a reference (as demanded by the structural concept) but through an intermediate class *Transition*. The right of Fig. 5 shows the in-place application of this unflattening to a Statechart, which abstracts all states with a transition triggered by “e” to *S4*.

**Source parallel.** It abstracts a maximal set of objects which is referenced from the same source objects. Thus, this abstraction is like the previous one, but considers input parallel objects instead of target ones. The concept to bind in this case is therefore similar to the target parallel one, but there is a reference *source* from class *Context* to *Item* instead of the reference *target*.

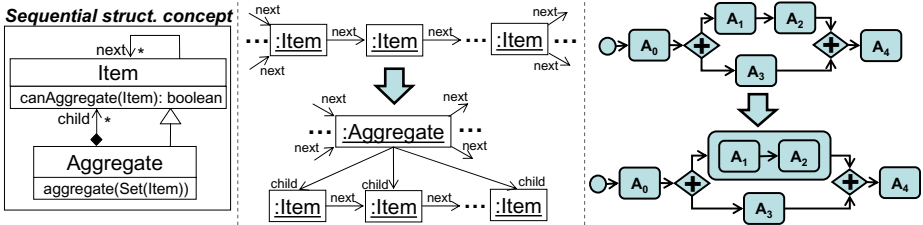
**Parallel.** It abstracts a maximal set of objects with the same source and target elements. The concept for this abstraction is shown to the left of Fig. 6 in its variant *merge* (i.e. the abstracted items are substituted by another item). There are also structural and hybrid versions of this concept for the *aggregate* variant, which include a class *Aggregate*. The middle of the figure shows an application of this abstraction to the meta-model of (non-deterministic)



**Fig. 6.** Parallel merge abstraction: concept (left), binding to non-deterministic automata (middle), and application (right)

automata. The aim is simplifying automata by merging sets of equivalent states, which are those that can reach, and can be reached from, the same states through transitions with the same symbols. In such a case, we can merge the equivalent states and obtain a simpler automaton which preserves behaviour, as the figure illustrates to the right. For this purpose, we bind all classes in the concept to class `State`, and override the `canAggregateS` and `canAggregateT` hook methods to check that transitions to/from the same states have the same symbol. As in the previous example, we have used the hybrid version of the concept because states are inter-connected through the intermediate class `Transition`, not directly through a reference.

**Sequential.** It abstracts a maximal sequence of linearly connected objects, and admits both variants *merge* and *aggregate*. The left of Fig. 7 shows the structural concept for the sequential *aggregate* variant, and the center of the figure illustrates how the abstraction works: it creates an aggregate for a sequence of items, and copies the connections of the first and last item of the sequence to the aggregate. The figure shows to the right an application to BPMN 2.0, where we have mapped `Item` to `FlowNode` and `Aggregate` to `SubProcess` (see Fig. 1). Moreover, we have customized the `canAggregate` hook method to forbid having gateways as first or last element in the abstracted sequence.



**Fig. 7.** Sequential aggregation abstraction: concept (left), behaviour (middle) and application to BPMN (right)

**Similarity.** It abstracts a maximal set of objects considered similar by a custom-defined operation. The left of Fig. 8 shows the structural concept for the *aggregate* variant of this abstraction. The right of the figure shows an example application to BPMN 2.0 models. As in the abstraction heuristic presented



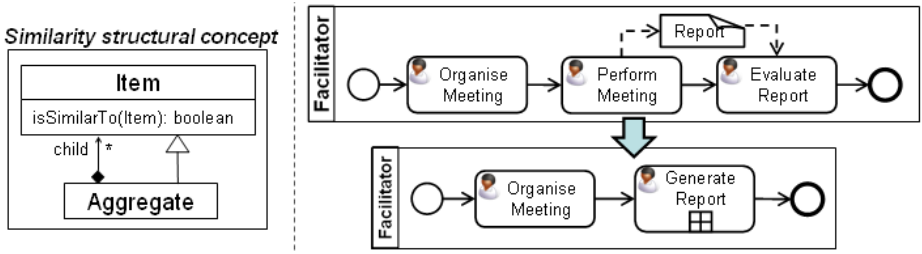


Fig. 8. Similarity aggregation abstraction: concept (left), application to BPMN (right)

in [3], tasks are considered similar if they produce or consume the same sets of Data Objects and use the same resources (in the model example, they are performed by the same HumanPerformer). Thus, we configure this abstraction by binding class *Item* in the concept to *FlowNode*, *Aggregate* to *SubProcess*, and implementing the *isSimilarTo* hook method.

**Loop removal.** It removes loops between two types of objects. The associated concept is shown in Fig. 9(a). We consider two variants: the first removes one of the references (*source* or *target*) to break the loop, whereas the other one removes the object with type *Item* and its references. The second variant is illustrated in Fig. 9(b). Fig. 9(c) shows an application of this abstraction that simplifies Petri nets, while preserving their behaviour, through the *elimination of self-loop places* [1]. Thus, a place is removed if it has at least one token and has no further input or output arcs. Fig. 9(d) shows an application to YAWL nets [4] that eliminates self-loop tasks. A different binding into YAWL would enable the elimination of self-loop conditions [4].

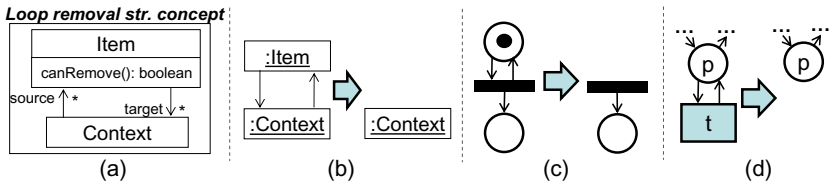


Fig. 9. Loop removal abstraction: concept (a), behaviour (b), application to Petri nets (c) and application to YAWL nets (d)

### 4.2 Domain-Specific Abstractions

Oftentimes, abstractions are specifically designed for a particular domain, such as Petri nets [1] or process models [3, 4]. Defining them over concepts makes them meta-model independent, and therefore reusable for other languages with similar semantics. For example, Fig. 10 shows the concept *Workflow* for process modelling languages, which expresses the requirements for the *block* abstraction operation. This abstraction aggregates into a subprocess a block delimited by two complementary gateways. An example application of this abstraction to BPMN is shown in Fig. 10.

If an abstraction for a particular domain is complex, we can certainly build it by reusing and customizing horizontal abstractions (like the ones in our catalogue). However, then it is sensible to specialize it as a domain-specific abstraction over a concept that reflects the particularities of the domain and can be bound to languages with close semantics without much configuration effort. For instance, once we have built the similarity abstraction that aggregates activities sharing data and resources (cf. Fig. 8), we can specialize it for their reuse for any workflow language. For this purpose, we would redefine the abstraction operation over a concept like the one in Fig. 11 but extended with tasks, resources and data objects. This concept (and abstraction) becomes easily reusable for families of process languages, like BPMN and Activity Diagrams.

## 5 Tool Support

The approach and abstractions presented in this paper have been implemented and validated using METADEPTH, a meta-modelling framework which supports multi-level meta-modelling and textual modelling [5]. It is integrated with the Epsilon family of languages [2], which permits defining in-place model manipulations, model-to-model transformations and code generators. All these operations can be made generic by their definition over (structural or hybrid) concepts or mixins. Moreover, we have extended METADEPTH's genericity support with the possibility of defining hook methods with default implementations in concepts.

Concepts, mixins and bindings are specified textually in METADEPTH. As an example, Listing 1 shows the definition of the structural concept shown in Fig. 11. The definition of a concept is similar to a meta-model definition, but its elements are considered variables and their definition is preceded by "&". The concept has a list of parameters (lines 2–3) in order to ease the binding to meta-models, as we will see later.

```

concept Workflow
  (&M, &WNode, &WEdge, &Aggregate,
   &Gateway, &child, &src, &tar) {
  Model &M {
    abstract Node &WNode {}
    Node &WEdge {
      &src : &WNode;
      &tar : &WNode;
    }
    Node &Aggregate : &WNode {
      &child : &WNode[*];
    }
    Node &Gateway : &WNode {}
  }
}

```

**Listing 1.** Structural concept

```

operation blockAbstraction() : Boolean {
  var comp : &WNode = null;
  for (gw in &WNode.allInstances())
    if (gw.isSplit()) {
      comp := getJoin(gw);
      if (comp<>null) {
        var sq : Sequence(&WNode);
        sq.addAll(getAllInBetween(gw, comp));
        createAggregateFor(sq, gw, comp);
        return true;
      }
    }
  return false;
}
...

```

**Listing 2.** Block abstraction operation

Once the concept is defined, we can build operations that use the variable types defined in the concept. In METADEPTH, the abstraction operations that modify the models in-place are defined using the Epsilon Object Language

<sup>2</sup> <http://eclipse.org/gmt/epsilon/>

(EOL), whereas the *view*-generating abstraction operations are defined with the Epsilon Transformation Language (ETL). Listing 2 shows an excerpt of the *block* abstraction operation using EOL (auxiliary operations and methods are not shown), which uses the types of the *Workflow* concept of Listing 1.

Finally, in order to apply the generic operation, a binding from the concept to a specific meta-model needs to be specified. Listing 3 shows the binding of the concept in Listing 1 to the meta-model of BPMN2.0. In this case, the *SubProcess* type of BPMN acts as aggregate (fourth parameter of the binding, corresponding to the *&Aggregate* variable of the concept).

```
bind Workflow ( BPMN, BPMN::FlowNode, BPMN::SequenceFlow,
               BPMN::SubProcess, BPMN::Gateway, BPMN::FlowElementsContainer::flowElements,
               BPMN::SequenceFlow::sourceRef, BPMN::SequenceFlow::targetRef )
```

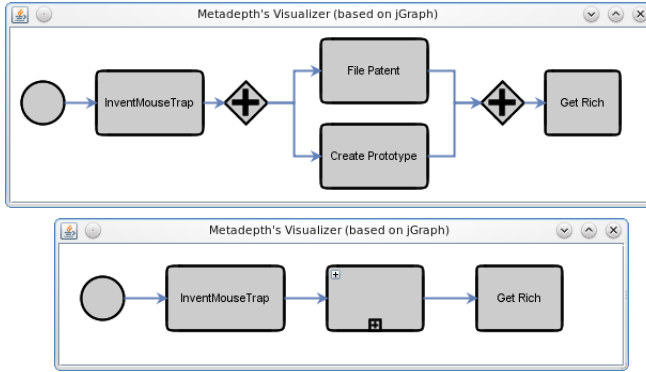
**Listing 3.** Binding to meta-model

The user of a generic abstraction only has to define the binding from the concept to his modelling language. The binding in Listing 3 allows for the reuse of the *block* abstraction operation, which consists of 200 lines of EOL code. Thus, the user has to specify fewer lines of code (3) to benefit from a proven abstraction operation. Actually, the real definition of our *Workflow* concept is slightly larger and has associated further domain-specific abstractions, so that the reuse opportunities are larger. Moreover, the user of the abstraction is not confronted with understanding the code being reused and a subsequent manual adaptation of the operation for a particular meta-model, which is error-prone. Instead, he only deals with the operation “interface” (the concept), and so the different elements in the concept become similar to roles in design patterns.

So far we have considered abstractions at the abstract syntax level. However, modelling languages have a concrete syntax, typically a graphical one. When an abstraction is applied, the visualization has to be updated accordingly. As a proof of concept, we have built a visualization engine for METADEPTH models with support for grouping together elements into aggregate objects. We have created a meta-model to define graphical concrete syntaxes which includes the notion of grouping. Hence, in order to define the visualization of a language, a simple mapping from its meta-model to our graphical syntax meta-model must be given. For those elements added by a mixin to a meta-model (and which therefore are not defined in the meta-model) we use a generic visualization. Then, our engine interprets concrete syntax models and renders their visualization with the jGraph library. Fig. 10 shows the application of the *block* abstraction to a BPMN model.

## 6 Related Work

Abstraction has been recognized as one of the key techniques for the model-based engineering of software and systems in the field of multi-paradigm modelling [6]. However, to our knowledge, there are no works aimed at systematizing and providing a framework to reuse abstractions for DSMLs in MDE.



**Fig. 10.** Example of a BPMN model, before and after applying an abstraction

Two kinds of models are distinguished in [7]: type and token models. The latter capture singular elements of a system (e.g. a road map is a token model), and the former capture universal aspects of it through classification (e.g. a meta-model is a type model). While a type model can be seen as an abstraction of the set of its valid instance models, there is a lack of proposals – like the one we give here – for systematic abstraction of token models.

Model slicing has been proposed as a model comprehension technique inspired by program slicing. It involves extracting a subset of a model, called a slice, which retains some properties of interest. Slicers are typically bound to concrete meta-models, for instance UML [10]. This technique can be seen as a particular case of our view abstraction, when the obtained view conforms to the original meta-model. In [11], a language to define and generate model slicers is proposed, but the obtained slicers are not reusable for different meta-models in MDE.

There are also works that aim at the description of generic model refactorings [12]. Although they do not explicitly deal with abstractions, they could be used to abstract models. However, they lack constructs like mixins or hybrid concepts which we use to broaden the applicability of abstractions. Similarly, [13] describes a comprehensive set of *change patterns* for process models for the purpose of comparing the change frameworks provided by process-support technologies. Some of the described patterns (e.g. Extract Sub Process) can be interpreted as abstractions. Our work is especially directed to abstractions, and is not tied to process modelling languages, being generic. While the goal of [13] is to provide a systematic comparison framework, our goal is to offer automatic support for abstracting DSMLs in MDE. This is achieved through a catalogue of abstractions, defined using concepts, which are reusable by means of bindings.

Most abstraction techniques are specifically designed for particular notations. For example, there are techniques tailored for abstracting process models in BPMN [3]. Abstractions are also heavily used in verification [1] to obtain simpler models. However, their implementation is frequently tied to the specificities of a language, and is hardly reusable even for similar notations.

Abstraction has also been studied theoretically. Hobbs [14] suggests that, in the course of reasoning, we conceptualize the world at different levels of granularity. He defines the notion of indistinguishability that allows us to map a complex theory of the world to a simpler theory for a particular context. This work has been used as a foundation to build theories about abstraction. For example, in [15], an abstraction is defined as a surjective, computable function between two first-order languages that preserves semantics. *Granularity* abstractions are defined as those non-injective mappings collapsing several symbols into a unique, abstracted one. Abstraction has also been used in many areas of Artificial Intelligence and Logic [16], e.g. to ease automated deduction. Keet [17] uses abstraction to help the comprehension of ontologies. In [18], granularity abstraction is applied to natural language processing. Natural language is represented as logical forms that are mapped to coarse-grained forms to enable sentence analysis. Kascheck [19] develops a theory of abstraction for information systems introducing *cohesion* predicates ( $m$ -ary relations) and abstractions of these (consistent  $n$ -ary relations, with  $n < m$ ).

Henderson-Sellers and González-Pérez have explored these theories of abstraction and granularity for conceptual modelling [8, 9]. For example, in [8], they consider granularity for whole/part, generalization and instantiation relations, and develop best-practices when adopting a meta-model for method engineering.

The field of information and diagram visualization also makes use of abstraction techniques. For example, in [20], the authors develop an ad-hoc semantic-zooming technique to ease the navigation in complex UML diagrams, and some visual language editors like DIAGEN enable the definition of abstractions [21]. However, the only purpose of these abstractions is visualization (i.e. they do not change the underlying model), they have to be manually programmed and are not reusable across different languages.

Altogether, even though abstraction has been studied in many different fields, our work is the first one proposing mechanisms for the development of reusable abstractions for modelling languages, in the context of MDE.

## 7 Conclusions and Future Work

In this paper, we have used generic techniques to define reusable abstractions, applicable to several modelling languages in a meta-model independent way. We have presented a catalogue of horizontal abstractions and domain-specific ones for process modelling languages. We have implemented the approach in the METADEPTH tool, which provides support for visualization of aggregate objects.

As future work, we plan to improve METADEPTH's support for genericity, e.g. making the binding more flexible. We also plan to explore the use of these techniques to define generic model slicers, to extend our catalogue of abstractions, and to provide built-in support for similarity techniques (e.g., Formal Concept Analysis) and for detecting non-confluent abstraction applications.

**Acknowledgements.** Work funded by the Spanish Ministry of Economy and Competitivity (TIN2011-24139), and the R&D programme of Madrid Region (S2009/TIC-1650). We thank the referees for their valuable comments.

## References

1. Murata, T.: Petri nets: Properties, analysis and applications. *Proceedings of the IEEE* 77(4), 541–580 (1989)
2. Polyvyanyy, A., Smirnov, S., Weske, M.: The Triconnected Abstraction of Process Models. In: Dayal, U., Eder, J., Koehler, J., Reijers, H.A. (eds.) *BPM 2009*. LNCS, vol. 5701, pp. 229–244. Springer, Heidelberg (2009)
3. Smirnov, S., Reijers, H.A., Weske, M.: A Semantic Approach for Business Process Model Abstraction. In: Mouratidis, H., Rolland, C. (eds.) *CAiSE 2011*. LNCS, vol. 6741, pp. 497–511. Springer, Heidelberg (2011)
4. Wynn, M.T., Verbeek, H.M.W., van der Aalst, W.M.P., ter Hofstede, A.H.M., Edmond, D.: Reduction rules for YAWL workflows with cancellation regions and or-joins. *Inf. & Softw. Techn.* 51(6), 1010–1020 (2009)
5. de Lara, J., Guerra, E.: From types to type requirements: Genericity for model-driven engineering. In: *SoSyM (2011)* (in press)
6. Mosterman, P.J., Vangheluwe, H.: Computer automated multi-paradigm modeling: An introduction. *Simulation* 80(9), 433–450 (2004)
7. Kühne, T.: Matters of (meta-)modeling. *SoSyM* 5(4), 369–385 (2006)
8. Henderson-Sellers, B., Gonzalez-Perez, C.: Granularity in Conceptual Modelling: Application to Metamodels. In: Parsons, J., Saeki, M., Shoval, P., Woo, C., Wand, Y. (eds.) *ER 2010*. LNCS, vol. 6412, pp. 219–232. Springer, Heidelberg (2010)
9. Henderson-Sellers, B.: Random Thoughts on Multi-level Conceptual Modelling. In: Kaschek, R., Delcambre, L. (eds.) *The Evolution of Conceptual Modeling*. LNCS, vol. 6520, pp. 93–116. Springer, Heidelberg (2011)
10. Lano, K., Kolahdouz, S.: Slicing techniques for UML models. *Journal of Object Technology* 10, 11:1–11:49 (2011)
11. Blouin, A., Combemale, B., Baudry, B., Beaudoux, O.: Modeling Model Slicers. In: Whittle, J., Clark, T., Kühne, T. (eds.) *MODELS 2011*. LNCS, vol. 6981, pp. 62–76. Springer, Heidelberg (2011)
12. Moha, N., Mahé, V., Barais, O., Jézéquel, J.M.: Generic Model Refactorings. In: Schürr, A., Selic, B. (eds.) *MODELS 2009*. LNCS, vol. 5795, pp. 628–643. Springer, Heidelberg (2009)
13. Weber, B., Rinderle, S., Reichert, M.: Change Patterns and Change Support Features in Process-Aware Information Systems. In: Krogstie, J., Opdahl, A.L., Sindre, G. (eds.) *CAiSE 2007*. LNCS, vol. 4495, pp. 574–588. Springer, Heidelberg (2007)
14. Hobbs, J.R.: Granularity. In: *IJCAI 1985*, pp. 432–435. M. Kaufmann (1985)
15. Ghidini, C., Giunchiglia, F.: A semantics for abstraction. In: *ECAI*, pp. 343–347. IOS Press (2004)
16. Giunchiglia, F., Walsh, T.: A theory of abstraction. *Artif. Intell.* 57(2-3), 323–389 (1992)
17. Keet, C.M.: Enhancing Comprehension of Ontologies and Conceptual Models Through Abstractions. In: Basili, R., Paziienza, M.T. (eds.) *AI\*IA 2007*. LNCS (LNAI), vol. 4733, pp. 813–821. Springer, Heidelberg (2007)

18. Mani, I.: A theory of granularity and its application to problems of polysemy and underspecification of meaning. In: KR 1998, pp. 245–255. M. Kaufmann (1998)
19. Kaschek, R.: A little theory of abstraction. In: Modellierung. LNI, vol. 45, pp. 75–92. GI (2004)
20. Frisch, M., Dachselt, R., Brückmann, T.: Towards seamless semantic zooming techniques for UML diagrams. In: SOFTVIS, pp. 207–208. ACM (2008)
21. Köth, O., Minas, M.: Structure, Abstraction, and Direct Manipulation in Diagram Editors. In: Hegarty, M., Meyer, B., Narayanan, N.H. (eds.) Diagrams 2002. LNCS (LNAI), vol. 2317, pp. 290–304. Springer, Heidelberg (2002)

# Logical Invalidations of Semantic Annotations

Julius Köpke and Johann Eder

Department of Informatics-Systems, Alpen-Adria Universität Klagenfurt, Austria

`firstname.lastname@aau.at`

`http://www.aau.at/isys`

**Abstract.** Semantic annotations describe the semantics of artifacts like documents, web-pages, schemas, or web-services with concepts of a reference ontology. Application interoperability, semantic query processing, semantic web services, etc. rely on a such a description of the semantics. Semantic annotations need to be created and maintained. We present a technique to detect logical errors in semantic annotations and provide information for their repair. In semantically rich ontology formalisms such as OWL-DL the identification of the cause of logical errors can be a complex task. We analyze how the underlying annotation method influences the types of invalidations and propose efficient algorithms to detect, localize and explain different types of logical invalidations in annotations.

**Keywords:** Semantic annotation, ontology evolution, annotation maintenance, logical invalidation.

## 1 Introduction

Semantic annotations are used to attach semantics to various kinds of artifacts like documents, web-pages, schemas, or web-services. Semantic annotations are used in information systems engineering in various ways: To enable semantic interoperability of information systems, to generate transformations of documents between heterogenous information systems, to support correct schema integration, selection and composition of web services, etc. - see e.g. [10,15,16]. In this paper we focus on annotations on the schema level rather than on the instance level (e.g. RDF [9]). Schema level annotations are better suited for high volume data because all documents that are instances of an annotated schema can be interpreted with the reference ontology. XML-Schemas can be annotated according to the W3C Recommendation Semantic Annotations for WSDL and XML Schema (SAWSDL) [6] which provides two different methods: declarative annotations with model references that attach concepts of a reference ontology to elements or types of a schema, as well as references to lifting and lowering mappings (scripts) that actually transform XML-instance data to individuals of the reference ontology. In [7] we proposed an annotation technique which is fully declarative but allows to express semantics more precisely than mere concept references.

The aim of this work is to support the process of annotating schemas and of maintaining annotations that got invalid after ontology evolution [3]. We do not



assume that invalid annotations can be corrected automatically as this typically requires real world domain knowledge. Annotations need to be validated during the creation or maintenance process on three levels:

1. Structure: The referenced concepts or properties have to exist in the reference ontology and satisfy basic structural constraints [7].
2. Logics: The representation of a structurally valid annotation as a concept must not contradict with the reference ontology.
3. Semantics: The annotations correctly describe the real-world domain. We define semantic invalidations as changes of the ontology that have consequences on the interpretation of instance data of annotated schemas. This is different from logical invalidation. We have presented an approach for the detection of semantic invalidations that is based on the explicit definition of change-dependencies in [8].

The contribution of this paper is an in depth analysis of logical invalidations, resulting in algorithms and methods to (a) discover whether an annotation path expressions is logically invalid, (b) which part of an annotation path expression is invalid, and (c) the cause of this invalidation. This information should enable annotators to correct the annotation.

In the following we discuss semantic annotations of XML-schemas. However, the annotation method and the algorithms for validation of the annotations are not restricted to XML Schemas, but can as well be used for all other types of artifacts like web services, relational schemas, etc.

## 2 Annotation Method

We will briefly introduce the declarative annotation method we proposed in [7]. The goal of the annotation method is to describe the annotated element in much more detail than the direct annotation with existing concepts of the reference ontology while being declarative in contrast to rather procedural lifting/lowering mappings. The proposed annotation method has two representations: Path expressions over concepts and properties of an ontology that are directly used to annotate XML-Schema elements or types in form of SAWSDL [6] model references. These annotation paths are translated to complex OWL formulas representing the annotation in the ontology. We first define the structure of an annotation path expression and then show how a path is translated to an OWL formula. For details we refer to [7].

**Definition 1.** *Annotation Path:* An annotation path  $p$  is a sequence of steps. Each step is a triple  $s=(uri, type, res)$ , where  $s.uri$  is some URI of an element of the reference ontology  $O$ ; the type  $s.type$  defines the type of ontology element that is addressed by  $s.uri$ . It can be  $cs$  for a concept-step,  $op$  for an object-property step or  $dp$  for a datatype-property step.

Only concept-steps may have a set of restrictions  $s.res$ . Each restriction  $r \in s.res$  can either be an individual or a restricting path expression. Such a path

expression adds a restriction to the corresponding step  $s$ . If  $s.res$  contains multiple restrictions they all apply to the corresponding step  $s$  (logical and). Each annotation path  $p \in P$  has a type  $\in \{ConceptAnnotation, DataTypePropertyAnnotation\}$ . The type is defined by the last step.

A structurally valid annotation path expression must comply with the following restrictions:

- All steps refer to existing URIs of the ontology.
- The first step must refer to a concept.
- The last step must refer to a concept or to a datatype property.
- A step that refers to an object property can only occur between two concept steps.
- A step that refers to a concept must be followed by an object- or datatype-property step or nothing.
- A step that refers to a datatype property can only exist as the last step.
- Only steps that refer to concepts may have additional restrictions.

Structurally valid annotation path expressions can automatically be transformed to OWL [11] concepts that extend the reference ontology. The following example shows an annotation path and its corresponding concept in the reference ontology.

The annotation path  $p = Order/billTo/Buyer[Mr\_Smith]/hasCountry/Country$  is used to annotate a *country* element for some XML-Schema for *order* documents. It describes a subconcept of a *country* with an inverse relation *hasCountry* to some *Buyer* that has an inverse *billTo* relation to some *Order*. The buyer has a restriction to state that Buyer is a specific buyer with the name/URI *Mr. Smith*. The corresponding class definition  $p.c$  is shown in listing 1.1.

```

1 Class: Order/billTo/Buyer[Mr_Smith]/hasCountry/Country
2 EquivalentClasses (
3     ConceptAnnotation and Country and inv
4     (hasCountry) some
5     (Buyer and {Mr_Smith} and inv (billTo) some (Order)
6     ))

```

**Listing 1.1.** Representation of an annotation path in OWL

Structurally valid annotation path expression can be transformed to OWL concepts with the following mappings:

- *Concept-steps* are directly mapped to OWL-concepts.
- Restrictions on *concept-steps* are mapped to enumerated classes or restrictions over the corresponding concept.
- *Object-property-steps* are mapped to inverse OWL *some values from* restrictions between concepts on that specific property.
- *Datatype-property-steps* are mapped to OWL *some values from* restrictions of the last concept step on that specific datatype-property.

The annotation method allows different types of annotations, that we now define in order to describe the possible invalidations for each type in the following sections. *Simple concept annotations* consists of only one concept. *Simple datatype annotations* consists of only one concept and one datatype property. *3-step concept annotations* consists of a concept, an object-property and another concept. *General annotations* consist of more than 3 steps.

### 3 Logical Invalidation of Annotation Paths

An annotation path is logically invalid, if the corresponding annotation concept is not satisfiable in the reference ontology. Thus, the detection of logically invalid annotations is a classical reasoning task. The root concept of OWL is *Thing*. Any subconcept of this concept is satisfiable in the ontology. A satisfiable concept can contain individuals. Concepts that are not subclasses of *Thing* are not satisfiable and are subclasses of the concept *Nothing*, which is the complement of *Thing*.

**Definition 2.** *Logical Invalidation of an Annotation:* A structurally valid annotation path  $p$  is logically invalid if the corresponding annotation concept  $p.c$  is unsatisfiable in the reference Ontology  $O$ .  $O \cup p.c \rightarrow p.c \not\sqsubseteq \text{Thing}$

Why can't we use standard tools for validation and repair such as [12] or [14]? First, we want to determine which steps of the annotation path are responsible for the invalidation rather than determining a set of axioms of the (extended) ontology causing the invalidation. Second, repairs can only change annotation paths, and not axioms of the ontology. For debugging annotations we have the following requirements:

- The ontology is assumed to be consistent and therefore, free of contradictions before the annotation concept is added.
- The structure of the annotation concepts is strictly defined by the annotation method.
- Repairs can change annotation path expression but not the ontology.
- In case of annotation maintenance we require that the annotation concept was valid in the previous ontology version.

Therefore, we need to find the error in the steps of the annotations rather than in their OWL representation. This limits the usefulness of standard OWL debugging methods (see section 6). If an ontology evolves, annotation maintenance means to identify those annotation paths which became logically invalid due to the changes in the ontology and to identify those steps in the annotation path which cause the invalidation. An expert then can repair the invalid annotation paths efficiently using the information about the cause of the invalidation.

In OWL logical contradictions boil down to a limited set of contradictions [12]: **Atomic** - An individual belongs to a class and its complement. **Cardinality** - An individual has a max cardinality restriction but is related to more distinct individuals. **Datatype** - A literal value violates the (global or local) range restriction of a datatype property.

These clashes also apply for unsatisfiable classes. Thus, for example a class is unsatisfiable if it is defined as an intersection with its complement or if it has contradicting cardinality- or datatype-restrictions. Of course such invalidations can be produced by non-local effects. In the next sections we discuss how the different annotation types can be logically invalid.

### 3.1 Invalidation of Simple Concept Annotations

A simple concept annotation consists of only one concept. Thus, a concept with the name  $prefix + conceptUri$  is generated, where  $prefix$  is some unique identifier that is not used in the ontology  $O$ , with the equivalent class definition (*ConceptAnnotation and conceptUri*).

**Theorem 1.** A simple concept annotation that is structurally valid is also logically valid.

*Proof.* We require that all concepts of the reference ontology are satisfiable. Thus, there is only one case, where the union of *ConceptAnnotation* and *conceptURI* can result in an unsatisfiable concept: The class with the URI *ConceptAnnotation* is disjoint from the concept with the URI *conceptURI*. This is impossible because the primitive concept *ConceptAnnotation* does not exist in the reference ontology before the annotations are added. Thus, there cannot be an axiom in the ontology that contradicts with it.

### 3.2 Invalidation of Simple Datatype Annotations:

A simple datatype annotation of the form  $/c/datatypeProperty$  consists of a concept and a restriction over some datatype-property of the form: (*datatypeAnnotation and c and datatypeProperty some rdf:Literal*).

**Theorem 2.** There exists no invalid simple datatype annotation that does not violate one of the following conditions:

1. **Invalid-domain:** The intersection of the domain of the property with the concept is not a subclass of  $OWL : Thing$ .  
 $c \sqcap domain(datatypeProperty) \not\sqsubseteq Thing$
2. **Invalid-restriction:** The intersection of the concept and the restriction over the datatype-property is not a subclass of  $OWL : Thing$ .  
 $c \text{ and } datatypeProperty \text{ some } Literal \not\sqsubseteq Thing$

*Proof.* Obviously, case 2 of an invalidation is equivalent to the satisfiability-check of the whole annotation concept. There is only one additional case for an invalidation where the concept with the URI *datatypeAnnotation* is disjoint from  $c$ , which is impossible in analogy to theorem 1. Thus, every logically invalid simple datatype annotation is captured.

According to theorem 2 every simple datatype annotation that is invalid due to an *invalid-domain* is also invalid due to an *invalid-restriction*. Thus, in order to detect the cause of the error in more detail we need to investigate the reasons for the invalid restriction. This can be realized by additionally checking the first case. In addition the restriction clash is not yet atomic. In OWL there are the following scenarios for invalid restrictions over datatype-properties:

1. The datatype of the restriction does not comply with a datatype that is required by an existing restriction in  $O$ .
2. There is a cardinality clash between the existential restriction of the annotation path and an existing restriction in  $O$ .

**Theorem 3.** An invalidation of a simple datatype annotation due to a conflicting datatype restriction is impossible.

*Proof.* A contradicting datatype must be disjoint from the datatype in the existential restriction. This is impossible because every datatype is a subtype of  $rdfs:literal$ , which is used for the existential restriction in the annotation concept. No subtype can be disjoint from its supertype.

Cardinality clashes are possible, when there is a restriction on the class ( $c \sqcap datatypeProperty$ ) of the form:  $datatypeProperty \text{ max } n \text{ type}$ , where  $type$  is  $rdfs : Literal$  or any subtype of it.

### 3.3 Invalidation of 3-Step Concept Annotations

A 3-step concept annotation is a triple of the form  $concept/property/otherconcept$ . It is represented as an OWL equivalent class expression  $otherconcept \text{ and } inv (property) \text{ some } concept$ . Such an expression can be invalid due to *domain-invalidation*, *range-invalidation* and *restriction-invalidation*.

**Definition 3.** *Domain-Invalidation:*

An annotation triple of the form  $concept/Property/otherconcept$  is unsatisfiable due to a *domain-invalidation*, iff:  $domain(Property) \sqcap concept \not\sqsubseteq Thing$

**Definition 4.** *Range-Invalidation*

An annotation triple of the form  $concept/Property/otherconcept$  is unsatisfiable due to a *range-invalidation*, iff:  $range(Property) \sqcap otherconcept \not\sqsubseteq Thing$

**Definition 5.** *Restriction-Invalidation:*

An annotation triple of the form  $concept/Property/otherconcept$  is unsatisfiable due to a *restriction-invalidation*, iff:  $otherconcept \sqcap inv (Property) \text{ some } concept \not\sqsubseteq Thing$

**Theorem 4.** There exists no invalid 3-step concept annotation that does not introduce a *domain-invalidation*, *range-invalidation* or *restriction-invalidation*.

*Proof.* A *restriction-invalidation* is defined as  $otherconcept \sqcap inv(hasProperty)$  some  $concept \not\sqsubseteq Thing$ . This is equivalent to the satisfiability requirement for the whole annotation path because the intersection of *otherconcept* and *ConceptAnnotation* cannot result in a clash (see proof of theorem [11](#)). Thus, there exists no invalid 3-step annotations that are not captured by the enumerated invalidations.

While the domain or range invalidations are already atomic there can be different causes for invalid restrictions: A restriction can be invalid because the range of the restriction is disjoint from another *allvaluesFrom* restriction on *concept* or it can be invalid because there is a cardinality restriction on *concept* of the form *property max n otherconcept*. Therefore, the *invalid-restriction* problem can be divided into *invalid-value-restriction* and *invalid-cardinality-restriction*.

OWL2 allows the definition of object properties to be functional, inverse functional, transitive, symmetric, asymmetric, reflexive, and irreflexive. Since the existence of the object property is defined by the existential quantification of the inverse of the property these characteristics can influence the satisfiability of the annotation. For example, given an annotation path  $p = /A/hasB/B$ , the path is invalid, if *hasB* is defined as inverse functional and *B* has an inverse *hasB* restriction in *O* to some other class that is disjoint from *A*.

We can summarize that a *3-step concept annotation* can be invalid because of the restrictions that are formulated over the corresponding annotation concept. Definition [5](#) is sufficient but the root cause can be found in property characteristics or cardinality or value clashes.

## 4 Invalidation of General Annotations

In the last section we defined all local invalidations that can occur in annotations that consists of 3 steps. A general annotation consists of a sequence of 3-step concept annotations called triples. The last step can be a 3-step concept annotation or a simple datatype annotation. We will first show that all local invalidation types also apply to general concept annotations and then discuss additional kinds of invalidations that are only possible in general annotations.

### 4.1 Invalidation of General Annotation Due to Local Invalidations

**Definition 6.** *Local-Invalidations:* The invalidation types *domain-invalidation* (see def. [3](#)), *range-invalidation* (see def. [4](#)) and *restriction-invalidation* (see def. [5](#)) are local invalidations, that are defined in the context of a triple.

**Theorem 5.** A locally invalid 3-step annotation cannot get valid, when it occurs as a triple in a general annotation path.

*Proof.* A general annotation path has the form:  $/c_1/p_2/c_3/\dots/c_{n-2}/p_{n1}/c_n/$ . We now assume that there exists a triple  $C_{inv} = c_x/p_y/c_z$ , in the path that is invalid, when it is inspected separately (local invalidation), but the entire annotation

concept ...  $c_{-2}/p_{-1}/c_x/p_y/c_z/p_1/c_2$  ... is valid. This implies that either  $c_x$  or  $c_z$  were implicitly changed to classes that are not still causing local invalidations in  $C_{inv}$ . When the triple  $C_{inv}$  is added to the annotation concept this is realized by an expression of the form:

...  $c_2$  and  $(inv) p_1$  some  $(c_z$  and  $inv (p_y)$  some  $(c_x$  and  $p_{-1}$  some ...

Thus,  $z_x$  is implicitly replaced with an intersection of  $z_x$  and  $(p_1$  some ...) that we now call  $z_{x2}$ .  $c_z$  gets implicitly replaced with  $c_z$  and  $(range (p_1)$  that we now call  $c_{z2}$ . In order to achieve a satisfiable triple  $C_{inv}$  in  $p$ ,  $c_{x2}$  must not be a subclass of  $c_x$  or  $c_{z2}$  must not be a subclass of  $c_z$ . This is a contradiction because they are logically subclasses of  $c_x$  and  $c_z$ .

**Theorem 6.** A general concept annotation that contains an invalid triple is itself logically invalid.

*Proof.* A general concept annotation path consists of triples:  $t_1/t_2/t_3/.../t_n$ . We will now show via induction that as soon as one of its triples is unsatisfiable, the whole annotation concept is unsatisfiable. Beginning with an annotation  $p_1$  that only consists of  $t_n$ . If  $t_n$  is itself unsatisfiable, then the whole path cannot be satisfiable because it is represented as a subclass of  $t_n$  in  $p_1.c$ . We now assume that  $p_1$  is satisfiable and we add  $t_{n-1}$ , which is supposed to be unsatisfiable. The addition renders the whole annotation path unsatisfiable because the connection between  $p_n$  and  $t_{n-1}$  is represented in form of an existential restriction. This step can be repeated by adding an unsatisfiable triple to a longer and longer valid path, until  $t_1$  is reached. Therefore, if any triple of a general concept annotation is locally invalid the whole annotation concept must be logically invalid.

As a conclusion all previously discussed local invalidations also apply to general annotations. Additionally there are invalidations that only occur in general annotations: direct-triple-disjointness and arbitrary non local invalidations.

### 4.2 Direct-Triple-Disjointness

One kind of invalidation that does not exist for 3-step annotations can be caused by the concatenation of two annotation triples. This means the concept that is implicitly created by the first triple is disjoint from the concept which is required by the second triple. An example for such a scenario is shown in Figure 1. The corresponding reference ontology is shown in listing 1.2.

```

1 Order isA Document
2 Invoice isA Document
3 Disjoint (Order , Invoice )
4 Domain (SendsOrder) = Customer
5 Range (SendsOrder) = Order
6 Domain (hasInvoiceNumber) = Invoice
7 Range (hasInvoiceNumber) = InvoiceNumber
    
```

**Listing 1.2.** Example ontology for direct-triple-disjointness invalidations



**Fig. 1.** Example of direct-triple-disjointness

In *Customer/sendsOrder/Document/hasInvoiceNumber/InvoiceNumber* each triple is valid individually, but the combination of the triples leads to an unsatisfiable concept. The reason for this invalidation is that the subclass of *Document* that is produced by the range of *sendsOrder* in the first triple is disjoint from the subclass of *Document* that is produced by the domain of *hasInvoiceNumber* in the second triple.

**Theorem 7.** *Direct-Triple-Disjointness:* An annotation path  $p = /t_1/\dots/t_n/$  is invalid, if there exist two logically valid neighbored triples  $t_n = c_n/p_n/c_m$  and  $t_m = c_m/p_m/c_{m+1}$ , where  $range(p_n) \sqcap restriction(c_n, p_n) \sqcap domain(p_m) \sqcap c_m \not\sqsubseteq Thing$ .

*Proof.* The intersection class  $range(p_n) \sqcap restriction(c_n, p_n) \sqcap domain(p_m) \sqcap c_m$  describes the implicit concept between two annotation triples, that is responsible for the concatenation of the triples. If this intersection concept is unsatisfiable any class with an existential restriction for this concept becomes unsatisfiable.

### 4.3 Non-local Invalidation

Local- and direct-triple-disjointness invalidations can be located precisely. That means the step in the path that causes the invalidation can be annotated with the type of the clash and the reason for the invalidation. This can be valuable information for a user who has to repair the annotation path.

In case of general annotation paths which consist of two or more triples additional invalidations can occur which are not necessarily induced by neighboring triples. We will now first present an example in listing [1.3](#) and then define the problem in general.

The example contains the annotation concept *MyAnnotation* that represents the path *BusinessCustomer/sends/Order/has/Itemlist/contains/PrivateProduct/hasPrice/Price*. The annotation concept is free of local- or direct-triple- disjointness invalidations. Nevertheless, it is logically invalid because according to the ontology, business customers may only send business orders and business orders may only contain business products. Business products are disjoint from private products. This renders the whole annotation path unsatisfiable. Now our goal is to find the steps in the path that are responsible for the invalidation. In this case the steps that are responsible for the clash are:

*BusinessCustomer/sends/Order/has/Itemlist/contains/PrivateProduct.*



```

1 Class (BusinessCustomer )
2 Class (Order), Class (BusinessOrder), Class (PrivateOrder)
3 Class (Itemlist )
4 Class (PrivateProduct), Class (BusinessProduct )
5 Class (Price)
6 Class (ClassMyAnnotation )
7 ObjectProperty (sends)
8 ObjectProperty (contains)
9 ObjectProperty (has)
10 ObjectProperty (hasPrice)
11 EquivalentClass (BusinessCustomer ,
12     BusinessCustomer and sends only BusinessOrder)
13 EquivalentClass (BusinessOrder , BusinessOrder and has only
14     (Itemlist and contains only BusinessProduct))
15 EquivalentClass (MyAnnotation , Price and inv (hasPrice) some
16     (PrivateProduct and inv (contains) some
17         (Itemlist and inv (has) some
18             (Order and inv
19                 (sends) some BusinessCustomer))))
20 disjoint (BusinessOrder , PrivateOrder)
21 disjoint (BusinessProduct , PrivateProduct)

```

**Listing 1.3.** A non local invalidation

To define such invalidations we first introduce a normalized representation form of an annotation concept.

**Definition 7.** *Normalized Annotation Concept*

An annotation path  $p = /c_1/p_2/c_3/...c_{n-2}/p_{n-1}/c_n/$  is represented as an annotation concept  $p.c = c_n$  and  $inv (p_{n-1})$  some  $(c_{n-2} \dots$  and  $inv (p_2$  some  $c_1) \dots)$ . This annotation concept uses nested anonymous concepts. In contrast a normalized annotation concept of  $p.c$  uses named concepts of the form:

$$\begin{aligned}
 p.c &= Ac_0 = c_n \text{ and } inv (p_{n-1}) \text{ some } Ac_1 \\
 Ac_1 &= c_{n-2} \text{ and } inv (p_{n-3}) \text{ some } Ac_2 \\
 Ac_2 &= c_{n-4} \text{ and } inv (p_{n-5}) \text{ some } Ac_3 \\
 &\dots \\
 Ac_j &= c_3 \text{ and } inv (p_2) \text{ some } c_1
 \end{aligned}$$

**Definition 8.** *Chain of Restrictions of an Annotation Path.*

A chain of restrictions of a normalized annotation concept  $p.c$  of an annotation path  $p$  is any set of succeeding named concepts  $Ac_x \dots Ac_{x+n}$  of  $p.c$ , where  $n \geq 1 \wedge x \geq 0 \wedge x + n < |p.c| - 1$ .

**Theorem 8.** *Non Local Invalidations:* When a path is invalid and it is free of local and direct-triple-disjointness invalidations, then there must exist at least one sub-path of two or more triples that conflicts with the ontology.

*Proof-Sketch:* Given a logically invalid annotation path  $p_{inv}$  that is free of local- and direct-triple-disjointness invalidations of  $m$  triples of the form

$/t1/..t_m$ . From the absence of local invalidations follows that each triple  $t \in p_{inv}$  is valid separately. From the absence of intra-triple-disjointness invalidations follows that the intermediate concept that is build by every neighbored pair of triples is satisfiable. Thus, the unsatisfiability of  $p_{inv}$  cannot have a local reason. Non-local invalidations are induced by chains of restrictions that conflict with the reference ontology. Each chain of restriction of an annotation path can also be represented as a sub-path of  $p_{inv}$ .

**Definition 9.** *Minimal Invalid Sub-path (MIS):* An invalid sub-path  $p_s$  that is free of local or triple disjointness invalidations of an annotation path  $p$  is minimal, iff the removal of the first or last triple of  $p_s$  yields a satisfiable concept of  $p_s.concept$  in  $O$ .

We will now discuss which OWL constructs can cause non local invalidations.

- **Chain of Restrictions:** There is a chain of restrictions defined on concepts in  $O$  that produces a clash with the chain of restrictions of the *MIS*. The chain can be created analogues to our annotation concept or it can be realized with named concepts. The definition may be defined inverse as our annotation concepts or non-inverse. An example was shown in listing [1.3](#)
- **Transitive Properties:** Transitive properties may also result in an invalidation of an annotation path. An example is shown in listing [1.4](#). The annotation concept of the annotation path  $A/has/B/has/C/has/D$  is unsatisfiable due to the transitivity of the property *has*, which has the consequence that  $D$  has an inverse property definition for *has* to  $B$  and  $A$  implicitly. Because  $D$  may only have an inverse relation *has* to  $C$  the annotation concept is unsatisfiable.
- **Property Chains:** A property chain has the form  $p1 \ o \ p2 \ \rightarrow \ p3$ . It expresses that if there is a chain where some individual  $i1$  has a property  $p1$  to another individual  $i2$  and this individual has a property  $p2$  to an individual  $i3$ , then the individual  $i1$  has an assertion for the property  $p3$  to  $i3$ . Of course the chain can have an arbitrary length. This can be seen as a flexible case of transitivity, where the properties in the chain do not need to have an equivalent or sub-property relation in order to produce a transitive chain. Thus, property chains can also be used to produce non-local invalidations.

```

1 Class(A), Class(B), Class(C), Class(D)
2 Class(ClassMyAnnotation)
3 ObjectProperty(has)
4 transitive(has)
5 EquivalentClass(D and inv(has) only C)
6 EquivalentClass(MyAnnotation, D and inv(has)
7   some(C and inv(has) some(B and inv(has) some(A)))
8 disjoint(C,B)

```

**Listing 1.4.** Example ontology for inter-triple invalidations by transitive properties

#### 4.4 An Algorithm for the Detection of a Minimal Invalid Sub-Path

An algorithm for the detection of a minimal invalid sub-path of an annotation path  $p$  can be based on a structural search over conflicting axioms in the reference ontology. The last section has shown that such non local conflicts can occur due to many different OWL constructs. Of course a *MIS* can be the result of a combination of the described causes, which makes an exhaustive search even more complex. The efficiency of such an algorithm is further reduced by the fact that reasoning over sub-, super-, and equivalent-properties and -classes is required. In addition, for such a detection method the first and last triple of a sub-path are not known in advance. This makes another approach that directly operates on definition 9 of a minimal invalid sub-path more efficient. The corresponding algorithm is shown in in listing 1.5. The algorithm takes an invalid path  $p$  that is free of local and direct-triple disjointness invalidations as input and returns the index of the start and end triple of the detected *MIS*.

```

1 (int , int) getMinimalInvalidSubpath (p,O) {
2   r = p.tripleCount()-1;
3   // Find the right border of the MIS
4   while (O.unsatisfiable(createSubPath(p,1,r))
5     r--;
6   }
7   l = 2;
8   // Find the left border of the MIS
9   while (O.unsatisfiable(createSubPath(p,l,r+1))
10    l++;
11  }
12  return(l-1,r+1)
13 }
```

**Listing 1.5.** An algorithm for the detection of the minimal invalid sub-path

The algorithm uses some helper methods. The method  $p.tripleCount()$  returns the number of triples of  $p$ ,  $createSubPath(p,l,r)$  returns the OWL expression of a sub-path of  $p$  that starts at index  $l$  and ends at index  $r$ . The methods assumes that the leftmost triple of  $p$  has the index 1 and the last triple of  $p$  has the index  $p.tripleCount()$ . The method  $O.unsatisfiable(owlExp)$  returns *true*, if the OWL expression  $owlExp$  is unsatisfiable in the ontology  $O$ .

The first loop is used to find the right boundary of a *MIS*. This is realized by sequentially creating a sub-path of  $p$  that begins at position 1 and end at position  $r$ , where  $r$  is decremented in each iteration. The loop terminates as soon as the created sub-path gets satisfiable. Therefore, the right boundary of the *MIS* must be at position  $r + 1$ . The reason for this is that analogues to theorem 6, there can exist no complete *MIS* before position  $r$ . Otherwise  $r$  cannot be satisfiable. After the right boundary was found it is guaranteed that the sub-path between 1 and  $r + 1$  is invalid. However, it is not yet sure that it is minimal. Therefore, the left boundary of the *MIS* needs to be found. This is

realized by creating a sub-path that begins at position  $l$  and ends at position  $r + 1$ , where  $l$  starts at 2 and it is incremented in each iteration. As soon as such a sub-path gets satisfiable the left boundary of the *MIS* has been found at position  $l - 1$ .

The detected *MIS* complies with definition 9 because both iterations guarantee that the removal of the first or last triple of the *MIS* result in a valid sub-path of  $p$ . The algorithm guarantees that it can find one *MIS*. If a path contains multiple *MIS*, we propose to remove them iteratively with the help of the proposed algorithm.

**Theorem 9.** When the algorithm of listing 1.5 is used on a path that is free of local and direct-triple-disjointness invalidations that contains multiple *MIS*, then the leftmost inner *MIS* is detected.

*Proof-Sketch:* Given an annotation path  $p = /t_1/t_2/.../t_n$ . In the first iteration sub-paths starting at  $t_1$  of  $p$  are created. The unsatisfiable sub-path with the minimum number of triples is considered to be a *MIS*-candidate. According to theorem 6 there can exist no other complete *MIS* in the path that ends before the *MIS* candidate. It is only possible that there exists another *MIS* that starts before and ends after or at the same position as the detected one. In the next loop the minimality of the *MIS* is guaranteed by chopping elements from the start. As a consequence the algorithm detects the leftmost inner-*MIS*.

## 5 Implementation Considerations

In this paper we have defined error-types on annotation paths. The goal is to tell the user, which steps of a path are responsible for the invalidation including an explanation of the type of invalidation. The detection of most invalidation types is straight forward. It is just a query to the reasoner that is equivalent to the definition of the specific invalidation type. Non local invalidations can be tracked by the proposed *MIS* algorithm. However, testing each triple of every invalid annotation path can be an expensive task. Therefore, we will briefly discuss properties of annotation path that can be used to enormously reduce the number of queries to the reasoner. In a typical scenario there is a set of valid annotations  $V$  and a set of invalid annotations  $I$ . Both sets are a direct result of the classification of the reference ontology with the added annotation concepts. We now define properties that hold between the elements of  $V$  and  $I$ .

**Theorem 10.** *Globally-valid path-postfix:* Given a set of valid annotations  $V$  and one invalid annotation  $i$ . If there exists an annotation  $v$  in  $V$  with a common postfix (ending with the same sequence of triples) with  $i$ , then the corresponding sub-path of  $i$  cannot introduce local or direct-triple-disjointness invalidations.

*Proof.* No annotation path  $\in V$  can contain local or direct-triple-disjointness invalidations. Otherwise it would not be satisfiable. If a path is satisfiable also every postfix of it must be satisfiable due to the monotonicity of OWL. An annotation concept is a specialization of the last concept-step. The longer a path

is, the more specific is the annotation concept. When there exists an annotation path  $v \in V$  which has the same postfix  $f$  as  $i$ , then  $i.concept$  is a more specific concept than  $f.concept$ . Thus, the additional specialization must induce the error. It is represented by the prefix of  $i$  which does not match  $f$ .

**Theorem 11.** *Globally-valid-triples:* A triple that is an element of a path  $\in V$  cannot produce a local invalidation when it is used in a path in  $I$ .

*Proof.* The proof of theorem [11] is a direct consequence of theorem [6]. Any triple that is an element of a valid path cannot be logically invalid because otherwise the path would be invalid.

These two properties of a globally valid postfix and triples can be used to find local invalidations or intra-triple-disjointness invalidations very efficiently. As a first step the longest common postfix from  $i$  and the annotations in  $V$  can be detected. If such a postfix is found it is guaranteed that the corresponding postfix in  $i$  cannot contain local or direct-triple disjointness invalidations. In addition all triples in all paths of  $V$  can be considered as locally valid triples. Thus, if they occur in  $i$  they do not need to be tested for local invalidations.

Finally, when the annotation concepts are represented in form of normalized concepts (see definition [7]) in the ontology, it is guaranteed that all triples that correspond to satisfiable named concepts are locally valid and that no direct-triple-disjointness invalidations can exist between succeeding triples, that correspond to satisfiable named concepts in the normalized representation.

All these considerations can lead to a major speedup for the detection of invalidations because triples and combinations of triples that are known to be valid do not need to be checked for specific error-types (domain-invalidation, range-invalidation, ...) and in order to guarantee that the input path of the *MIS* algorithm is free of local or direct triple-disjointness invalidations, only the potentially invalid triples and combinations of triples need to be checked.

## 6 Related Work

The basis of this research is a declarative annotation method for XML-Schema published in [7]. The annotation method has two representations: path expressions and complex OWL formulas. Only the path expressions can be changed by the annotators. Therefore, we have proposed methods to track errors in annotation paths. In order to find errors in the corresponding complex OWL formula also general ontology debugging solutions could be used. However, preliminary experiments with the well known OWL1 tool Swoop [12] have shown that Swoop was not able to detect the root-cause of many non local invalidations that only used OWL1 language constructs. In this case the concept was detected to be invalid but no explanation could be generated. When explanations could be generated it was very tedious for the annotator to actually discover which elements in the path were responsible for the problem. The integrated repair tool of Swoop could not help either. In contrast our method can precisely track, which elements in the path are responsible for the invalidation and it is a reasoner-independent

black-box approach. In addition we have defined error-types that indicate the reason for the invalidation with respect to the steps of the path.

A fundamental publication in the field of ontology debugging is [14]. It introduces the term of minimal unsatisfiable sub Tboxes (MUPS). A MUPS is a minimal set of axioms that is responsible for a concept to be unsatisfiable. When one axiom gets removed from the MUPS the concept gets satisfiable unless there are additional MUPS for the concept. This definition is somehow analogous to our definition of the minimal invalid sub-path. In [5] an optimized black box algorithm for the computation of the MUPS is presented. The Black-Box algorithm basically tries to find the MUPS in a trial and error fashion, which requires a high number of expensive reclassifications. In order to get all justifications the authors calculate a first justification (MUPS) and afterwards use a variant of the Hitting Set Algorithm [13] to obtain all other justifications for the unsatisfiability. The goal of general ontology debugging approaches is: Given an ontology with at least one unsatisfiable concept find a set of axioms that need to be removed in order to obtain a coherent ontology. There can be multiple sets of such axioms (also called diagnoses). Therefore, it is beneficial to rank the possible repairs either by assuming that the set of removed axioms should be minimal [14] or by selecting the diagnosis [4] that best fits the modeling intention by asking an oracle/user. This is a major difference to the annotation maintenance scenario, where the ontology cannot be changed by the annotator and only changes of the the path expression are allowed. Therefore, we search especially for steps in the path that lead to an error.

An alternative approach to debug ontologies are patterns/anti-patterns (in particular logically detectable anti-patterns) as proposed in [2,1]. Those patterns concentrate on common modeling errors that are made on ontology artifacts such as concepts. They can provide well understandable explanations for common errors on simple concepts. Because the subject of such patterns is a concept and not an annotation path their usefulness for annotation paths is limited to simple cases.

## 7 Conclusion

Semantic annotation is an important technique for semantic processing of information, in particular for interoperability of heterogeneous information systems. Annotation paths are a declarative yet highly expressive way to describe the semantics of artifacts like schemas or documents. We propose methods and algorithms for the identification of deficits in annotation paths, which is based on an in depth analysis of the possible causes for invalid annotations. We expect that experts who annotate artifacts will be much more efficient, if the position and the cause of errors in annotations paths is automatically determined. This technique is also particularly useful for annotation maintenance as a consequence of an evolution of the reference ontology as it not only identifies the annotations which became logically invalid, but also narrows the inspection area to the shortest possible path and gives indications on the causes of the invalidation in form of invalidation types. The proposed algorithm and methods are built upon the functionality usually provided by generic reasoners for OWL ontologies, so they are not restricted to a specific reasoner or ontology management system.

## References

1. Corcho, O., Roussey, C., Blazquez, L.M.V., Perez, I.: Pattern-based owl ontology debugging guidelines. In: Proc. of WOP 2009. CEUR-WS.org., vol. 516 (November 2009)
2. Corcho, O., Roussey, C., Zamazal, O., Scharffe, F.: SPARQL-based Detection of Antipatterns in OWL Ontologies. In: Proc. of EKAW 2010 Poster and Demo Track (October 2010)
3. Eder, J., Koncilia, C.: Modelling Changes in Ontologies. In: Meersman, R., Tari, Z., Corsaro, A. (eds.) OTM Workshops. LNCS, vol. 3292, pp. 662–673. Springer, Heidelberg (2004)
4. Friedrich, G., Shchekotykhin, K., Rodler, P.: Balancing brave and cautions query strategies in ontology debugging. In: Proc. of DX-2011 Workshop, pp. 122–130. DX Society (2011)
5. Kalyanpur, A., Parsia, B., Horridge, M., Sirin, E.: Finding All Justifications of OWL DL Entailments. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 267–280. Springer, Heidelberg (2007)
6. Kopecký, J., Vitvar, T., Bournez, C., Farrell, J.: Sawsdl: Semantic annotations for wsdl and xml schema. IEEE Internet Comp. 11(6), 60–67 (2007)
7. Köpke, J., Eder, J.: Semantic Annotation of XML-Schema for Document Transformations. In: Meersman, R., Dillon, T., Herrero, P. (eds.) OTM 2010. LNCS, vol. 6428, pp. 219–228. Springer, Heidelberg (2010)
8. Köpke, J., Eder, J.: Semantic Invalidation of Annotations Due to Ontology Evolution. In: Meersman, R., Dillon, T., Herrero, P., Kumar, A., Reichert, M., Qing, L., Ooi, B.-C., Damiani, E., Schmidt, D.C., White, J., Hauswirth, M., Hitzler, P., Mohania, M. (eds.) OTM 2011, Part II. LNCS, vol. 7045, pp. 763–780. Springer, Heidelberg (2011)
9. Miller, E., Manola, F.: RDF primer. W3C recommendation, W3C (February 2004), <http://www.w3.org/TR/2004/REC-rdf-primer-20040210/>
10. Missikoff, M., Schiappelli, F., Taglino, F.: A controlled language for semantic annotation and interoperability in e-business applications. In: Proc. of ISWC 2003, pp. 1–6 (2003)
11. W3C OWL Working Group. OWL 2 Web Ontology Language: Document Overview. W3C Recommendation, October 27 (2009), <http://www.w3.org/TR/owl2-overview/>
12. Parsia, B., Sirin, E., Kalyanpur, A.: Debugging owl ontologies. In: Proc. of 14th Int. Conf. WWW, pp. 633–640. ACM Press (2005)
13. Reiter, R.: A Theory of Diagnosis from First Principles. In: Siekmann, J.H. (ed.) CADE 1986. LNCS, vol. 230, pp. 153–153. Springer, Heidelberg (1986)
14. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: IJCAI, pp. 355–362 (2003)
15. Sheth, A.P., Ramakrishnan, C.: Semantic (web) technology in action: Ontology driven information systems for search, integration and analysis. IEEE Data Eng. Bull. 26(4), 40–48 (2003)
16. Vujasinovic, M., Ivezic, N., Kulvatunyou, B., Barkmeyer, E., Missikoff, M., Taglino, F., Marjanovic, Z., Miletic, I.: Semantic mediation for standard-based b2b interoperability. IEEE Internet Comp. 14(1), 52–63 (2010)

# Uniform Access to Non-relational Database Systems: The SOS Platform

Paolo Atzeni, Francesca Bugiotti, and Luca Rossi

Dipartimento di informatica e automazione  
Università Roma Tre

atzeni@dia.uniroma3.it, franbugiotti@yahoo.it, luca@rossi.net

**Abstract.** Non-relational databases (often termed as NoSQL) have recently emerged and have generated both interest and criticism. Interest because they address requirements that are very important in large-scale applications, criticism because of the comparison with well known relational achievements. One of the major problems often mentioned is the heterogeneity of the languages and the interfaces they offer to developers and users. Different platforms and languages have been proposed, and applications developed for one system require significant effort to be migrated to another one. Here we propose a common programming interface to NoSQL systems (and also to relational ones) called SOS (Save Our Systems). Its goal is to support application development by hiding the specific details of the various systems. It is based on a metamodeling approach, in the sense that the specific interfaces of the individual systems are mapped to a common one. The tool provides interoperability as well, since a single application can interact with several systems at the same time.

**Keywords:** Non-Relational Databases, NoSQL, Interoperability.

## 1 Introduction

Relational database systems (RDBMSs) dominate the market by providing an integrated set of services that refer to a variety of requirements, which mainly include support to transaction processing but also refer to analytical processing and decision support. From a technical perspective, all the major RDBMSs on the market show a similar architecture (based on the evolutions of the building blocks of the first systems developed in the Seventies) and do support SQL as a standard language (even though with dialects that differ somehow). They do provide reasonably general-purpose solutions that balance the various requirements in an often satisfactory way.

However, some concerns have recently emerged towards RDBMSs. First, it has been argued that there are cases where their performances are not adequate, while dedicated engines, tailored for specific requirements (for example decision support or stream processing) behave much better [12] and provide scalability [11]. Second, the structure of the relational model, while being effective for



many traditional applications, is considered to be too rigid or not useful in other cases, with arguments that call for *semistructured* data (in the same way as it was discussed since the first Web applications and the development of XML [1]). At the same time, the full power of relational databases, with complex transactions and complex queries, is not needed in some contexts, where “simple operations” (reads and writes that involve small amount of data) are enough [11]. Also, in some cases, *ACID* consistency, the complete form of consistency guaranteed by RDBMSs, is not essential, and can be sacrificed for the sake of efficiency. It is worth observing that many Internet application domains, for example, the social networking domain, require both scalability (indeed, Web-size scalability) and flexibility in structure, while being satisfied with simple operations and weak forms of consistency.

With these motivations, a number of new systems, not following the RDBMS paradigm (neither in the interface nor in the implementation), have recently been developed. Their common features are scalability and support to simple operations only (and so, limited support to complex ones), with some flexibility in the structure of data. Most of them also relax consistency requirements. They are often indicated as *NoSQL* systems, because they can be accessed by APIs that offer much simpler operations than those that can be expressed in SQL. Probably, it would be more appropriate to call them *non-relational*, but we will stick to common usage and adopt the term NoSQL.

There is a variety of systems in the NoSQL arena [6,11], more than fifty, and each of them exposes a different interface (different model and different API). Indeed, as it has been recently pointed out, the lack of standard is a great concern for organizations interested in adopting any of these systems [10]: applications and data are not portable and skills and expertise acquired on a specific system are not reusable with another one. Also, most of the interfaces support a lower level than SQL, with record-at-a-time approaches, which appear to be a step back with respect to relational systems.

The observations above have motivated us to look for methods and tools that can alleviate the consequences of the heterogeneity of the interfaces offered by the various NoSQL systems and also can enable interoperability between them together with ease of development (by improving programmers’ productivity, following one of the original goals of the relational databases [8]). As a first step in this direction, we present here *SOS (Save Our Systems)*, a programming environment where non-relational databases can be uniformly defined, queried and accessed by an application program.

The programming model is based on a high-level description of the interfaces of non-relational systems by means of a generic and extensible meta-layer, based on principles that are inspired by those our group has used in the MIDST and MIDST-RT projects [2,3,4]. The focus in our previous work was on the structure of models and was mainly devoted to the definition of techniques to translate from a representation to another one. Here, the meta layer refers to the basic common structure and it is therefore concerned with the methods that can

be used to access the systems. The meta-layer represents a theoretical unifying structure, which is then instantiated (indeed, implemented) in the specific underlying systems; we have experimented with various systems and, in this work, we will discuss implementations for three of them with rather different features within the NoSQL family: namely, Redis<sup>1</sup>, MongoDB<sup>2</sup> and HBase<sup>3</sup>.

Indeed, the implementations are transparent to the application, so that they can be replaced at any point in time (and so one NoSQL system can be replaced with another one), and this could really be important in the tumultuous world of Internet applications. Also, our platform allows for a single application to partition the data of interest over multiple NoSQL systems, and this can be important if the application has contrasting requirements, satisfied in different ways by different systems. Indeed, we will show a simple application which involves different systems and can be developed in a rapid way by knowing only the methods in our interface and not the details of the various underlying systems.

To the best of our knowledge, the programming model we present in this paper is original, as there is no other system that provides a uniform interface to NoSQL systems. It is also a first step towards a seamless interoperability between systems in the family. Indeed, we are currently working at additional components that would allow code written for a given system to access other systems: this will be done by writing a layer to translate proprietary code to the SOS interface; then, the tool proposed here would allow for the execution on one system of code developed for another one.

The rest of this paper is organized as follows. In Section 2 we illustrate the approach by defining a common interface. In Section 3 we illustrate our meta-layer and in Section 4 we present the structure of the platform. In Section 5 we illustrate a simple example application. Then, we briefly discuss related work (Section 6) and draw our conclusions (Section 7).

## 2 The Common Interface

As we said, the goal of our approach is to offer a uniform interface that would allow access to data stored in different data management systems (mainly of the NoSQL family, but possibly also relational), without knowing in advance the specific one, and possibly using different systems within a single application. In this section we discuss on the desirable features of such an interface and then present our proposal for it. In the next section we will then describe the underlying architecture that allows for mapping it to the various systems.

NoSQL systems have been developed independently from one another, each with specific application objectives, but all with the general goal to offer rather simple operations, to be supported in a scalable way, with possibly massive

---

<sup>1</sup> <http://redis.io>

<sup>2</sup> <http://www.mongodb.org>

<sup>3</sup> <http://hbase.apache.org>

throughput.<sup>4</sup> Indeed, there are many proposals for them, and effort have been devoted to the classification of them into various categories, the most important of which have been called key-value stores, document stores and extensible record stores, respectively [6]. The three families share the idea of handling individual items (“objects” or “rows”) and on the need for not being rigid on the structure of the items, while they differ on the features that allow to refer to internal components of the objects.<sup>5</sup> Most importantly, given the general goal of concentrating on simple operations, they are all based on simple operations for inserting and deleting the individual items, mainly one at the time, and retrieving them, one at the time or a set at the time. Therefore, a simple yet powerful common interface can be defined on very basic and general operations:

- **put** to insert objects
- **get** to retrieve objects
- **delete** to remove objects

Crucial issues in this interface are (i) the nature of the objects that can be handled, which in some systems are allowed to have a rather complex structure, not fixed in advance, but with components and some forms of nesting, and in others are much simpler and (ii) the expressivity of the get operation, which in some cases can only refer to identifiers of objects and in others can be based on conditions on values. This second issue is related to the fact that operations can also be specified at high level with reference to a single field of a structured object. The platform infers from the meta-layer all the involved structures and defines coherently the sequence of operations on them.

The simple interface we have just described is the core component of the architecture of the SOS system, which is organized in the following modules (Figure 11):

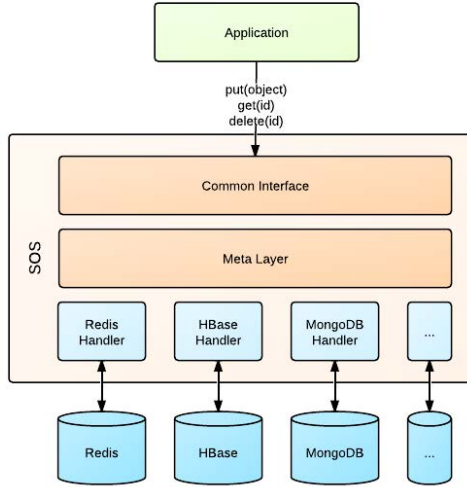
- the common interface that offers the primitives to interact with NoSQL stores;
- the meta-layer that stores the form of the involved data;
- the specific handlers that generate the appropriate calls to the specific database system.

The SOS interface exposes high level operations with reference to the general constructs defined in the meta-layer. The meta-layer stores data and provides the interface with a uniform data model for performing operations on objects.

---

<sup>4</sup> Our interest here is in the features related to how data is modeled and accessed. So, we will not further refer to the attention to scalability nor to another important issue, the frequent relaxation of consistency, which are both orthogonal to the aspects we discuss. Because of this orthogonality, our approach preserves the benefits of scalability and relaxation of consistency.

<sup>5</sup> We will give some relevant details for the three families and their representative systems in the next section, while discussing how the common interface can be implemented in them.



**Fig. 1.** Architecture of SOS

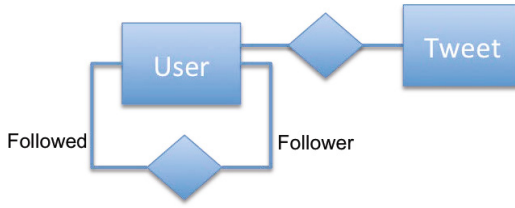
The specific handlers support the low level interactions with specific NoSQL storage systems mapping meta-layer generic calls to system specific queries.

In order to show how our proposed system can support application development, in this paper we will refer to an example regarding the definition of a simplified version of Twitter<sup>6</sup> the popular social network application. We will adopt the perspective of a Web 2.0 development team that wants to benefit from the use of different NoSQL systems. Transactions are short-lived and involve little amount of data, so the adoption of NoSQL systems is meaningful. Also, let us assume that quantitative application needs have led the software architect to drive the decision towards the use of several NoSQL DBMSs, as it turned out that the various components of the application can benefit each from a different system<sup>7</sup>.

The data of interest for the example have a rather simple structure, shown in Figure 2: we have users, with login and some personal information, who write posts; every user “follows” the posts of a set of users and can, in turn, “be followed” by another set of users. In the example, in Section 5, we will show how this can be implemented by using three different NoSQL systems in one single application.

<sup>6</sup> <http://twitter.com/>

<sup>7</sup> For the sake of space here the example has to be simple, and so the choice of multiple systems is probably not justified. However, as the various systems have different performances and different behavior in terms of consistency, it is meaningful to have applications that are not satisfied with just one of them.



**Fig. 2.** The schema for the data in the example

### 3 The Meta-layer

In this section we give some formal explanation of the meta-layer structure and we contextualize it with respect to the NoSQL system our platform handles.

Our approach leverages on the genericity of the data model to allow for a standard development practice that is not bound to a specific DBMS API, but to a generic one.

Application programming interfaces are built over and in terms of the constructs of the meta-layer (without explicit reference to lower level constructs); in this way, programs are modular and independent of the particular data model; reuse is maximized.

According to the literature, a data model can be represented as the collection of its characterizing constructs, a set of constraints and a set of operations acting over them [14]. A construct is an entity that has a conceptual significance within the model. A construct can be imagined as the elementary outcome of a structural decomposition of a data model. A construct is relevant in a data model since it is a building block of its logical structures.

Thus, the aim of the meta-layer is to reconcile the relevant descriptive elements of mainstream NoSQL databases: key-value stores, document stores and record stores. In the following paragraphs, we will describe concisely their data models, and we will show how their constructs and operations can be effectively modeled through our meta-layer.

In key-value stores, data is organized in a map fashion: each value is identified by a unique key. Keys are used to insert, retrieve and remove single values, whereas operations spanning multiple ones are often not trivial or not supported at all. Values, associated with keys, can be simple elements such as Strings and Integers, or structured objects, depending on the expressive power of the DBMS considered. We chose Redis as a representative of key-value stores, being one of the richest in terms of data structures and operations. Redis supports various data types such as Set, List, Hash, String and Integer, and a collection of native operations to manipulate them.

Document stores handle collections of objects represented in structured formats such as XML or JSON. Each document is composed of a (nested) set of fields and is associated with a unique identifier, for indexing and retrieving purposes. Generally, these systems offer a richer query language than those in other NoSQL categories, being able to exploit the structuredness of the objects they

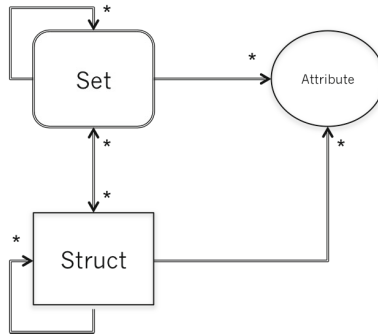
store. Among document stores, MongoDB is one of the most adopted, offering a rich programming interface for manipulating both entire documents and individual single fields.

Extensible record stores offer a relaxation of the relational data model, allowing tables to have a dynamic number of columns, grouped in families. Column families are used for optimization and partitioning purposes. Within a table, each row is identified by a unique key: rows are usually stored in lexicographic order, which enables single accesses and sequential scans as well. The progenitor of this category is Google BigTable [7] and we consider here HBase, which is modeled after it and supports most of the features described above.

Moving from the data models described above, our meta-layer is designed for dealing with them effectively, allowing to manage collections of objects having an arbitrarily nested structure. It turns out that this model can be founded on three main constructs: Struct, Set and Attribute.

Instances of each construct are given a name associated to a value. The structure of the value depends on the type of construct itself: Attributes contain simple values, such as Strings or Integers; Structs and Sets, instead, are complex elements whose values may contain both Attributes and Sets or Structs as well, as shown in Figure 3.

Each database instance is represented as a Set of collections. Each collection is a Set itself, containing an arbitrary number of objects. Each object is identified by a key, which is unique within the collection it belongs to.



**Fig. 3.** The meta-layer

As it turns out, specific non-relational models can be represented as a particular instance of the meta-layer, where generic constructs are used in well-defined combinations and, if necessary, renamed. Simple elements, such as key-values pairs or single qualifiers can be modeled as Attributes. Groups of attributes, as HBase column families or Redis hashes, are represented by Structs. Finally, collections of elements, as HBase tables or MongoDB collections, are modeled by Sets. According to the specific structures the various NoSQL systems implement, we will define a translation process, from meta-layer constructs, to coherent

system-specific structures. In the remainder of this section it will be shown how this translation and data memorization works, moving from the model generic representation to the system-specific ones, with reference to the example introduced in Section 2. The meta-layer representation of the example is shown in Figure 4. A Struct Tweet represents a tweet sent by a User and a Set Tweets contains all the tweets of a User. Finally a simple Attribute is used for every non-structured information item of Users and Tweets (SSN, Name, Surname, ...).

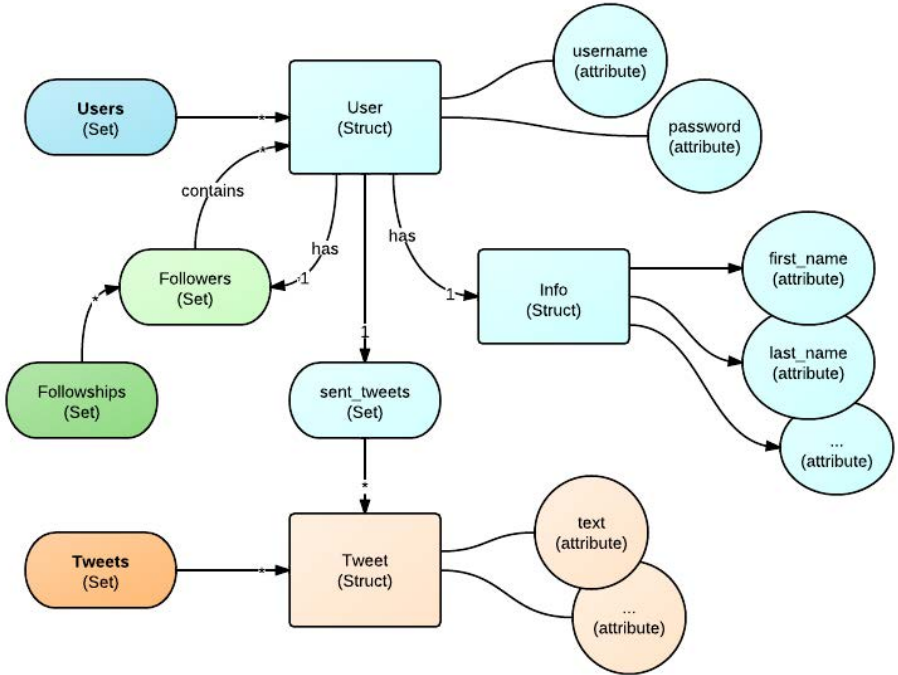


Fig. 4. The meta-layer representation

### 3.1 MongoDB

MongoDB handles Collections of structured documents represented in BSON<sup>8</sup> and identified by a global key. In our translation, every SOS Collection is implemented as a MongoDB Collection. Each object in the meta-layer is then represented as a single document within the collection itself. The structure of each object, implemented in JSON, fits closely the BSON format, allowing seamless translations between the two models.

<sup>8</sup> BSON: a binary encoding of the popular JSON format.

### 3.2 HBase

In this context we model every SOS Collection as a Table. Objects within a Collection are implemented as records in the corresponding Table: the structure of each object establishes which HBase constructs are involved in the translation. Top-level Attributes are stored in a reserved column family named `_top` within a reserved column (qualifier) named `_value`; top-level Sets generate a reserved column family (named `_array[]`) that groups all the fields they further contain. Finally, each top-level Struct is represented as a single column family; deeper elements it contains are further stored in qualifiers. In order to store a nested tree in a flat structure (i.e. the qualifiers map) each field of the tree is given a unique qualifier key made of the whole path in the tree that goes from the root Struct of the column family, to the field itself.

Users				
<code>_top</code>	<code>Tweet[]</code>	<code>Info</code>		
Username: xyz Password: *****	<table border="1"> <tr> <td>Id:1 Content: "abc"</td> </tr> <tr> <td>Id: 2 Content: "def"</td> </tr> </table>	Id:1 Content: "abc"	Id: 2 Content: "def"	Name: Foo Surname: Foo ...
Id:1 Content: "abc"				
Id: 2 Content: "def"				

Fig. 5. HBase Example

An example of translation is shown in Figure 5. The choice of storing top-level elements into different column families is driven by some data modeling considerations. In HBase, column families correspond to the first-class concepts each record is made of; in fact, data partitioning and query optimization are tuned on a per-column-family basis, considering each column family as an independent conceptual domain. In tree documents, we assume that top-level structs and sets correspond to the most significant concepts in the model, and therefore we represent each of them as a single column family.

### 3.3 Redis

Redis, among the three DBMSs we consider, is arguably the most flexible and rich in constructs: it is a key-value store where values can be complex elements such as Hashes, Sets<sup>9</sup> or Lists.

For every object of the SOS Collections of the meta-layer, in Redis two R-Sets are defined. The first one is used to store keys, therefore it indexes the elements. The second one contains data: a Hash named `_top` with Attributes directly related to the Struct and a different Hash for every Struct or Set it contains. The name of those Hashes will be the concatenation of the name of the elements and the contained Struct.

<sup>9</sup> In order to avoid ambiguity, we will use the term R-Set to refer to a Redis set.



The top Hash will contain all the first level attributes as following:

```

hash key:          user:10001
hash values:       [_top, info, twees[], ]

hash key:          user:10001:\_top
hash values:       username: foo
                   password: bar

hash key:          user:10001:info
hash values:       firstName: "abc"
                   lastName:  "def"

hash key:          user:10001:tweets[]
hash values:       [0].tweet.id = "1234",
                   [0].tweet.content = "...",
                   [1].tweet.id = "1235"
                   [1] ...

```

Assumptions we made for defining Redis translation are somehow close to HBase ones, discussed above. In fact, in our translation, Redis Hashes roughly correspond to HBase column families (each containing the qualifiers map). Nevertheless, object data in Redis is spread throughout many keys, whereas in HBase it is contained in a single record. As a consequence, we have to define in Redis support structures to keep track of the keys associated with each object, such as the R-set containing the hash names.

## 4 The Platform

For the definition of the SOS platform, we featured a Java API that exposes the following methods corresponding to the basic operations illustrated in Section 2:

- void put (String collection, String ID, Object o)
- void delete (String collection, String ID)
- Object get (String collection, String ID)
- Set<Object> get (Query q)

The core class is `NonRelationalManager`. It supports `put`, `delete` and `get` operations. Primitives are based on object identifiers, however multiple retrievals are also possible by means of simple conjunctive queries (the second form of `get`). These methods handle arbitrary Java objects and are responsible for their serialization into the target NoSQL system. This process is based on the meta-layer, the data model pivoting the access to the systems. In the current version of the tool, we implemented the meta-layer in JSON, as there are many off-the-shelf libraries for Java object serialization into JSON. The implementation is based on the following mapping between the meta-layer and JSON format:

- Sets are implemented by Arrays
- Structs by Objects
- Attributes by Values

As the final step, each request is encoded in terms of native NoSQL DBMS operations, and the JSON object is given a suitable, structured representation, specific for the DBMS used. The requests and the interactions are handled by technology-specific implementations acting as adapters for the DBMS API.

We have implementations for this interface in the three systems we currently support. The classes that directly implement the interface are the “managers” for the various systems, which then delegate to other classes some of the technical, more elaborate operations.

For example, the following code is the implementation of the `NonRelationalManager` interface for MongoDB. It can be noticed that `put` links a content to a resource identifier, indeed creates a new resource. The adapter wraps the conversion to a technical format (this responsibility is delegated to `objectMapper`) which is finally persisted in MongoDB.

```
public class MongoDBNonRelationalManager
    implements NonRelationalManager {

    public void put(String collection, String ID, Object object) {
        DBCollection coll = db.getCollection(collection);

        ByteArrayOutputStream baos =
            new ByteArrayOutputStream();
        this.objectMapper.writeValue(baos, object);

        this.mongoMapper.persist(coll, getId(ID),
            new ByteArrayInputStream(baos.toByteArray()));
    }
}
```

As a second implementation of `NonRelationalManger`, let us consider the one for Redis. As for MongoDB, it contains the specific mapping of Java objects into Redis manageable resources. In particular, Redis needs the concept of collection, defining a sort of hierarchy of resources, typical in resource-style architectures. It can be seen that the hierarchy is simply inferred from the ID coming from the uniform interface.

```
public class RedisNonRelationalManager
    implements NonRelationalManager {

    public void put(String collection, String ID, Object object) {
        Jedis jedis = pool.getResource();

        try {

            // the object is stored in the meta-layer
            ByteArrayOutputStream baos =
```

```

        new ByteArrayOutputStream();
    this.objectMapper.writeValue(baos, object);

    ByteArrayInputStream bais =
        new ByteArrayInputStream(baos.toByteArray());
    this.databaseMapper.persist(
        jedis, collection, ID, bais);

    baos.close();
    bais.close();

} catch(JsonParseException ex) {
    ex.printStackTrace();

} finally {
    pool.returnResource(jedis);
}
}

```

## 5 Application Example

In this Section we present the actual implementation of the Twitter example mentioned in Section [2](#).

The application can be implemented by means of a small number of classes, one for users, with a method for registering new ones and for logging in, one for tweets with methods for sending them, and finally one for the “follower-followed” relationship, for updating it and for the support to listening. Each of the classes is implemented by using one or more database objects, which are instantiated according to the implementation that is desired for it (MongoDB for users, Redis for tweets, and HBase for the followships). More precisely, the database objects are indeed handled by a support class that offers them to all the other classes.

As an example, let us see the code for the main method, `sendTweet()` for the class that handles tweets. We show the two database objects of interest, `tweetsDB` and `followshipsDB` of the `NonRelationalManager` with the respective constructors, used for the storage of the tweets and of the relationships, respectively. Then, the operations that involve the tweets are specified in a very simple way, in terms of `put` and `get` operations on the “DB” objects.

```

NonRelationalManager tweetsDB =
    new RedisDbNonRelationalManager();
NonRelationalManager followshipsDB =
    new HBaseNonRelationalManager();
...

public void sendTweet(Tweet tweet) {

    // ADD TWEET TO THE SET OF ALL TWEETS
    tweetsDB.put("tweets", tweet.getId(), tweet);
}

```

```

// ADD TWEET TO THE TWEETS SENT BY THE USER
Set<Long> sentTweets =
    tweetsDB.get("sentTweets", tweet.getAuthor());
sentTweets.add(tweet.getId());
tweetsDB.put("sentTweets", tweet.getAuthor(), sentTweets);

// NOTIFY FOLLOWERS
Set<Long> followers =
    followshipsDB.get("followers", tweet.getAuthor());

for(Long followerId : followers) {
    Set<Long> unreadTweets =
        tweetsDB.get("unreadTweets", followerId);
    unreadTweets.add(tweet.getId());
}

tweetsDB.put("unreadTweets", followerId, unreadTweets);
}

```

It is worth noting that the above code refers to the specific systems only in the initialization of the objects `tweetsDB` and `followshipsDB`. Thus, it would be possible to replace an underlying system with another by simply changing the constructor for these objects.

In a technical context, it is clear that an application such as the one described above, can be easily implemented from scratch, given the managers for the various systems. It is important to notice that systems built on this programming model address modularity, in the sense that the NoSQL infrastructure can be easily replaced without affecting the client code.

## 6 Related Work

To the best of our knowledge SOS is the first proposal that aims to provide support to the management of heterogeneity of NoSQL databases.

The approach for SOS we describe here uses the meta-layer as the principal means to support the heterogeneity of different data models. The idea of a pivot model finds its basis in the MIDST and MIDST-RT tools [43]. In MIDST, the core model (named “supermodel”), is the one to which every other model converges. Whereas MIDST faces heterogeneity through explicit translations of schemas, in SOS schemas are implied and translations are not needed. The pivot model, the meta-layer, is used to point out a common interface. Several other differences exist between the two approaches, as we already remarked in the introduction.

The need for a runtime support to interoperability of heterogeneous systems based on model and schema translation was pointed out by Bernstein and Melnik [5] and proposals in this direction, again for traditional (relational and object-oriented) models were formulated by Terwilliger et al. [13] and by

Mork et al. [9]. With reference to NoSQL models, SOS is the first proposal in the runtime direction: in fact, the whole algorithm takes place at runtime and direct access to the system is granted.

From a theoretical point of view, the need for a uniform classification and principle generalization for NoSQL databases is getting widely recognized; it was described by Cattell [6], reporting a detailed characterization of non-relational systems.

Stonebraker [10] presents a radical approach tending to diminish the importance of NoSQL systems in the scientific contest. Actually, Stonebraker denounces the absence of a consolidated standard for NoSQL models. Also, he uses the absence of a formal query language as a supplementary argument for his thesis. Here we move from the assumption that non-relational systems have a less strict data model which cannot be subsumed under a fixed set of rules as easily as for the relational system. However, we notice that commonalities and structural concepts among the various systems can be detected and this can be exploited to build a common meta-layer.

## 7 Conclusions

In this paper we introduced a programming model that enables homogeneous treatment of non relational schemas.

We provided a meta-layer that allows the creation and querying of NoSQL databases defined in MongoDB, HBase and Redis using a common set of simple atomic operation. We also described an example where the interface we provide enables the simultaneous use of several NoSQL databases in a way that it is transparent for the application and for the programmers.

It could be observed that such elementary operations might reduce the expressive power of the underlying databases. Actually, in this paper we do not deal with a formal analysis of information capacity of the involved models. However, it is apparent that when lower level primitives are involved, expressive power is not limited with reference to the whole language, but only to the single statement. This means that a query that can be expressed as one statement in HBase, for example, will require two or more statements in the common query language.

However, what is noticeable is that the standardization is not achieved by adding a supplementary level of abstraction but by spotting common concepts in the models which are deemed general.

To the best of our knowledge this is the first attempt to reconcile NoSQL models and their programming tactics within a single framework. Our approach both offers a theoretical basis for unification and provides a concrete programming interface to address widespread problems.

We enable a sort of federated programming where different NoSQL databases can be used together in a single program. This might be even adopted in the future development of data integration tools where adapters towards NoSQL databases still lack.

The solution proposed here adds an indirection layer to the generally bare-boned NoSQL databases. Obviously, this homogenization presents performance tradeoffs that need to be carefully studied.

## References

1. Abiteboul, S., Buneman, P., Suciu, D.: Data on the web: from relations to semistructured data and XML. Morgan Kaufmann (2000)
2. Atzeni, P., Bellomarini, L., Bugiotti, F., Celli, F., Gianforme, G.: A runtime approach to model-generic translation of schema and data. *Inf. Syst.*, pp. 269–287 (2012)
3. Atzeni, P., Bellomarini, L., Bugiotti, F., Gianforme, G.: A runtime approach to model-independent schema and data translation. In: EDBT Conference, pp. 275–286. ACM (2009)
4. Atzeni, P., Cappellari, P., Torlone, R., Bernstein, P., Gianforme, G.: Model-independent schema translation. *VLDB Journal* 17(6), 1347–1370 (2008)
5. Bernstein, P.A., Melnik, S.: Model management 2.0: manipulating richer mappings. In: SIGMOD Conference, pp. 1–12 (2007)
6. Cattell, R.: Scalable SQL and NoSQL data stores. *SIGMOD Record*, 12–27 (2010)
7. Chang, F., Dean, J., Ghemawat, S., Hsieh, W.C., Wallach, D.A., Burrows, M., Chandra, T., Fikes, A., Gruber, R.E.: Bigtable: A distributed storage system for structured data. *ACM Trans. Comput. Syst.* 26(2), 4:1–4:26 (2008)
8. Codd, E.: Relational database: A practical foundation for productivity. *Commun. ACM* 25(2), 109–117 (1982)
9. Mork, P., Bernstein, P., Melnik, S.: A schema translator that produces object-to-relational views. Technical Report MSR-TR-2007-36, Microsoft Research (2007), <http://research.microsoft.com>
10. Stonebraker, M.: Stonebraker on NoSQL and enterprises. *Commun. ACM*, 10–11 (2011)
11. Stonebraker, M., Cattell, R.: 10 rules for scalable performance in ‘simple operation’ datastores. *Commun. ACM*, 72–80 (2011)
12. Stonebraker, M., Madden, S., Abadi, D.J., Harizopoulos, S., Hachem, N., Helland, P.: The end of an architectural era (it’s time for a complete rewrite). In: VLDB, pp. 1150–1160 (2007)
13. Terwilliger, J.F., Melnik, S., Bernstein, P.A.: Language-integrated querying of XML data in SQL server. In: PVLDB, pp. 1396–1399 (2008)
14. Tsichritzis, D., Lochovski, F.: Data Models. Prentice-Hall, Englewood Cliffs (1982)

# Variability as a Service: Outsourcing Variability Management in Multi-tenant SaaS Applications

Ali Ghaddar<sup>1,2</sup>, Dalila Tamzalit<sup>2</sup>, Ali Assaf<sup>1</sup>, and Abdalla Bitar<sup>1</sup>

<sup>1</sup> BITASOFT, Nantes, France

{ali.ghaddar, ali.assaf, abdalla.bitar}@bitasoft.com

<sup>2</sup> Université de Nantes, LINA, France

ali.ghaddar@etu.univ-nantes.fr, Dalila.Tamzalit@univ-nantes.fr

**Abstract.** In order to reduce the overall application expenses and time to market, SaaS (*Software as a Service*) providers tend to outsource several parts of their IT resources to other services providers. Such outsourcing helps SaaS providers in reducing costs and concentrating on their core competences: software domain expertises, business-processes modeling, implementation technologies and frameworks etc. However, when a SaaS provider offers a single application instance for multiple customers following the multi-tenant model, these customers (or *tenants*) requirements may differ, generating an important variability management concern. We believe that variability management should also be outsourced and considered as a service. The novelty of our work is to introduce the new concept of *Variability as a Service (VaaS) model*. It induces the appearance of VaaS providers. The objective is to relieve the SaaS providers looking forward to adopt such attractive multi-tenant solution, from developing a completely new and expensive variability solution beforehand. We present in this paper the first stage of our work: the VaaS meta-model and the VariaS component.

**Keywords:** SaaS, multi-tenant, variability.

## 1 Introduction

In recent years, the tendency towards outsourcing IT resources that are not the key competencies of an enterprise has caused the appearance of new services providers type. These new services providers develop and maintain, on their infrastructures, such outsourceable IT resources while offering their access to enterprises through the web. The business model of such resources offering aims at providing a pay-as-you-go payment model for their use, where these resources can be provisioned and un-provisioned on demand. Such new outsourcing principle is gaining wide acceptance, especially in the small and medium enterprises (SME) segment. The reason for this acceptance is purely economic. Indeed, the majority of SME consider taking less risks by contracting the implementation of parts of their IT resources to a more experienced provider and find this option more cost effective. The term *Cloud Computing* summarises these outsourcing

efforts and provisioning of services on demand. This term has emerged as the run-time platform to realise this vision [13]. More concretely, Cloud Computing is viewed as a stack containing these new types of services to be provided. That is to provide an Infrastructure as a Service (IaaS), also to provide a development Platform as a Service (PaaS), to finally provide a Software as a Service (SaaS). In the following, we will focus on this last service type.

SaaS can be defined as: "Software deployed as a hosted service and accessed over the Internet" [5]. In the model of this service, the provider offers a complete application, ready to be used by prospective customers. These customers subscribe to the application and use it via the web through a simple browser. Full details on the adopted implementation technology and the application deployment server are transparent. This type of solution is offered by SaaS providers as an alternative to the traditional software applications that require installation on the client side, as well as significant investments in terms of materials and physical resources. However, from a SaaS provider perspectives, the real benefits of SaaS begins when it is possible to host multiple customers on the same application instance, without the need of a dedicated application to be deployed and separately maintained for each customer. This approach is called *multi-tenant*, where tenant means a customer organisational structure, grouping certain number of users [3]. In fact, application tenants could be distributed across different geographic zones and belong to different industry verticals, and thereby, their requirements from the application may differ, increasing the need to support these specific requirements in addition to common ones. This inevitably generates an additional and important *variability* [15] management concern, which obviously distract SaaS providers focus from their core competences.

According to our experience, variability management in multi-tenant applications generally implies: (1) modeling such variability (explicitly documenting the variable features and the possible alternatives), (2) binding the variability model and storing each tenant customisations and choice of alternatives, (3) adapting the appropriate application behavior at run-time for a tenant, according to its stored customisations. This last key concern is tied to the application's architecture. Our work is conducted in a SOA context where the targeted multi-tenant applications are designed and developed following the service oriented architecture (SOA) model. So, such application adaptation usually concern its look and business-processes (services composition an orchestration). As these variability requirements may be distracting and disturbing issues for SaaS providers, we are convinced that they will gain to outsource, if their is a possibility, such variability management concern to a specific service provider. For this purpose, we introduce in this paper the concept of *Variability as a Service (VaaS)* model (in the same spirit of IaaS and PaaS models in terms of outsourced resources). The VaaS model implies the necessity of a new type of service providers: the VaaS providers. A VaaS provider that follows the VaaS model we propose, will permit to its customers (i.e. multi-tenant SaaS providers) to model and to resolve their multi-tenant applications variability on his infrastructure. We will explain



our approach and its context in the following. The remainder of the paper is structured as follows.

Section 2 presents the needs of modeling variability in terms of concepts for modeling and managing; it mainly identifies the limitations that the variability modeling techniques from software product line engineering encounters to realise our objectives from variability outsourcing. Section 3 presents our suggested architecture, process and meta-model for the variability outsourcing and the VaaS model. Section 4 presents a case study from the food industry that we have revisit in order to apply the new approach. In Section 5, discussions and future works are detailed. Section 6 shows related works and Section 7 concludes the paper.

## 2 Needs of Variability Modeling

When many variable features in a software system are identified, variability modeling will become an important requirement to express these features. Generally, variability modeling consists on explicitly documenting, in the software system, what does vary and how does it vary, allowing thus to improve traceability of the software variability, as well as to improve variability communication with the customers [12]. However, since software systems are composed from different development artifacts, a variability concern may impact and cut across these different artifacts (i.e. in SOA-based systems, the same variability in the business-process may have effects on the GUI and may also requires the development and deployment of additional services). Modeling variability in each artifact separately and differently is a very bad idea because it makes us loose all the sense of variability tracking, and it makes representing dependencies between variability in these different artifacts very hard. In addition, the different way of modeling variability in each development artifact will complicates the reasoning about variability in the entire system. Therefore, in order to avoid these problems and to ensure strong expressiveness and management of variability, it is necessary to guarantee its representation in a separated, central and uniform model, while maintaining its relation with impacted artifacts.

### 2.1 Orthogonal Variability Models

Based on the need of tracing dependencies between variability in different development artifacts and to maintain the uniformity in variability definition and modeling, orthogonal variability models such as the OVM have been introduced in the software product line engineering (SPLE) [1, 12] domain. These orthogonal variability models represent variability in one central and separated model, and then link each variability information to one or many development artifacts. In [6], we have shown how OVM can be used to model variability in multi-tenant SOA-based applications. One of our main goals was to show that concepts for variability modeling from the SPLE can be used in a multi-tenant SaaS context. These OVM modeling concepts are resumed in the following:

- *Variation point (VP) and variant*: A *VP* represents one or many locations in the software where a variation will occur, indicating the existence of different alternatives, each of them may result in a different software behavior. A *Variant* represents an alternative of a *VP*.
- *Artifact dependency*: this concept relates the defined variability in the variability model to different development artifacts, indicating which artifacts are representing *VPs* and which ones are realising variants.
- *Constraints dependency*: constraints in OVM can be operated between variants, between *VPs* and between variants and *VPs*: an *exclude* constraint specifies a mutual exclusion; e.g., if variant1 at *VP1* excludes variant2 at *VP2*, the variant2 can not be used at *VP2* if variant1 is used at *VP1*. A *requires* constraint specifies an implication; i.e. if a variant is used, another variant has to be used as well.
- *Internal and external variability*: these two concepts separate between the variability that is only visible to the developers (*internal*), and the variability that is communicated to the customers (*external*).

However, according to our objectives from variability outsourcing, the OVM is not well-suited. In fact, OVM encounters certain limits that prevent its adoption as a variability modeling solution on a VaaS provider infrastructure. These limits result from the differences between the SPLE domain and the multi-tenant SaaS model. In SPLE, the software adaptation to customers requirements is realised through a customisation task that is usually done by the same organisation that has built the software. While in the multi-tenant SaaS model, since all tenants are using the same software instance, it is more practical to give these tenants the control to customise the software to their needs (interventions from the SaaS provider still always required to guide the tenants through the customisation of the application).

## 2.2 Enabling the Software Customisation by Tenants

The *direct* customisation of the software by tenants is one of our objectives since it relieves, in many cases, the SaaS providers from doing this heavy customisation tasks each time a new tenant subscribe to the application. However, such customisation capabilities implies that SaaS providers restrict, for certain tenants, the possibility to choose certain variants which, for example, cannot be selected in the application test period (tenants usually pass through an application test period to determine if the software can provide the service they expect). In addition, some variants may be highly expensive and they must be restricted for certain tenants (in use period) with limited budget. The possibility to define, depending on the tenants contexts, the required restriction conditions for variants and *VPs* is not the focus of OVM, as it assume that the software customisation will be always done by the software creator. In addition, another main reason to propose variability as a service is minimising the number of interventions that SaaS providers have to make for managing variability, and especially, providing concrete data values for variants. Therefore, our variability

modeling solution gives SaaS providers the possibility to define variants with free values, thus tenants can provide the informations they like, such as their own application logo, title, background color etc., without having to always use those offered and predefined by the SaaS providers. Once again, this option is not supported by the OVM. Finally, since the application will be customised directly by individual tenants, it will be important to specify, in the variability model, the order in which the variation points must be bound (Some VPs may depend on each other, which is not always intuitive for tenants). In OVM, such ordering could be done by exploiting the constraints dependencies between VPs, but since constraints in OVM were not designed for this purpose, using them to order the VPs binding will be a workaround. We suggest in the next section, a new variability meta-model that covers the limitations of OVM and others SPLE variability modeling techniques, by bringing additional variability modeling concepts, allowing the SaaS providers to give their tenants certain customisation controls and capabilities in a simple, secure and flexible manner.

### 3 Variability as a Service: Architecture, Process and Meta-model

In this section we will explain in details the *VaaS model* and its core element: the *VariaS component*. The VariaS component provides different interfaces for SaaS providers to create and manage their applications variability models, as well as to resolve variation places in the applications. The VariaS component is deployed and maintained on a VaaS provider infrastructure. The high-level architecture and different actors of VaaS are presented in figure [1](#).

#### 3.1 Architecture and Variability Outsourcing Process

As depicted in figure [1](#), the VaaS architecture defines certain steps that must be performed by SaaS providers as well as by tenants in the variability outsourcing process. These steps are ordered in time and they are divided under *Specification* and *Execution* steps. All mentioned steps ((step 1), (step 2)etc.) and elements ((a), (b), etc.) are shown in the figure.

**Specification.** First of all, the SaaS providers have to instantiate the variability meta-model (a) offered by the VaaS provider (step 1). The variability model resulting from such instantiation is stored in a variability models repository (b). Prospective tenants of the SaaS application (having outsourced its variability management) must bind its variability model (choosing the variants that meet their needs), before they can start its use (step 2). Such variability binding has to be made through the SaaS provider application and then redirected to the VariaS component, since it is more easy and safe that tenants continue to communicate only with their SaaS providers. In addition, such binding (through an integrated variability binding tool) has to be controlled thus tenants will not be able to select non authorised variants and will not have access to certain

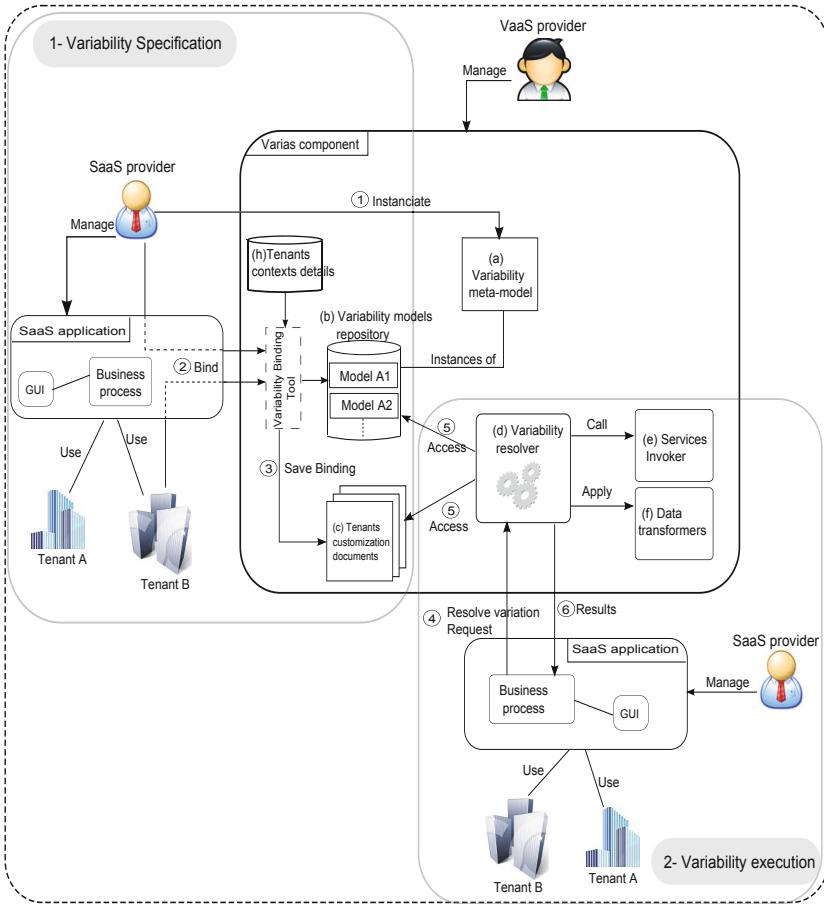


Fig. 1. High-level VaaS Architecture

VPs depending on their contexts and details (h). However, after binding the application variability model, tenants variants choices are saved as customisation documents (c) (step 3). Each customisation document concern one tenant, it indicates their variants chosen at the variation points. Having done so, the tenants can start using their SaaS application and expect from it to behave as they have customised it. Therefore, the application behavior must be adapted at run-time according to these customisations.

**Execution.** In order to adapt the multi-tenant application behavior according to its defined variability model and the tenants customisations, the SaaS developers must request the VariaS component at each variation place they have identified in the application. Each variation place must have a corresponding variation point in the application variability model that is stored on the VaaS

provider side, thus the VariaS component can identify such variation (step 4). These variability resolution requests are in specific forms and use specific protocols defined by the VaaS provider. When a request is received, it will be handled by a variability resolver (d). Such resolver accesses the stored variability models as well as the tenants customisation documents in order to identify the variability action that must be performed (step 5). This action will result in a specific value which is either equivalent to the variant value (selected by the tenant) or deduced from it in case that additional computations based on the variant value must be performed. The variability resolution results will be returned to the application that sent the resolve variation request(step 6). These results must always have the same data format expected by the SaaS application at the corresponding variation place. The non respect of this format may break down the application. In this case, the additional computations to perform on a variant value would be, for example, to transform its data format to the one expected by the application. The expected data format from the resolution results of each variation point must be defined in the variability model.

### 3.2 Variability Meta-model

A variability meta-model is essential for the VaaS model, since it defines how SaaS providers describe their applications variable parts, and how they indicate restricted variants, those accessible to different customisations possibilities. In addition, the variability meta-model has to provide a technical and code level solution for SaaS providers, to help them resolving their application variations while still independent from these applications implementation details. Thus, the VaaS model, through such a meta-model, deals with applications based on standard technologies, such as web-services, and use common applications code artifacts such as texts, files and expressions. In section 2, we have discussed the need of an orthogonal variability model, and shown the limitations that the SPLE modeling techniques encounters for realising our variability outsourcing vision. In the following, new variability modeling concepts related to the VaaS model as well as reused concepts from OVM will be presented. Figure 2 shows our variability meta-model.

The variability modeling approach we present, aims to provide certain level of flexibility to SaaS providers, making them able to deal with complex variability modeling situations in a simple manner. Complex modeling situations such as constraining (by restricting or obligating) the choice of certain variants as well as the binding of certain VPs depending on the tenants contexts are needed. Being able to do so, the *Activation Condition* concept is introduced. Such concept allows SaaS providers to define conditions, which if they evaluate to *true*, their related *Constraints* will be activated. In this way, and depending on the conditions evaluation results, the same variant may be restricted for some tenants and enabled for others. A condition semantic is expressed by a SaaS provider the same as in *If-Then-Else* statements in programming languages. These conditions

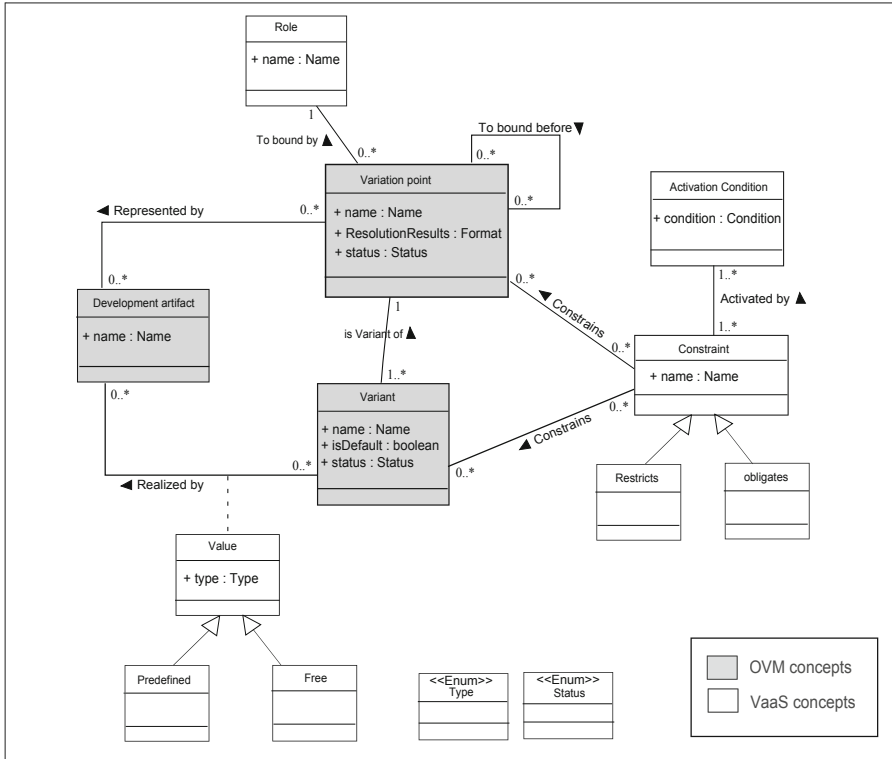


Fig. 2. Variability Meta-model of VaaS

evaluate on informations such as tenant identity, status, already chosen variants and other informations may be useful to limit or extend the software usability for tenants depending on the SaaS provider commercial offer.

The variation point, variant and development artifact concepts are reused from OVM. As mentioned above, variability informations (variants and VPs) are represented in an abstract and artifacts-independent manner, while their relations with the impacted development artifacts is maintained through so called *Artifact Dependency*. However, for each variant (realised by one or many artifacts) its important to indicate its corresponding *value* in each artifact, thus such value can be treated by our service and returned to the SaaS application when needed. The same variant can have different values in different artifacts, each value must have a *type*, thus our variability service can determine the variability action to perform on resolution requests, as well as the appropriate GUI in which the SaaS provider can provide these values. For example, if a SaaS provider creates a value instance for a particular variant with the *Web-Service* type, automatically, a relevant GUI will be opened to provide the web-service description, end point address, operation to invoke, input and output messages

types etc. The same, if a value instance is created with a *File* type, the GUI opened would be a simple file input. These is beneficial for SaaS providers, as *free* values can be added by tenants through dedicated and simple GUI's. Finally, the *Role* concept has been added and associated to a variation point in order to differentiate between VPs to bound by the tenants and others to bound by the SaaS providers. In addition, such role concept is beneficial in case one or many application resellers exist. In fact, the SaaS provider may find more efficient that application resellers make decisions on certain VPs that are related to a given segment specificities which some prospective tenants are parts of. Generally, such resellers are more experienced in these segments standards and rules. In the following, we present a case study that shows an instantiation example of these variability concepts.

## 4 Case Study: A Food Industry Application

In this section, we will revisit a food-industry application (FIA for short) that we have developed as presented in [6]. FIA is a SOA-based multi-tenant application. It allows its food industry tenants looking for foods production and quality improvement to predict the future expenses and benefits from their potential recipes, before executing a real and expensive manufacturing process. FIA evaluates, by relying on an internally developed simulation service, the recipes manufacturing time and cost, as well as their quality characteristics such as nutritional value, taste and smell. FIA has also an integrated shipping service for foods, from the warehouse of the foods supplier to the tenants manufactures. Figure 3 left side shows the FIA business process.

The back-end business logic of the application is implemented using a Business Process Execution Language (BPEL) [9] based solution, and the front-end for the tenants is a Web application which gathers data and hands it over to the BPEL engine for processing the recipes simulation requests. When a tenant user accesses the application, he can manage his existing recipes or decide to create a new one. In this last case, the user has to provide the foods composing the new recipe as well as their respective percentages in it. The user must also describe the recipe cooking process, by sending a pre-negotiated XML structure, or he can rely on a FIA integrated cooking process drawing tool. When finishing, the user must click on simulate button and wait for results. When the application receives the recipe details, the BPEL process invoke the foods supplier service, asking for foods prices and quality informations. These informations, as well as the recipe cooking process allows the simulation service to calculate exactly the final recipe cost and quality. However, when the simulation ends the process sends back a simulation report to the user containing the recipe simulation results. In case user validate the results, the process saves the recipe details in the database and invoke the shipping service thus the foods (composing the recipe) will be shipped from the warehouse of the foods supplier to the tenant manufacture.

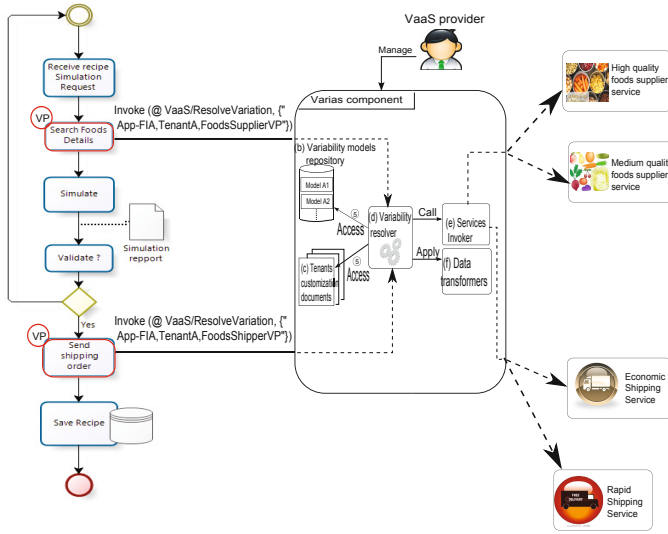


Fig. 3. FIA Business-process and its Interaction with the Varias Component

### 4.1 Modeling FIA Variations Following the VaaS Approach

In order to detect possible variations in the application we have presented its business process to some selected prospective tenants. Three variations have been detected: 1) some tenants are interested by the software but they have asked for another foods supplier with higher foods quality. 2) Some other tenants asked for another foods shipper because the existing one is costly. Taking into account this shipping variation, we have decided to add an economic shipping service to the application but with a higher shipping time. The costly shipping service is very rapid, but it does not supports international shipments. On the other side, we have found that high quality foods suppliers have their own shipping services and that will excludes the need of the shippers proposed by our application. 3) Finally, some tenants want their own company-specific logos and specific titles for the application. Beside these tenant-oriented variations, a segment variation is added to indicate the different simulation models that the application supports. Currently, we are supporting the French and the American models which are the most used worldwide. However, these FIA variations was modeled in first time by relying on the OVM model, and the application customisation was always done by a member of our development team. In figure 4 the FIA variations are modeled following the VaaS approach.

### 4.2 Resolving FIA Variations

In this section we only focus on the business-process variability, as it requires additional computations to perform by the Varias component in order to resolve



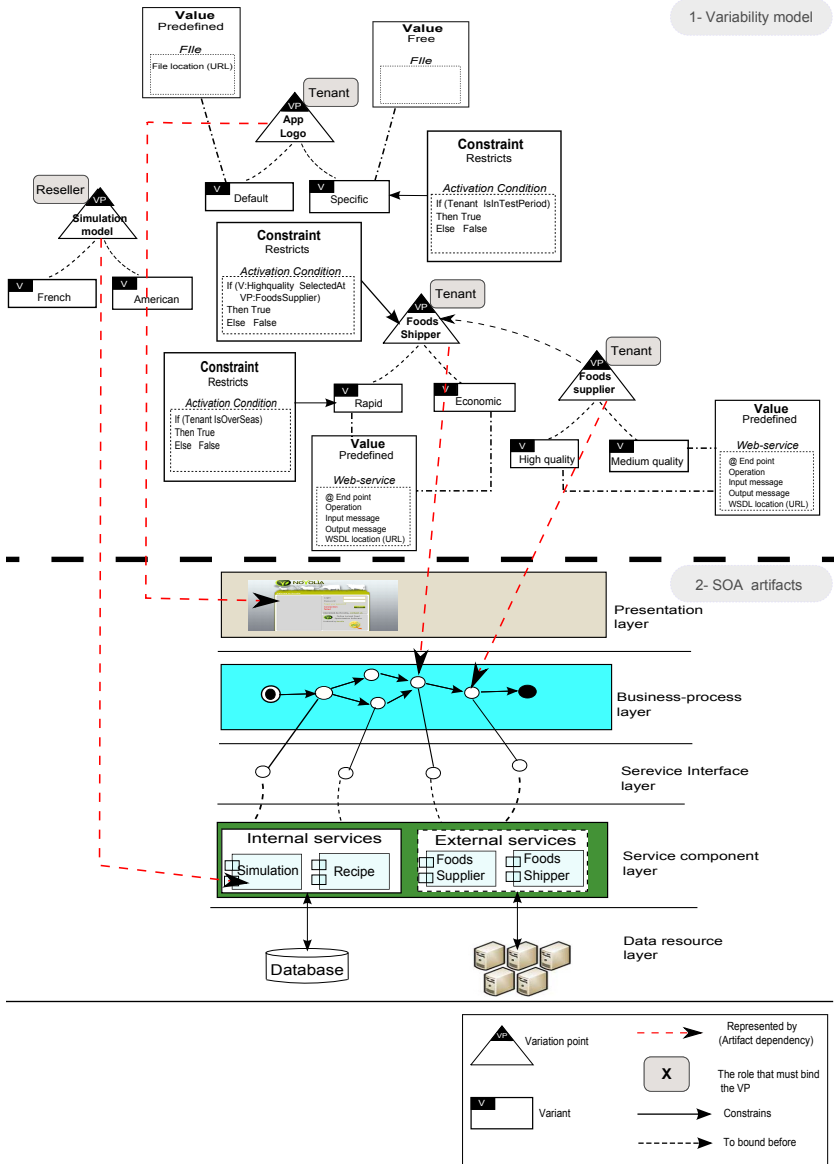


Fig. 4. Modeling FIA Variability Following the VaaS Meta-model

them. In fact, as previously mentioned, we consider multi-tenant applications as SOA-based. In SOA, the application business-process is realised by composing and orchestrating available web-services achieving the desired functionality. For each service in the process there may be different alternatives which implement the service with different implementation logics or quality attributes. In this

case, the variability occurs on selecting the most appropriate service depending on the tenants requirements. If a SaaS provider defines these different services alternatives as variants at a particular variation point, the VariaS component will be responsible of invoking the appropriate service (being able to invoke the appropriate service variant requires providing the informations needed for its invocation, such as end point address, input and output messages, operations etc.) and compensating the data mismatch at the input and output messages of different services, if exists. The services invoker (e) and the data transformers (f) elements in the VariaS component (see figure 1) are designed for this purpose. As depicted in figure 3, the two variation points at different services invocations are resolved by sending requests to the Varias component. These requests are in a fixed form which abstract for SaaS providers the concrete services addresses. In addition, such abstraction is also beneficial for SaaS providers since they always interact with variable services in the same manner and expect the same output messages format.

## 5 Discussion and Future Work

In this paper, we presented the first stages of our work: VaaS or outsourcing *Variability as a Service* with its associated architecture and meta-model. We also presented the VaaS model in its specification and execution steps (see section 3.1) consider SOA applications. We see them as a set of several artifacts, possibly variable. It is important to outline two main hypothesis we voluntarily considered as implicit in this paper:

- *Specification*: a first implicit hypothesis of our work is that variability models of an application and its business design are defined at quite the same time. So the application and its variability models are both defined from scratch and the variability concern is considered at the first stages of the design of the application. This is what we called *early variability* in 6.
- *Execution*: a second implicit hypothesis of our work in its current stage is that the executed application has a stable design and a stable variability. This hypothesis implies that only variants need to be detected, uploaded, executed and potentially evolved. More precisely, in the execution step, we consider only source-code variable artifacts, those executed at run-time. The FIA application is an illustration example through its executable artifacts (i.e. Business-process, web-services, GUI etc.). We also consider that (ii) the application design and its variability models, essentially the View Points, are stable.

At this stage, we aim to stabilise the VaaS concept and VariaS component before exploring the following important future issues:

- *Validation*: we successfully developed multi-tenant applications by including and managing variability through VP and variants. This development has been successfully tested and achieved. The next validation step is to implement the VaaS architecture and its VariaS component.

- *Non-code artifacts*: rather than supporting only code, we aim to generalise the approach to other artifacts like requirements, design and test.
- *VaaS and existing applications*: the objective is to enable the VaaS architecture able to support variability as well for new applications as for existing ones. We thus have to look how one can re-engineer an existing application in order to inject variability in terms of specification and management.
- *Evolution*: applications are in essence evolutive. Managing evolution in a VaaS approach is our ultimate objective. It can be seen through three main concerns: (i) variability evolution, (ii) co-evolution of the application and its variability models and (iii) the co-evolution of the variability meta-model and its variability models.

## 6 Related Work

### 6.1 SOA Variability Management

Several authors studied the variability concern in the context of SOA systems. In [4] authors identify four types of variability which may occur on SOA. In [14] authors present a framework and related tool suite for modeling and managing the variability of Web service-based systems. They have extended the COV-AMOF framework for the variability management of software product families. In [8] authors describe an approach to handle variability in Web services, while [10] focus on variability in business-process by proposing a VxBPEL language. However, all these approaches focus on variability management in SOA systems as a part of the software providers responsibilities which is different from our VaaS and variability management outsourcing approach.

### 6.2 SaaS and Multi-tenancy

In [7] authors provide a catalog for customisation techniques that can guide developers when dealing with multi-tenancy, they have identified two types of customisation: Model View Controller (MVC) customisation and system customisation. In [2] the authors propose some architectural choices to make when building multi-tenant applications, while in [3] authors discuss their experiences with re-engineering an existing industrial single-tenant application into a multi-tenant one. In [11], authors propose a variability modeling technique for SOA-based multi-tenant applications, they differentiate between internal variability only visible to the developers, and external variability that is communicated to the tenants of the application. Once again, they do not propose the variability as a service which we have treated in this paper.

## 7 Conclusion

In this work, we have shown the importance and the complexity of supporting variability in SOA-based multi-tenant applications. We have also motivated the need of outsourcing the variability management to a VaaS provider. Architecture

and meta-model supporting our approach have been also provided. In addition, we have revisited a multi-tenant application case study and transforming its existing OVM variability model into VaaS model. Our approach aims to decrease the variability management complexity, and to relieve the SaaS providers looking forward to adopt a multi-tenant solution, from developing an expensive variability solution beforehand.

## References

- [1] Bayer, J., Gerard, S., Haugen, O., Mansell, J., Moller-Pedersen, B., Oldevik, J., Tessier, P., Thibault, J.P., Widen, T.: Consolidated product line variability modeling (2006)
- [2] Bezemer, C.P., Zaidman, A.: Multi-tenant saas applications: maintenance dream or nightmare? In: Proceedings of the Joint ERCIM Workshop on Software Evolution (EVOL) and International Workshop on Principles of Software Evolution (IWPSE), pp. 88–92. ACM (2010)
- [3] Bezemer, C.P., Zaidman, A., Platzbeecker, B., Hurkmans, T., 't Hart, A.: Enabling multi-tenancy: An industrial experience report. In: 2010 IEEE International Conference on Software Maintenance (ICSM), pp. 1–8. IEEE (2010)
- [4] Chang, S.H., Kim, S.D.: A variability modeling method for adaptable services in service-oriented computing. In: 11th International Software Product Line Conference, SPLC 2007, pp. 261–268. IEEE (2007)
- [5] Chong, F., Carraro, G.: Architecture strategies for catching the long tail. MSDN Library, Microsoft Corporation, pp. 9–10 (2006)
- [6] Ghaddar, A., Tamzalit, D., Assaf, A.: Decoupling variability management in multi-tenant saas applications. In: 2011 IEEE 6th International Symposium on Service Oriented System Engineering (SOSE), pp. 273–279. IEEE (2011)
- [7] Jansen, S., Houben, G.-J., Brinkkemper, S.: Customization Realization in Multi-tenant Web Applications: Case Studies from the Library Sector. In: Benattallah, B., Casati, F., Kappel, G., Rossi, G. (eds.) ICWE 2010. LNCS, vol. 6189, pp. 445–459. Springer, Heidelberg (2010)
- [8] Jiang, J., Ruokonen, A., Systa, T.: Pattern-based variability management in web service development. In: Third IEEE European Conference on Web Services, ECOWS 2005, p. 12. IEEE (2005)
- [9] Jordan, D., Evdemon, J., Alves, A., Arkin, A., Askary, S., Barreto, C., Bloch, B., Curbera, F., Ford, M., Goland, Y., et al.: Web services business process execution language version 2.0. OASIS Standard 11 (2007)
- [10] Koning, M., Sun, C., Sinnema, M., Avgeriou, P.: Vxbpel: Supporting variability for web services in bpel. *Information and Software Technology* 51(2), 258–269 (2009)
- [11] Mietzner, R., Metzger, A., Leymann, F., Pohl, K.: Variability modeling to support customization and deployment of multi-tenant-aware software as a service applications. In: Proceedings of the 2009 ICSE Workshop on Principles of Engineering Service Oriented Systems, pp. 18–25. IEEE Computer Society (2009)
- [12] Pohl, K., Bockle, G., Van Der Linden, F.: *Software product line engineering: foundations, principles, and techniques*. Springer-Verlag New York Inc. (2005)

- [13] Sengupta, B., Roychoudhury, A.: Engineering multi-tenant software-as-a-service systems. In: Proceeding of the 3rd International Workshop on Principles of Engineering Service-Oriented Systems, pp. 15–21. ACM (2011)
- [14] Sun, C., Rossing, R., Sinnema, M., Bulanov, P., Aiello, M.: Modeling and managing the variability of web service-based systems. *Journal of Systems and Software* 83(3), 502–516 (2010)
- [15] Svahnberg, M., Van Gurp, J., Bosch, J.: A taxonomy of variability realization techniques. *Software: Practice and Experience* 35(8), 705–754 (2005)

# Configurable Process Models for the Swedish Public Sector

Carl-Mikael Lönn, Elin Uppström, Petia Wohed, and Gustaf Juell-Skielse

Department of Computer and Systems Sciences, Stockholm University, Sweden  
{cml,elinu,petia,gjs}@dsv.su.se

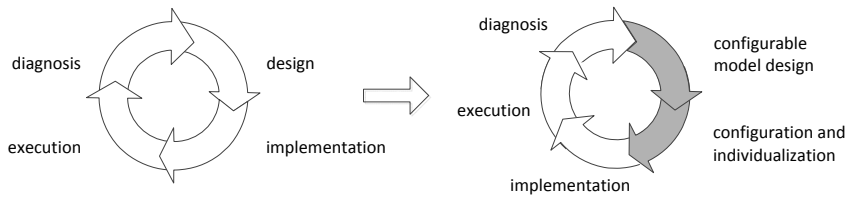
**Abstract.** Process orientation and e-services have become essential in revitalizing local government. Although most municipalities offer similar services there is little reuse of e-services or underlying process models among municipalities. Configurable process models represent a promising solution to this challenge by integrating numerous variations of a process in one general model. In this study, design science is used to develop a configurable process model to capture the variability of a number of different processes. The results include a validated configurable process model for social services, a benefits analysis and directions for future development. Although the results are perceived useful by municipal officials, there are several challenges to be met before the benefits of configurable process models are fully utilized.

**Keywords:** Configurable process models, public administration processes, e-services, e-government.

## 1 Introduction

Local governments in Sweden are responsible for a large share of public services. Care for elderly and disabled account for almost 30% of municipal spending. To meet the challenges arising with an aging population [19], municipalities are working on (i) streamlining administration associated with provision of the public services and (ii) developing e-services. Important goals with the development of e-services is to provide 24/7 accessibility, offer self-service capabilities to citizens and increase transparency between citizens and municipalities. This is made possible by the provision of e-forms and automated, positive, criteria-based decisions where feedback and decisions are electronically communicated almost instantly. To deal with both (i) and (ii) above, we have been involved in process analyses carried out within a project (ÖST) at the social services administration at Järfälla municipality [9]. Järfälla is one of 290 Swedish municipalities and is an average sized municipality situated 20 km northwest of Stockholm.

During the ÖST (Open Social e-Services) project three open social e-services for applications of emergency phone, companion and part time successor services were developed. For the development of each service a thorough analysis of the underlying process was performed with officials from the municipality. During the process analysis a large overlap between the three processes was observed.



**Fig. 1.** The configurable process model life cycle (Source: [15])

To capture the similarities and keep the differences, we postulated that a configurable process model will provide a model at a suitable level of abstraction.

A configurable process model is a general process model that integrates a number of process variants. Decisions regarding variation points in the model depends on requirements of the organization and does not take into account data values that will be available when the process is executed [10]. The configurable process model also maintains domain data on how the individual variants can be derived from the general model. Such derivation is typically called individualization (or configuration) and is done in design time, a priory deployment and run-time, by splitting the PAIS design phase into a configurable model design phase and a configuration and individualization phase as seen in Figure 1.

Configurable models are suitable for the creation of reference models, i.e. general models that aim at capturing processes specific for certain domains. Configurable models are also applicable when a process is with small variation implemented in organizations with different geographical locations where the variations reflect cultural, legislative, or other differences. The situation in the public sector is similar where comparable processes are applied within different administrative regions in a country.

The main point with a configurable process modeling approach is to support process variability. This can be contrasted to the provision of standard solutions aimed at covering the needs of everyone. Within the ÖST project we encountered similar processes and decided to explore the benefits of considering these processes as variants of a general process. Therefore we applied a configurable process modeling approach. In particular we utilized the approach by Gottschalk et al. [4]. Here we report our experiences from this work. Our study is similar to the study reported in [5]. The similarity lies in the utilization of the same technology for creating configurable models. The difference lies in the way the technology is applied. While [5] creates a configurable model for one and the same process running at different municipalities, we create a configurable model based on three different processes sharing a portion of similarities. By doing so we envisage the following benefits.

1. Provide a *comprehensive process model* describing the work routines associated with application handling for social services. This general model will serve as a basis for shared understanding by the stakeholders. It will unite the commonalities and at the same time preserve the differences between the considered processes.

2. Offer *support for individualization*. From the configurable model different individual models capturing the specific processes can easily be produced. To support the individualization we have build a domain model summarizing the knowledge on the differences of the models. In this way the configurable model will be individualized by its stakeholders (without any process modeling training).
3. Simplify *process change management*. General changes, e.g. changes reflecting updates in a law, can be applied on the configurable model, which makes model maintenance easier. Instead of introducing a change into three or several similar models (running the risk of introducing errors or losing consistency) the change is introduced in one place only, i.e. the general model.
4. Increase *process applicability*. Even if we are currently focusing at the needs of one municipality, the configurable model may potentially be beneficial for the other municipalities in the country (as is or after some minor modifications further turning it to a yet more general solutions).

The paper is structured as follows. Section 2 presents the method of our work. Section 3 gives some background knowledge to the tool we have used, i.e. YAWL. Section 4 presents the configurable process model and the domain model we have developed. Section 5 discusses the evaluation of the results. Section 6 presents some challenges we met and how they were tackled. Section 7 presents directions for future work and Section 8 concludes the paper.

## 2 Method

We have followed the Design Science research approach [7]. Using the vocabulary of Hevner et al. [7], the artefact developed during our work is an *instantiation*, i.e. a *proof-of-concept configurable model* for applications of social services in the Swedish public sector. To ensure *research rigor*, we utilize the configurable approach defined in Gottschalk et al. [4] and its implementation in the YAWL environment [3], which we will refer to as C-YAWL. Some other approaches for dealing with variability within business process management have been developed for different process modeling notations, e.g., UML Activity Diagram [18,14], EPC [16], or notation independent approaches such as PROVOP [6]. We selected C-YAWL because: (i) it is an *executable* environment. This feature makes it easier to demonstrate and test practical implications and methodology to users that are unfamiliar with process modeling [4]; (ii) The C-YAWL environment provides model *verification* functionality; (iii) it is *open-source*, which is essential for spreading the results to the other municipalities in the country and (iv) the YAWL notation was earlier used in the project resulting in a number of *process models already being documented in YAWL* and the *familiarity* of the municipal officials with the notation.

The *development* was carried out iteratively. First the three processes models, developed in earlier phases of the project, were integrated to a configurable model. Then the solution was validated through two workshops in which four



municipal officials took part. During the first workshop, the model was presented and discussed with the business users, who were also asked to annotate the solution with any presumptive improvements (which they also did). Based on the input from the users, the model was improved and revisited during the second workshop.

The results were *evaluated* through a combination of analytical and observational evaluation methods [7]. The analytical evaluation was done through the verification functionality in YAWL [8], i.e. we ensured that the produced configurable model is sound. The observational evaluation was carried out through a case study [13]. During the case study the *quality* of the model (i.e. how well it captures the domain under consideration) and the *generality* of the model were studied by exploring the model's capability for capturing a fourth process. When exploring the generality of the model, the postulated benefits 1-3 from Section II were studied.

The case study was carried out through four seminars and one workshop. The four seminars were attended by a working group containing five municipal officials, three customer relationship officers and two financial assistants, representing both the elderly and the handicapped unit at the municipality. During the seminars the configurable model for the three services was extended to cover the fourth service for the control-flow and resource perspectives. Then the data perspective was analyzed and documented. After this, the configuration points were presented and validated. During the workshop, the solution created in the seminars was presented to a broader audience. Initially, the workshop was scheduled to be attended by all municipal officials dealing with applications for social services (i.e. 25-30 employees), but due to unexpected external reasons which we could not influence, the workshop was attended by nine officials only. Among the participants were two managers for the elderly and handicapped administration, one program manager from the handicapped unit, one representative from the IT department, three of the officials from the working group (i.e. customer relationship officers and financial assistants), and the two project managers for the project.

## 3 Background

### 3.1 The C-YAWL Notation

In YAWL [8] a number of constructs are used to create workflow specifications. Figure 2 shows the relevant constructs for this work. These are divided into conditions, tasks and flows. Start and end conditions indicate the beginning and the end of a process. Tasks might have decorators. The decorators are used to capture joins and splits of the control flow. There are three types of decorators - AND, XOR and OR decorators - each coming in two variants, join and split. We have used colors to show the different roles responsible for the execution of the tasks.

C-YAWL [8] introduces the concept of port, i.e. a possible incoming or outgoing path of a task. For instance a XOR-join task with two incoming flows have



**Fig. 2.** The C-YAWL notation

two ports associated to it, one for each alternative paths enabling the task, while an AND-join task have one incoming port associated to it. C-YAWL contains in addition two settings, *blocked* and *hidden*, that may be used on the ports. *Hidden* on an incoming port indicates that the relevant task will be skipped and the flow continued. *Blocked* on an outgoing port indicates that the subsequent tasks on the path are disabled and will never execute. To keep the configuration process sound, certain consistency between the configuration settings on incoming and outgoing ports is necessary. For example, a blocked outgoing port on a task will entail a blocked incoming port on the consecutive task(s).

### 3.2 Steering Configurations through Questionnaires

In [17] it is argued that the variability of a domain should be separated from the process model. This is to relieve the need for modeling expertise during the individualization of a configurable model for a specific context. Furthermore, it would be rather complex to evaluate the soundness and correctness of a *process model* and associated *domain constraints* restricting it at the same time. Therefore a mechanism for keeping a process model apart from the domain constraints associated to it is proposed in [17].

A *domain questionnaire model* contains a set of *questions* with related facts, both specified in natural language. The questions may be dependent on each other, hence the order in which they should be presented (or asked) to business users needs to be defined. The *facts* are boolean variables used to describe different features of a specific domain. They can be either disabled or enabled and they shape the space of all possible answers to a question. By answering a specific question the values of all facts related to it are set at once. A domain questionnaire model encapsulates all domain specific constraints needed to configure a general process model capturing the same domain.

To capture the relationship between process model and constraints, a *restraints mapping* using standard logical expressions is used. Apart from solving the interdependency issues it provides the model with a certain flexibility, making it possible to individualize a number of different process models for a given domain in an automated fashion without compromising the correctness of the domain or the process model.

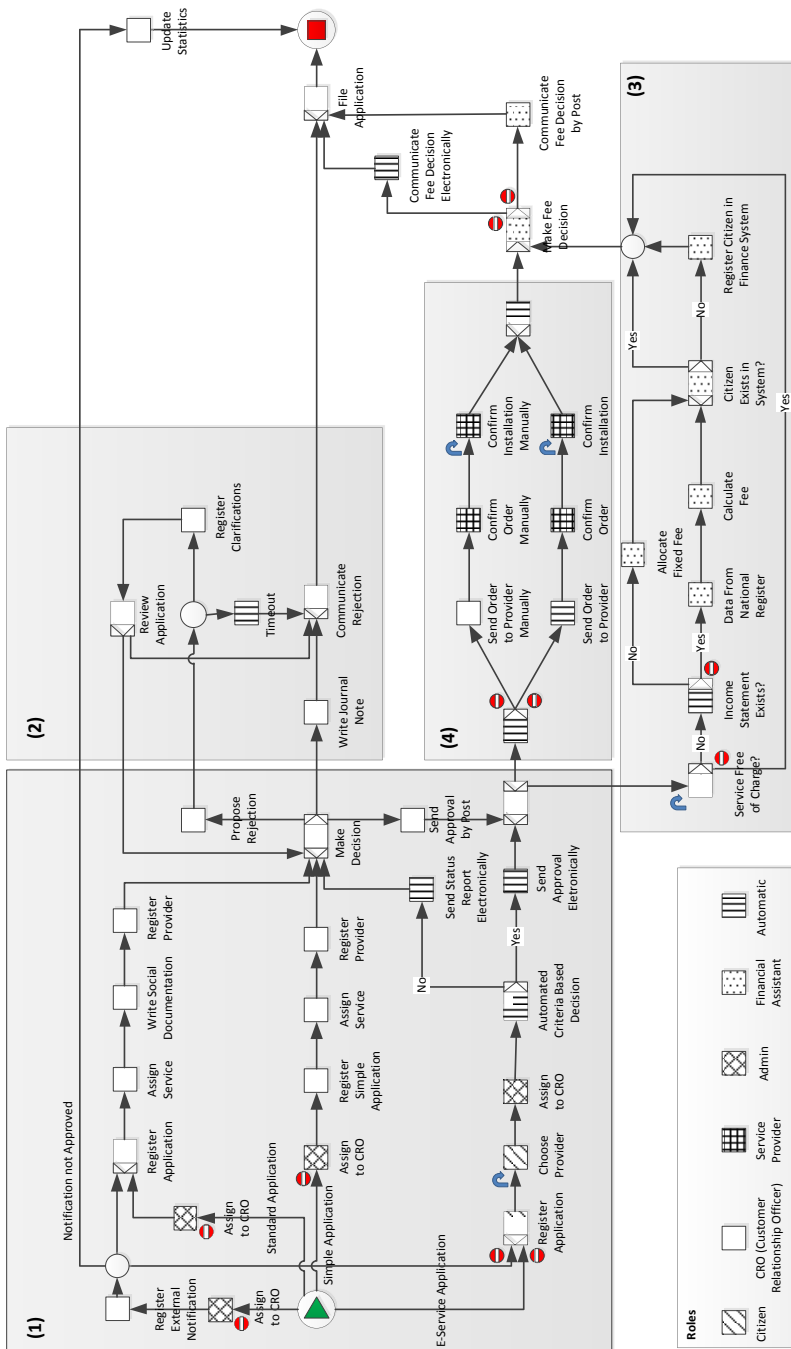


Fig. 3. The configurable process model

## 4 The Configurable Process Model

### 4.1 Process Model

In this work, a configurable model was created by merging three processes for handling of citizens applications for three social services namely *part-time successor*, *companion service* and *emergency phone*.

The *part-time successor* is a relief service for people who care for an elderly or disabled person. The *companion service* offers assistance to elderly or disabled people to participate in activities outside of their homes. These two social services are restricted to a certain amount of hours per month. They are payed by the subscribers based on the amount of utilized hours. The *emergency phone* service means that service subscribers can call for assistance 24/7 if something unexpected happens which they can not deal with on their own. This service requires extra equipment in form of an alarm watch linked to a phone. The installation is taken care of by the service provider company. The service costs a fixed monthly fee.

These are social services, which means that they are sponsored by the local government. The role of the municipality is to determine which applicants are entitled for the services and how much they should pay (economically weak applicants are entitled to pay less). Also to ensure that the service subscriptions are allocated to the correct service provider and to follow up on the provision of the services are obligations carried out by the municipality. The applications for the services can come in three different forms, as an *e-service application*, as a *simplified application* or as a *standard application*. The underlying processes with their different variations are depicted in Figure 3. Area (1) in this figure shows the three variants of receipt and registration of an application, as well as the decision made for an application. Positive decisions lead to two parallel treads of activities: contacting a service provider (Area (4) in the figure) and calculating and deciding on the service fee (most of the relevant activities in this part of the process are captured in Area (3) in the model). Finally, the activities associated to negative decisions are depicted in Area (2) in the model.

A common feature for the three e-services is that positive decisions are made automatically, based on a set of predefined criteria. The criteria are specific for each service and visible for the applicants. Some overlaps are observed between the criteria for the three services, but there are also differences. The intention with the predefined criteria is to increase the transparency towards the citizens as well as to filter and quickly handle the obviously positive decisions. Due to the availability of criteria and automated positive decisions, the services are called open social services. Negative decisions are never taken automatically.

The model was developed following the process modelling guidelines in [11]. We have used as few elements as possible in the model but a few tasks have been duplicated. This was done to reduce the number of routing paths and to reinforce the comprehensiveness (clearness) of the model.

## 4.2 Domain Questionnaire Model

To develop the domain questionnaire model for the case study we used the questionnaire designer tool in the Synergia toolset [15]. The model was derived by using material gathered from workshops and in depth interviews with representatives from all parts of the social services department at the municipality.

A *companion service* application is processed in the same fashion as a *part-time successor* application with the difference that the service is not offered free of charge, therefore a question mapping to this variation point is included in the domain questionnaire model. An *emergency phone* application is handled similarly, except that a provider for the service can not be chosen but is predefined, and thus a question regarding the possible choice of provider is also included in the domain questionnaire model. The service requires equipment installation, which is not needed in the other services and again this question exist in the domain questionnaire model. Furthermore, when citizens apply for *emergency phone* service through the e-service the fee is fixed and cannot be reduced. To capture this variation, there are two questions where one depends on the other in the domain questionnaire model. Besides these differences the municipality performs the same flow of activities for all three services. However, some questions that guide for example different type of services also have to be included in the domain questionnaire model. One of the individualized models, the *emergency phone* can be seen in Figure 5.

The questionnaire model is depicted in Figure 4, green boxes represents domain questions, yellow ones represents the facts connected to a question and arrows imply that a dependency exist between questions. To relate the domain questionnaire and process configuration models a mapping between the two was performed in two steps: first the process model was used to map constraints into the domain questionnaire model and then the questions in the created domain questionnaire model were mapped to each variation point in the process model. These mappings were done manually as the mapping tool described in [17] for YAWL was released just after the study took place. Therefore the mapping verification functionality in the tool could not be utilized but instead the verification was carried out analytically.

## 5 Evaluation

The evaluation of the results was carried out through a case study, during which we validated the *quality*, *generality* and the *perceived usefulness* of the general model and the configuration approach.

### 5.1 Quality

The soundness of the configurable model was ensured through the verification functionality in YAWL. The ability of the model to capture the domain was ensured through input from the municipal officials and middle level management.

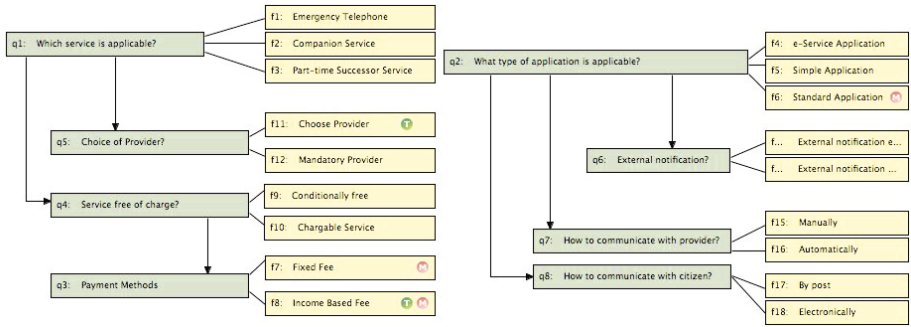


Fig. 4. Domain Questionnaire Model

To ensure quality in the received responses it was important to control that the respondents had a good understanding of the general model and the configuration of it. For this reason we used control question to check their understanding and we asked them to make suggestions for changes directly in the model. The respondents answered the control questions properly and also managed to apply suggested improvements directly in the model. They also asked very precise and logical questions e.g. “Is it possible to automate the Register Citizen in Finance System task?”, “Make Fee Decision task has now been changed to be carried out by economy assistants!”, “The task Write Social Documentation is done before Register Provider task” which showed they understood the model well.

### 5.2 Generality

The results from the study show that the configurable process model is general enough to cover a number of processes within the elderly and handicapped sector at Järfälla municipality. However, the model is not yet proven to be complete, as there is one additional process which we have not yet studied.

The generality of the model was evaluated by studying the extensibility of the model for accommodating a fourth process, i.e. the application handling process

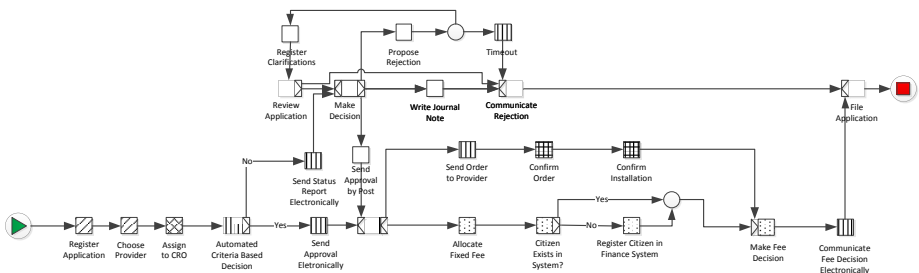


Fig. 5. The emergency phone e-service application - individualized process

for *home-care* service. The application handling is similar to the handling of an *emergency phone*, *part-time successor*, and *companion* services applications. A difference is that the *home-care* service includes a number of different service efforts, e.g. cleaning, window-cleaning, laundry, grocery shopping and meal delivery. It is possible to apply for one or several of these service efforts in the same application, however different levels of granularity did not have to be considered due to the fact that the decision made is for the *home-care* service as a whole. It was concluded that neither the control flow nor the allocation of resources in the configurable model had to be adapted in order to accommodate the *home-care* service. The configurable model was general enough in those two perspectives to completely capture the new process. Modifications in the data perspective, to accommodate decision criteria specific for the *home-care* service and the choice of what specific service efforts to apply for were on the other hand needed. As a new process always implies new data that will have to be taken care of, this would not have been possible to avoid.

Regarding the completeness of the solution, it should be noted that the set of processes studied so far is not exclusive. To make the analysis exclusive, a fifth process needs to be added to the solution, namely the process of handling applications for the *personal-care* service. In contrast to the other services incorporated in the model so far, the municipality will not offer the *personal-care* service as an open service, i.e. automated positive decision making based on criteria will not be implemented for this service. This means that to provide the application of *personal-care* service as an e-service, the model in Figure 3 would need to be extended with a flow from the e-service application thread (bottom left corner in the model) to the manual “Make Decision” task in the middle of the model, and a hiding configuration would be added to the “Automated Criteria Based Decision” task. I.e. the configurable model could incorporate the new process with only minor modifications. This extension was discussed with the municipal officials in order to highlight the difference between open-services and e-services - two concepts that until now were used as synonyms by the municipality. In order to fully extend the solution for the *personal-care* service, the analysis for the data perspective need to be added. Nonetheless, the small impact on the model for incorporating the proposed extension from open e-services to e-services confirms the expected benefit on process change management (see Section 1).

To validate the questions in the domain model, we highlighted the configuration points and demonstrated the questions and how the answers individualized the configurable model. The individualization example we used was for the emergency phone e-service application, depicted in Figure 5. The questions got a good acceptance by the officials. They were considered as a natural way to capture the variations between the processes. This confirmed our expectation of easiness to produce configurations and support the envisaged individualization benefit discussed in Section 1.

### 5.3 Perceived Usefulness

The configurable process model is perceived useful since it promotes shared understanding and simplifies process analysis (see the discussion on the comprehensive process model benefit in Section 4). With regard to perceived usefulness, we distinguish between impressions from the officials that were earlier involved in the analysis of the three other application processes and the officials that were new in the project. The first group of officials emphasized that it was less time-consuming to analyze the underlying process for *home-care* application than for the first three processes. The officials believed that this is due to the fact that the configurable process model served as a “great starting point”. Another contributing factor to the timesavings was reuse of concepts and terms that had already been defined and built into the process model. The configurable process model was also appreciated by the officials new in the project. They valued the model because it gave them a good and comprehensive picture of the work routines and the rules in place.

To further test the *perceived usefulness* of the configurable model we asked the municipal officials to comment on the usefulness of the integrated process model. The immediate reaction from three officials was “It is nice to see such a flow describing what we actually do”, “It gives a comprehensive picture of our work”, and “It is good because our managers can see what we are doing”. We also asked if the municipal officials would like to have posters with the processes, and if so which ones i.e. the general configurable process or the four configurations for the e-services. The answers from the working group were unanimous. The officials wanted all five models, i.e. both the general model and the four e-services configurations of it, as posters. This answer confirms the perceived usefulness of the configurable model for the social sector at Järfälla municipality.

## 6 Challenges

During our work we met a number of challenges that can be classified into the following areas: challenges related to the process modeling and execution paradigm, challenges associated to the configurable approach we have utilized, and challenges associated to the tools we used.

One of the main challenges related to process modeling and the execution paradigm during our work has been the difficulty in getting an understanding of the software support that a business process management system offers to a local government. At Järfälla municipality work has been made for streamlining the processes, however, the current (legacy) IT system at the municipality does not offer any support for these new processes. Therefore it is a challenge to reach an understanding of potential benefits in utilizing a process centric tool such as YAWL beyond that of being able to conceptualize and make the work processes visible to the municipal officials. The same difficulty has been observed regarding requirements specification for the e-services. Even though a validated process model with data and roles has been developed and used as requirements



documentation towards the IT-service providers involved in the project, the process model has not been used as a blueprint for the actual development of the e-services it describes.

Challenges associated to the configuration approach mainly have to do with difficulties in capturing all variation points in the general model while at the same time maintaining an understandable and sound model. Since YAWL does not support configuration of data or resources we had to make workaround solutions that enlarged the model and duplicated a number of tasks. This was necessary in order to make the role configuration visible on a conceptual level. On the data level we made the decision of not capturing the configurations by multiplying tasks where data differed since this would have made the general model complex and difficult to understand. Also, since the municipality does not have a business process management system, the necessity of capturing data configurations in the model using a workaround solution was not considered as vital at this stage, although one of the most important arguments for choosing YAWL, that it is an executable notation, thereby became weaker. However, we still consider the executable feature of the YAWL notation an important argument for continuous use. The reason is that configuration on resource- and data level has already been solved conceptually [16] and only the implementation in the YAWL engine remains.

Challenges associated to the tools we used were mostly due to the fact that these are still at a prototype stage of development. For instance, apart from the configurable model capturing all variation points, we had to construct nine configurable models, one for each individualization we wanted to define. This was because the configuration feature of the YAWL editor only supports an on/off switch to apply all or none of the configurations made on a model. When the configuration is turned on all hides and blocks in the model are applied. Furthermore, the verification of models containing OR -splits and -joins is a known problem that is highlighted in [212]. In order to use the verification functionality in YAWL to guarantee the correctness of the configurable model, we worked around all OR-splits and OR-joins, present in the model initially, by replacing them with corresponding constructs. By ensuring the correctness of the configurable model, we could guarantee the correctness of all individualized models extracted from it [1].

## 7 Where Next?

Future work can be undertaken in two directions. The first line of work is to further extend the configurable model presented in this paper. The model now covers four processes in three different variants each. A beneficial extension would be to expand the general model to include e-services for processes dealing with the applications for *personal-care* services. As mentioned in Section 5, further configurations of resources and changes in the control-flow is expected to be minor. However a more detailed analysis of data may lead to more significant extensions of the process model, hence such extensions would deserve a dedicated



## 8 Conclusion

In this paper we have utilized a configurable process modeling approach, suggested in [4] and developed a general process model covering three different processes for the social services department at a municipality in Sweden. While this approach has earlier been used for integrating variants of one process for example in [5], the uniqueness of our work is that the approach has successfully been implemented to integrate a number of different but highly similar processes. We envisaged four potential benefits of developing a general model for the social services sector using a configurable approach.

The model and the approach have been evaluated in workshops where the understanding of the solution and benefits with it have been studied. Also a fourth process have been incorporated into the general model, which was possible without any changes to the control-flow and resource perspectives of the model. The evaluation shows that using configurable process models facilitate understanding and organizational change within the social services department. The general model is regarded as valuable for gaining a mutual understanding of how the work should be carried out and for educating new employees. Thereby the first benefit has been shown to be fulfilled. The use of questions that capture the domain restrictions applicable to the model has also been found valuable and is regarded as easily understood by the social services personnel wherefore we also consider the second benefit fulfilled.

During the evaluation work, and the incorporation of a fourth service into the general model, some legal changes has been observed. Also discussions on how to further utilize the electronic services by expanding their use to include services other than the open social e-services and the visualization of these envisioned changes into the general model points towards the third benefit being realistic, although the full advantage of it would be greater if a business process management software, supporting the processes, was used in the municipality.

Our work also shows that in order to take full advantage of the configurable approach, software support that enables the use of executable processes is needed. One of the main challenges identified is the old legacy systems used in the municipality, systems that do not support process orientation and are a great obstacle in making the social services enabled for e-government. The old IT-systems also work as a hurdle against the understanding of how IT can be a support for the local government. Today most social services officials consider IT more as a hindrance, slowing down and creating unnecessary frustration in their daily work. In our further work to realize the fourth benefit we expect this to be a challenge since the usage of old IT-systems is a reality in most municipalities in Sweden. The conclusion is thereby that out of the four envisaged benefits, two are considered fulfilled, the third considered realistic and the fourth needs further study in order to be proved.

A configurable process model can potentially become complex for example depending on the complexity of the domain and the number of variations points. It is also shown in [20] that the complexity of a configurable model is related to the similarity of the models that has been integrated to create the configurable

model. If the dissimilarities between the integrated models are too many the configurable model becomes too complex to verify and validate. We have not seen any technical limitations of the configurable approach due to complex models, which might be due to the relatively small size (46 tasks) of the developed model. It could be argued that the generality of the study to cover larger and more complex models are questionable. However we consider it to be representative for two reasons: (1) The model makes use of all elements available in the configurable process approach. (2) The size of the model is not unusual if compared to other models collected from the public sector [20,5].

During our work with the configurable approach used in this study we have not recognized any concrete pitfalls or limitations affecting the feasibility of configurable models in the domain of the Swedish public sector. The question remains as to whether it is possible to incorporate a larger number of processes into the general model or if the dissimilarities between these models are so great that it becomes an obstacle? Our belief is that the developed configurable model will be able to cover a large share of processes within the public sector domain in Sweden.

**Acknowledgement.** We would like to thank all officials from Järfälla municipality who participated in the project for their deep involvement in the work.

## References

1. van der Aalst, W.M.P., Dumas, M., Gottschalk, F., ter Hofstede, A.H.M., La Rosa, M., Mendling, J.: Correctness-Preserving Configuration of Business Process Models. In: Fiadeiro, J.L., Inverardi, P. (eds.) FASE 2008. LNCS, vol. 4961, pp. 46–61. Springer, Heidelberg (2008)
2. van der Aalst, W.M.P., Dumas, M., Gottschalk, F., ter Hofstede, A.H.M., La Rosa, M., Mendling, J.: Preserving correctness during business process model configuration. *Formal Asp. Comput.* 22(3-4), 459–482 (2010)
3. YAWL foundation. YAWL: Yet Another Workflow Language, [www.yawlfoundation.org](http://www.yawlfoundation.org) (accessed November 2011)
4. Gottschalk, F., van der Aalst, W.M.P., Jansen-Vullers, M.H., La Rosa, M.: Configurable Workflow Models. *Int. J. Cooperative Inf. Syst.* 17(2), 177–221 (2008)
5. Gottschalk, F., Wagemakers, T.A.C., Jansen-Vullers, M.H., van der Aalst, W.M.P., La Rosa, M.: Configurable Process Models: Experiences from a Municipality Case Study. In: van Eck, P., Gordijn, J., Wieringa, R. (eds.) CAISE 2009. LNCS, vol. 5565, pp. 486–500. Springer, Heidelberg (2009)
6. Hallerbach, A., Bauer, T., Reichert, M.: Guaranteeing Soundness of Configurable Process Variants in Provop. In: Hofreiter, B., Werthner, H. (eds.) Prof. of IEEE Conference on Commerce and Enterprise Computing, pp. 98–105. IEEE Computer Society (2009)
7. Hevner, A.R., March, S.T., Park, J.: Design Science in Information Systems Research. *MIS Quarterly* 28(1), 75–106 (2004)
8. ter Hofstede, A., van der Aalst W.M.P., Adams, M., Russell, N. (eds.): *Modern Business Process Automation: YAWL and its Support Environment*. Springer (2010)

9. Juell-Skielse, G., Wohed, P.: Design of an Open Social E-Service for Assisted Living. In: Wimmer, M.A., Chappelet, J.-L., Janssen, M., Scholl, H.J. (eds.) EGOV 2010. LNCS, vol. 6228, pp. 289–300. Springer, Heidelberg (2010)
10. Dumas La Rosa, M., van der Aalst, W.M.P., ter Hofstede, A.H.M.: Questionnaire-based Variability Modeling for System Configuration. *Software and Systems Modeling (SoSyM)* 8(2) (2009)
11. Mendling, J., Reijers, H.A., van der Aalst, W.M.P.: Seven process modeling guidelines (7pmg). *Information & Software Technology* 52(2), 127–136 (2010)
12. Mendling, J., van Dongen, B.F., van der Aalst, W.M.P.: Getting Rid of the OR-Join in Business Process Models. In: 11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007), pp. 3–14. IEEE Computer Society (2007)
13. Neale, P., Thapa, S., Boyce, C.: *Preparing A Case Study: A Guide for Designing and Conducting a Case Study for Evaluation Input*. Pathfinder International (May 2006) (accessed November 2011)
14. Razavian, M., Khosravi, R.: Modeling Variability in Business Process Models Using UML. In: Proc. of 5th Int. Conf. on Information Technology: New Generations. IEEE Computer Society (2008)
15. La Rosa, M.: Process Configuration.com, [www.processconfiguration.com](http://www.processconfiguration.com) (accessed November 2011)
16. La Rosa, M., Dumas, M., ter Hofstede, A.H.M., Mendling, J., Gottschalk, F.: Beyond Control-Flow: Extending Business Process Configuration to Roles and Objects. In: Li, Q., Spaccapietra, S., Yu, E., Olivé, A. (eds.) ER 2008. LNCS, vol. 5231, pp. 199–215. Springer, Heidelberg (2008)
17. La Rosa, M., Lux, J., Seidel, S., Dumas, M., ter Hofstede, A.H.M.: Questionnaire-driven Configuration of Reference Process Models. In: Krogstie, J., Opdahl, A.L., Sindre, G. (eds.) CAISE 2007. LNCS, vol. 4495, pp. 424–438. Springer, Heidelberg (2007)
18. Schnieders, A., Puhlmann, F.: Variability Mechanisms in E-Business Process Families. In: Abramowicz, W., Mayr, H.C. (eds.) Proc. of 9th Int. Conf. on Business Information Systems. LNI, vol. 85, pp. 583–601. GI (2006)
19. Statistics Sweden. The future population of Sweden (2009), [www.scb.se/statistik/\\_publikationer/BE0401\\_2009I60\\_BR\\_BE51BR0901ENG.pdf](http://www.scb.se/statistik/_publikationer/BE0401_2009I60_BR_BE51BR0901ENG.pdf) (accessed July 15 2010)
20. Vogelaar, J., Verbeek, H.M.W., Luka, B., van der Aalst, W.M.P.: Comparing Business Processes to Determine the Feasibility of Configurable Models: A Case Study, pp. 50–61 (2011)

# Aligning Software Configuration with Business and IT Context

Fabiano Dalpiaz<sup>1</sup>, Raian Ali<sup>2</sup>, and Paolo Giorgini<sup>1</sup>

<sup>1</sup> University of Trento, Italy

{dalpiaz,paolo.giorgini}@disi.unitn.it

<sup>2</sup> University of Bournemouth, United Kingdom  
rali@bournemouth.ac.uk

**Abstract.** An important activity to maximize Business/IT alignment is selecting a software configuration that fits a given context. Feature models represent the space of software configurations in terms of distinguished characteristics (features). However, they fall short in representing the effect of context on the adoptability and operability of features and, thus, of configurations. Capturing this effect helps to minimize the dependency on analysts and domain experts when deriving a software for a given business and IT environment. In this paper, we propose *contextual feature models* as a means to explicitly represent and reason about the interplay between the variability of both features and context. We devise a formal framework and automated analyses which enable to systematically derive products aligned with an organizational context. We also propose FM-Context, a support tool for modeling and analysis.

**Keywords:** Variability, Product Lines, Business/IT Alignment.

## 1 Introduction

Business/IT (B/I) alignment concerns the effective usage of Information Technology (IT) in a business environment [9]. Homogenizing software with the business and technical context of an organization is a major B/I alignment challenge which requires a joint effort of business managers and IT administrators. This task is particularly difficult due to cultural differences between the involved actors (the so-called B/I gap [14]) and the evident—though fuzzy—mutual impact between IT and business aspects. A main challenge for information systems engineering is to bridge this gap minimizing mismatches and maximizing alignment.

Software product lines [3] and feature models [10] are a development paradigm and a modeling notation that support the configuration of software products from reusable assets. A product is generated for each organization depending on its business and IT profile. Existing configuration approaches, e.g., [11, 5], provide limited automated support to achieve B/I alignment. Thus, they heavily rely on the skills of analysts and domain experts. This calls for the development of systematic approaches and CASE tools to identify a configuration that fits an organizational context both at the business and IT level.

The contribution of this paper is a modeling and analysis framework which enables the configuration of a software aligned with an organizational context. We propose Contextual Feature Models (CFMs), which extend traditional feature models with the concept of *context*. This enables minimizing the efforts of domain experts when configuring a product to a specific environment: CFMs express how the organizational context affects the inclusion/exclusion and the applicability of features. We distinguish between two types of context:

- *Business context* concerns the organizational characteristics of an enterprise and helps to determine whether a feature is required or advisable. For instance, a feature like “LDAP server” may be advisable in a business context where “the company has multiple branches”;
- *Technical context* concerns the technical infrastructure required to operationalize a feature. For example, a feature “bug tracking system” may require a technical contexts “MySQL” and “PHP  $v \geq 5$ ” to be successfully deployed.

We develop automated analysis techniques to examine the interplay between features and context. Our techniques enable maximizing B/I alignment, as they support the selection of a configuration that fits well with a given environment:

- **Configurations:** given an organizational environment, in terms of technical and business contexts, we identify possible software configurations. Among these configurations, the analyst can select the best-quality one;
- **Business-to-IT alignment:** we identify an IT infrastructure that supports all software configurations fitting with a given business context, as well as the minimal IT technical requirements to support the business context;
- **IT-to-Business alignment:** given an IT infrastructure, described in terms of technical contexts, we identify the business contexts that support such infrastructure. This analysis serves to compare candidate IT infrastructures so to maximize the support of the business contexts the client cares about.

We also develop FM-Context, an Eclipse-based CASE tool which allows for drawing CFMs and implements our analysis techniques on top of a Datalog solver. We illustrate our framework using the following scenario.

**Motivating Scenario.** Drupal<sup>1</sup> is an open-source content management system with wide industrial adoption. Several web development companies base their products upon a Drupal-based product line (over 13.000 modules are available for use). The challenge is about identifying a configuration of Drupal which is aligned with the business and technical context of a client organization. We model Drupal configurations in a CFM, and illustrate how our automated analyses support the derivation of a configuration highly aligned with an organizational context. □

The paper is structured as follows. Sec. 2 introduces CFMs and applies them to the Drupal scenario. Sec. 3 describes a formal framework for CFMs and presents automated analysis techniques built on top of it. Sec. 4 details FM-Context and reports on preliminary scalability results. Sec. 5 presents related work, while Sec. 6 draws our conclusions and presents future directions.

<sup>1</sup> [www.drupal.org](http://www.drupal.org)

## 2 Contextual Feature Models (CFMs)

Feature models are a compact and intuitive modeling language to represent variability—the existence of multiple configurations—in software product lines. Many features, however, are not always applicable or advisable: they are context-dependent. A feature may be *advisable* only in a business context: “semantic search engine” could be advisable only in business context “the customer has a vast catalog”. A feature may be *applicable* only in a technical context: “role-based access control” could require technical context “mysql v≥5”. Such constraints apply to configurations too, as a configuration is a set of features.

Existing feature modeling approaches pay no or limited attention (see Sec. 5) to the interplay between features and context. Most approaches rely on the analyst’s expertise or on informal communication with stakeholders and, thus, make B/I alignment hard to achieve. To address these shortcomings, we introduce contextual feature models, which explicitly represent the effect of business and technical contexts on features and configurations.

There is an active discussion concerning the structure of feature models (e.g. graph vs. tree), and the semantics of a feature (e.g. is a feature user-visible?) [15]. Our focus in this paper is to demonstrate the importance of weaving context together with features and, consequently, with configurations variability. Therefore, we adopt a fairly simple structure of feature model and add context to it. Our notation can, however, be extended to support more expressive relations between features, e.g. cardinality constraints over features [6].

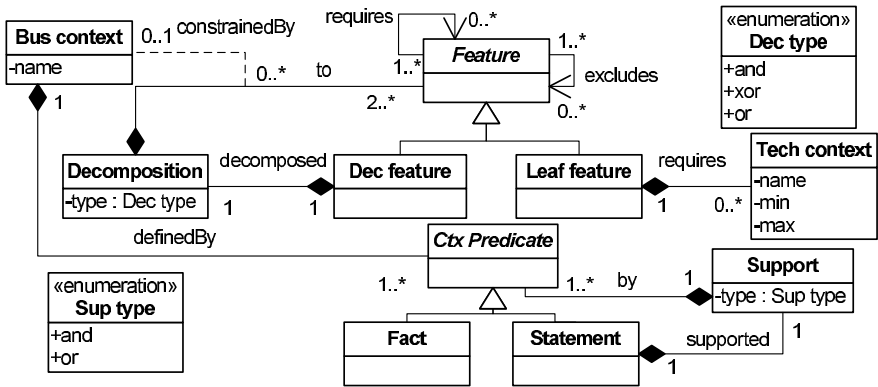


Fig. 1. Meta-model of contextual feature models

The meta-model of CFMs is shown in the class diagram of Fig. 1. A Feature is an abstract class with two concrete manifestations: (i) a decomposed feature Dec feature is further decomposed into sub-features, and is not directly implemented; (ii) a Leaf feature is directly implemented and is a leaf in a feature tree.



Features are hierarchically structured via **Decomposition** relations. A decomposed feature  $f$  has exactly one decomposition, which has a type (**Dec type**): (i) *and*: all sub-features shall be selected to select  $f$ ; (ii) *xor*: exactly one sub-feature; (iii) *or*: at least one sub-feature. In addition to their hierarchical decomposition, features can be connected via two additional relationships. The **requires** relation says that the inclusion of a feature mandates the inclusion of another feature. The **excludes** relation is used to represent mutual exclusion between two features.

Unlike non-contextual feature models, each branch of a decomposition can be constrained by a business context (**Bus context**). A business context refers to social, business, or organizational characteristics of the deployment environment. Some examples are: “the company is a small enterprise”, “the company has branches in different countries”, “customers speak different languages”, and “the revenue trend is negative”. Business contexts define the information to be acquired to decide if a certain sub-feature is advisable or required. Business contexts do not include technical prerequisites (those are technical contexts). The effect of business context on a sub-feature depends on the decomposition type:

- *and*: the sub-feature is *required* when the context holds;
- *or/xor*: the sub-feature is *selectable* only when the context holds.

Business contexts could be communicated as high-level statements whose validity is hard to judge objectively. For instance, the validity of  $bc_1$  = “the company has an international profile” cannot be judged in an objective way: what exactly makes a company profile international? Thus, a business context would need refinement to a formula of objectively observable facts that reifies it. This is essential to avoid misleading judgments of business contexts. For example, while two analysts might give two different answers about  $bc_1$ , they would give the same judgment if we refine  $bc_1$  to a formula of observable facts such as  $fact_1 \vee fact_2$  where  $fact_1$  = “the company has branches in different countries” and  $fact_2$  = “the company has employees from different countries”. We adapt our previous work on *context analysis* [1] to analyze business contexts and ultimately identify a formula of observable facts. Accordingly, we define context as a formula of contextual predicates (**Ctx predicate**) of statements and facts: statements are hierarchically refinable to formulae of facts which support their evidence:

- **Statement**: it is a contextual predicate which is not observable per se;
- **Fact**: it is a contextual predicate which is observable per se.

Leaf features are implemented features and can be associated with a *technical context* (**Tech context**). Technical contexts specify technical prerequisites for an implemented feature. If not met, they inhibit the operationalization of a leaf feature. Unlike business ones, technical contexts need no context analysis, as the development team of an implemented feature knows—and typically documents—its technical prerequisites. For instance, a technical context for a web-based text editor may be using the Internet Explorer browser. We characterize a technical context by attributes **min** and **max**, which specify the range of admitted values. For instance, given the Internet Explorer context, **min** and **max** may refer to its

version: (min = 6, max = ∞) indicates that the feature requires a version equal to or greater than 6; (min = 6, max = 8) a version between 6 and 8; and (min = 0, max = ∞) no version constraints.

Fig. 2 shows a partial CFM for the Drupal scenario presented in Sec. 1. We focus on features related to content management, i.e. how users can create and modify different types of web pages. To illustrate our approach using a small model, we introduce some features which are not currently available in Drupal.

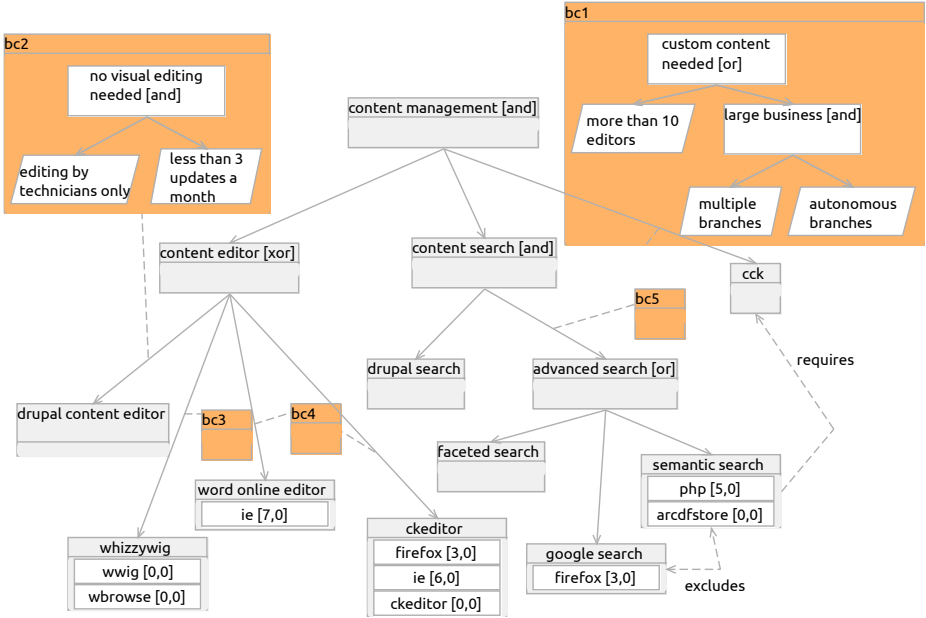


Fig. 2. Contextual feature model for the Drupal scenario

The root feature **content management**—a sub-feature in the complete CFM for Drupal—is and-decomposed to three sub-features: **content editor** to create and modify web pages, **content search** to search information, and **cck** (content construction kit) to enable the creation of custom content. **cck** is context-dependent, as it is required only when business context **bc1** holds. The context analysis for **bc1** says that custom content is needed if the website has at least ten editors (fact), or the customer has a large business (statement). In turn, **large business** is and-supported if there are multiple branches and branches are autonomous (there is no organizational policy on how each branch creates content).

The feature **content editor** is xor-decomposed: only one content editor can be in a configuration. The **drupal content editor** has no technical contexts, while other editors have prerequisites. For example, **word online editor** requires internet explorer  $v \geq 7$ ; **ckeditor** requires firefox ( $v \geq 3$ ), internet explorer ( $v \geq 6$ ), and the **ckeditor** binaries. In the sub-tree rooted by **content search**, the features **google**

search and semantic search are mutually exclusive (*excludes* relation). The feature semantic search needs technical contexts php ( $v \geq 5$ ) and arcdfstore, as well as the selection of feature cck (*requires* relation). Due to space limitations, business contexts bc3, bc4, and bc5 are not analyzed in Fig. 2. Context bc3 stands for “basic visual editing is needed”, bc4 indicates “advanced visual editing is needed”, and bc5 means “customer has a large products catalog”.

### 3 Reasoning about B/I Alignment with CFMs

We devise the formal framework to represent CFMs (Sec. 3.1) and then we detail and illustrate the proposed techniques that help maximizing B/I alignment in feature-oriented software configuration (Sec. 3.2).

#### 3.1 Formal Framework

We rely upon the following assumptions: (i) every feature decomposition is contextual, and such context may be trivially true; (ii) we refer to a single contextual feature model  $\mathcal{M}$ : in the following, all features, contexts, and relations are part of  $\mathcal{M}$ ; (iii) a business context BC is defined as a set of facts  $\{\text{fact}_1, \dots, \text{fact}_m\}$  that hold; (iv) a technical context TC= $\{\text{tc}_1, \dots, \text{tc}_q\}$  is a set of technical contexts (tcname, min, max); and (v) excludes is a symmetric relation.

A configuration CFG for  $\mathcal{M}$  is a set of features that are aligned with the business context (they are adequate to the deployment business environment), with the technical context (the technical infrastructure enables their deployment), and, together, support the root feature of  $\mathcal{M}$ .

**Definition 1 (Configuration).** *Given BC and TC, a set of features  $\{f_1, \dots, f_n\}$  is a configuration CFG for  $\mathcal{M}$  with respect to BC and TC, formally  $\text{CFG} \vdash_{\text{BC}, \text{TC}} \mathcal{M}$ , if and only if  $\text{root}(\mathcal{M}) \in \text{CFG}$  and (1-4) hold  $\forall f' \in \text{CFG}$ :*

1.  $\text{dec}(f', D = \{\langle f_1, \text{bc}_1 \rangle, \dots, \langle f_n, \text{bc}_n \rangle\}, \text{type}) \rightarrow$   
 $\nexists \langle f_j, \text{bc}_j \rangle \in D : \neg \text{support}(\text{BC}, \text{bc}_j) \wedge f_j \in \text{CFG}, \text{ and}$   
 $- \text{type} = \text{and} : (\exists 1 \leq k \leq n : f_k \in \text{CFG}) \wedge (\forall \langle f_i, \text{bc}_i \rangle \in D : \text{support}(\text{BC}, \text{bc}_i) \rightarrow f_i \in \text{CFG})$   
 $- \text{type} = \text{or} : \exists \langle f_i, \text{bc}_i \rangle \in D : \text{support}(\text{BC}, \text{bc}_i) \wedge f_i \in \text{CFG}$   
 $- \text{type} = \text{xor} : \exists ! \langle f_i, \text{bc}_i \rangle \in D : \text{support}(\text{BC}, \text{bc}_i) \wedge f_i \in \text{CFG}$
2.  $\forall f'' : \text{requires}(f', f'') \rightarrow f'' \in \text{CFG}$
3.  $\forall f'' : \text{excludes}(f', f'') \rightarrow f'' \notin \text{CFG}$
4.  $\text{is-leaf}(f') \rightarrow \forall \langle \text{tcname}, \text{min}, \text{max} \rangle \in \text{techctx}(f')$   
 $\exists \langle \text{tcname}, \text{min}', \text{max}' \rangle \in \text{TC} : \text{min}' \geq \text{min} \wedge \text{max}' \leq \text{max} \quad \square$

In other words, CFG is a configuration if it includes the root feature and: (1) if a feature is decomposed, no sub-feature whose business context is not supported is in CFG and, depending on the decomposition type and on the contexts supported by BC, one or more sub-features are in CFG; (2) for any feature  $f'$  in CFG, all its required features are in CFG too; (3) if  $f'$  is in CFG, all mutually exclusive features are not in CFG; and (4) the technical contexts of leaf features are compatible

with TC. The predicate **support** is defined in our previous work [1] and, roughly, returns true if the formula ( $\text{fact}_1 \wedge \dots \wedge \text{fact}_n$ ) gives enough evidence to the truth of bc (as specified via context analysis).

**Table 1.** Disjunctive datalog rules to enable configurations generation

Id	Rule definition
1	$\text{active}(F) :- \text{anddecomposed}(F), 0 = \# \text{count}\{Fi: \text{dec}(F, Fi, Ca), \text{holds}(Ca), \neg \text{active}(Fi)\},$ $\text{not noExtraAct}(F), \text{dec}(F, Fj, Cb), \text{active}(Fj), \text{holds}(Cb).$
2	$\text{noExtraAct}(F) :- \text{dec}(F, Fi, Ca), \text{active}(Fi), \text{not holds}(Ca).$
3	$\text{active}(F) :- \text{ordecomposed}(F), \text{dec}(F, Fi, Ca), \text{active}(Fi), \text{holds}(Ca), \text{not noExtraAct}(F).$
4	$\text{active}(F) :- \text{xordecomposed}(F), \text{dec}(F, Fi, Ca), \text{holds}(Ca), \text{active}(Fi), \text{not actdiff}(F, Fi).$
5	$\text{actdiff}(F, Fi) :- \text{xordecomposed}(F), \text{active}(Fi), \text{dec}(F, Fi, \_), \text{dec}(F, Fj, \_), \text{active}(Fj),$ $Fi \neq Fj.$
6	$\neg \text{active}(Fi) :- \text{requires}(Fi, Fj), \neg \text{active}(Fj).$
7	$\neg \text{active}(Fj) \vee \neg \text{active}(Fi) :- \text{excludes}(Fi, Fj).$
8	$\neg \text{active}(X) :- \text{anddecomposed}(X), \text{not active}(X).$
9	$\neg \text{active}(X) :- \text{ordecomposed}(X), \text{not active}(X).$
10	$\neg \text{active}(X) :- \text{xordecomposed}(X), \text{not active}(X).$
11	$\neg \text{active}(Y) :- \text{dec}(X, Y, C), \neg \text{active}(X).$
12	$\text{holds}(TC) :- \text{tc}(\_, TC), \text{not noPartInactive}(TC).$
13	$\text{noPartInactive}(TC) :- \text{tpart}(TC, P, Vmin, Vmax), \text{not istrue}(P, Vmin, Vmax).$
14	$\text{holds}(BC) :- \text{anddec}(BC), \text{not subUnsat}(BC).$
15	$\text{subUnsat}(BC) :- \text{fdec}(BC, SUB), \text{not holds}(SUB).$
16	$\text{holds}(BC) :- \text{ordec}(BC), \text{fdec}(BC, SUB), \text{holds}(SUB).$
17	$\text{active}(X) \vee \neg \text{active}(X) :- f(X), \text{tc}(X, C), \text{holds}(C).$
18	$\text{active}(X) \vee \neg \text{active}(X) :- f(X), 0 = \# \text{count}\{C: \text{tc}(X, C)\}.$
19	$\neg \text{active}(X) :- f(X), \text{tc}(X, C), \text{not holds}(C).$
20	$f(X) :- \text{dec}(\_, X, \_), 0 = \# \text{count}\{Z: \text{dec}(X, Z, \_)\}.$
21	$\text{act}(X) :- \text{active}(X), f(X).$
22	$\text{bc}(X) :- \text{dec}(\_, \_, X).$
23	$\text{tch}(X) :- \text{tc}(\_, X), \text{holds}(X).$
24	$\text{bch}(X) :- \text{dec}(\_, \_, X), \text{holds}(X), X \neq \text{true}.$
25	$\text{holds}(\text{true}).$

Based on Definition 1, Table 1 lists a set of rules [2] in disjunctive datalog [7] that form the basis of the analysis techniques which we propose in Sec. 3.2. These rules are a generic framework that, given a set of constraints about technical and business contexts, supports configurations generation.

Rules 1-2 say that an and-decomposed feature is in a configuration (is *active*) if (i) all its sub-features with a holding business context on the respective decomposition branch are active; (ii) no sub-feature whose business context does not hold is active; and (iii) at least one sub-feature is active. Rule 3 handles or-decomposition (at least one sub-feature is active), while rules 4-5 manage xor-decomposition (exactly one sub-feature is active).

<sup>2</sup> We adopt the syntax of DLV [13]: <http://www.science.at/proj/dlv/>

Rules 6-7 handle the effect of the *requires* and *excludes* relations, respectively: (i) if the required feature is inactive, then the requiring feature has to be inactive too; (ii) one of the mutually exclusive features has to be inactive. Rules 8-10 are technicalities to deal with true negation in disjunctive datalog: if a decomposed feature is not active, then it is inactive. If a decomposed feature is inactive (rule 11), its sub-features has to be inactive too (we support feature trees).

The preconditions of a leaf feature are met (rules 12-13) if all its technical contexts hold. Rules 14-16 deal with business contexts. A business context (or a statement) is *and-supported* if all statements/facts supporting it hold. A statement is *or-supported* if at least one statement/fact supporting it holds.

Rules 17-18 say that a leaf feature, in the context of configurations generation, can be either active or inactive if its technical contexts hold (rule 18 handles the case of a leaf feature having no technical context). Rule 19 says that a feature whose technical contexts do not hold has to be inactive.

Rules 20-24 define utility predicates: (i) *f* for leaf features; (ii) *act* for active leaf features; (iii) *bc* for business contexts; (iv) *tch* for a holding technical context; and (v) *bch* for a holding business context. A true context always holds (rule 25).

### 3.2 Reasoning Techniques

We present and illustrate reasoning techniques that use the formal framework of Sec. 3.1. These techniques analyze CFMs to answer questions which support the B/I alignment decisions. In the rest of this section, *f* is the root feature of  $\mathcal{M}$ , and we list only leaf features in configurations to maximize readability.

#### Configurations Generation

**Problem 1 (conf-gen).** *Given TC and BC, return all the configurations  $CFG_i$  such that  $CFG_i \vdash_{BC,TC} \mathcal{M}$ .*

This is an essential query an analyst would make: given information concerning a prospective deployment environment—the important business contexts to support and the available technical contexts—, which are the candidate configurations of  $\mathcal{M}$  that are aligned with such environment? We solve *conf-gen* by extending the datalog formalization of Table 1. For each technical context  $tc = \langle tcname, min, max \rangle \in TC$ , we add rule *istrue(tcname,min,max)*. For each fact  $\in BC$ , we add rule *holds(fact)*. The query is *active(f)*?

*Example 1 (conf-gen).* Consider company  $\alpha$  that supports browser internet explorer  $v \geq 8$ , php  $v5$ , and arcdfstore.  $\alpha$  needs advanced visual editing (*bc4*), has a large catalog (*bc5*), and more than 10 editors. There are three configurations aligned with the business and technical context of such a company:

- $CFG_1 = \{cck, drupal\ search, word\ online\ editor, semantic\ search\}$
- $CFG_2 = \{cck, drupal\ search, faceted\ search, word\ online\ editor, semantic\ search\}$
- $CFG_3 = \{cck, drupal\ search, faceted\ search, word\ online\ editor\}$

These configurations share features `cck`, `drupal search`, and `word online editor`, while they differ in the type of search features: `faceted`, `semantic`, or both. The analyst can suggest either the configuration with minimal cost (associating a cost with features), the one that maximizes a set of qualities, or the one company  $\alpha$  prefers. Previous work by Benavides [2]—which enriches feature models with attributes related to cost and quality and reasons about such models using constraint programming—can be used in conjunction with our framework to rank the derived configurations based on cost and qualities.  $\square$

### Business-to-IT Alignment

Configurations generation is useful to explore the space of configurations that match the business and technical context of an enterprise. We present now techniques to identify an IT infrastructure which accommodates configurations that are aligned with a given business context. Specifically, we address the problems of determining alternative IT infrastructures (Problem 2), choosing a minimal infrastructure that enables all software configurations which are aligned with a certain business context (Problem 3), and identifying the core infrastructure elements supporting at least one configuration (Problem 4). These techniques can be complemented by cost analysis to rank alternative infrastructures.

**Problem 2 (bus-to-it).** *Given BC, return all the configurations and technical contexts  $\langle \text{CFG}_i, \text{TC}_i \rangle$  such that  $\text{CFG}_i \vdash_{\text{BC}, \text{TC}_i} \mathcal{M}$ .*

This problem presumes that the analyst has information about the business context that a customer wants to support. The analyst aims to identify the possible configurations for such business context and to compare the technical prerequisites, so to recommend a configuration to the customer which could be based on what infrastructure is already available in the customer’s organization. We solve *bus-to-it* by adding the following rules. Each technical context can be selected or not:  $\text{istrue}(X, \text{Min}, \text{Max}) \vee \text{-istrue}(X, \text{Min}, \text{Max}) \text{- tcpart}(Y, X, \text{Min}, \text{Max})$ ,  $\text{tc}(Z, Y)$ ,  $\text{act}(Z)$ ; for each  $\text{fact} \in \text{BC}$ , add rule  $\text{holds}(\text{fact})$ . The query is  $\text{active}(f)?$ .

*Example 2 (bus-to-it).* Through context analysis, company  $\beta$  has identified facts that support business contexts `bc4` (advanced editing features) and `bc5` (vast product catalog). Given this input, six solutions exist (S1-S6 in Table 2):

**Table 2.** Solutions to the *bus-to-it* problem of Example 2

Feature / Tech context	S1	S2	S3	S4	S5	S6
<code>f1 = drupal search</code>	✓	✓	✓	✓	✓	✓
<code>f2 = google search</code>	✓	×	✓	×	✓	✓
<code>f3 = faceted search</code>	✓	✓	✓	✓	×	×
<code>f4 = word online editor</code>	✓	✓	×	×	✓	×
<code>f5 = ckeditor</code>	×	×	✓	✓	×	✓
<code>tc1 = firefox</code>	$\geq 3$	×	$\geq 3$	$\geq 3$	$\geq 3$	$\geq 3$
<code>tc2 = ckeditor</code>	×	×	✓	✓	×	✓
<code>tc1 = ie</code>	$\geq 7$	$\geq 7$	×	$\geq 6$	$\geq 7$	$\geq 6$

The analyst can now compare the technical contexts to determine the most adequate configuration. He may conduct cost analysis that takes into account the existing IT infrastructure as well as the cost of buying, upgrading, and deploying the infrastructure to satisfy missing technical prerequisites.  $\square$

We describe two techniques that provide insights about Business-to-IT alignment, especially when many solutions are identified by solving Problem 2.

**Problem 3 (high-variability).** *Given BC, which is the minimal set of technical contexts that enables the deployment of a high-variability product including all configurations for BC?*

The problem is to determine the requirements for a high-variability product, that includes all possible configurations for the business context BC. The deployment of high-variability products is useful either to create *adaptive* systems—which are able to switch to an alternative configuration when the current one fails or is ineffective—or to support *customization* to different users, preferences, and profiles. We solve Problem 3 by post-processing the output of Problem 2: all technical contexts are returned, and the minimum-maximum values should satisfy all solutions. If a technical context appears with value range  $[3, \infty]$  in a solution, and with range  $[2, 5]$  in another, the minimal range to solve Problem 3 will be  $[3, 5]$ . In case such set is inconsistent (e.g. a same technical context appears with ranges  $[0, 2]$  and  $[3, 4]$ ), manual analysis is required.

*Example 3 (high-variability).* Take the solution of Example 2. The minimum set of technical contexts for a high-variability product is  $\{\text{ckeditor } [0, \infty], \text{firefox } [3, \infty], \text{ie } [7, \infty]\}$   $\square$

**Problem 4 (core-tech-ctx).** *Given BC, identify the core sets of technical contexts. Each core set CS enables at least one configuration CFG for  $\mathcal{M}$  ( $\text{CFG} \vdash_{\text{BC}, \text{CS}} \mathcal{M}$ ), and is such that, removing any technical context tc from CS, there exists no configuration CFG' such that  $\text{CFG}' \vdash_{\text{BC}, \text{CS} \setminus \{\text{tc}\}} \mathcal{M}$ .*

Each core set of technical contexts defines the minimal requirements for an IT infrastructure supporting business context BC. Notice that many of these sets may exist, each defining a candidate technical environment to support BC. Due to space limitations, we do not describe the algorithm, which is implemented in our CASE tool FM-Context (see Sec. 4).

*Example 4 (core-tech-ctx).* Take the solution of Example 2. Two core sets that support business contexts bc4 and bc5 are identified:  $\text{CS}_1 = \{\text{ie} \geq 7\}$  and  $\text{CS}_2 = \{\text{ie} \geq 6, \text{ckeditor}, \text{firefox} \geq 3\}$ . If the analyst aims at supporting all business contexts, he will select one or more configurations that support either  $\text{CS}_1$  or  $\text{CS}_2$ , upon obtaining feedback from the customer about the feasibility and cost of each core set of technical contexts.  $\square$

## IT-to-Business Alignment

We propose techniques that, given an IT infrastructure, provide insights about the business contexts supported by such infrastructure. These analyses are useful if a customer is not willing to change its infrastructure, as well as to conduct *what-if* studies about the business efficacy of alternative infrastructures. To improve readability, the output of these techniques is shown in terms of holding business contexts ( $bc_1, bc_2, \dots$ ) rather than of facts.

**Problem 5 (it-to-bus).** *Given TC, return all the configurations and business contexts  $\langle CFG_i, BC_i \rangle$  such that  $CFG_i \vdash_{BC_i, TC} \mathcal{M}$ .*

The analyst knows the technical infrastructure of the customer, and aims to understand which are the configurations and business contexts that such infrastructure supports. We solve *it-to-bus* by extending the formalization of Table 1. For each technical context, we add a rule `istrue(name,min,max)`. Then, we add a rule stating that every business context has to either hold or not: `holds(Y) v -holds(Y) :- bc(Y)`, and one saying that if the sub-feature on a contextual decomposition branch is not active, then the business context is not active: `-holds(Y) :- bc(Y), 0=#count{X: dec(_X,Y), active(X)}`. The query is `active(f)?`.

*Example 5 (it-to-bus).* Company  $\gamma$  has many customers with legacy web browsers. As a consequence,  $\gamma$  wants to make as few assumptions as possible about the browser customers use: its technical infrastructure includes only `php v5` and `arcdfstore`. Given this technical context, six solutions exist (see Table 3):

**Table 3.** Solutions to the *it-to-bus* problem of Example 5

Feature / Business context	S1	S2	S3	S4	S5	S6
f1 = drupal search	✓	✓	✓	✓	✓	✓
f2 = drupal content editor	✓	✓	✓	✓	✓	✓
f3 = faceted search	×	✓	✓	×	✓	×
f4 = cck	×	×	✓	✓	✓	✓
f5 = semantic search	×	×	✓	✓	×	×
bc1 = custom content needed	×	×	✓	✓	✓	✓
bc2 = no visual editing needed	✓	✓	✓	✓	✓	✓
bc5 = large product catalog	×	✓	✓	✓	✓	×

The solutions are quite different one from another. For example, S1 supports only business context `bc2`, while solutions S3-S5 support `bc1`, `bc2`, and `bc5`. Notice that, given the technical contexts in input, business contexts `bc3` and `bc4` are never supported. If those contexts are important to the management of  $\gamma$ , the analysis could be repeated by extending the technical infrastructure. That will, however, require  $\gamma$  to impose more technical prerequisites to its customers.  $\square$



**Problem 6 (bus-support).** *Given TC and a ranking  $R=bc_i>bc_j>bc_k>\dots$  that defines the relative priority of the business contexts in  $\mathcal{M}$ , which are all the supported maximal sets of business contexts? Which is the relative order of these maximal sets according to  $R$ ?*

The problem is to identify the maximal sets of business contexts supported by the technical infrastructure. Since there may be many of these sets, the analyst is asked to rank the relative priority of all the business contexts in BC. This problem is solved by post-processing the output of Problem 5 and returning all maximal sets of business contexts (those not included in any other set). These sets are ranked according to  $R$ : for any constraint  $bc>bc'$ , a set including  $bc$  and not including  $bc'$  is ranked before any set including  $bc'$  and not including  $bc$ .

*Example 6 (bus-support).* Company  $\theta$  supports only technical context ie  $v\geq 7$ ; let  $R=bc1>bc4>bc3>bc5>bc2$ . There are two maximal sets:  $\{bc1, bc5, bc4\}$  and  $\{bc1, bc2, bc5\}$ . These maximal sets allow for supporting either  $bc2$  (no visual editing needed) or  $bc4$  (advanced editing needed), but not both. Given that  $R$  states that  $bc4$  is preferred to  $bc2$ , the former maximal set is recommended.  $\square$

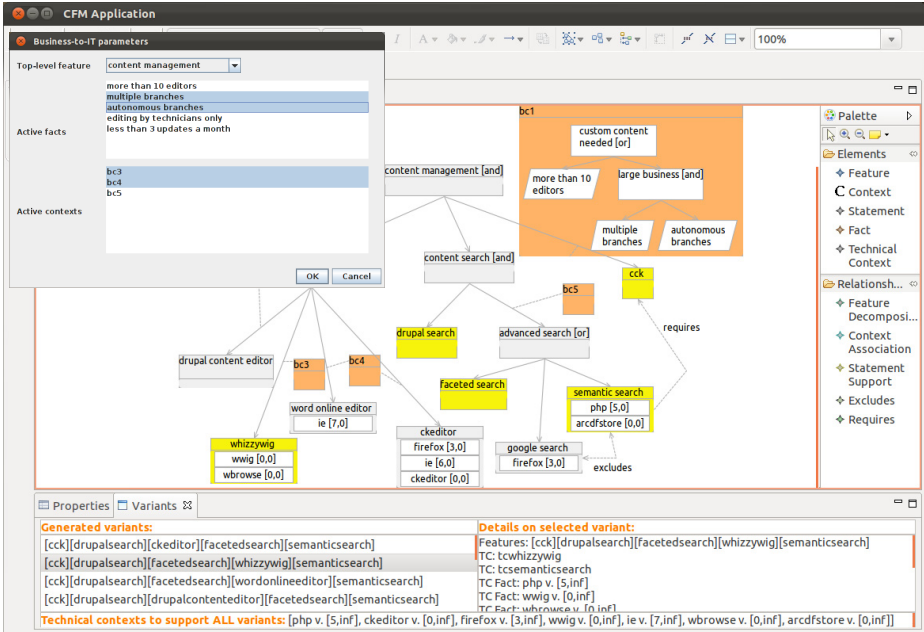
Our reasoning techniques are a basic toolset to address problems about B/I alignment in software configuration, and can be complemented with cost analysis, elaborated quality-based reasoning, and what-if analysis.

## 4 FM-Context: A Support Tool for CFMs

FM-Context is our developed CASE tool for CFMs. The tool enables the graphical creation of CFMs, and implements algorithms to solve the Problems 1-6 of Sec. 3. FM-Context is an Eclipse Rich Client Platform application—built on Eclipse Galileo 3.5.2—implemented according to the meta-model driven development framework provided by the Eclipse Graphical Modeling Project<sup>3</sup>. Its reasoning engine is based on the disjunctive datalog solver DLV. FM-Context currently supports Windows and Linux machines both 32 and 64 bits. FM-Context is freely available from the web at <http://goo.gl/wx3Vl>.

Figure 3 shows a screen-shot of FM-Context while prompting the user for input to solve the *bus-to-it* problem. FM-Context allows for a more flexible input than that described in Problem 2: the analyst can choose also non-analyzed business contexts. In the foreground window, the analyst has selected not only active facts, but also non-analyzed contexts  $bc3$  and  $bc4$ . These selectable lists are populated automatically by analyzing the CFM at hand (in the background window). The central part of the background window is occupied by the contextual feature model of Fig. 2. On the right side is the drawing palette that allows for adding elements and relations to the diagram canvas. At the bottom are details about the output obtained by a previously ran automated analysis.

<sup>3</sup> <http://www.eclipse.org/modeling/gmp/>



**Fig. 3.** Screen-shot of FM-Context: the user is prompted for the input to Problem 2

In Table 4 we outline preliminary results about the scalability of the automated reasoning in FM-Context. Specifically, we ran the configurations generation technique (to solve Problem 1) on CFMs of growing size. Starting from the Drupal CFM, we constructed five CFMs of different sizes (in terms of features *Feat*, statements *Stat*, facts *Fact*, and technical contexts *Tech*) and hypothesized that all contexts (business and technical) hold; for each CFM, we executed four tests: (i) *1-var*: all decompositions are of type “and” (a single variant); (ii) *Low-var*: prevalence of xor-decompositions; (iii) *Mid-var*: a balanced number of xor- and or-decompositions; and (iv) *High-var*: all or-decompositions.

**Table 4.** Preliminary performance evaluation of the configurations generation reasoning with FM-Context. Time in milliseconds.

Feat	Stat	Fact	Tech	Nodes	1-var			Low-var			Mid-var			High-var		
					Time	Vars	time_var	Time	Vars	time_var	Time	Vars	time_var	Time	Vars	time_var
7	4	5	2	18	16	7	22	3.14	9	24	2.67	31	31	1.00		
16	10	13	6	45	20	127	59	0.47	319	101	0.31	2047	487	0.24		
24	15	17	8	64	25	251	96	0.38	1799	508	0.28	18431	6712	0.36		
32	20	26	12	90	33	1023	648	0.63	8191	3400	0.42	32767	19968	0.61		
64	40	52	24	180	51	3615	2954	0.81	14415	12518	0.86	54960	66048	1.21		

For each experiment we report the number of variants (*Vars*), overall time (*Time*), and time per variant (*time/var*). Since experiment *1-var* includes only one variant, we show only overall time. We repeated each test three times and present the average time in Table 4. The results of our evaluation are promising: the main criteria to increase reasoning time is the number of variants, and not the number of nodes. In a realistic environment, moreover, the analyst would not typically obtain so many variants as in our example, given that the deployment environment would constrain the possible configurations. A CFM with much non-contextual variability would make the configuration task very difficult.

## 5 Related Work

The relation between context and features has been studied in literature. Lee and Kang [12] propose to analyze usage contexts to derive the factors—qualities and technical constraints—that drive feature selection. They rely on a set of feature diagrams that represent the product line, usage context, qualities, and technical constraints. Similarly, Hartmann and Trew [8] complement feature diagrams with a context variability model, so to support multiple product lines. The two models are linked via constraints (the context constrains applicable features). These models are merged into a feature model for a specific context. We share the same motivation and vision. However, we rely upon richer conceptual models to represent context: business contexts are specified via context analysis, while technical contexts are expressed as prerequisites to leaf features. Moreover, our reasoning helps maximizing B/I alignment during product configuration.

Tun et al. [16] address the configuration problem based on a refinement approach that consists of multiple feature models, which analyze requirements, problem world contexts, specifications, and expression of quantitative constraints. We do not focus on the refinement process here: our purpose is to identify and reason about the relationships between features, business, and technical contexts.

The original feature modeling notation of FODA [10] does not explicitly support context modeling. However, it documents decision points with “issues”, specific questions that an analyst should answer in order to choose the most adequate configuration. Thurimella et al. [17] extend FODA by adopting a rationale management framework to express issues associated with decision points. Our approach goes one step further and uses a formal—as opposed to a textual—language to document such rationale; moreover, we focus on B/I alignment.

Czarnecki et al. [5] propose a method—called staged configuration—wherein the best configuration is achieved via stepwise specialization of feature models, with multiple people involved in such process. This approach could be used in conjunction with CFMs to guide the selection process.

Our approach is in the spirit of supporting automated reasoning on feature models (Benavides et al. [2]). They propose a framework that supports attribute-extended feature models to express technical constraints. Unlike them, we consider the business context and propose a different set of reasoning mechanisms.

In their empirical study, Chan et al. [4] provide valuable insights about the relationship between planning (among which, information systems configuration),

and B/I alignment. In particular, their field study evidences that planning sophistication does not directly improve B/I alignment, while it improves shared domain knowledge. Such knowledge, in turn, improves B/I alignment. There exist s, thus, an indirect relationship between planning and alignment. While CFMs provide directions about the B/I alignment problem, their output still necessitates interpretation by analysts and domain experts.

In previous work [1], we have proposed contextual goal models to represent and reason about the interplay between requirements at the intentional level and context. That modeling framework helps to detect whether there exists a set of system requirements that satisfies the goals of stakeholders in a specific context. Differently, CFMs apply when a software product family already exists, and help to derive a configuration that fits well in a prospective deployment environment.

## 6 Conclusion and Future Directions

In this paper, we introduced contextual feature models, which enrich traditional feature models with information about contextual constraints—both at the business and at the technical level. Importantly, we formally represent the interplay between contexts and features. Our automated reasoning techniques can be exploited to configure the product line while maximizing B/I alignment. Both modeling and reasoning are supported by our CASE tool FM-Context.

Our approach is a means to support analysts in selecting a configuration that fits well with a prospective deployment environment. CFMs can be used as a sophisticated planning tool that increases shared domain knowledge [4]. However, they do not replace the role of designers. The suggestions about configurations and contexts of our analyses are used by analysts and domain experts, who will take the final decision about which configuration to deploy.

This paper opens the doors to several research directions that will be part of our future work: (i) devise a methodology analysts can use to fully benefit from CFMs; (ii) refine and extend the modeling notation (e.g. support cardinality and more complex technical contexts); (iii) develop and implement new algorithms to support analysts; and (iv) evaluate the applicability of the approach by conducting experimentation on industrial case studies.

**Acknowledgment.** The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grants no 257930 (Aniketos), 256980 (Nessos), and 258109 (FastFix), and by the Science Foundation Ireland under grant 10/CE/I1855.

## References

1. Ali, R., Dalpiaz, F., Giorgini, P.: A Goal-based Framework for Contextual Requirements Modeling and Analysis. *Requirements Engineering* 15(4), 439–458 (2010)
2. Benavides, D., Trinidad, P., Ruiz-Cortés, A.: Automated Reasoning on Feature Models. In: Pastor, Ó., Falcão e Cunha, J. (eds.) *CAiSE 2005*. LNCS, vol. 3520, pp. 491–503. Springer, Heidelberg (2005)

3. Bosch, J.: Design and Use of Software Architectures: Adopting and Evolving a Product-Line Approach. Addison-Wesley Professional (2000)
4. Chan, Y.E., Sabherwal, R., Bennet Thatcher, J.B.: Antecedents and outcomes of strategic IS alignment: an empirical investigation. *IEEE Transactions on Engineering Management* 53(1), 27–47 (2006)
5. Czarnecki, K., Helsen, S., Eisenecker, U.: Staged Configuration Using Feature Models. In: Nord, R.L. (ed.) *SPLC 2004*. LNCS, vol. 3154, pp. 266–283. Springer, Heidelberg (2004)
6. Czarnecki, K., Helsen, S., Eisenecker, U.: Formalizing Cardinality-based Feature Models and their Specialization. *Software Process: Improvement and Practice* 10(1), 7–29 (2005)
7. Eiter, T., Gottlob, G., Mannila, H.: Disjunctive Datalog. *ACM Transactions on Database Systems* 22(3), 364–418 (1997)
8. Hartmann, H., Trew, T.: Using Feature Diagrams with Context Variability to Model Multiple Product Lines for Software Supply Chains. In: *Proceedings of the 12th International Software Product Line Conference (SPLC 2008)*, pp. 12–21 (2008)
9. Henderson, J.C., Venkatraman, N.: Strategic alignment: Leveraging information technology for transforming organizations. *IBM Systems Journal* 32(1), 4–16 (1993)
10. Kang, K.C., Cohen, S.G., Hess, J.A., Novak, W.E., Spencer Peterson, A.: Feature-Oriented Domain Analysis (FODA) Feasibility Study. Technical Report CMU/SEI-90-TR-21, Carnegie Mellon University (1990)
11. Kang, K.C., Kim, S., Lee, J., Kim, K., Shin, E., Huh, M.: FORM: A Feature-oriented Reuse Method with Domain-specific Reference Architectures. *Annals of Software Engineering* 5, 143–168 (1998)
12. Lee, K., Kang, K.C.: Usage Context as Key Driver for Feature Selection. In: Bosch, J., Lee, J. (eds.) *SPLC 2010*. LNCS, vol. 6287, pp. 32–46. Springer, Heidelberg (2010)
13. Leone, N., Pfeifer, G., Faber, W., Eiter, T., Gottlob, G., Perri, S., Scarcello, F.: The DLV System for Knowledge Representation and Reasoning. *ACM Trans. Comput. Log.* 7(3), 499–562 (2006)
14. Peppard, J., Ward, J.L.: “Mind the Gap”: Diagnosing the Relationship between the IT Organisation and the Rest of the Business. *The Journal of Strategic Information Systems* 8(1), 29–60 (1999)
15. Schobbens, P.-Y., Heymans, P., Trigaux, J.-C.: Feature Diagrams: A Survey and a Formal Semantics. In: *Proceedings of the 14th IEEE International Requirements Engineering Conference (RE 2006)*, pp. 136–145 (2006)
16. Tun, T.T., Boucher, Q., Classen, A., Hubaux, A., Heymans, P.: Relating Requirements and Feature Configurations: a Systematic Approach. In: *Proceedings of the 13th International Software Product Line Conference (SPLC 2009)*, pp. 201–210 (2009)
17. Thurimella, A.K., Bruegge, B., Creighton, O.: Identifying and Exploiting the Similarities between Rationale Management and Variability Management. In: *Proceedings of the 12th International Software Product Line Conference (SPLC 2008)*, pp. 99–108 (2008)

# Mining Inter-organizational Business Process Models from EDI Messages: A Case Study from the Automotive Sector\*

Robert Engel<sup>1</sup>, Wil M.P. van der Aalst<sup>2</sup>, Marco Zapletal<sup>1</sup>,  
Christian Pichler<sup>1</sup>, and Hannes Werthner<sup>1</sup>

<sup>1</sup> Vienna University of Technology  
Institute of Software Technology and Interactive Systems  
Electronic Commerce Group

{engel,marco,pichler,werthner}@ec.tuwien.ac.at

<sup>2</sup> Eindhoven University of Technology  
Department of Computer Science  
w.m.p.v.d.aalst@tue.nl

**Abstract.** Traditional standards for *Electronic Data Interchange* (EDI), such as EDIFACT and ANSI X12, have been employed in Business-to-Business (B2B) e-commerce for decades. Due to their wide industry coverage and long-standing establishment, they will presumably continue to play an important role for some time. EDI systems are typically not “process-aware”, i.e., messages are standardized but processes simply “emerge”. However, to improve performance and to enhance the control, it is important to understand and analyze the “real” processes supported by these systems. In the case study presented in this paper we uncover the inter-organizational business processes of an automotive supplier company by analyzing the EDIFACT messages that it receives from its business partners. We start by transforming a set of observed messages to an event log, which requires that the individual messages are correlated to process instances. Thereby, we make use of the specific structure of EDIFACT messages. Then we apply process mining techniques to uncover the inter-organizational business processes. Our results show that inter-organizational business process models can be derived by analyzing EDI messages that are exchanged in a network of organizations.

**Keywords:** process mining, inter-organizational business processes, BPM, EDI, event correlation.

## 1 Introduction

Recent academic research on inter-organizational business processes has mainly focused on Web service choreographies and related XML-based technologies.

---

\* This research has been conducted in the context of the *EDImine* project and has been funded by the Vienna Science and Technology Fund (WWTF) through project ICT10-010.

Nevertheless, many inter-organizational systems are still realized by traditional Electronic Data Interchange (EDI) [12,7,16] standards, such as EDIFACT [2] or ANSI X12. Presumably they will continue to be the primary data formats in Business-to-Business (B2B) interaction for years to come [24].

However, such traditional EDI systems are usually process-unaware, meaning that they are solely responsible for sending and receiving messages. When companies intend to analyze their inter-organizational business processes they generally have to rely on a-priori models, if models documenting the business processes exist at all. In case there are models, those may describe the business processes as they were planned, which is not necessarily in sync with the real-world business processes. To address this shortcoming, we seek to derive models of inter-organizational business processes from EDI message exchanges [8]. Thereby we employ and extend existing *process mining* [19,21] techniques, which so far have concentrated on business processes within single organizations.

In this paper we present a case study where we analyze the EDI-based inter-organizational business processes of an automotive supplier company. These processes include activities for placing and changing orders as well as for the management of a just-in-time supply chain. Our results show that inter-organizational business process models can be derived by analyzing EDI messages that are exchanged between companies. This implies that we can apply a wide range of process mining techniques. Besides discovering the “real” processes, we can discover bottlenecks, discover deviations from some predefined behavior (conformance checking), predict performance (e.g., predict the remaining flow time for running instances), etc. [19].

To apply process mining techniques we need to convert a collection of EDI messages to an event log. In such an event log, each sent or received EDI message represents an event. Process mining algorithms generally presuppose that individual events can be assigned to process instances (i.e., cases) [19]. However, in practice many legacy implementations of EDI systems do not provide case identifiers in individual messages. This is also the case for the data set under consideration in this case study. Hence, in order to enable the application of process mining techniques in such settings, it is necessary to *correlate individual EDI messages to process instances*. Consequently, our approach for the analysis of EDI messages for deriving inter-organizational business process models comprises two main steps: (i) correlation of individual, unlinked EDI messages to process instances in order to attain an event log suitable for process mining and (ii) application of a suitable process mining algorithm on the event log to derive the inter-organizational business process models.

The remainder of this paper is organized as follows. In Section [2], related work is discussed. In Section [3], we describe our approach of correlating individual EDI messages to process instances as a necessary prerequisite for the application of process mining algorithms on the data set used in this case study. In Section [4], the actual case study as well as an interpretation of the results are provided. Finally, in Section [5], a critical discussion of the results and an outlook on future work are given.

## 2 Related Work

In [8] we introduce our overall approach of extending process mining techniques for inter-organizational business processes by means of examining EDI message exchanges. Process mining techniques [19,21,11,3,4] extract knowledge about business processes by analyzing event logs. So far, the main focus has been on the analysis of processes inside single organizations. The few publications on process mining in an inter-organizational context tend to focus on the area of Web services [20,22,6,13,15,17]. For example, in [20] conformance checking techniques are applied to the message logs of Oracle BPEL. Another example may be found in [22] where process mining techniques are applied in the context of IBM's WebSphere.

Dustdar et al. [6] proposed techniques for services interaction mining, i.e., applying process mining techniques to the analysis of service interactions, and in [14, pp. 30-32], the suitability of general data mining methods [11] for correlating messages is discussed, where, however, several serious limitations are identified. In [18], an algorithm for mining sequential patterns from databases is presented.

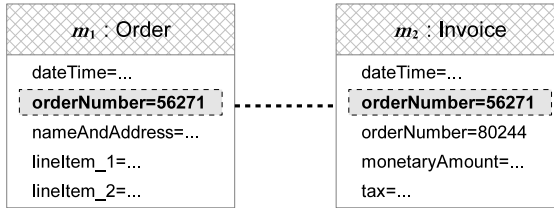
Nezhad et al. [13,15] developed techniques for event correlation and process discovery from web service interaction logs. The authors introduce the notion of a "process view" which is the result of a particular event correlation. However, they argue that correlation is subjective and that multiple views are possible. A collection of process views is called the "process space". With regard to message correlation, Nezhad et al. [14] also describe *correlation rule patterns*, a formalization of corresponding correlation rules as well as heuristics to derive such rules for correlating a set of messages to conversations. The correlation approach presented in the aforementioned work can be seen as a generalization of the mechanism for message correlation used in this paper. For example, *key/value-correlation*, as introduced in this paper, corresponds to *key-based correlation* in [14]; the *synonymy* and *concatenation* rules introduced in this paper can also be expressed with *reference-based correlation* rules as described in [14]. However, the correlation mechanism presented in this paper - while potentially applicable to other domains - was derived from specific observations on real-world EDI messages.

In [17], a technique is presented for correlating messages with the goal to visualize the execution of Web services.

In practice, however, neither explicit choreography modeling nor Web services are widely employed in electronic business transactions. Rather, traditional approaches to EDI such as EDIFACT [2] still play an overwhelmingly dominant role [24]. Therefore, we developed techniques for the correlation of EDI messages.

The problem of recognizing process instances from a set of EDI messages corresponds to the requirement of *event correlation* in process mining [19, p. 113]. Literature on this topic is generally sparse with the notable exception of [9], where a probabilistic approach for correlating events to cases without any a-priori information is proposed. However, in the case of EDI messages one can work with richer information by examining the actual content of messages sent and received by EDI systems. This is a new approach to process mining, since it





**Fig. 1.** Example for correlating EDI messages by matching the `orderNumber` data elements and their values

considers also the “content” within the execution of a process instance instead of treating the sending and receiving of messages as opaque events.

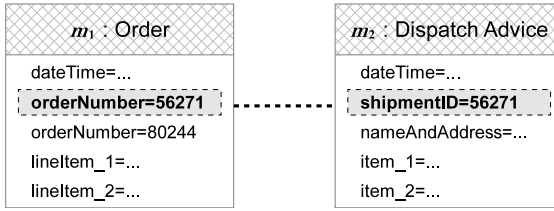
Related to mining process models from exchanged messages is mining from I/O operations as described in [5]. In [10], an approach is presented that accounts for the identification of different variants of processes by clustering traces that share similar behavior patterns.

### 3 Correlation of EDI Messages to Process Instances

Contrary to process-based approaches such as BPEL, traditional EDI messages usually do not contain explicit *case identifiers* that map individual messages to process instances. However, traditional EDI standards such as EDIFACT define data elements that may contain back references to previously sent or received messages of the same business case. For example, the *Order* and *Invoice* messages of the purchase order transaction shown in Fig. 1 both contain the data element `orderNumber`. Additionally, both messages contain the same value in this data element (56271). Hence, in this case the order number can be seen as a case identifier that allows for the conclusion that both messages belong to the same process instance. Typically, it is not as simple as shown in Fig. 1. Therefore, we discuss the basic correlation mechanisms and then present a concrete algorithm.

#### 3.1 Key/Value-Correlation

We define a *trace* as a set of EDI messages that represents a process instance or a fragment thereof. *Note that although in process mining the order of events is generally crucial, we define a trace as an unordered set of messages because the order is given implicitly in the timestamps of the messages and can thus be recovered after correlation.* Furthermore, we define a *correlator* as a key for a specific data element defined in one or more EDI message types that can be used for correlating messages to traces by matching the values contained in the data element. Messages are assigned to a trace when each of them contains the data element denoted by the correlator and the values in the data elements match. We refer to this basic mechanism as *key/value-correlation*.



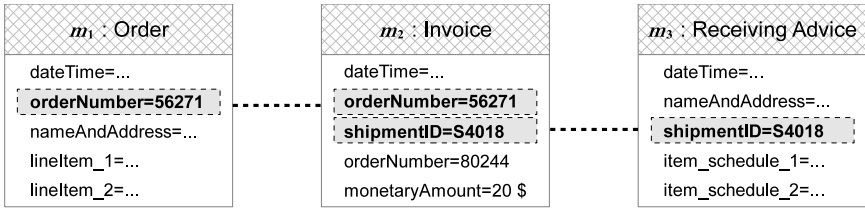
**Fig. 2.** Example for synonymous correlators

Note that not all data elements can serve as correlators. Some data elements may contain repeating values across multiple messages by coincidence, or the contained information may be unrelated to actual process instances (e.g., an address). We define a *set of potential correlators*  $\mathbb{C}$  as a set of keys for data elements that can be potentially used for correlation in some EDI standard. In the case of the EDIFACT standards, a number of different *segments* is defined that group related data elements. For example, the RFF (for *Reference*) segment is intended to hold information that references some other message, business document or other business entity. This can be exploited for correlating EDIFACT messages. Observations on the sample data set used in this case study reveal that many data elements in the RFF segment<sup>1</sup> represent indeed potentially effective correlators (e.g., order numbers are usually contained in RFF segments). Hence, one may use the various types of references defined in the RFF segment as a starting point for building a set of potential correlators (cf. Section 4). In practice, however, varying implementations of EDI standards require further narrowing of the set of potential correlators to a subset that suffices for the correlation of a specific EDI implementation. We refer to such a subset as a *correlator set*  $\mathbb{C} \subseteq \mathbb{C}$ .

### 3.2 Synonymous Keys

As will be shown in detail in Section 4, the data set used for this case study reveals that distinct data elements sometimes contain identical information. For example, a seller receives an *Order* message ( $m_1$ ) from a customer with a specific order number (56271) (cf. Fig. 2). The seller replies with a *Dispatch Advice* message ( $m_2$ ) indicating that ordered goods have been shipped. To facilitate the alignment of orders with shipments, the seller references the order numbers received from the customer in shipment IDs of corresponding shipments. The shipment ID (56271) of the *Dispatch Advice* matches with the order number that was originally assigned by the customer in the *Order* message (56271). If correlators `orderNumber` and `shipmentID` are defined as *synonymous*, then they are regarded as identical keys in key/value-correlation, i.e.,  $m_1$  and  $m_2$  are correlated. Note that every correlator is reflexively synonymous, i.e., every correlator is synonymous to itself.

<sup>1</sup> See <http://live.unece.org/trade/untdid/d10b/tred/tred1153.htm>



**Fig. 3.** Example for concatenation of traces

### 3.3 Concatenation of Traces

We also consider that results may be further improved by concatenating traces at points where they intersect (i.e., overlap) with respect to individual messages. Consider, for example, the three messages shown in Fig. 3 representing a purchase order transaction. An *Order* message ( $m_1$ ) of a customer is followed by a corresponding *Invoice* message ( $m_2$ ) sent by the seller. Because both messages refer to the same **orderNumber** (56271), the trace  $\langle m_1, m_2 \rangle$  can be recognized by applying key/value-correlation. However, in the *Invoice* message, the seller assigns a new **shipmentID** (S4018) to a shipment of goods corresponding to the order. When the customer receives the shipment, it advises the seller by sending a *Receiving Advice* message ( $m_3$ ) containing the same value in the **shipmentID** data element (S4018). Hence,  $\langle m_2, m_3 \rangle$  can also be recognized as a trace by key/value-correlation. From a business point of view, all three messages may be regarded as part of the same process instance since they belong to the same purchase order transaction. The larger trace  $\langle m_1, m_2, m_3 \rangle$  can be reconstructed by concatenating the traces  $\langle m_1, m_2 \rangle$  and  $\langle m_2, m_3 \rangle$  as a result of recognizing  $m_2$  as the intersection of the message sets. In this example the overlap occurs with respect to two different correlators (**orderNumber** and **shipmentID**), but overlaps may also occur in traces of more than two correlators or a single correlator in case the corresponding data element occurs multiple times in individual EDI messages.

Note that this correlation rule may lead to undesired results in cases where concatenation is not appropriate. This implies that subsets of the correlator set need to be selected for which the concatenation rule should be applied. Such *concatenation groups* of correlators specify groups of correlators. When any combination of one or more correlators in a concatenation group is encountered in a single message that is contained in multiple traces resulting from key/value-correlation, the respective traces are concatenated. This means that a new trace is built from the union of the respective message sets and the original traces get discarded. This also includes cases where just one of the correlators from the concatenation group occurs repeatedly in one message that is contained in multiple traces.

### 3.4 Correlation Algorithm

In the following we present the algorithm for correlating EDI messages to process instances that is used in the case study presented in Section 4. It applies key/value-correlation and selectively the rules for synonymous correlators and concatenation of traces on a set of observed messages. The algorithm has to be parameterized with a correlator set and a corresponding specification of relationships between the correlators with respect to synonymies and concatenation. These relationships are modeled by means of *synonymy sets* and *concatenation sets* as described below.

**Potential Correlators, Correlator Set.**  $\mathbb{C}$  is the set of potential correlators, i.e., all keys for data elements that can potentially be used for correlating EDI messages.  $\mathcal{C} \subseteq \mathbb{C}$  is the selected set of correlators. The keys in the set  $\mathbb{C} \setminus \mathcal{C}$  will be discarded when correlating messages.

**Synonymy Group, Synonymy Set.** A *synonymy group*  $\mathcal{SG} \subseteq \mathcal{C}$  is a set of correlators that are considered to be synonymous. For example,  $\mathcal{SG} = \{\text{orderNumber}, \text{shipmentID}\}$  is a synonymy group for the transaction shown in Fig. 2. A synonymy group may be a singleton set, e.g.,  $\mathcal{SG} = \{\text{nameAndAddress}\}$ . This implies that correlator `nameAndAddress` is only synonymous with itself.

A *synonymy set*  $\mathcal{SS} = \{\mathcal{SG}_1, \mathcal{SG}_2, \dots, \mathcal{SG}_m\} \subseteq \mathcal{P}(\mathcal{C})$  is a set of synonymy groups partitioning  $\mathcal{C}$ , i.e.,  $\mathcal{SG}_i \cap \mathcal{SG}_j \neq \emptyset$  implies that  $i = j$  (pairwise disjoint) and  $\bigcup \mathcal{SS} = \mathcal{C}$ .

**Concatenation Group, Concatenation Set.** A *concatenation group*  $\mathcal{CG} \subseteq \mathcal{SS}$  is a set of synonymy groups for which the traces that result from correlation with these synonymy groups are to be concatenated in case of overlaps. For example,  $\mathcal{CG} = \{\{\text{orderNumber}\}, \{\text{shipmentID}\}\}$  is a concatenation group for the transaction shown in Fig. 3. A concatenation group may contain only a single synonymy group; in this case concatenation of traces is applied with regard to overlaps in traces that were correlated by this synonymy group. This may be useful when an EDI message contains multiple instances of the same data element (e.g., multiple order numbers).

A *concatenation set*  $\mathcal{CS} = \{\mathcal{CG}_1, \mathcal{CG}_2, \dots, \mathcal{CG}_n\} \subseteq \mathcal{P}(\mathcal{SS})$  is a set of concatenation groups. Unlike synonymy sets, a concatenation set  $\mathcal{CS}$  does not necessarily contain all correlators from  $\mathcal{C}$ ; synonymy groups that are inappropriate for the concatenation rule are left out of concatenation sets. This is to prevent the concatenation of traces of single synonymy groups where this behavior is undesired.

**Messages, Strings, Values.**  $\mathbb{M}$  is the universe of EDI messages, i.e., all potential messages.  $\mathbb{S}$  is the set of all strings, i.e., possible data values.  $\mathcal{MS} \subseteq \mathbb{M}$  is a concrete set of messages. For modeling actual values in data elements of EDI messages we define the *values function*  $val : \mathbb{M} \times \mathcal{SS} \rightarrow \mathcal{P}(\mathbb{S})$ . The values function returns the values from a specific message and synonymy group.  $val(m, \mathcal{SG})$  is a set of strings corresponding to message  $m \in \mathbb{M}$  and synonymy group  $\mathcal{SG} \in \mathcal{SS}$ .

<sup>2</sup>  $\mathcal{P}(X)$  denotes the powerset of some set  $X$ .  $\bigcup X$  denotes the union of the subsets in a set of sets  $X$ .

As certain data elements may occur repeatedly in a single EDI message, the values function might also yield multiple values for a particular message and synonymy group. Each of these values has to be considered for correlation. Thus, the values function returns a set of strings<sup>3</sup>. For example, for messages  $m_1$  and  $m_2$  shown in Fig. 2 and a synonymy group  $\mathcal{SG} = \{\text{orderNumber}, \text{shipmentID}\}$ , both  $val(m_1, \mathcal{SG}) = \{"56271", "80244"\}$  and  $val(m_2, \mathcal{SG}) = \{"56271"\}$  hold.

**Correlation.** For computing the traces that result from the application of key/value-correlation and the synonymy rule using a single synonymy group  $\mathcal{SG} \in \mathcal{SS}$  on a set of observed messages  $\mathcal{MS} \subseteq \mathbb{M}$ , we define the *correlation function*  $corr : \mathcal{P}(\mathbb{M}) \times \mathcal{SS} \rightarrow \mathcal{P}(\mathcal{P}(\mathbb{S}) \times \mathcal{P}(\mathbb{M}))$  as

$$corr(\mathcal{MS}, \mathcal{SG}) = \left\{ (V, \mathcal{MS}') \in \mathcal{P}(\mathbb{S}) \times \mathcal{P}(\mathcal{MS}) \mid \right. \\ \left. V = \bigcap_{m \in \mathcal{MS}'} val(m, \mathcal{SG}) \setminus \bigcup_{m \in \mathcal{MS} \setminus \mathcal{MS}'} val(m, \mathcal{SG}) \neq \emptyset \right\}.$$

$(V, \mathcal{MS}') \in corr(\mathcal{MS}, \mathcal{SG})$  represents a trace<sup>4</sup> containing messages  $\mathcal{MS}'$  correlated using the set of values  $V$ . Note that the resulting traces, which are sets of messages, need not be disjoint: as a single message may yield multiple values for a single synonymy group, it might be contained in multiple traces of this synonymy group as well.

The set of all traces with respect to a synonymy set  $\mathcal{SS}$  and message set  $\mathcal{MS}$  can be obtained as follows:

$$\mathcal{T}_{\mathcal{SS}, \mathcal{MS}} = \bigcup_{\mathcal{SG} \in \mathcal{SS}} corr(\mathcal{MS}, \mathcal{SG}).$$

**Concatenation.** Let  $\mathcal{CS}$  be the concatenation set used to concatenate traces. Two traces  $(V_1, \mathcal{MS}_1), (V_2, \mathcal{MS}_2) \in \mathcal{T}_{\mathcal{SS}, \mathcal{MS}}$  are related, denoted  $(V_1, \mathcal{MS}_1) \sim_{\mathcal{CS}} (V_2, \mathcal{MS}_2)$ , if and only if:

$$\exists CG \in \mathcal{CS} \exists \mathcal{SG}_1, \mathcal{SG}_2 \in CG \exists m \in \mathcal{MS}_1 \cap \mathcal{MS}_2 V_1 \subseteq val(m, \mathcal{SG}_1) \wedge V_2 \subseteq val(m, \mathcal{SG}_2).$$

$\sim_{\mathcal{CS}}^*$  is the reflexive transitive closure of  $\sim_{\mathcal{CS}}$ , i.e., two traces  $(V_1, \mathcal{MS}_1)$  and  $(V_2, \mathcal{MS}_2)$  are related, i.e.  $(V_1, \mathcal{MS}_1) \sim_{\mathcal{CS}}^* (V_2, \mathcal{MS}_2)$ , if  $(V_1, \mathcal{MS}_1) = (V_2, \mathcal{MS}_2)$  or there exists a trace  $(V, \mathcal{MS}) \in \mathcal{T}_{\mathcal{SS}, \mathcal{MS}}$  such that  $(V_1, \mathcal{MS}_1) \sim_{\mathcal{CS}} (V, \mathcal{MS})$  and  $(V, \mathcal{MS}) \sim_{\mathcal{CS}}^* (V_2, \mathcal{MS}_2)$ .

All traces related through  $\sim_{\mathcal{CS}}^*$  are merged into larger traces. The set of traces resulting from the application of key-/value-correlation and selectively the rules

<sup>3</sup> However, if the same value occurs repeatedly in the same data element in a single message, this will not be reflected by the result of the values function as the function's codomain is a set rather than a multiset.

<sup>4</sup> Note that earlier we described a trace as a set of messages (the ordering can be derived based on the timestamps). Here, we refer to a trace as a pair  $(V, \mathcal{MS}')$  where  $\mathcal{MS}'$  is the set of time-ordered messages and  $V$  is the set of strings used to correlate the messages in the trace.

for synonymous correlators and concatenation of traces with respect to a message message set  $\mathcal{MS}$ , a synonymy set  $\mathcal{SS}$  and a concatenation set  $\mathcal{CS}$  can be obtained as follows:

$$\mathcal{T}_{\mathcal{SS},\mathcal{MS}}^{\mathcal{CS}} = \left\{ \bigcup \left\{ \mathcal{MS}_2 \mid (V_2, \mathcal{MS}_2) \in \mathcal{T}_{\mathcal{SS},\mathcal{MS}} \wedge (V_1, \mathcal{MS}_1) \sim_{\mathcal{CS}}^* (V_2, \mathcal{MS}_2) \right\} \mid (V_1, \mathcal{MS}_1) \in \mathcal{T}_{\mathcal{SS},\mathcal{MS}} \right\}.$$

An event log suitable for the subsequent application of process mining techniques can be obtained by generating an event for each message from  $\mathcal{MS}$ . There, the messages are grouped to process instances according to the traces in  $\mathcal{T}_{\mathcal{SS},\mathcal{MS}}^{\mathcal{CS}}$  and sorted according to their timestamps.

## 4 Case Study

For the case study at hand we examined a data set of 410 EDIFACT messages that has been supplied by an automotive supplier company. The data set consists of 180 DELFOR (*Delivery schedule message*), 75 DELJIT (*Delivery just in time message*), 28 GENRAL (*General purpose message*), 28 ORDCHG (*Purchase order change request message*), 2 ORDERS (*Purchase order message*) and 97 RECADV (*Receiving advice message*) messages received by the company in a period of approximately two months. The set does not contain any messages sent by the company. Note that we do not presume any further knowledge about the company's processes nor availability of any other meta-information about the data set, in particular with regard to completeness.

### 4.1 Methodology

We analyzed the data set with regard to the process models that can be mined by applying existing process mining algorithms on sets of process instances that have been recognized using the approach presented in Section 3. For interpreting the mined process models we referred to the EDIFACT standards and the therein specified purposes of particular message types. For evaluating the results we discussed the mined process models as well as our interpretation of these models with representatives from the company that has supplied the data set.

For performing the case study we implemented the correlation algorithm described in Section 3 in a plug-in<sup>5</sup> for ProM 6<sup>6</sup> [23]. We will further on refer to the plug-in as *correlation tool*. The correlation tool comes with a pre-defined set of 778 potential correlators for EDIFACT messages. These potential correlators

<sup>5</sup> The plug-in is publicly available in the *EDImine* package of ProM 6 nightly builds as of November 2011.

<sup>6</sup> ProM is developed at the Eindhoven University of Technology and currently regarded as the most prevalent tool in the area of process mining.

<http://www.processmining.org>

**Table 1.** Potential correlators occurring in the data set with average trace lengths resulting from key/value-correlation

Correlator	Avg. trace length [msgs.]
Profile_number	30.0
Order_document_identifier_buyer_assigned	14.7
Delivery_schedule_number	9.6
Previous_delivery_instruction_number	4.5
SID_Shipper_s_identifying_number_for_shipment	1.8
Release_number	1.0
Previous_delivery_schedule_number	1.0
Customer_reference_number	1.0
Contract_number	1.0
Shipment_reference_number	1.0

were automatically derived from the code list that defines the possible values an RFF (*Reference*) segment can hold according to the UN/EDIFACT standard, Release D10B<sup>7</sup>. We use this set of 778 potential correlators in our case study<sup>8</sup>.

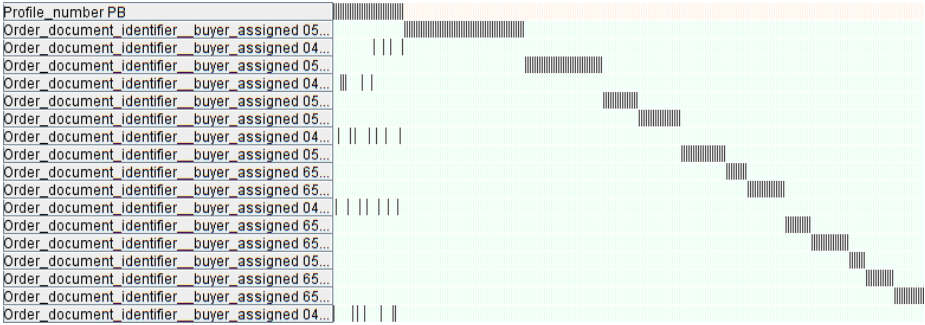
## 4.2 Correlation

A first examination of the messages in the data set reveals that ten of the 778 potential correlators derived from the possible RFF segment qualifiers actually occur in the examined data set. These ten correlators are shown in Table 1 ordered by average length of the traces (i.e., number of contained messages/activities) that result from applying key/value-correlation on the data set using the corresponding correlator. The average length of the traces may give an indication for the suitability of the corresponding correlator; however, this metric can also be misleading. For example, adding `Profile_number` and `Order_document_identifier_buyer_assigned` to the correlator set and applying the correlation algorithm without any non-reflexive synonymies and an empty concatenation set results in the set of 18 recognized traces visualized in Fig. 4. These 18 traces comprise 250 of the 410 messages. The visualization reveals that `Profile_number` is not a suitable correlator because the data element it denotes contains always the same value and is only present in ORDER and ORDCHG messages that also contain order numbers (denoted by `Order_document_identifier_buyer_assigned`). Thus, it can be removed from the correlator set without losing any traces.

A closer look at the data set reveals that most DELFOR (68%) and all RECADV messages contain the data element denoted by `Order_document_identifier_buyer_assigned`, and all DELJIT messages

<sup>7</sup> See <http://live.unece.org/trade/untdid/d10b/trsd/trsdrff.htm>

<sup>8</sup> However, the correlation tool allows a user to define and use any set of potential correlators.



**Fig. 4.** Recognized traces in a condensed visualization (screen-shot). Columns represent individual EDI messages and rows represent recognized traces. The labels on the left hand side show the correlator that was used for matching the messages of a particular trace. Dark markings in the matrix on the right hand side indicate membership of a message in a trace. Note that some traces exhibit partial overlaps on the left hand border of the matrix.

contain the data element denoted by `Delivery_schedule_number`. Thus, the correlator `Delivery_schedule_number`, which is rated third in terms of average trace length, is added to the correlator set. Also, with respect to the aforementioned three message types, the values contained in these two data elements match in 6 cases (of 12 resp. 28 cases total when comparing DELJIT with DELFOR and RECADV messages), giving a strong indication of the presence of a synonymy relationship between these two correlators. Hence, by putting these correlators in a synonymy group it is possible to connect traces consisting of DELFOR and RECADV messages with traces consisting of just DELJIT messages. Guided by this insight, the two correlators are added to a synonymy group resulting in a synonymy set  $SS = \{\{\text{Order\_document\_identifier\_buyer\_assigned}, \text{Delivery\_schedule\_number}\}\}$ . Note that although some of the remaining potential correlators may also lead to interesting results, we leave them unconsidered in this case study for simplicity. Thus, the resulting correlator set is  $\mathcal{C} = \{\text{Order\_document\_identifier\_buyer\_assigned}, \text{Delivery\_schedule\_number}\}$ .

The absence of partial overlaps in a new visualization of the preliminarily computed traces (as, for example, contrary to Fig. 4) indicates that the concatenation rule is not applicable to any of the previously defined correlators/synonymy groups. Hence, the concatenation set is left empty, i.e.,  $CS = \emptyset$ . Applying the correlation algorithm with this parameterization ( $CS, SS$ ) on the data set  $\mathcal{MS}$  yields 39 individual process instances  $\mathcal{T}_{SS, \mathcal{MS}}^{CS}$  comprising 382 messages in total, where each message is assigned to exactly one trace. This amounts exactly to the number of messages in the data set excluding messages of the GENRAL (*General purpose message*) message type. Hence, at this point the messages of the GENRAL message type are discarded and not included in subsequent process mining.



In order to derive the inter-organizational business process model, a process mining algorithm has to be applied. Hence, an event log suitable for the application of process mining algorithms is built from the messages in the data set according to the discovered process instances  $\mathcal{T}_{SS,MS}^{CS}$ . In this event log each message represents an event that is assigned to a specific process instance. Furthermore, the events are ordered according to the timestamps of the corresponding messages. Then a suitable process mining algorithm has to be selected, as discussed in the following.

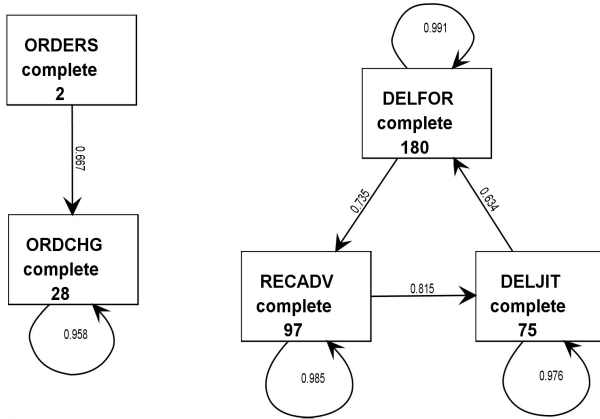
### 4.3 Mining the Inter-organizational Business Process Model

Different process mining algorithms exhibit different strengths and weaknesses with regard to properties such as representational bias, robustness (i.e., ability to deal with noise) or assumptions about completeness [19, pp. 159]. *Heuristic mining algorithms* consider frequencies of events and sequences when deriving a process model. Together with their favorable properties regarding representational bias, these factors make heuristic mining algorithms much more robust than most other approaches [19, p. 163].

The data set under consideration shows apparent signs that it is incomplete. For example, the appearance of multiple traces in the generated event log which consist of repeated ORDCHG messages without corresponding ORDERS messages is an indication that the data set is missing the initial ORDERS messages. This is likely due to the limited time frame under which messages were collected. Furthermore, knowing that the EDI messages in the data set were observed in a limited time window allows for the assumption that process instances may be missing leading or trailing messages/activities. This assumed incompleteness of the data set suggests that a relatively robust mining algorithm should be used. Hence, we choose the *Heuristics Miner* plug-in of ProM 6 [25] for mining a business process model from the correlated EDI messages.

Providing the previously generated event log as input to the *Heuristics Miner* plug-in yields the inter-organizational process model shown in Fig. 5.

A possible interpretation of the figure is that the observed EDI messages are artifacts of two separate sub-processes. The first process appears to be a *purchase order* process; it consists of an activity to place orders (*ORDERS*), succeeded by multiple requests for changing these orders (*ORDCHG*). The second process appears to be a continuous (i.e., looping) supply chain management (SCM) process, where customers continuously send *delivery forecasts* (*DELFOR*) and *just-in-time delivery* requests (*DELJIT*). When goods are received, customers send *receiving advices* (*RECADV*). Note that the GENERAL messages from the data set do not appear in the mined process model because the business information they convey is not covered by any of the correlators we have used in this case study.



**Fig. 5.** Process model derived from the resulting traces with the *Heuristics Miner* plug-in of ProM 6. The arrows represent dependency relations between activities; the associated values are measures for the strength of the dependency relation, where values close to 1 indicate a strong positive dependency. The values next to the activity labels indicate the number of events in the event log (i.e., the number of messages) that led to the recognition of the corresponding activity (see [25] for a detailed description of the elements in the graph).

## 5 Conclusion and Reflection

According to the EDIFACT standards a *Delivery just in time* message (DELJIT) is intended to provide “firm deliver instructions” with fine-grained time granularity (i.e., days, hours) whereas *Delivery schedule* messages (DELFOR) are intended for forecasts of supply needs with a more coarse-grained time granularity (i.e., months, weeks). Also, common sense dictates that a *Receiving advice* message (RECADV) should be received after corresponding DELJIT and/or DELFOR messages. These considerations suggest that a corresponding SCM process should look similar to the (fictional) process model illustrated in Fig. 6. The mined SCM sub-process model visualized in Fig. 5 differs from the process model illustrated in Fig. 6 insofar that (i) the causality relation (i.e., the “control flow”) represented by the arrows is in the opposite (and counter-intuitive) direction, (ii) that it contains repeated occurrences of individual messages/activities (length-one loops) and (iii) that it does not “end” with a receiving advice, but is a continuous (looping) process. Furthermore, one could ask why there is no link (i.e., no arrow in the graph) between the purchase order and SCM sub-processes. In the following, possible reasons for these differences are discussed.

A discussion of our results with representatives of the company that supplied the data set has led to some interesting insights. First, the company stated that it actually only uses the information from received DELFOR and DELJIT messages; received messages of other types are not processed. This means that



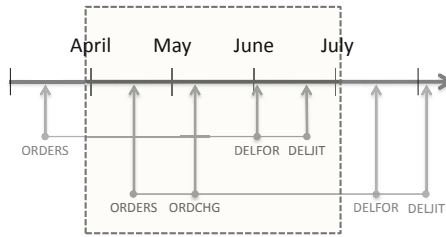
**Fig. 6.** Model of a fictional supply chain management process consisting of delivery forecasts, just-in-time delivery requests and receiving advices

the remaining message types are artifacts of processes that were originally designed and implemented unilaterally by customers of the company. Secondly, the company confirmed that a supply-chain-management process implemented by DELFOR, DELJIT and RECADV messages, including length-one loops as featured in the mined model, is indeed employed. However, as already conjectured before, the order of the messages/activities (the direction of the arrows) does not correspond to the real-world process. The question of what causes the missing link between the order and SCM sub-processes still remained open after the discussion. We assume that this might be due to the limited time frame in which the messages for the data set were collected. If the delay between ORDERS and ORDCHG messages on the one hand and DELFOR, DELJIT and RECADV messages on the other hand is long enough as to span over a significant portion of the period in which the messages were collected, this would prevent us from finding references between these groups of messages in the data set (cf. Fig. 7).

Regarding the inaccurate order of activities in the SCM sub-process we consider the possibility that this is a result of the observed data set being incomplete *within* the observed time window. In other words, the data set may not contain *all* messages that were indeed received by the company in a specific timespan. This in turn could lead to distortions in the mined process model such as an incorrect order of activities in a looping process. However, one cannot draw such a conclusion with certainty solely by analyzing a supposedly incomplete data set itself when no other constraints are a priori known.

The reason that the *General purpose* messages (GENERAL) do not lead to a corresponding activity in the mined process model is rather trivial: the GENERAL messages in the data set contained only free-text fields with maintenance information for administrators of EDI systems. Therefore, they were not picked up by any of the potential correlators that we have used in the case study. This accurately reflects the fact that the GENERAL messages in the data set do not represent tangible business activities.

In this paper we presented an initial case study using our approach for mining inter-organizational business process models from EDI messages. Discussions with representatives of the company that provided the data set for this case study have shown that the mined process model allowed for a number of substantial insights. The above mentioned peculiarities and limitations of the mined process model are assumed to appear due to the relatively small size and assumed incompleteness of the used data set. We expect that more reliable statements about the generalizability of our approach can be made when using a larger data set.



**Fig. 7.** Long delays between messages of the procurement and SCM sub-processes may exceed the time window in which messages were collected for the data set

However, based on our results we assume that such larger sets with longer periods of observation and completeness of the data would increase the usability of our approach.

We expect that our approach will help companies to rediscover and document the relationships in their business network and enable the subsequent application of Business Process Management methods on inter-organizational business processes realized by means of EDI. Our future research plans include - besides working with larger sets - complexity analyses of the used correlation algorithm as well as the development of methods for assessing the quality of recognized process instances. Furthermore, we intend to determine how inter-organizational process models can be most effectively visualized to account for the inter-organizational character of the mined process models.

## References

1. Agrawal, R., Gunopulos, D., Leymann, F.: Mining Process Models from Workflow Logs. In: Schek, H.-J., Saltor, F., Ramos, I., Alonso, G. (eds.) EDBT 1998. LNCS, vol. 1377, pp. 469–483. Springer, Heidelberg (1998)
2. Berge, J.: The EDIFACT Standards. Blackwell Publishers, Inc. (1994)
3. Cook, J.E., Wolf, A.L.: Discovering Models of Software Processes from Event-Based Data. *ACM Transactions on Software Engineering and Methodology* 7(3), 215–249 (1998)
4. Datta, A.: Automating the Discovery of AS-IS Business Process Models: Probabilistic and Algorithmic Approaches. *Information Systems Research* 9(3), 275 (1998)
5. Diniz, P.C., Ferreira, D.R.: Automatic Extraction of Process Control Flow from I/O Operations. In: Dumas, M., Reichert, M., Shan, M.-C. (eds.) BPM 2008. LNCS, vol. 5240, pp. 342–357. Springer, Heidelberg (2008)
6. Dustdar, S., Gombotz, R.: Discovering Web Service Workflows Using Web Services Interaction Mining. *International Journal of Business Process Integration and Management* 1(4), 256–266 (2006)
7. Emmelhainz, M.A.: EDI: A Total Management Guide, 2nd edn. John Wiley & Sons, Inc., New York (1992)
8. Engel, R., Krathu, W., Zapletal, M., Pichler, C., van der Aalst, W.M.P., Werthner, H.: Process Mining for Electronic Data Interchange. In: Huemer, C., Setzer, T. (eds.) EC-Web 2011. LNBIP, vol. 85, pp. 77–88. Springer, Heidelberg (2011)

9. Ferreira, D.R., Gillblad, D.: Discovering Process Models from Unlabelled Event Logs. In: Dayal, U., Eder, J., Koehler, J., Reijers, H.A. (eds.) BPM 2009. LNCS, vol. 5701, pp. 143–158. Springer, Heidelberg (2009)
10. Greco, G., Guzzo, A., Pontieri, L., Sacca, D.: Discovering Expressive Process Models by Clustering Log Traces. *IEEE Transactions on Knowledge and Data Engineering* 18(8), 1010–1027 (2006)
11. Hand, D.J., Smyth, P., Mannila, H.: *Principles of Data Mining*. MIT Press, Cambridge (2001)
12. Hill, N., Ferguson, D.: Electronic Data Interchange: A Definition and Perspective. *EDI Forum: The Journal of Electronic Data Interchange* 1, 5–12 (1989)
13. Nezhad, H.R.M., Saint-paul, R., Benatallah, B., Casati, F.: Deriving Protocol Models from Imperfect Service Conversation Logs. *IEEE Transactions on Knowledge and Data Engineering* 20(12), 1683–1698 (2008)
14. Nezhad, H.R.M., Saint-paul, R., Benatallah, B., Casati, F., Andritsos, P.: Message Correlation for Conversation Reconstruction in Service Interaction Logs. Technical Report DIT-07-010, University of Trento (2007)
15. Nezhad, H.R.M., Saint-paul, R., Casati, F., Benatallah, B.: Event Correlation for Process Discovery from Web Service Interaction Logs. *VLDB Journal* 20(3), 417–444 (2011)
16. Nurmilaakso, J.-M.: EDI, XML and e-business frameworks: A survey. *Computers in Industry* 59(4), 370–379 (2008)
17. Pauw, W.D., Lei, M., Pring, E., Villard, L., Arnold, M., Morar, J.: Web Services Navigator: Visualizing the Execution of Web Services. *IBM Systems Journal* 44(4), 821–845 (2005)
18. Srikant, R., Agrawal, R.: Mining Sequential Patterns: Generalizations and Performance Improvements. In: Apers, P.M.G., Bouzeghoub, M., Gardarin, G. (eds.) EDBT 1996. LNCS, vol. 1057, pp. 3–17. Springer, Heidelberg (1996)
19. van der Aalst, W.: *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer (2011)
20. van der Aalst, W., Dumas, M., Ouyang, A.R.C., Verbeek, H.: Conformance Checking of Service Behavior. *ACM Transactions on Internet Technology* 8(3), 29–59 (2008)
21. van der Aalst, W., Reijers, H.A., Weijters, A.M., van Dongen, B., Alves de Medeiros, A., Song, M., Verbeek, H.: Business Process Mining: An Industrial Application. *Information Systems* 32, 713–732 (2007)
22. van der Aalst, W., Verbeek, H.: Process Mining in Web Services: The WebSphere Case. *IEEE Bulletin of the Technical Committee on Data Engineering* 31(3), 45–48 (2008)
23. Verbeek, H.M.W., Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: XES, XESame, and ProM 6. In: Soffer, P., Proper, E. (eds.) CAiSE Forum 2010. LNBIP, vol. 72, pp. 60–75. Springer, Heidelberg (2011)
24. Vollmer, K., Gilpin, M., Stone, J.: *B2B Integration Trends: Message Formats*. Forrester Research (2007)
25. Weijters, A., van der Aalst, W.: Rediscovering Workflow Models from Event-Based Data using Little Thumb. *Integrated Computer-Aided Engineering* 10(2), 151–162 (2003)

# Data Transformation and Semantic Log Purging for Process Mining\*

Linh Thao Ly<sup>1</sup>, Conrad Indiono<sup>2</sup>, Jürgen Mangler<sup>2</sup>, and Stefanie Rinderle-Ma<sup>2</sup>

<sup>1</sup> Institute of Databases and Information Systems, Ulm University, Germany  
thao.ly@uni-ulm.de

<sup>2</sup> University of Vienna, Faculty of Computer Science, Austria  
{conrad.indiono,juergen.mangler,stefanie.rinderle-ma}@univie.ac.at

**Abstract.** Existing process mining approaches are able to tolerate a certain degree of noise in the process log. However, processes that contain infrequent paths, multiple (nested) parallel branches, or have been changed in an ad-hoc manner, still pose major challenges. For such cases, process mining typically returns “spaghetti-models”, that are hardly usable even as a starting point for process (re-)design. In this paper, we address these challenges by introducing data transformation and pre-processing steps that improve and ensure the quality of mined models for existing process mining approaches. We propose the concept of semantic log purging, the cleaning of logs based on domain specific constraints utilizing semantic knowledge which typically complements processes. Furthermore we demonstrate the feasibility and effectiveness of the approach based on a case study in the higher education domain. We think that semantic log purging will enable process mining to yield better results, thus giving process (re-)designers a valuable tool.

**Keywords:** Process mining, Data transformation, Log purging, Process constraints.

## 1 Introduction

Process Mining has developed as promising technique for analyzing process-oriented data in various ways: process discovery refers to the extraction and establishment of process models, process analysis addresses the analysis and comparison of processes. Both kinds of techniques are based on logs that store temporally ordered events about process executions. Though the Process Mining Manifesto [1] states that process event logs should be treated as “first class citizens” by enterprises, reality is often different: data are distributed over several sources and are often not directly available as temporally structured event logs. However, neglecting such “second class level” data sources might lead to missing out relevant and valuable analysis results as well as limit the applicability of process mining techniques dramatically. Hence, in order to utilize application

---

\* The work presented in this paper has been partly conducted within the project I743-N23 funded by the Austrian Science Fund (FWF).

data sources, the development of adequate methods for *data transformation* as pre-processing phase of a process mining project becomes essential.

Similar to data mining, data quality is crucial for process mining. In this case, the heterogeneity of log data sources (including applications that enable non-sequential or flexible execution of tasks) might account for the following cases that usually hamper the application of process mining techniques:

1. Incorrect/incomplete log data (in the following addressed as noise).
2. Log data contributed by parallel branches.
3. Infrequent traces.
4. Log data contributed by ad-hoc changed instances.

Case (1) has been mainly tackled at the algorithmic level. The *Genetic Miner* [2] and the *Heuristics Miner* [3], for example, are by design able to tolerate a certain degree of noise in the logs. We do not focus on such kinds of incorrect / incomplete log data and rely on the features of the process mining tool. For case (2), post-processing methods in order to reduce the "spaghetti-degree" of the resulting process models have been proposed [4]. A special case worth mentioning is processes which include late modeling (e.g. BPMN ad-hoc subprocesses). Allowing to invoke a set of optional process steps in arbitrary order is identical to the modeling of parallel branches for all possible sequence variations of these steps. Detection of such cases is, to the best of our knowledge, currently not possible. (3) Infrequent traces are the result of normal process execution, but sparsely occur in process logs because they depend on rare conditions (e.g. credit lines over €100 million have to be signed by the board of directors). Sometimes they are implemented using automatic exception handling logic as can be defined in current process execution languages (i.e. planned exception handling). Current frequency-based approaches would discard cases of category (3). Case (4), to the best of our knowledge, is currently not covered by existing approaches: results may range from wrong process models to complete graphs (for parallel or ad-hoc changed instances). Ad-hoc changed instances typically involve the intervention of humans (i.e. unplanned exception handling).

In this paper, we will address the following two research questions:

1. *Data transformation*: How can we support the (automatic) transformation of temporal application data to process logs?
2. *Data quality*: Is it possible to increase the process mining quality for existing mining algorithms through semantic pre-processing?

The contribution of this paper is two-fold. First, we will present a method for query-based data collection and transformation of temporal application data into process logs. Secondly, a data cleaning method based on *semantic log purging* will be proposed. This method utilizes semantic knowledge on the processes in the form of process constraints. Based on the process constraints, (1) process logs

can be checked for instances that violate one or more process constraints. (2) The log can be filtered according to expected behavior thus allowing to untangle ad-hoc or highly parallel sections. In both cases, filtering out constraint violations can lead to an improvement of the process mining results in terms of reduced “spaghetti-degree” [5] of the processes.

In order to evaluate our approaches on data transformation and semantic log-purging for data cleaning, we apply them to a realistic data set from the higher education domain (HEP project, [www.wst.univie.ac.at/communities/hep/](http://www.wst.univie.ac.at/communities/hep/)). The HEP data set consists of ten different process types (reflecting different courses). For this paper we selected one course, which took place in three consecutive years. Collecting data for this course yielded 330 instances (students) with 18511 events. We also, together with the instructors, collected a set of process constraints, that served as the basis for the semantic log purging. In the end we used a goal process to compare the result quality of the mined processes.

The paper is structured as follows: In Sect. 2, we introduce the applied methodology and provide fundamentals on the HEP data set. Sect. 3 addresses the transformation of temporal application data to process logs. Semantic log purging is introduced in Sect. 4. The results from our study are summarized in Sect. 5. Related work is discussed in Sect. 6. The paper closes with a summary and an outlook in Sect. 7.

## 2 Applied Methodology and the HEP Data Set

As mentioned in the introduction, this paper focuses on new techniques for data transformation and cleaning in order to improve the quality of mined processes, embedded within a case study to show the feasibility of the approach. In this section, we describe considerations regarding the applied methodology as well as the raw data set and the associated reference process models.

### 2.1 Methodology

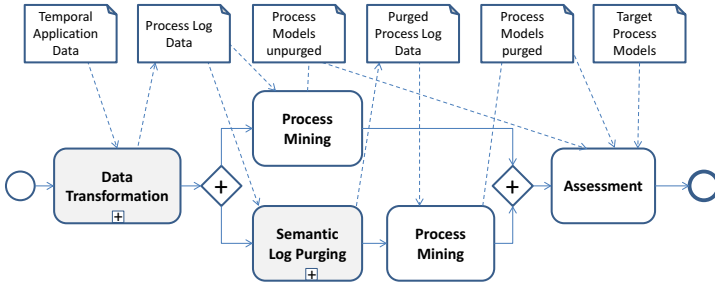
Figure 1 depicts the methodology applied in this paper, comparing process mining results with and without pre-processing by semantic log purging.

**Data transformation.** Traditionally, the raw log data is transformed into some data format. In the case of applications aggregating several process steps, this may also include transformation and import of data into a database.

**Process mining.** In order to obtain a process model from the logs, we apply the *Heuristics Miner* using the ProM process mining framework [6]. Being the mining approach most resilient towards noise [3], the Heuristics Miner is not as prone to deriving “spaghetti models” as for example the *Alpha Algorithm*. All mining algorithms in general seem to have problems with parallel executions which are seldom correctly detected and lead to a cobweb of intertwined tasks.

**Assessment.** In the final step, models are typically analyzed and sanity-checked.





**Fig. 1.** Methodology for evaluating the effectiveness of semantic log purging

As outlined in the introduction, our goal is to clean the log in order to improve the overall quality of the mined process models. We assume that the quality of a mined process is high when an assessment concludes that: (i) Infrequent traces are correctly included in the result. (ii) Parallel branches are correctly identified. Logs stemming from processes which include late modeling yield meaningful results (a small set of branches). (iii) Ad-hoc changed instances stemming from manual repair are not incorporated in the process model. This implies that no tasks and/or edges have to be removed or added.

Our idea is that with a small set of constraints we are able to (1) filter the logs on a semantic basis without purging infrequent traces, (2) separate parallel branches, and (3) select typical execution patterns for ad-hoc execution. Thus we introduced the lower branch in Fig. 1 where we redefine/add the steps:

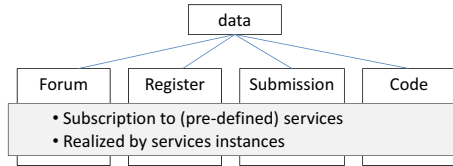
**Data transformation.** In order to allow for a wider range of data sources, we introduce a DSL (domain specific language) that allows for performant, stream-based live-accumulation and transformation of log data to arbitrary formats (MXML, XES, CSV).

**Semantic log purging.** Based on expert interviews, we identify a set of fundamental constraints that a process has to obey. These constraints are used to semantically clean the log data obtained from the transformation step. In particular, the constraints help to enforce certain expected behaviors in the mined models. If it is possible to identify parallel branches, the constraints can also be used to separate these branches, with the intention to mine them separately.

## 2.2 Raw Data from a Blended Learning Environment

In order to evaluate our approach we utilized data collected through our university's SOA learning platform [7]. In particular, we selected a set of 10 different computer science courses (process) that were known to (1) be conducted every year, (2) have a reasonably stable underlying process, and (3) utilize one or more of 4 learning services (Forum, Submission and Grading, Registration, Program

Code Evaluation). For the purpose of this paper, we selected one course, which took place 3 times over the last three years, was attended by over 330 students (some of them attended multiple times) and yielded a total of 18511 events.



**Fig. 2.** Abstract view on HEP data

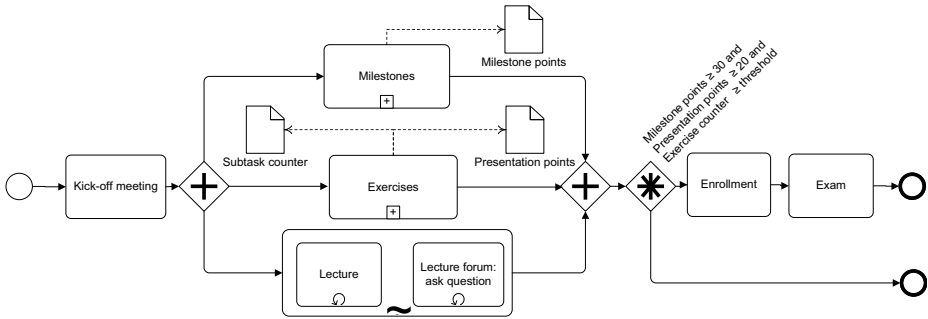
As depicted in Fig. 2, the data was collected from a set of existing learning services. The particular course we selected consisted of Forum (to ask questions), Registration (for appointments and topics) and Submission (for uploading exercises). For this paper, we created a snapshot and anonymized the original data.

### 2.3 The Reference Process Model

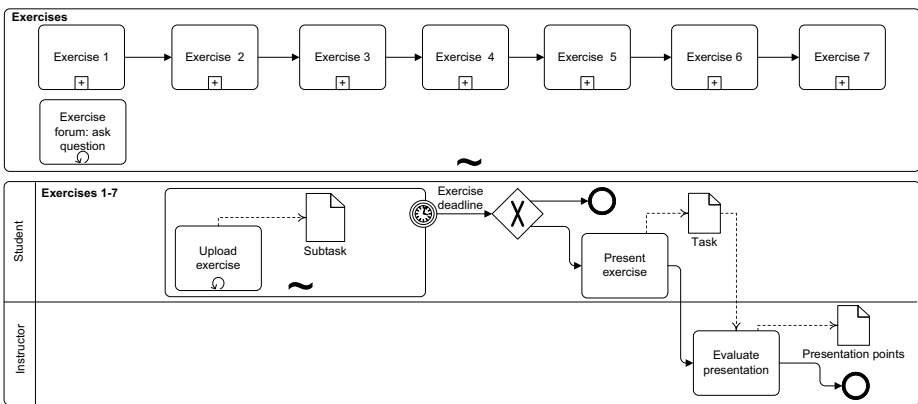
From interviews with the involved actors in selected learning process, we created the following reference process model (cf. Fig. 3 and Fig. 4) in order to be able to compare it to the results of the mining. The model describes the process as envisioned by the three course instructors. For each student (process instance) the course starts with a **Kickoff Meeting**, followed by the parallel execution of three subprocesses. Within **Exercises** the students have to solve up to 7 tasks consisting of different subtasks within a certain time frame (cf. Fig. 4). After the timely **Upload** of a solution, the student **presents** the solution. This presentation is **evaluated** by the instructor. In parallel to all subtasks, the students might ask questions in the **Forum**. A similar procedure is executed for the **Milestone** subprocess during which students upload and present solutions that are evaluated by the instructor. For both, **Exercises** and **Milestones** the student collects points. This process, though quite small, offers possible challenges for process mining, i.e., parallelism, ad hoc process fragments, and loops.

## 3 Transformation of Temporal Application Data to Process Logs

As described in the previous section, in the real world logs often are not readily available but have to be aggregated from a multitude of data sources and formats. Sometimes, processes include applications that, in turn, allow for a series of steps (i.e. subprocesses). In order to incorporate such information into process logs, often a series of complex transformation steps involving databases has to be carried out (which cannot be done at runtime). The goal is to generate chronological log files suitable for process mining.



**Fig. 3.** Reference course process model (super process in BPMN notation)

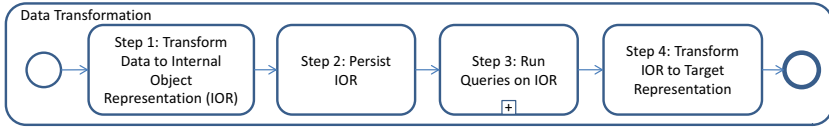


**Fig. 4.** Reference course process model (subprocess **Exercises** in BPMN notation)

In order to avoid the above mentioned shortcomings, we designed a data extraction and transformation method building on a functional query-based domain specific language (DSL) written in Haskell. The full tool-chain used for extracting the data for this paper is available under [https://github.com/indygemma/uni\\_bi2](https://github.com/indygemma/uni_bi2). Advantages of our approach include:

- Access data from heterogeneous sources without manual intermediate steps.
- Utilization of data streams instead of tables. This leads to efficient memory utilization for large data sets.
- For the data streams, arbitrary nesting of selections, projections, updates, and joins becomes possible.
- Once defined, transformations can be reused for equal or similar data sources.

In the following, we give an overview of the involved concepts. We choose a CSV based format as it can be imported into process mining and semantic log purging tools. The CSV file consisted of: (1) timestamp, (2) instance id (i.e student id),



**Fig. 5.** The data transformation subprocess

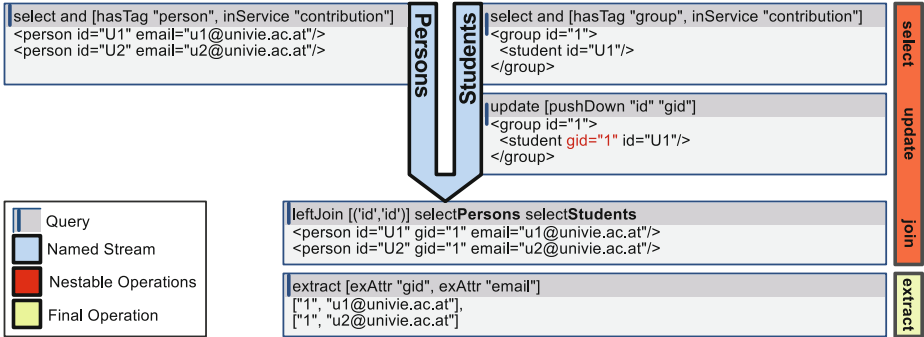
(3) agent id (i.e. originator of the event), (4) role (i.e. student, lecturer, tutor), (5) task name, and (6) data elements encoded as JSON (Javascript Object Notation). We use the four-step process depicted in Fig. 5 to break up transformation tasks for single data sources into a series of subtasks.

*Step 1 - Transform to IOR:* In order for queries to work uniformly on heterogeneous data, we transform the original data sets into an internal object representation (IOR). Data formats that are relevant for our example are: CSV, XML and raw file metadata (access time, file size). Since some of the data is deeply hierarchical, one of the design goals was to keep the hierarchical nature intact in the IOR. In order to also enable data stream operations, children are enriched with data properties of their parents (push down). For our data, an object instance corresponds to a single XML/CSV element (e.g. student). The IOR is thus populated by crawling once over the initial data set and creating object instances for each encountered element.

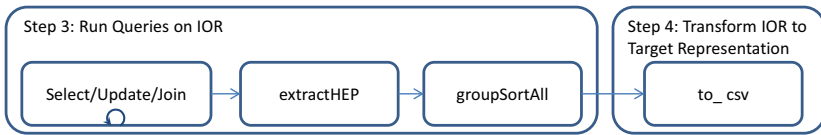
*Step 2 - Persist IOR:* In step 2, the IOR is compressed and serialized to disk. For step 3, files are only read when required. The resulting list of files represents the index on which queries can operate in the next step.

*Step 3 - Run Queries on IOR:* In this step, we define query functions that operate on the previously defined object streams. The idea is to apply arbitrary transformations on the objects before extracting them and passing them on for conversion to the target representation. The index is iterated until all the objects required for the subsequent step are present. The basic available operations are derived from relational algebra: select and join. Furthermore, it is possible to update (transform) objects according to custom rules. All operations return object streams, thus allowing for arbitrary nesting of operations. The DSL further supports a *Fuzzy Join*, which works on value ranges or value functions. The final operation in step 3 typically is the data extraction, that transforms objects into data rows. Fig. 6 shows a simple example, where a person record and a group record are selected, then joined (after the group records hierarchical structure is enriched (update)) and finally extracted. The extraction function required in our case was *groupSortAll*. Altogether, for our course data set, the algorithm as depicted in Fig. 7 was applied.

*Step 4 - Transform IOR to Target Representation:* In step 4, the extracted object rows are transformed into the final log (format).



**Fig. 6.** A simple query example



**Fig. 7.** Algorithm applied within step 3

A related tool for data transformation is XESame [6], which enables domain experts without programming experience to create transaction logs from data sources. This is achieved by a GUI-driven interface, wherein the user defines the mapping of source fields to target fields in the log. To make this mapping possible, XESame expects the input data to be tabular. Our approach does not make assumptions on the input nor on the output data, allowing all kinds of structured data formats – be they hierarchical or tabular – to serve as input. Employing an embedded DSL does require users to be comfortable with programming; but this allows for arbitrary complex data transformations in the form of flexible queries applied uniformly on heterogeneously structured data.

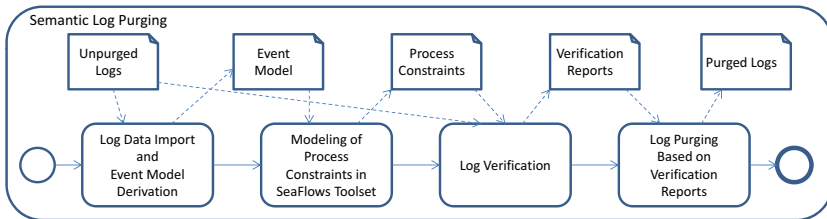
## 4 Semantic Log Purging

In contrast to existing algorithmic approaches to deal with noise data such as the Heuristics Miner [3], semantic log purging aims at cleaning the log at the semantic level. The basic idea of semantic log purging is to improve the quality of the logs by checking the encoded cases for compliance with fundamental domain-specific constraints. The logs representing cases that are incorrect with respect to these constraints are purged from the log set. It is notable that while noise stemming from spurious data (e.g., missing events) is often characterized by its low frequency of occurrence within the logs, cases violating semantic constraints can even outnumber the cases complying with the constraints if this behavior

is enabled by the system. Thus, semantic log purging may favor less frequent behaviors (e.g., *infrequent traces* as described in Sect. 11) and introduces a bias with respect to desired properties of the mined process model. Unlike frequency-based log cleaning, our approach does not purge infrequent traces unless they violate imposed constraints.

The specific semantic constraints relevant to the application can be obtained for example by interviewing domain experts. Clearly, the choice of semantic constraints used for log purging heavily affects the mined process models. Therefore, experimenting with constraints with different enforcement levels (e.g., high or low) can be helpful to identify the constraint set that leads to the best results with respect to the desired process model properties described in Sect. 2.1. Thus, an iterative approach is recommended.

The overall process of semantic log purging is illustrated in Fig. 8. The particular steps are detailed in the following.



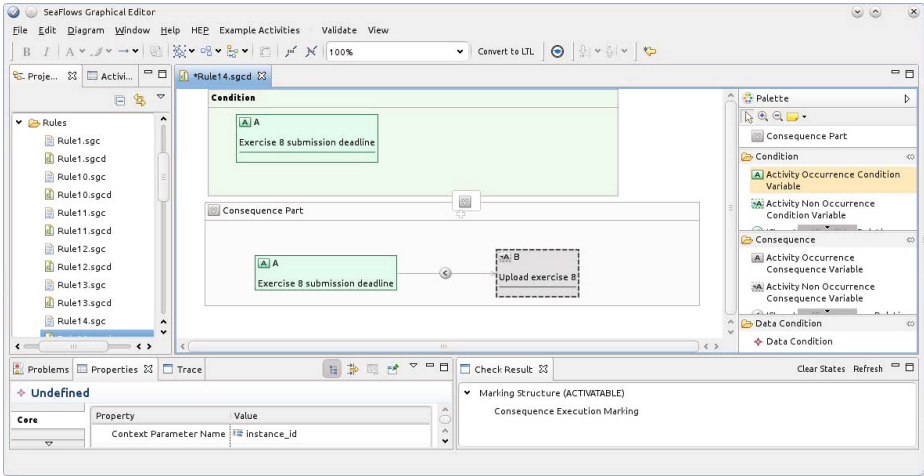
**Fig. 8.** The overall process of semantic log purging

*Step 1 - Log Import:* We opted for the SeaFlows Toolset [8] in order to conduct semantic log purging. SeaFlows Toolset is a framework for verifying process models, process instances, and process logs against imposed constraints. It enables graphical modeling of constraints based on the events and their parameters (e.g., originator or event-specific data) contained in the logs. In an initial step, the logs in the form of CSV files are imported into SeaFlows Toolset where an event model for the logs is established. In particular, all events and their parameters are identified and can be used for specifying constraints.

*Step 2 - Constraint Specification:* Based on interviews of a domain expert, we identify fundamental semantic constraints that have to be obeyed by the cases. Table 1 summarizes these constraints. We modeled the identified constraints in SeaFlows using the event model of the process logs. Fig. 9 shows a screenshot of the SeaFlows graphical constraint editor where the constraints are designed. In SeaFlows, a constraint is modeled as an directed graph consisting of a condition part and consequence parts. Fig. 10 depicts the modeled constraint  $c_2$  and the corresponding logic formula. More details on the underlying formalism called *compliance rule graphs* is provided in [9].

**Table 1.** Examples of semantic constraints imposed on the example course

Constraint	Enforcement level
c <sub>1</sub> For each milestone, no upload must take place after the corresponding milestone deadline.	high
c <sub>2</sub> For each exercise, no upload must take place after the corresponding exercise deadline.	high
c <sub>3</sub> For each uploaded milestone, the instructor gives feedback.	low



**Fig. 9.** Screenshot of the SeaFlows Graphical Constraint Editor

*Step 3 - Log Verification:* In the third step, the cases of the log are automatically checked for compliance with the modeled constraints. In order to select specific branches for mining, there need to be additional constraints to remove tasks from unwanted branches. In our example we created separate logs for exercises and milestones. As shown in Fig. 11, SeaFlows Toolset then provides a detailed report on the detected violations of each case. In particular, for each case the violation report details which constraints are violated through which events.

*Step 4- Log Purging:* Based on the detailed verification reports, we purged the log. For the case study, we removed only those cases violating constraints with strict enforcement level as only these really conflict with the expected process behavior. As illustrated by Fig. 12, the resulting log set is then used for the process mining (i.e. the Heuristics Miner which is resilient towards noise [3]).

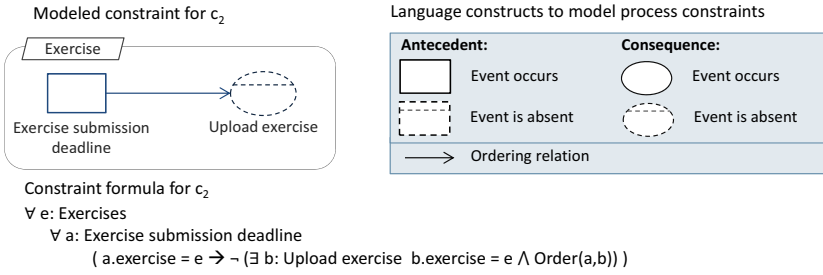


Fig. 10. Example of constraint  $c_2$

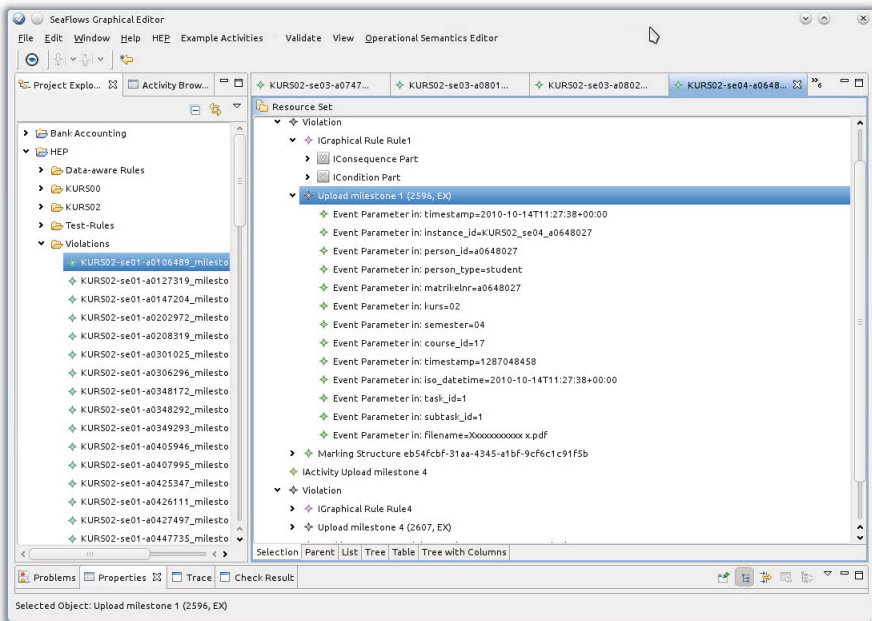


Fig. 11. Screenshot of the violation view

## 5 Evaluation Based on HEP

As discussed in Section 2.1 and illustrated in Fig. 1, we compare the process models mined from the original and from the purged logs against the expected reference models. For the overall case study, we analyzed ten course types (among them a course on advanced database systems (DBS) that serves as example in the paper). The example course has 18511 events and more than 330 cases. For brevity, we will present details on only the example course in the following.



### 5.1 Observations and Findings

Table 2 compares the process models mined from the original and from the purged log with each other. As discussed in Sect. 4, we exploited semantic knowledge about the subprocesses to mine them separately as their parallel nature makes it very difficult to mine meaningful process models otherwise. For each semester, in which the course took place, we mined a separate process model, respectively, as the particular process may vary in each semester.

**Table 2.** Quantitative comparison of the mined process models

	Process model: original		Process model: purged	
Subprocess	Tasks	Edges	Tasks	Edges
Exercise (SE01)	21	31	21	30
Milestone (SE01)	20	22	20	26
Exercise (SE03)	29	38	29	36
Milestone (SE03)	27	37	27	34
Exercise (SE04)	29	40	26	34
Milestone (SE04)	23	32	23	32

Qualitative analysis revealed two cases: the process models mined from purged logs are equal or smaller (case **A**)/ bigger (case **B**) than the original models (w.r.t. the nodes and edges). Interestingly, all models mined from purged logs are closer to the expected reference models (cf. Section 2.3) than the models mined from the original logs. In case **A**, spurious edges were removed in the models from purged logs reducing the “spaghetti-degree” of the models. In case **B**, the original models were overabstracted. Here, the models mined from purged logs contain more expected edges. In the Exercise subprocess (SE04), optional activities were removed from the model. In the Milestone subprocess (SE04), a spurious edge was replaced by an expected edge.

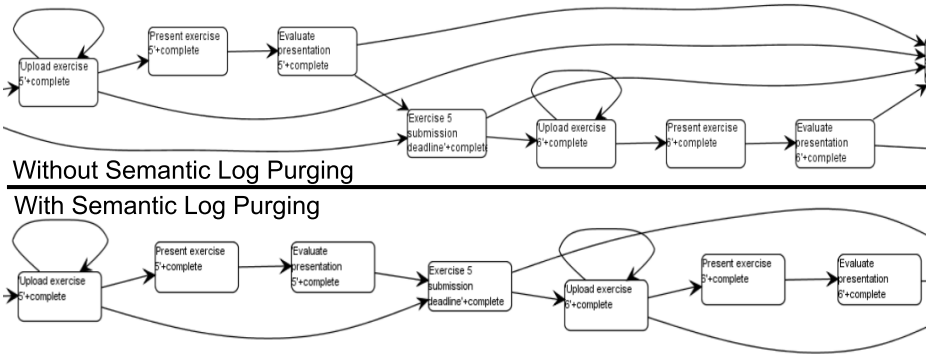
A partial process model of the DBS course obtained from mining using original logs



A partial process model of the DBS course obtained from mining using purged logs



**Fig. 12.** Original vs. purged logs (Exercise subprocess)



**Fig. 13.** Original mined process model vs. process model mined from purged logs

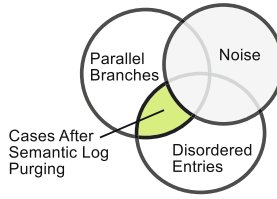
*Example.* Fig. 12 depicts the Exercise subprocess (SE03) mined from the original process logs and the one mined from purged logs. The process models cover the activities of the example course concerning the exercises conducted in the course. As the figure shows, both models mainly differ in the last part where the originally mined process model is more complex (i.e., has more edges) than the process model mined from the purged log.

In the detail view of these processes in Fig. 13, we can see that the model mined from the original logs contain edges that are not allowed by the reference process model. For example, from the submission deadline of Exercise 4 there is a path to the submission deadline of Exercise 6 without passing the submission deadline of Exercise 5. As the deadlines are system generated events, this cannot occur in practice. Such paths in the mined process model are caused by the submission events occurring after the submission deadline, which violates the imposed constraints (cf. Table 1). In contrast, the process model mined from the purged logs does not contain such edges as the cases with submission events after the deadline are removed from the log set.

### 5.2 Lessons Learned

We observed that semantic log purging indeed improves the quality of the mined models with respect to the properties described in Sect. 2.1. We observed that with a small set of constraints the process models mined from the purged logs are already quite close to the reference models. Due to the semantic log filtering, ad-hoc changed instances that violate the constraints were purged from the data set (cf. case (4) in Sect. 1) and thus do not contribute to the mined models.

As we also used constraints to extract branches (for separate mining), it was interesting to see that after removing all unnecessary events and incorrect / incomplete cases (noise), often only a small fraction of cases remained (see Fig. 14). The separate mining of parallel branches has proven to be a valid approach to reduce the “spaghetti-degree” of process models mined from log data contributed



**Fig. 14.** Fraction of log entries used for mining

by parallel branches (cf. case **(2)** in Sect. [11](#)). As the approach does not rely on frequency, infrequent traces are not filtered out and thus, can be incorporated in the mined process models (cf. case **(3)** in Sect. [11](#)).

Altogether, semantic log purging is an approach to clean the log with respect to expected properties and thus to support the mining of reference models from the process logs. It introduces a bias with respect to the resulting process model. Therefore, it “guides” the mining process.

It should be noted that the collection of process constraints can happen independently of whether or not expected reference models are created before the mining process. We also experimented with bigger sets of constraints containing also constraints with a lower enforcement level (nice-to-have constraints). As most logs violate these constraints, the data set would have become too small. Therefore, a lesson learned is that an iterative approach (consisting of choosing constraints, log purging, mining, re-choosing constraints, ...) is helpful to determine a suitable set of constraints. We can also consider the automatic application of constraint subsets in order to find an optimal set.

## 6 Related Work

In general, data quality is a crucial issue for applying analysis techniques such as data or process mining. For preparing data accordingly different cleaning techniques have been developed during the last decades, cf. [\[10\]](#). In particular, for cleaning data from errors or impossible values, integrity constraints have been utilized in the database area [\[11\]](#). Though the basic idea is similar to the one of this paper, there are two basic differences. First of all, no process-oriented data (represented by process logs) has been subject to data cleaning approaches so far. Process-oriented data might be particularly complex, considering at least control and data flow, time-relations, and organizational assignments. Accordingly, the associated process constraints might be quite complex as well, e.g., regulatory packages such as BASEL III. Further on, integrity constraints are mostly independent of the application context, hence correcting data errors can be accomplished without further domain knowledge (e.g., extinguishing data with  $AGE < 0$  [\[11\]](#)). In this paper, we extend these application-independent techniques by a data cleaning approach that utilizes knowledge on the application context of the data, representing, for example, knowledge on the order of certain teaching courses. In practice, the existence of such knowledge can often

be assumed, for example, medical guidelines [12], internal quality controls, or contractual policies [13].

Process mining offers a multitude of techniques to analyze logs from past process executions [1]. By now, a variety of tools for process analysis is comprised by the process mining framework ProM [6]. Key to the effectiveness of process mining is the acquisition and (pre-)processing of suitable logs. Here, particular challenges arise from the various sources of log data in real-world applications. In [14], Funk et al. describe their setup for product usage observation by means of process mining. Their approach allows for semantically annotating the logged data using ontologies. Mans et al. report on their experience from a case study on the application of process mining in the hospital context in [15]. To receive intelligible models, they had to abstract from low-level events. In our case, we rather had to provide more low-level events in order to obtain meaningful process models. As data sources are heterogeneous, no general approach can be provided for log pre-processing besides some fundamental strategies such as filtering particular events.

A common strategy to deal with “spaghetti-models” is to abstract from infrequent behavior (considered as noise) to yield simpler models, for example employed by the Heuristics Miner [3]. Here, we would like to stress that our approach does not rely on the frequency of observed behavior but rather introduces expectations with respect to the mined model. In [4], Fahland et al. introduce an approach to structurally simplify a mined process model while preserving certain equivalence notions. Such approaches are orthogonal to our work.

For auditing process logs with respect to certain properties (e.g., the constraints for semantic log purging), ProM offers the LTL Checker [16], a tool that allows for checking properties specified in *linear temporal logic* (LTL). The original LTL Checker only works at the granularity of activity labels. Thus, additional data conditions as used by some of the constraints we experimented with are not directly supported. An extension of the LTL Checker is introduced in [17] in the context of *semantic process mining* that allows for using concepts from ontologies as parameters of an LTL formula. Application of this approach requires the establishment of an ontology.

## 7 Summary and Outlook

This paper provides new techniques for data transformation and cleaning embedded within a case study to show the feasibility of the approach. In particular, we introduced a query-based data transformation approach that is able to transform temporal application data into process logs in different target representations (e.g., CSV). We further propose semantic log purging as an approach to improve the quality and intelligibility of the mined process model. In contrast to algorithmic approaches that deal with noise data by for example applying heuristics and thus can only capture incorrect process executions occurring infrequently, our approach cleans the log with respect to expected properties. In the case study within the HEP project, we were able to confirm the feasibility of our approaches. In future work, we will additionally examine the information on constraint-violating cases to analyze the reasons for deviations from the process.

## References

1. van der Aalst, W.M.P., et al.: Process Mining Manifesto. In: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) BPM Workshops 2011, Part I. LNBIP, vol. 99, pp. 169–194. Springer, Heidelberg (2012)
2. De Medeiros, A.K.A., Weijters, A.J.M.M.: Genetic process mining: an experimental evaluation. *Data Mining and Knowledge Discovery* 14 (2007)
3. Weijters, A., van der Aalst, W.M.P.: Rediscovering workflow models from event-based data using little thumb. In: ICAE, vol. 10, pp. 151–162 (2003)
4. Fahland, D., van der Aalst, W.M.P.: Simplifying Mined Process Models: An Approach Based on Unfoldings. In: Rinderle-Ma, S., Toumani, F., Wolf, K. (eds.) BPM 2011. LNCS, vol. 6896, pp. 362–378. Springer, Heidelberg (2011)
5. van der Aalst, W.M.P.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer, Heidelberg (2011)
6. Verbeek, H.M.W., Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: XES, XESame, and ProM 6. In: Soffer, P., Proper, E. (eds.) CAiSE Forum 2010. LNBIP, vol. 72, pp. 60–75. Springer, Heidelberg (2011)
7. Derntl, M., Mangler, J.: Web services for blended learning patterns. In: Proc. IEEE International Conference on Advanced Learning Technologies, pp. 614–618 (2004)
8. Ly, L.T., Knuplesch, D., Rinderle-Ma, S., Göser, K., Pfeifer, H., Reichert, M., Dadam, P.: SeaFlows Toolset – Compliance Verification Made Easy for Process-Aware Information Systems. In: Soffer, P., Proper, E. (eds.) CAiSE Forum 2010. LNBIP, vol. 72, pp. 76–91. Springer, Heidelberg (2011)
9. Ly, L.T., Rinderle-Ma, S., Dadam, P.: Design and Verification of Instantiable Compliance Rule Graphs in Process-Aware Information Systems. In: Pernici, B. (ed.) CAiSE 2010. LNCS, vol. 6051, pp. 9–23. Springer, Heidelberg (2010)
10. Rahm, E., Do, H.: Data cleaning: Problems and current approaches. *IEEE Data Engineering Bulletin* 23(4), 313 (2000)
11. Heiko Müller, J.F.: Problems, methods, and challenges in comprehensive data cleansing. Technical Report 164, Humboldt University Berlin (2003)
12. Dunkl, R., Fröschl, K.A., Grossmann, W., Rinderle-Ma, S.: Assessing Medical Treatment Compliance Based on Formal Process Modeling. In: Holzinger, A., Simonic, K.-M. (eds.) USAB 2011. LNCS, vol. 7058, pp. 533–546. Springer, Heidelberg (2011)
13. Rinderle-Ma, S., Mangler, J.: Integration of process constraints from heterogeneous sources in Process-Aware information systems. In: Int'l. Workshop Enterprise Modelling and Information Systems Architectures, EMISA (2011)
14. Funk, M., Rozinat, A., Alves de Medeiros, A.K., van der Putten, P., Corporaal, H., van der Aalst, W.M.P.: Improving Product Usage Monitoring and Analysis with Semantic Concepts. In: Yang, J., Ginige, A., Mayr, H.C., Kutsche, R.-D. (eds.) UNISCON 2009. LNBIP, vol. 20, pp. 190–201. Springer, Heidelberg (2009)
15. Mans, R.S., Schonenberg, H., Song, M., van der Aalst, W.M.P., Bakker, P.J.M.: Application of Process Mining in Healthcare - A Case Study in a Dutch Hospital. In: Fred, A., Filipe, J., Gamboa, H. (eds.) BIOSTEC 2008. CCIS, vol. 25, pp. 425–438. Springer, Heidelberg (2008)
16. van der Aalst, W.M.P., de Beer, H.T., van Dongen, B.F.: Process Mining and Verification of Properties: An Approach Based on Temporal Logic. In: Meersman, R., Tari, Z. (eds.) OTM 2005. LNCS, vol. 3760, pp. 130–147. Springer, Heidelberg (2005)
17. de Medeiros, A.K.A., van der Aalst, W.M.P., Pedrinaci, C.: Semantic process mining tools: Core building blocks. In: Proc. ECIS 2008, pp. 1953–1964 (2008)

# Improved Artificial Negative Event Generation to Enhance Process Event Logs

Seppe K.L.M. vanden Broucke, Jochen De Weerd, Bart Baesens, and Jan Vanthienen

Department of Decision Sciences and Information Management, KU Leuven, University of Leuven, Naamsestraat 69, B-3000, Leuven, Belgium

`firstname.lastname@econ.kuleuven.be`

**Abstract.** Process mining is the research area that is concerned with knowledge discovery from event logs. Process mining faces notable difficulties. One is that process mining is commonly limited to the harder setting of unsupervised learning, since negative information about state transitions that were prevented from taking place (i.e. negative events) is often unavailable in real-life event logs. We propose a method to enhance process event logs with artificially generated negative events, striving towards the induction of a set of negative examples that is both correct (containing no false negative events) and complete (containing all, non-trivial negative events). Such generated sets of negative events can advantageously be applied for discovery and evaluation purposes, and in auditing and compliance settings.

**Keywords:** process mining, process discovery, event logs, negative events.

## 1 Introduction

Many of today's organizations are currently confronted with an information paradox: the more business processes are automated, the harder it becomes to understand and monitor them. While information support systems such as Enterprise Resource planners (ERP) and modern Workflow Management systems (WfMS) provide some analysis and visualization tools in order to monitor and inspect processes – often based on key performance indicators, an abundance of data about the way people conduct day-to-day practices still remains untapped and concealed in so called event logs, capturing exactly which business activities happened at certain moments in time. The research area that is concerned with deep knowledge discovery from event logs is called process mining [1] and is often situated at the intersection of the fields of data mining and Business Process Management (BPM). Most of the attention in the process mining literature has been given to process discovery techniques [2], which focus specifically on the extraction of control-flow models from event logs [2, 3, 4, 5, 6]. One of the particular difficulties process discovery is faced with is that the learning task is commonly limited to the harder setting of unsupervised learning, since information about state transitions (e.g. starting, completing) that were prevented from

taking place (i.e. negative events) is often unavailable in real-life event logs and consequently cannot guide the discovery task [7]. In addition, process models frequently display complex structural behavior such as non-free choice, invisible activities (which were executed but not recorded in an event log) and duplicate activities, which make the hypothesis space of process mining algorithms harder to navigate. A third difficulty is the presence of noise in event logs, which often leads to the discovery of models which overfit the given log. In this paper, we focus our attention towards the first difficulty, namely the absence of negative examples in event logs.

Generating a robust set of negative events boils down to finding an optimal set of negative examples under the counteracting objectives of correctness and completeness. Correctness implies that the generation of false negative events has to be prevented, while completeness entails the induction of “non-trivial” negative events, that is, negative events which are based on constraints imposed by complex structural behavior, such as non-free choice constructs. The existence of the trade-off between these two goals is due to a “completeness assumption” one has to make over a given event log when generating artificial negative events. Under its most strict form, the completeness assumption states that the given event log contains all possible traces that can occur. Without some assumption regarding the completeness of a given event log, it would be impossible to induce any negative events at all, since no single candidate negative event can be introduced in the knowledge that the given log does not cover all possible cases. Note that process discovery algorithms make a similar assumption, in order to derive models which are not overly general.

In this paper, we refine an artificial negative event generation method first introduced in [7]. In the original version of this algorithm, a configurable completeness assumption is defined by proposing a window size parameter and a negative event injection probability. We aim to make the completeness assumption more configurable and the generation procedure more robust so that the induction of false negative events in cases where an event log does not capture all possible behavior is prevented (correctness), while also remaining able to derive “non-trivial” negative events, that is, negative events following from complex structural behavior (completeness).

Enhancing a process log with a correct and complete set of negative events as shown hereafter can prove beneficial for a number of reasons. First, supervised learners can now be employed in order to perform a process discovery task. Multi-relational learning techniques have already been applied in this context, using inductive logic programming in order to learn a process model from a given event log supplemented with negative events. We refer to [8, 9, 10, 11, 12] for a detailed overview of inductive logic programming and its applications in the field of process mining. Second, event logs supplemented with negative events can also be applied towards evaluation purposes, in order to assess fitness and precision of process discovery algorithms. For instance, the metrics in [7, 13] make use of event logs containing negative events to compose a confusion matrix in concordance with standard metric definitions in the field of data mining.

Third, since negative events capture which actions in a process could not occur, compliance and conformance related analysis tasks present themselves as a natural area of application for negative events. For example, auditors can cross-check a set of induced negative events with the expected behavior of real-life processes to determine if prohibited actions were indeed captured in this generated set of rejected activity state transitions. Another specific example is access rule mining. Often, users are not so much interested in the actual control policy already present in a specific process, as this policy might already have been formally specified, but rather in a more restrictive policy that reveals which access rules are unnecessary. Negative events can then stipulate that a particular agent did never perform a particular activity at a given time, even although it could be the case that this agent officially had the rights to do so. In this way, learners could distinguish access rules that are actually needed, instead of leaving the establishment of such rules and modifications of policies to modelers and business practitioners alone. As shown hereafter, our contributions provide some important benefits with regard to correctness and completeness when compared with earlier techniques, allowing to improve the obtained results when executing the aforementioned analysis tasks.

## 2 Related Work

Process mining is a relatively young research area. Cook and Wolf [14], Agrawal et al. [15], Lyytinen and Datta et al. [16] and van der Aalst et al. [2] can be considered as fundamental works in the field.

The notion of negative events was developed in the context of the application of machine learning and classification techniques towards process discovery. Maruster et al. [17] were among the first to investigate the use of rule-induction techniques to predict dependency relationships between activities. The authors do not rely on artificial negative event generation, but apply a uni-relational classification learner on a table of direct metrics for each process activity in relation to other activities. Ferreira and Ferreira [18] apply a combination of inductive logic programming and partial-order planning techniques, where negative events are collected from users and domain experts. Similarly, Lamma et al. [19] apply an logic programming towards declarative process discovery. Unlike the approach in [18], the authors assume the presence of negative events, without providing an immediate answer to their origin. In [9], the authors extend this approach and define a “negation response” constraint in order to derive information about which activities were prohibited to occur after some other activity type. Our approach differs from these methods, since we apply a window based trace comparison technique in order to derive negative events based on a given process log, leading to a larger set of negative examples. Furthermore, instead of deriving information about prohibited state transitions in the form of declarative rules, we immediately inject the negative examples in the given traces.

Goedertier et al. [7] represent the process discovery task as a multi-relational first-order classification learning problem and use the TILDE inductive logic



programming learner for their AGNEsMiner algorithm to induce a declarative control flow model. To guide the learning process, an input event log is supplemented with artificial negative events by replaying the positive events of each process instance and by checking if a state transition of interest corresponding to a candidate negative event could occur, more specifically by investigating if other traces can be found in the event log which do allow this state transition, and present a similar history of completed activities.

### 3 Preliminaries

Before outlining the artificial negative event generation algorithm, some important concepts and notations that are used in the remainder of this paper are discussed.

An event log consists of events that pertain to process instances. A process instance is defined as a logical grouping of activities whose state changes are recorded as events. We thus assume that it is possible to record events such that each event refers to a task (a step in a process instance), a case (the process instance) and that events are totally ordered.

As such, let  $X$  be a set of event identifiers,  $P$  a set of case identifiers, the alphabet  $A$  a set of activity types and the alphabet  $E$  a set of event types corresponding with activity life cycle state transitions (e.g. *start*, *assign*, *restart*, *complete*, etc.). An event can then be formulated as a predicate  $Event(x, p, a, e, t)$  with  $x \in X$  the event identifier,  $p \in P$  the case identifier and  $a \in A$  the activity type,  $e \in E$  the event type and  $t \in \mathbb{N}$  the position of the event in its process sequence. Note that events commonly store much more information (timestamps, originators, case data, etc.), but since we do not necessarily have to deal with this additional information in the context of generating artificial negative events, this information is left out in the remainder of this paper. Furthermore, we will assume that the state transition  $e$  for each logged event equals *complete*. The function  $Case \in X \cup L \mapsto P$  denotes the case identifier of an event or sequence. The function  $Activity \in X \mapsto A$  denotes the activity type of an event.

Let event log  $L$  be a set of sequences. Let  $\sigma \in L$  be an event sequence;  $\sigma = \{x | x \in X \wedge Case(x) = Case(\sigma)\}$ . The function  $Position \in X \times L \mapsto \mathbb{N}_0$  denotes the position of an event in its sequence. The set  $X$  of event identifiers has a complete ordering, such that  $\forall x, y \in X : x < y \vee y < x$  and  $\forall x, y \in \sigma, x < y : Position(x) < Position(y)$ . Let  $x.y \subseteq \sigma$  be two subsequent event identifiers within a sequence  $\sigma$ ;  $x.y \Leftarrow \exists x, y \in \sigma : x < y \wedge \nexists z \in \sigma : x < z < y$ . We will use this predicate in the context of single sequence which is therefore left implicit. Furthermore, a sequence of two events  $x.y$  with activity types  $a$  and  $b$  respectively can be abbreviated as  $\langle a, b \rangle$ . Each row  $\sigma_i$  in the event log now represents a different execution sequence, corresponding to a particular process instance, and can be depicted as  $\langle a, b, c, \dots, z \rangle$  with  $a, b, c, \dots, z$  the activity types of the ordered events contained in the sequence.

## 4 Artificial Negative Event Generation Algorithm

In this section, the artificial negative event generation algorithm is discussed in more detail. Since we extend the negative event generation algorithm as introduced in AGNEsMiner, we will avoid describing some parts in excessive detail, instead referring to [7] for a more detailed overview of some steps, allowing us to focus on new contributions.

A high level overview of the algorithm can be given as follows. Negative events record that at a given position in an event sequence, a particular event cannot occur. At each position in each event trace in the log, it is examined which negative events can be recorded for this position. In a first step, frequent temporal constraints are mined from the event log. Secondly, structural information is derived from these frequent temporal constraints. Finally, negative events are induced for each grouped process instance, based on two complementing generation techniques: a window based trace comparison routine and a structural introspective technique which derives negative events from dependency information mined from an event log.

### 4.1 Step 1: Mining Frequent Temporal Constraints

Frequent temporal constraints are constraints that hold in a sufficient number of sequences  $\sigma$  within an event log  $L$ . The following list of predicates express temporal constraints that either hold or not for a particular sequence  $\sigma \in L$ , with  $a, b, c \in A$  activity types:

$$\begin{aligned}
 \textit{Existence}(1,a,\sigma) &\Leftrightarrow \exists x \in \sigma: \textit{Activity}(x)=a \\
 \textit{Absence}(2,a,\sigma) &\Leftrightarrow \nexists x,y \in \sigma: \textit{Activity}(x)=a \wedge \textit{Activity}(y)=a \wedge x \neq y \\
 \textit{Ordering}(a,b,\sigma) &\Leftrightarrow \forall x,y \in \sigma, \textit{Activity}(x)=a, \textit{Activity}(y)=b: x.y \\
 \textit{Precedence}(a,b,\sigma) &\Leftrightarrow \forall y \in \sigma: \textit{Activity}(y)=b, \exists x \in \sigma: \textit{Activity}(x)=a \wedge x < y \\
 \textit{Response}(a,b,\sigma) &\Leftrightarrow \forall x \in \sigma: \textit{Activity}(x)=a, \exists y \in \sigma: \textit{Activity}(y)=b \wedge x < y \\
 \textit{ChainPrec}(a,b,\sigma) &\Leftrightarrow \forall y \in \sigma: \textit{Activity}(y)=b, \exists x \in \sigma: \textit{Activity}(x)=a \wedge x.y \\
 \textit{ChainResp}(a,b,\sigma) &\Leftrightarrow \forall x \in \sigma: \textit{Activity}(x)=a, \exists y \in \sigma: \textit{Activity}(y)=b \wedge x.y \\
 \textit{ChainSeq}(a,b,c,\sigma) &\Leftrightarrow (\forall z \in \sigma: \textit{Activity}(z)=c, \exists x,y \in \sigma: \textit{Activity}(x)=a \wedge \textit{Activity}(y)=b \wedge x.y.z) \vee \\
 &\quad (\forall y \in \sigma: \textit{Activity}(y)=b, \exists x,z \in \sigma: \textit{Activity}(x)=a \wedge \textit{Activity}(z)=c \wedge x.y.z) \vee \\
 &\quad (\forall x \in \sigma: \textit{Activity}(x)=a, \exists y,z \in \sigma: \textit{Activity}(y)=b \wedge \textit{Activity}(z)=c \wedge x.y.z)
 \end{aligned}$$

For an event log  $L$ , a temporal constraint  $C$  is considered frequent if its support is greater than or equal to a predefined threshold. Let  $C, D$  be temporal constraints. The support for a temporal constraint can be defined as:

$$\textit{Supp}_{\sigma \in L}(C,L) = \frac{|S|}{|L|}$$

for which  $S$  is a set containing the sequences  $\sigma$  for which  $C$  succeeds. Temporal constraints can also be combined to form temporal association rules of the form  $C \rightarrow D$ . The support of an association rule is defined as:

$$Supp_{\sigma \in L}(C \rightarrow D, L) = Supp_{\sigma \in L}(C, L)$$

The confidence of a temporal association rule is defined as:

$$Conf_{\sigma \in L}(C \rightarrow D, L) = \frac{Supp_{\sigma \in L}(C \wedge D, L)}{Supp_{\sigma \in L}(C, L)}$$

Temporal association rules are considered frequent if their support and confidence are greater than or equal to a predefined threshold. Since some activities occur more frequently than others in some event logs, the detection of frequent patterns must not be sensitive to the frequency of occurrence of a particular activity type in the event log. Consider for an example an event log  $L$  containing a large number of traces that do not contain activity type  $a$ . When checking the rule  $ChainResp(a, b, \sigma)$  over all traces in  $L$ , this expression will hold true for all traces that do not contain  $a$ . Since these traces make up for a large part of the event log, the support of  $ChainResp(a, b)$  would thus be high. To detect frequent patterns in an event log, it is therefore more important to look at the confidence of the association rule  $Existence(1, a, \sigma) \rightarrow ChainResp(a, b, \sigma)$ . The following frequent temporal association rules are derived from an event log  $L$  (left implicit):

$$Absence(2, a) \Leftarrow \forall a \in A: Supp_{\sigma \in L}(Absence(2, a, \sigma), L) \geq t_{absence}$$

$$Ordering(a, b) \Leftarrow \forall a, b \in A: Conf_{\sigma \in L}(Existence(1, a, \sigma) \wedge Existence(1, b, \sigma) \rightarrow Ordering(a, b, \sigma), L) \geq t_{ordering}$$

$$Precedence(a, b) \Leftarrow \forall a, b \in A: Conf_{\sigma \in L}(Existence(1, b, \sigma) \rightarrow Precedence(a, b, \sigma), L) \geq t_{succession}$$

$$Response(a, b) \Leftarrow \forall a, b \in A: Conf_{\sigma \in L}(Existence(1, a, \sigma) \rightarrow Response(a, b, \sigma), L) \geq t_{succession}$$

$$ChainPrec(a, b) \Leftarrow \forall a, b \in A: Conf_{\sigma \in L}(Existence(1, b, \sigma) \rightarrow ChainPrec(a, b, \sigma), L) \geq t_{chain}$$

$$ChainResp(a, b) \Leftarrow \forall a, b \in A: Conf_{\sigma \in L}(Existence(1, a, \sigma) \rightarrow ChainResp(a, b, \sigma), L) \geq t_{chain}$$

$$ChainSeq(a, b, c) \Leftarrow \forall a, b, c \in A: Conf_{\sigma \in L}(Existence(1, a, \sigma) \vee Existence(1, b, \sigma) \vee Existence(1, c, \sigma) \rightarrow ChainSeq(a, b, c, \sigma), L) \geq t_{triple}$$

Remark that our definition of the *Ordering* and *ChainSeq* temporal constraints and corresponding association rules differs from the original implementation so that the detection of these patterns is no longer sensitive to the frequency of occurrence of particular activity types. This is our first tangible improvement. We also define the following additional temporal constraint:

$$Iteration(l, n, v, \sigma) \Leftrightarrow \exists s \in \sigma, t = Position(s, \sigma): Activity(s) = Activity(u) \wedge \forall i \in [1, l], j \in [1, n]: \\ \forall x \in \sigma, y \in v, i = Position(y, v), t + i + j = Position(x, \sigma): \\ Activity(x) = Activity(y) \wedge l = |v|, v \subseteq \sigma, 1 = Position(u, v)$$

With the corresponding association rule:

$$\begin{aligned} \text{Iteration}(l,n,v) \Leftarrow \forall v \subseteq \sigma, \sigma \in L, l=|v|, n=2, 0 < l \leq 3: \\ \text{Conf}_{\sigma \in L}(\forall x \in v: \text{Existence}(1, \text{Activity}(x), \sigma) \rightarrow \\ \text{Iteration}(l,n,v,\sigma)) \geq t_{\text{iteration}} \end{aligned}$$

The *Iteration* temporal constraint holds in a sequence  $\sigma$  if another sequence  $v$  with length  $l$  iterates  $n$  times in  $\sigma$ . In order to keep the number of computations under control, we limit the discovery of iterations to length 3 and less, which repeat for minimally 2 times;  $n = 2$  and  $0 < l \leq 3$ .

The derivation of temporal frequent constraints can be compared with the more general problem of mining event sequences and episodes in an event log using apriori-like techniques, see for instance [20, 21, 22]. An alternative but ultimately similar method to mine structural behavior from an event log is suggested by Maggi et al. [23], who apply temporal logic property verification techniques, formalized in the Declare modeling language to discover structural behavior.

## 4.2 Step 2: Deriving Structural Information: Parallelism, Locality and Recurrence

Now that a set of temporal frequent constraints is constructed, it becomes possible to derive information about parallelism and locality of pairs of activities. Based on the constraints above, two derivation rules: *Parallel*( $a, b$ ) (symmetric) and *Local*( $a, b$ ) (non-symmetric), can be defined. The former denotes that two activities  $a, b \in A$  can occur concurrently, while the latter states that two activities occur in a serial manner, denoting an explicit dependency. For a more detailed description of these derivation rules, see [7].

Additionally, the *Iteration* temporal constraint as defined in the previous section allows us to formulate a loop discovery heuristic. However, loops cannot immediately be derived from the set of *Iteration* temporal constraints. To see why this is the case, consider the sequence  $\langle a, b, c, d, b, c, d, b, c, d, e \rangle$ . The following *Iteration* constraints hold in this sequence: *Iteration*(3, 2,  $\langle b, c, d \rangle$ ), *Iteration*(3, 2,  $\langle c, d, b \rangle$ ) and *Iteration*(3, 2,  $\langle d, b, c \rangle$ ). Since we are only concerned with the loop following from the first *Iteration* constraint, with the start activity in the correct, first position, we define *Loop*( $v$ ) as such:

$$\forall v \subseteq \sigma, \sigma \in L, 1 = \text{Position}(u, v), l = |v|: \text{Iteration}(l, 2, v) \wedge \exists a \in A: \text{Local}(a, u) \Rightarrow \text{Loop}(v)$$

## 4.3 Step 3: Generating Artificial Negative Events

**Window Based Negative Event Generation.** Once parallelism, recurrence and locality information is derived, negative examples can be introduced in given process instances. Negative events record that at a particular position in an event sequence, a particular event cannot occur. At each position  $k$  in each event sequence  $\tau_i$ , it is examined which negative events can be introduced at this

position. In a first step, the event log is made more compact by grouping process instances that have identical sequences  $\sigma \in L$  into grouped process instances  $\tau \in M$ . In the next step, all negative events are induced for each grouped process instance  $\tau_i$  (the “positive trace” under consideration) by checking at any given positive event  $x_k \in \tau_i$  whether another event of interest (a “candidate negative event”)  $z_k$  with activity type  $b \in A \setminus \{Activity(x_k)\}$  also could occur. Thus, for each positive event  $x_k \in \tau_i$ , it is tested whether there exists a sequence  $\tau_j \neq \tau_i \in M$  in the event log in which an event  $y_k$  has taken place that is similar to  $z_k$  and where both sequences present a similar history up until that point. Note that we consider two events being similar if their activity types are identical. If such a similar “disproving sequence” can not be found, such behavior is not present in the event log  $L$ , meaning that the candidate event indeed cannot occur. Consequently, candidate negative event  $z_k$  cannot be disproved and is added at position  $k$  in the positive sequence  $\tau_i$ . Finally, the negative events in the grouped process instances are used to induce negative events into the similar non-grouped sequences. Usually, a large number of negative events can be generated, so that a probability  $\pi$  is introduced as a threshold for injecting negative events into the ungrouped sequences.

To address the problematic nature of the completeness assumption of an event log under recurrent (loops) and concurrent (parallelism) behavior, we exploit the previously mined parallelism and loop information to generate parallel and looping variants of traces, by swapping parallel activity types and inserting or removing loops where possible. We now thus compare the positive trace under consideration  $\tau_i$  with each  $\tau_j^{\sim} \in AllVariants(\tau_j)$ , with *AllVariants* a function which returns the set of all parallel and loop variants of sequence  $\tau_j$ . Generating these variants results in a larger number of traces which will be used when evaluating a negative event, thus resulting in a weaker completeness assumption. The addition of the generation of variants based on recurrence is a second contribution.

Even when parallel and loop variants are considered, incorrect negative events could still be induced, due to the possible recursive and complex nature of recurrence and concurrency (nested loops, for example). Instead of trying to deal with this complex behavior by adjusting the *Loop* and *Iteration* constraints further, a window size parameter *windowSize* is introduced to limit the number of events which are compared when evaluating a candidate negative event (i.e. how far we look back before  $x_k \in \tau_i$  and  $y_k \in \tau_j^{\sim}$ ). The larger the window size, the less probable that a similar subsequence is contained by the other sequences in the event log, and the more probably that a candidate negative event will be introduced. Reducing the window size makes the completeness assumption less strict. An unlimited window size (a maximum-length comparison between sequences) results in the most strict completeness assumption. Note that history-dependent processes generally will require a larger window size to correctly detect all non-local dependencies. When the window size is limited to 1, it is no longer possible to take into account non-local dependencies, so that negative events following from this behavior will not be generated. This underlines the trade-off as discussed

before in the introduction. Using a higher window size leads to the generation of more valuable negative events, that is, negative events derived from non-local dependencies. On the other hand, using an increased window also leads to a more strict completeness assumption, so that candidate negative events are less likely to be disproved, leading to the possible introduction of incorrect negative examples.

Contrary to the strict manner of comparing windows found in the original definition of the window comparison routine found in [7] – which stated that the position of the completed event under consideration ( $x_k$ ) in its trace ( $\tau_i$ ) must be equal to the position of the event with activity type equal to the activity type of the proposed negative event ( $y_k$ ) in the candidate disproving trace ( $\tau_j^\sim$ ), we compare the window  $\tau_i$  of the original “positive” trace with *each* possible window in the “candidate disproving” trace  $\tau_j^\sim$ , meaning each sequence of events before an event with activity type equal to the activity type of the current candidate negative event under consideration. An example can clarify this principle. Consider the positive trace  $\tau_i = \langle a, b, c, d, f \rangle$  and  $\tau_j^\sim = \langle a, b, c, b, c, e, f \rangle$  a candidate disproving trace under consideration. Assume that we are currently checking to see if candidate activity  $e$  could also occur instead of  $d$  in  $\tau_i$ . In cases where a strict window is used, with size equal to, say, 2, the candidate disproving trace  $\tau_j^\sim$  fails to disprove the negative event, since  $Position(d, \tau_i) \neq Position(e, \tau_j^\sim)$ . Instead of doing so, we now compare the window  $\langle b, c \rangle$  in  $\tau_i$  with each possible window in  $\tau_j^\sim$ . In this case, the window of size 2 before activity  $e$  in  $\tau_j^\sim$  is also equal to  $\langle b, c \rangle$ . This leads to a correct rejection of the candidate negative event. The full artificial negative event generation algorithm using this “dynamic” window is listed in Algorithm 11.

Note that, depending on the window size configured, cases might now exist where the size of the window in the original trace  $\tau_i$  is unequal to the size of a window in a candidate disproving trace  $\tau_j^\sim$ . In cases where the window in  $\tau_j^\sim$  is larger than the window in  $\tau_i$ , an effective window with size equal to the window used in  $\tau_i$  is used. When the window in  $\tau_j^\sim$  is smaller than the window in  $\tau_i$ , using the smallest window could potentially lead to the rejection of negative events, even when a high window size parameter was set. An example can help to illustrate this. Consider again the positive trace  $\tau_i = \langle a, b, c, d, f \rangle$  and  $\tau_j^\sim = \langle a, b, c, b, c, e, f \rangle$  the candidate disproving trace under consideration. Assume now we are currently checking to see if candidate activity  $b$  could also occur instead of  $d$  in  $\tau_i$ . Based on this information, two windows can be defined in  $\tau_j^\sim$  which can be used to check the validity of the candidate negative event at hand:  $\langle a \rangle$  and  $\langle a, b, c \rangle$ . For the latter, no problem exists, as this window is as long as the window  $\langle a, b, c \rangle$  in  $\tau_i$ . On the other hand, the other window ( $\langle a \rangle$ ) is smaller than the window in  $\tau_i$ , and thus the similarity check between these two windows might differ from the actual window size parameter which was configured by the user. Although this does not lead to the generation of incorrect negative events (or any additional events, in fact), we define a parameter *minimumWindowSize* to denote a minimum required length for a window to be used in the negative

**Algorithm 1.** Artificial event generation algorithm with a dynamic window

---

```

-- group similar sequences  $\sigma \in L$  into  $\tau \in M$ 
 $M := \text{GroupLog}(L)$ 
-- generate artificial events
 $N := \emptyset$ 
for each  $\tau_i \in M$  do
  for each  $x_k \in \tau_i$  do
     $k := \text{Position}(x_k, \tau_i)$ 
    for each  $b \in A \setminus \{\text{Activity}(x_k)\}$  do
      if  $\nexists \tau_j \sim \tau_i : \forall \tau_j \in M \wedge \tau_j \sim \tau_i \wedge$ 
         $\forall y_k \in \tau_j, j = \text{Position}(y_k, \tau_j), \text{Activity}(y_k) = b :$ 
           $\forall y_l \in \tau_j, j - l = \text{Position}(y_l, \tau_j), k - l = \text{Position}(x_l, \tau_i),$ 
             $0 < l \leq \text{windowSize} : \text{Activity}(y_l) = \text{Activity}(x_l)$ 
        then  $z_k := \text{Event}(x, \text{Case}(\tau_i), b, \text{completeRejected}, k)$ 
          -- with  $x$  a new event identifier
           $N := N \cup \{\text{NegativeEvent}(z_k, k, \tau_i)\}$ 
-- induce negative events in non grouped sequences  $\sigma \in L$  with injection
frequency  $\pi$ 
 $L_n := \text{InduceEvents}(N, L, \pi)$ 

```

---

event rejection procedure. Setting this parameter to -1 denotes that the window in  $\tau_j \sim \tau_i$  should be at least of the same size as the current window in  $\tau_i$ .

Using this dynamic window extension now allows us to drastically weaken the completeness assumption made by the artificial negative event generation process. Using a dynamic window with size 1 indeed assumes only that each binary sequence of two activities is somewhere present in the given event log, or can be generated from the given event log using parallel and loop variant calculation as described above. This weakens the completeness assumption equal to the one made by the formal alpha-miner learner [2].

Finally, remark the absence of a “forward window”; we only consider the history of completed events when investigating which activities could not be completed at a certain point in the process instance. The reason for this is straightforward: at the time of investigating a current state transition, the future of the process instance at hand is still unknown. Therefore, only historical facts can be considered in the negative event generation process.

**Dependency Based Negative Event Generation.** As a fourth and final extension, instead of using a window based trace comparison algorithm, we present an alternative way of generating artificial negative events, based on discovered structural information as given by the temporal frequent constraints and association rules. Both explicit dependency (locality), and long-distance dependency information can be applied towards the induction of negative events.

*Explicit Dependencies (Locality)*. It is possible to generate negative events based on locality information (i.e. explicit dependencies). As a rule of thumb, negative events with a certain activity type can be added before a given completed event in a process instance at position  $k$  when this activity type is not locally dependent on the activity type of the event completed at the previous position ( $k - 1$ )<sup>1</sup>:

$$\forall \tau \in M, 1 \leq k \leq |\tau|, k = \text{Position}(x_k, \tau), a \in A, a \neq \text{Activity}(x_k), k-1 = \text{Position}(x_{k-1}, \tau): \\ \sim \text{Local}(\text{Activity}(x_{k-1}), a) \Rightarrow \text{Event}(x, \text{Case}(x_k), a, \text{completeRejected}, k)$$

Note that, when  $k = 1$ ,  $x_{k-1}$  will be set to a dummy, non-existing event, so that constraints involving this dummy event (e.g.  $\text{Local}(x_0, x_1)$ ) always evaluates as being false. This constraint is omitted from the above and following definitions for reasons of clarity.

*Long-Distance Dependencies, Implicit Dependencies*. We define the following structural derivation rules to mine all dependencies between activity types (both implicit and explicit), similar to *Local* (explicit dependencies only):

$$\forall a, b \in A: (\text{Precedence}(a, b) \vee \text{Response}(a, b)) \wedge \sim \text{Parallel}(a, b) \Rightarrow \text{Dependence}(a, b) \\ \forall a, b \in A: (\text{Precedence}(a, b) \wedge \text{Response}(a, b)) \wedge \sim \text{Parallel}(a, b) \Rightarrow \text{StrongDependence}(a, b)$$

Based on these dependencies, the rule of thumb defined above can be expanded. Even negative events with a certain activity type that is locally dependent on the activity type of the event completed at the previous position ( $k - 1$ ) can be added before a given completed event (position  $k$ ), so long as not *all* activities on which the negative event under consideration is *strongly* dependent (*StrongDependence*) on were completed before, or so long as *no single* activity on which the negative event under consideration is dependent on (*Dependence*) has completed before:

$$\forall \tau \in M, 1 \leq k \leq |\tau|, k = \text{Position}(x_k, \tau), a \in A, a \neq \text{Activity}(x_k), k-1 = \text{Position}(x_{k-1}, \sigma), \\ v = \{\text{Activity}(i) \mid i \in \sigma, \text{Position}(i, \sigma) < k\}; \exists b \in A, b \notin v: \\ \text{StrongDependence}(b, a) \Rightarrow \text{Event}(x, \text{Case}(x_k), a, \text{completeRejected}, k)$$

and:

$$\forall \tau \in M, 1 \leq k \leq |\tau|, k = \text{Position}(x_k, \tau), a \in A, a \neq \text{Activity}(x_k), k-1 = \text{Position}(x_{k-1}, \sigma), \\ v = \{\text{Activity}(i) \mid i \in \sigma, \text{Position}(i, \sigma) < k\}; \nexists b \in A, b \in v: \\ \text{Dependence}(b, a) \Rightarrow \text{Event}(x, \text{Case}(x_k), a, \text{completeRejected}, k)$$

<sup>1</sup> Remark that we use negation-as-failure ( $\sim$ ) rather than normal logical negation ( $\neg$ ) to denote that the absence of a frequent temporal constraint is derived from the absence of sequences in the event log that portray this behavior. In this context, the reader may ignore the exact semantic differences between the two notations, as  $\sim p \Rightarrow \neg p$  under a closed-world assumption.



Finally, we can also use the *StrongDependence* construct to suggest an optimal minimum window size, i.e. a window size which is able to capture pairs of activity types between which an implicit (long-distance) dependency relation exists. To do so, we need to restrict *StrongDependence* a bit further to drop strong dependencies which do not correspond with an implicit dependency in the underlying process model. For example, a *StrongDependence* construct can always be found between starting and ending activities. However, a starting activity is always followed by the ending activity, so that this dependency is not an implicit one. Deriving a window size from this construct would lead to a useless suggestion, as this would give the same result as when using an unlimited window. Therefore, we restrict our search to unique strong dependencies (derived with the *UniqueStrongDependence* rule below) where activity types are strongly dependent on one other activity type only, which is a good indication for the presence of an implicit dependency.

$$\forall a,b \in A: \text{StrongDependence}(a,b) \wedge \nexists c \in A, c \neq a: \\ \text{StrongDependence}(c,b) \Rightarrow \text{UniqueStrongDependence}(a,b)$$

$$\text{MinimalWindowSize}(M) = \text{Max}_{a,b} (\text{Min}_{x,y,\tau} (\text{Position}(x,\tau) - \text{Position}(y,\tau)))$$

with:  $a,b \in A, \text{UniqueLongDistanceDependence}(a,b),$

$$\tau \in M, x \in \tau, y \in \tau, \text{Activity}(x) = a, \text{Activity}(y) = b, \text{Position}(x,\tau) < \text{Position}(y,\tau)$$

Note that, when sequences are present in the event log which contain multiple events corresponding to the same activity type (e.g. for process models containing loops),  $\text{Position}(x,\tau) - \text{Position}(y,\tau)$  is computed such that the resulting difference between the two events is minimal and greater than zero.

## 5 Experimental Results and Discussion

We have implemented our revised artificial negative event generation technique in ProM6 [24]. We test the improvements above with the “DriversLicenseLoop” log, an artificial process log which has been used before by de Medeiros et al. [3] to evaluate the GeneticMiner discovery algorithm. The drivers license process is interesting, since it contains parallelism, recurrence, duplicate tasks and implicit dependencies. To compare the different settings of the artificial event generation algorithm, a process log containing 350 process instances was used (87 distinct traces, 11 activity types).

Table 1 lists the various parameter configurations used to evaluate the artificial event generation technique. For each of the configurations, all generated negative events were introduced in the given event log. The “Window Generation” parameter denotes the use of window based artificial negative event generation, “Window Size” denotes the size of the window, “Dynamic Window” denotes the use of the dynamic window improvement with a minimum

required window size “Minimum Window Size”. “Structural Generation” defines if dependency based artificial negative event generation is performed, either on its own (“strucOnly”), or together with window based negative event generation (“strucNonDynamicWs-1” and following are obtained by merging a window based generated set of negative events with the set of negative events obtained from “strucOnly”). We use window sizes -1 (unlimited), 3 (suggested by *MinimalWindowSize(M)*) and 1 (most limited) to test the event generation procedure. When a dynamic window is used, we use both -1 (candidate disprove window must be as long as window in positive trace) and 1 (no effective minimum) as required minimum window values. “Original” configuration identifiers correspond with a parameter setting which could be obtained with the original version (i.e., no improvements) of the artificial event generation algorithm.

**Table 1.** Used parameter setting configurations for the artificial event generation tests

Parameter Configuration Identifier	Window Generation	Window Size	Dynamic Window	Minimum Window Size	Structural Generation
originalWs-1	yes	-1	no	–	no
originalWs3	yes	3	no	–	no
originalWs1	yes	1	no	–	no
dynamicWs-1MinWs-1	yes	-1	yes	-1	no
dynamicWs3MinWs-1	yes	3	yes	-1	no
dynamicWs1MinWs-1	yes	1	yes	-1	no
dynamicWs-1MinWs1	yes	-1	yes	1	no
dynamicWs3MinWs1	yes	3	yes	1	no
dynamicWs1MinWs1	yes	1	yes	1	no
strucOnly	no	–	–	–	yes
strucNonDynamicWs-1	yes	-1	no	–	yes
strucNonDynamicWs3	yes	3	no	–	yes
strucNonDynamicWs1	yes	1	no	–	yes
strucDynamicWs-1MinWs-1	yes	-1	yes	-1	yes
strucDynamicWs3MinWs-1	yes	3	yes	-1	yes
strucDynamicWs1MinWs-1	yes	1	yes	-1	yes
strucDynamicWs-1MinWs1	yes	-1	yes	1	yes
strucDynamicWs3MinWs1	yes	3	yes	1	yes
strucDynamicWs1MinWs1	yes	1	yes	1	yes

Table 2 gives the results for each of the above defined parameter setting configurations. We compare the results for the various parameter configurations with two given sets of negative events. A “naive generation method” constructs a set of negative events by injecting at each position in a trace a negative event for each activity type, except the activity type equal to the (completed) event at the current position. Note that even when this naive method is used, the number of incorrect negative events in respect to the total negative events is rather low. This is an indication towards the fact that the given event log gives a good coverage of all possible execution traces as allowed by the underlying process model. The “Fully Correct Log” was constructed based on the given Petri net used to simulate the drivers license process. Of course, in real life cases, such a reference model is

**Table 2.** Results of the DriversLicenseLoop experiment under various configurations

Parameter Configuration	Incorrect	Total Negative		
Identifier	Negative Events	Events	Correctness	Completeness
Fully Correct Log	0	44866	100%	100%
Naive Generation Method	2484	47350	0%	100%
originalWs-1	642	45508	74,2%	100%
originalWs3	621	45487	75,0%	100%
originalWs1	621	44866	75,0%	98,6%
dynamicWs-1MinWs-1	642	45508	74,2%	100%
<b>dynamicWs3MinWs-1</b>	0	44866	<b>100%</b>	<b>100%</b>
dynamicWs1MinWs-1	0	42382	100%	94,5%
dynamicWs-1MinWs1	642	45508	74,2%	100%
<b>dynamicWs3MinWs1</b>	0	44866	<b>100%</b>	<b>100%</b>
dynamicWs1MinWs1	0	42382	100%	94,5%
strucOnly	0	40469	100%	90,2%
strucNonDynamicWs-1	642	45508	74,2%	100%
strucNonDynamicWs3	621	45487	75,0%	100%
strucNonDynamicWs1	621	45349	75,0%	99,7%
strucDynamicWs-1MinWs-1	642	45508	75,2%	100%
<b>strucDynamicWs3MinWs-1</b>	0	44866	<b>100%</b>	<b>100%</b>
strucDynamicWs1MinWs-1	0	43969	100%	98,0%
strucDynamicWs-1MinWs1	642	45508	74,2%	100%
<b>strucDynamicWs3MinWs1</b>	0	44866	<b>100%</b>	<b>100%</b>
strucDynamicWs1MinWs1	0	43969	100%	98,0%

unavailable, preventing the construction of a fully correct set of negative events by which the artificial induction results can be evaluated. For each parameter configuration, we calculate the correctness and completeness ratio. Correctness is defined as one minus the ratio of incorrect negative events to the number of incorrect negative events generated by the naive method. Completeness is defined as the ratio of correct negative events over the full number of possible, correct negative events, as given by the fully correct log.

The following conclusions can be derived from the results. First, the inherent trade-off between correctness and completeness becomes apparent here, as most configurations show an inverse relation between the two requirements. Second, we note that no single window size configuration is able to generate a set of negative events which is both correct and complete when using the original version of the artificial event generation algorithm. Next, using a strict window size (1) in combination with the dynamic window improvement leads to a set of negative events which is fully correct, albeit not complete. Constructing a set of negative events which is both complete and correct is possible if the window size is increased to 3 (suggested by investigating the structure of implicit dependencies – denoted in bold case in Table 2), or by using non-window dependency based generation, which also leads to an acceptable completeness value (98%). Moreover, using dependency-based generation ensures the addition of “non-trivial” negative events, derived from implicit dependencies, which proves especially helpful in a later phase when the set of negative events is used for evaluation or discovery tasks. The results also deal with another concern: even although we have defined a large number of parameters, two straightforward, well performing defaults can be suggested: either apply window based generation with

a dynamic window of size 1 in conjunction with dependency based event generation, or only apply window based generation with a window size equal to the suggested window size.

## 6 Conclusions and Future Work

In this paper, we propose an improved artificial negative event generation method, building upon [7] in order to derive sets of negative events which are both correct and complete. The construction of event logs with artificial negative events can be expected to be valuable in multiple settings: first, supervised learners can now be deployed in order to perform a process discovery task. Second, artificial induction of negative events can be applied towards evaluation purposes as well, where an event log, supplemented with negative events, is used to assess fitness and precision of process discovery algorithms. Finally, since negative events capture which actions in a process could not occur, compliance and conformance related analysis tasks present themselves as a natural area of application for negative events, although it should be noted that, in order to perform this last set of tasks, the artificial negative event generation technique as described here should be extended to take state transitions other than completed events into account, together with event originator (agent) and case data in order to fully utilize all available information. In future work, we aim at validating our novel approach, by extending the set of artificial logs included in the experiment and by examining the different parameter configurations. In this way, we aim at making event logs with artificially generated negative events widely available.

**Acknowledgements.** We would like to thank the KU Leuven research council for financial support under grand OT/10/010 and the Flemish Research Council for financial support under Odysseus grant B.0915.09.

## References

- [1] van der Aalst, W., Reijers, H., Weijters, A., van Dongen, B., Alves de Medeiros, A., Song, M., Verbeek, H.: Business process mining: An industrial application. *Information Systems* 32(5), 713–732 (2007)
- [2] van der Aalst, W., Weijters, A.J.M.M., Maruster, L.: Workflow mining: discovering process models from event logs. *IEEE Transactions on Knowledge and Data Engineering* 16(9), 1128–1142 (2004)
- [3] de Medeiros, A., Weijters, A., van der Aalst, W.: Genetic process mining: an experimental evaluation. *Data Mining and Knowledge Discovery* 14(2), 245–304 (2007)
- [4] de Medeiros, A., van Dongen, B., van der Aalst, W.: Process mining: Extending the alpha-algorithm to Mine Short Loops (2004)
- [5] Weijters, A., van der Aalst, W., de Medeiros, A.: Process mining with the heuristics miner-algorithm (2006)
- [6] Wen, L., van der Aalst, W., Wang, J., Sun, J.: Mining process models with non-free-choice constructs. *Data Mining and Knowledge Discovery* 15(2), 145–180 (2007)

- [7] Goedertier, S., Martens, D., Vanthienen, J., Baesens, B.: Robust Process Discovery with Artificial Negative Events. *Journal of Machine Learning Research* 10, 1305–1340 (2009)
- [8] Blockeel, H., De Raedt, L.: Top-down induction of first-order logical decision trees. *Artificial Intelligence* 101(1-2), 285–297 (1998)
- [9] Chesani, F., Lamma, E., Mello, P., Montali, M., Riguzzi, F., Storari, S.: Exploiting Inductive Logic Programming Techniques for Declarative Process Mining. In: Jensen, K., van der Aalst, W.M.P. (eds.) *Transactions on Petri Nets and Other Models of Concurrency II*. LNCS, vol. 5460, pp. 278–295. Springer, Heidelberg (2009)
- [10] Muggleton, S.: Inductive logic programming. In: *Proceedings of the 1st International Conference on Algorithmic Learning Theory*, pp. 42–62 (1990)
- [11] Lamma, E., Mello, P., Riguzzi, F., Storari, S.: Applying Inductive Logic Programming to Process Mining. In: Blockeel, H., Ramon, J., Shavlik, J., Tadepalli, P. (eds.) *ILP 2007*. LNCS (LNAI), vol. 4894, pp. 132–146. Springer, Heidelberg (2008)
- [12] Dzeroski, S., Lavrac, N.: *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood, New York (1994)
- [13] De Weerd, J., De Backer, M., Vanthienen, J., Baesens, B.: A robust f-measure for evaluating discovered process models. In: *IEEE Symposium Series in Computational Intelligence* (2011)
- [14] Cook, J., Wolf, A.: Discovering models of software processes from event-based data. *ACM Transactions on Software Engineering and Methodology* 7(3), 215–249 (1998)
- [15] Agrawal, R., Gunopulos, D., Leymann, F.: Mining Process Models from Workflow Logs. In: Schek, H.-J., Saltor, F., Ramos, I., Alonso, G. (eds.) *EDBT 1998*. LNCS, vol. 1377, pp. 467–483. Springer, Heidelberg (1998)
- [16] Lyytinen, K., Mathiassen, L., Ropponen, J., Datta, A.: Automating the discovery of as-is business process models: Probabilistic and algorithmic approaches. *Information Systems Research* 9(3), 275–301 (1998)
- [17] Maruster, L., Weijters, A., van der Aalst, W., van den Bosch, A.: A Rule-Based Approach for Process Discovery: Dealing with Noise and Imbalance in Process Logs. *Data Mining and Knowledge Discovery* 13(1), 67–87 (2006)
- [18] Ferreira, H., Ferreira, D.: An integrated life cycle for workflow management based on learning and planning. *International Journal of Cooperative Information Systems* 15(4), 485–505 (2006)
- [19] Lamma, E., Mello, P., Montali, M., Riguzzi, F., Storari, S.: Inducing Declarative Logic-Based Models from Labeled Traces. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) *BPM 2007*. LNCS, vol. 4714, pp. 344–359. Springer, Heidelberg (2007)
- [20] Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules. *VLDB* 12(1), 487–499 (1994)
- [21] Huang, K., Chang, C.: Efficient mining of frequent episodes from complex sequences. *Information Systems* 33(1), 96–114 (2008)
- [22] Mannila, H., Toivonen, H., Inkeri Verkamo, A.: Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery* 1(3), 259–289 (1997)
- [23] Maggi, F., Mooij, A., van der Aalst, W.: User-guided discovery of declarative process models. In: *IEEE Symposium on Computational Intelligence and Data Mining* (2011)
- [24] van der Aalst, W., van Dongen, B., Rozinat, A., Günther, C., Verbeek, E.: Prom: The process mining toolkit. In: de Medeiros, A.K.A., Weber, B. (eds.) *BPM (Demos)*. *CEUR Workshop Proceedings*, vol. 489, CEUR-WS.org (2009)

# Efficient Discovery of Understandable Declarative Process Models from Event Logs

Fabrizio M. Maggi\*, R.P. Jagadeesh Chandra Bose, and Wil M.P. van der Aalst

Eindhoven University of Technology, The Netherlands

{f.m.maggi, j.c.b.rantham.prabhakara, w.m.p.v.d.aalst}@tue.nl

**Abstract.** Process mining techniques often reveal that real-life processes are more variable than anticipated. Although declarative process models are more suitable for less structured processes, most discovery techniques generate conventional procedural models. In this paper, we focus on discovering *Declare* models based on event logs. A *Declare* model is composed of temporal constraints. Despite the suitability of declarative process models for less structured processes, their discovery is far from trivial. Even for smaller processes there are many potential constraints. Moreover, there may be many constraints that are trivially true and that do not characterize the process well. Naively checking all possible constraints is computationally intractable and may lead to models with an excessive number of constraints. Therefore, we have developed an Apriori algorithm to reduce the search space. Moreover, we use new metrics to prune the model. As a result, we can quickly generate understandable *Declare* models for real-life event logs.

**Keywords:** process mining, business process management, declarative process models.

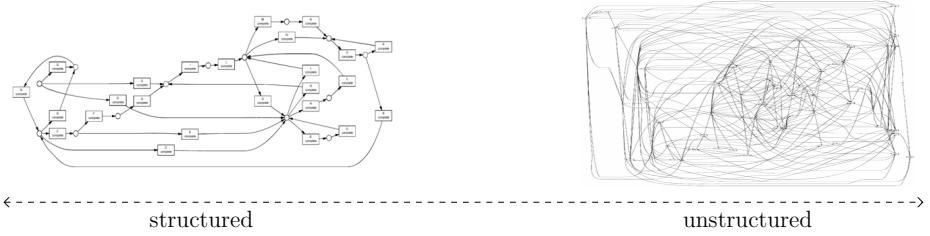
## 1 Introduction

The increasing availability of event data recorded by contemporary information systems makes *process mining* a valuable instrument to improve and support business processes [2]. Starting point for process mining is an *event log*. Each event in a log refers to an *activity* (i.e., a well-defined step in some process) and is related to a particular *case* (i.e., a *process instance*). The events belonging to a case are *ordered* and can be seen as one “run” of the process (often referred to as a *trace* of events). Event logs may store additional information about events such as the *resource* (i.e., person or device) executing or initiating the activity, the *timestamp* of the event, or *data elements* recorded with the event.

Typically, three types of process mining can be distinguished [2]: (a) *process discovery* (learning a model from example traces in an event log), (b) *conformance checking* (comparing the observed behavior in the event log with the

---

\* This research has been carried out as a part of the Poseidon project at Thales under the responsibilities of the Embedded Systems Institute (ESI). The project is partially supported by the Dutch Ministry of Economic Affairs under the BSIK program.



**Fig. 1.** Procedural models for structured and unstructured processes

modeled behavior), and (c) *model enhancement* (extending models based on additional information in the event logs, e.g., to highlight bottlenecks). In this paper, we focus on process discovery which is generally considered as the most challenging process mining task.

Fig. 1 shows two example models discovered based on two different event logs. The Petri net on the left shows a relatively structured process that is easy to understand. The Spaghetti-like model on the right is less structured and much more difficult to comprehend. In practice, process models are often less structured than anticipated. Therefore, one could argue that procedural models such as the ones depicted in Fig. 1 are less suitable. Nevertheless, almost all process discovery techniques [15,6,10,12,14,18] aim to discover procedural model expressed in terms of BPMN, UML activity diagrams, Petri nets, EPCs, and the like. In this paper, we use a different approach and aim to *discover declarative models* from event logs.



**Fig. 2.** Declare *response* constraint:  $\square(a \rightarrow \diamond b)$

Fig. 2 shows a *Declare* model [4,13,20] consisting of only one constraint. The arrow connecting activity  $a$  to  $b$  models a so-called *response* constraint, i.e., activity  $a$  is always followed by  $b$ . This response constraint is satisfied for traces such as  $\langle a, a, b, c \rangle$ ,  $\langle b, b, c, d \rangle$ , and  $\langle a, b, c, a, b \rangle$ , but not for  $\langle a, b, a, c \rangle$  because the second  $a$  is not followed by  $b$ . The semantics of Declare constraints are rooted in Linear Temporal Logic (LTL), e.g., the response constraint can be formalized as  $\square(a \rightarrow \diamond b)$  and checked or enforced automatically.

A Declare model consists of a set of constraints which, in turn, are based on *templates*. A template defines a particular type of constraint (like “response”). Templates have formal semantics specified through LTL formulas and are equipped with a user-friendly graphical front-end that makes the language easy to understand also for users that are not familiar with LTL [19,23]. Templates are parameterized, e.g., the response constraint in Fig. 2 is instantiated for activities  $a$  and  $b$ . Hence, in a process model with  $n$  events there are  $n^2$  potential response constraints.

In [17], the authors present a technique to automatically infer Declare constraints. This technique exhaustively generates all possible constraints and then checks them on the event log. This approach suffers from two important drawbacks:

- First of all, such an exhaustive approach is intractable for processes with dozens of activities. For example, in a process with 30 activities there are already 900 possible response constraints. Some of the other templates have four parameters, resulting in  $30^4 = 810,000$  potential constraints for a single template. Since there are dozens of templates in the standard Declare language, this implies that the log needs to be traversed millions of times to check all potential constraints. As event logs may contain thousands or even millions of events, this is infeasible in practice.
- Second, of the millions of potential constraints, many may be trivially true. For example, the response constraint in Fig. 2 holds for any event log that does not contain events relating to activity  $a$ . Moreover, one constraint may dominate another constraint. If the stronger constraint holds (e.g.,  $\square(a \rightarrow \diamond b)$ ), then automatically the weaker constraint (e.g.,  $\diamond a \rightarrow \diamond b$ ) also holds. Showing all constraints that hold typically results in unreadable models.

This paper addresses these two problems using a two-phase approach. In the first phase, we generate the list of candidate constraints by using an Apriori algorithm. This algorithm is inspired by the seminal Apriori algorithm developed by Agrawal and Srikant for mining association rules [7]. The Apriori algorithm uses the monotonicity property that all subsets of a frequent item-set are also frequent. In the context of this paper, this means that sets of activities can only be frequent if all of their subsets are frequent. This observation can be used to dramatically reduce the number of interesting candidate constraints. In the second phase, we further prune the list of candidate constraints by considering only the ones that are relevant (based on the event log) according to (the combination of) simple metrics, such as *Confidence* and *Support*, and more sophisticated metrics, such as *Interest Factor* (IF) and *Conditional-Probability Increment Ratio* (CPIR), as explained in Section 4. Moreover, discovered constraints with high CPIR values are emphasized like highways on a roadmap whereas constraints with low CPIR values are greyed out. This further improves the readability of discovered Declare models.

The paper is structured as follows. Section 2 introduces the Declare formalism using a running example. Section 3 describes how an Apriori algorithm can be applied to generate a list of candidate constraints. Section 4 explains how metrics proposed in the literature on association rule mining can be used to evaluate the relevance of a Declare constraint. Section 5 presents experimental results comparing our new discovery algorithm with the naive approach presented in [17]. Section 6 provides an illustrative case study. Section 7 concludes the paper.



## 2 Declare

In this paper, we present an approach to efficiently discover understandable Declare models. Therefore, we first introduce the Declare language [4,13,20] which is grounded in Linear Temporal Logic (LTL).

LTL can be used to specify constraints on the ordering of activities (see also [11]). For instance, a constraint like “whenever activity  $a$  is executed, eventually activity  $b$  is executed” can be formally represented using LTL and, in particular, it can be written as  $\Box(a \rightarrow \Diamond b)$ . In a formula like this, it is possible to find traditional logical operators (e.g., implication  $\rightarrow$ ), but also temporal operators characteristic of LTL (e.g., always  $\Box$ , and eventually  $\Diamond$ ). In general, using the LTL language it is possible to express constraints relating activities (atoms) through logical operators or temporal operators. The logical operators are: implication ( $\rightarrow$ ), conjunction ( $\wedge$ ), disjunction ( $\vee$ ), and negation ( $\neg$ ). The main temporal operators are: *always* ( $\Box p$ , in every future state  $p$  holds), *eventually* ( $\Diamond p$ , in some future state  $p$  holds), *next* ( $\bigcirc p$ , in the next state  $p$  holds), and *until* ( $p \sqcup q$ ,  $p$  holds until  $q$  holds).

LTL constraints are not very readable for non-experts. Declare [4,13,20] provides an intuitive graphical front-end together with a formal LTL back-end. In Fig. 2 we already showed the graphical Declare representation for the *response* constraint. There are dozens of different Declare constraints possible. Each type of constraint is described using a parameterized template. Besides the response constraint, we have constraints such as *responded existence* (formally:  $\Diamond a \rightarrow \Diamond b$ ) and *precedence* (formally:  $(\neg b \sqcup a) \vee \Box(\neg b)$ ).

### 2.1 Running Example

Fig. 3 shows a simple Declare model with some example constraints for an insurance claim process. The model includes eight activities (depicted as rectangles, e.g., *Contact Hospital*) and five constraints (shown as connectors between the activities, e.g., *co-existence*).

The *responded existence* constraint specifies that if *High Medical History* is executed also *High Insurance Check* is executed in the same process instance. The *precedence* constraint indicates that, if *Receive Questionnaire Response* is executed, *Send Questionnaire* must be executed before. In contrast, the *response*

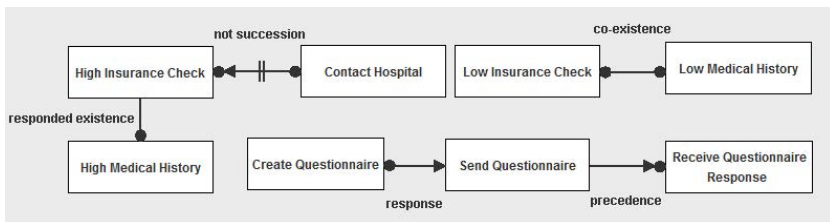


Fig. 3. Running example: Declare model consisting of five constraints

constraint indicates that if *Create Questionnaire* is executed this is eventually followed by *Send Questionnaire*. The *not succession* constraint means that *Contact Hospital* cannot be followed by *High Insurance Check*. Finally, the *co-existence* constraint indicates that if *Low Insurance Check* and *Low Medical History* occur in a process instance, they always coexist. We refer the reader to [4,13,20] for more details about the Declare language (graphical notation and LTL semantics).

## 2.2 Discovering Relevant Declare Constraints Using Vacuity Detection

As shown in [3], LTL constraints can be checked for a particular trace and therefore also for an entire event log. For example, one may find that responded existence constraint between *High Medical History* and *High Insurance Check* holds for 955 of 988 insurance claims. Hence, Declare models can be discovered by simply checking all possible constraints as shown in [17]. First, all possible constraints need to be constructed. Since there is a finite number of constraint templates and in a given setting there is also a finite set of activities, this is always possible. Then, the set of potential constraints can be pruned on the basis of simple metrics such as the percentage of process instances where the constraint holds, e.g., keep all constraints satisfied in at least 90% of cases.

Such an approach will result in the discovery of many constraints that are trivially valid. Consider for example a process instance  $\langle a, a, b, a, b, a \rangle$ . The constraint  $\Box(c \rightarrow \Diamond d)$  (“whenever activity  $c$  is executed, eventually activity  $d$  is executed”) holds. This constraint holds trivially because  $c$  never happens. Using the terminology introduced in [8,15], we say that the constraint is *vacuously satisfied*. In general, a formula  $\varphi$  is *vacuously satisfied* in a path  $\pi$ , if  $\pi$  satisfies  $\varphi$  and there is some sub-formula of  $\varphi$  that does not affect the truth value of  $\varphi$  in  $\pi$  [8]. In our example, the first term of the implication  $\Box(c \rightarrow \Diamond d)$  is always false. Therefore, sub-formulas  $\Diamond d$  and  $d$  do not affect the truth value of  $\Box(c \rightarrow \Diamond d)$  in  $\langle a, a, b, a, b, a \rangle$ .

To address this issue, in [17], the authors use techniques for *LTL vacuity detection* [8,15] to discriminate between instances where a constraint is generically non-violated and instances where the constraint is non-vacuously satisfied. Only process instances where a constraint is non-vacuously satisfied are considered *interesting witnesses* for that constraint. Roughly speaking, to ensure that a process instance is an interesting witness for a constraint, it is necessary to check the validity of the constraint in the process instance with some extra conditions. The authors in [15] introduce an algorithm to compute these extra conditions. For example, for the constraint  $\Box(c \rightarrow \Diamond d)$  the condition is  $\Diamond c$ . This means that the constraint is non-vacuously satisfied in all the process instances where “whenever activity  $c$  is executed, eventually activity  $d$  is executed” and “eventually activity  $c$  is executed”. The percentage of interesting witnesses for a constraint is a much better selection mechanism than the percentage of instances for which the constraint holds.

In [21], the authors extend the list of vacuity detection conditions to ensure that if a process instance is an interesting witness for a Declare constraint, no stronger constraint holds in that process instance. Table 1 specifies the list of vacuity detection conditions for some types of Declare constraints. For instance, a response constraint is non-vacuously satisfied in all the process instances where it is satisfied ( $\Box(a \rightarrow \Diamond b)$ ), the vacuity detection condition derived from [15] is valid ( $\Diamond a$ ) and, in addition, constraints that are stronger than response (i.e., succession and alternate response) do not hold ( $\neg((\neg b \Box a) \vee \Box(\neg b)) \wedge \neg(\Box(a \rightarrow \bigcirc(\neg a \Box b)))$ ). In this paper, we refer to this extended notion of vacuity detection.

For completeness we also mention the approach described in [16]. This approach also learns Declare models, but requires negative examples. This implies that everything that did not happen is assumed not to be possible. We consider this an unrealistic assumption as logs only contain example behavior.

### 3 Apriori Algorithm for Declare Discovery

Vacuity detection can be used to prune set of constraints, but this can only be done after generating a set of candidate constraints. As discussed in the introduction, even for smaller processes, there can be millions of potential constraints. Therefore, we adopt ideas from the well-known Apriori algorithm [7] for discovering association rules. Using an Apriori-like approach we can efficiently discover frequent sets of correlated activities in an event log.

Let  $\Sigma$  be the set of potential activities. Let  $\mathbf{t} \in \Sigma^*$  be a trace over  $\Sigma$ , i.e., a sequence of activities executed for some process instance. An event log  $\mathcal{L}$  is a multi-set over  $\Sigma^*$ , i.e., a trace can appear multiple times in an event log.

The *support* of a set of activities is a measure that assesses the relevance of this set in an event log.

**Definition 1 (Support).** *The support of an activity set  $A \subseteq \Sigma$  in an event log  $\mathcal{L} = [\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_n]$  is the fraction of process instances in  $\mathcal{L}$  that contain all of the activities in  $A$ , i.e.,*

$$\text{supp}(A) = \frac{|\mathcal{L}_A|}{|\mathcal{L}|}, \text{ where } \mathcal{L}_A = [\mathbf{t} \in \mathcal{L} \mid \forall_{x \in A} x \in \mathbf{t}]$$

An activity set is considered to be *frequent* if its support is above a given threshold  $\text{supp}_{\min}$ . Let  $\mathcal{A}_k$  denote the set of all frequent activity sets of size  $k \in \mathbb{N}$  and let  $\mathcal{C}_k$  denote the set of all *candidate activity sets* of size  $k$  that may potentially be frequent. The Apriori algorithm uncovers all frequent activity sets in an event log. The algorithm starts by considering activity sets of size 1 and progresses iteratively by considering activity sets of increasing sizes in each iteration and is based on the property *that any subset of a frequent activity set must be frequent*. The set of candidate activity sets of size  $k + 1$ ,  $\mathcal{C}_{k+1}$ , is generated by joining relevant frequent activity sets from  $\mathcal{A}_k$ . This set can be pruned efficiently using the property that a relevant candidate activity set of size  $k + 1$  cannot have an infrequent subset. The activity sets in  $\mathcal{C}_{k+1}$  that have a support above a given

**Table 1.** Vacuity detection conditions for some Declare constraints

<i>template</i>	<i>LTL semantics</i>	<i>vacuity detection conditions</i>
responded existence	$\diamond a \rightarrow \diamond b$	$\diamond a \wedge \neg(\Box(a \rightarrow \diamond b)) \wedge \neg(\diamond a \leftrightarrow \diamond b)$
co-existence	$\diamond a \leftrightarrow \diamond b$	$\diamond a \wedge \diamond b \wedge \neg(\Box(a \rightarrow \diamond b) \wedge (\neg b \sqcup a) \vee \Box(\neg b))$
response	$\Box(a \rightarrow \diamond b)$	$\diamond a \wedge \neg((\neg b \sqcup a) \vee \Box(\neg b)) \wedge \neg(\Box(a \rightarrow \Box(\neg a \sqcup b)))$
precedence	$(\neg b \sqcup a) \vee \Box(\neg b)$	$\diamond b \wedge \neg a \wedge \neg(\Box(a \rightarrow \diamond b)) \wedge \neg(((\neg b \sqcup a) \vee \Box(\neg b)) \wedge \Box(b \rightarrow \Box((\neg b \sqcup a) \vee \Box(\neg b))))$
not succession	$\Box(a \rightarrow \neg(\diamond b))$	$\diamond a \wedge \diamond b$

threshold  $supp_{\min}$  constitute the frequent activity sets of size  $k + 1$  ( $\mathcal{A}_{k+1}$ ) used in the next iteration.

We explain the Apriori algorithm with an example. Consider an event log  $\mathcal{L} = [\langle e, a, b, a, a, c, e \rangle, \langle e, a, a, b, c, e \rangle, \langle e, a, a, d, d, e \rangle, \langle b, b, c, c \rangle, \langle e, a, a, c, d, e \rangle]$  defined over the set of activities  $\Sigma = \{a, b, c, d, e\}$ . Let us try to find frequent activity sets whose support is above 50%. The Apriori algorithm starts by first considering activity sets of size 1, i.e., the individual activities. The candidate sets in  $\mathcal{C}_1$  correspond to the singletons of the elements of  $\Sigma$ . Fig. 4(a) depicts the candidate activity sets and their support. Among the candidate sets, activity  $d$  has a support of only 40% in the event log, which is below the specified threshold. Therefore, the frequent activity sets correspond to the singletons of the elements of  $\Sigma \setminus \{d\}$ . In the next iteration, the Apriori algorithm considers candidate activity sets of size 2,  $\mathcal{C}_2$ . Since the support of  $d$  is less than the specified threshold, all activity sets that involve  $d$  are bound to have their support less than the threshold. The Apriori algorithm elegantly captures this by deriving candidates at iteration  $k + 1$ ,  $\mathcal{C}_{k+1}$ , from frequent activity sets of iteration  $k$ . Fig. 4(b) depicts such candidate activity sets of size 2 along with their support values. Only 4 activity sets among  $\mathcal{C}_2$  satisfy the minimum support criteria and hence are considered frequent (see  $\mathcal{A}_2$  in Fig. 4(b)). Proceeding further, we get only one frequent activity set of size 3 as depicted in Fig. 4(c). The algorithm terminates after this step as no further candidates can be generated. The frequent activity sets in  $\mathcal{L}$  are  $\mathcal{A}_1 \cup \mathcal{A}_2 \cup \mathcal{A}_3$ .

The Apriori algorithm returns frequent activity sets, which indicate that the activities involved in an activity set are correlated. However, it doesn't specify the type of correlation. Declare templates capture different relationships between activities. For instance, for any frequent activity set  $\{a, b\}$ , one can generate constraints such as the response  $\Box(a \rightarrow \diamond b)$ .

In general, to discover constraints deriving from a Declare template with  $k$  parameters, we have to generate frequent activity sets of size  $k$ . Afterwards, we generate the list of candidate constraints. To do that, we instantiate the considered template by specifying as parameters all the possible permutations of each frequent set. For instance, for the frequent activity set  $\{a, b\}$ , we generate the response constraints  $\Box(a \rightarrow \diamond b)$  and  $\Box(b \rightarrow \diamond a)$ .

Limiting ourselves to frequent activity sets drastically reduces the number of candidate constraints to be checked. For instance, if we take the example in Fig. 4(b), to generate the candidate constraints deriving from a template with

candidate activity sets		frequent activity sets		candidate activity sets		frequent activity sets		candidate activity sets		frequent activity sets	
$\mathcal{C}_1$	supp	$\mathcal{A}_1$	supp	$\mathcal{C}_2$	supp	$\mathcal{A}_2$	supp	$\mathcal{C}_3$	supp	$\mathcal{A}_3$	supp
a	80	a	80	{a, b}	40	{a, c}	60	{a, b, c}	40	{a, c, e}	60
b	60	b	60	{a, c}	60	{a, e}	80	{a, b, e}	40		
c	80	c	80	{a, e}	80	{b, c}	60	{a, c, e}	60		
d	40	e	80	{b, c}	60	{c, e}	60	{b, c, e}	40		
e	80			{b, e}	40						
				{c, e}	60						

(a)
(b)
(c)

**Fig. 4.** Discovering frequent activity sets using the Apriori algorithm in the event log  $\mathcal{L}$ . The support values are expressed in %.

2 parameters we consider all the permutations of each frequent activity set in  $\mathcal{A}_2$  and we generate 12 candidate constraint (we also include pairs  $(a, a)$ ,  $(b, b)$ ,  $(c, c)$ ,  $(e, e)$  by considering repetitions of elements of frequent activity sets of size 1). In contrast, with the naive approach we need to consider all the dispositions of length 2 of 5 activities  $(a, b, c, d, e)$ , i.e., 25 ( $5^2$ ) candidate constraints. In general, to discover constraints deriving from a Declare template with  $k$  parameters from a log with  $n$  activities, the state space exploration using the naive approach always leads to  $n^k$  candidate constraints. In contrast, applying our Apriori algorithm and considering only the frequent activity sets, the number of candidate constraints depends on the support value we specify for the Apriori algorithm. This number is often significantly lower than  $n^k$  because we ignore the item sets with low support.

One could also look at negative events (non-occurrence) within the Apriori setup. Such information might be useful for inferring, for instance, events that act as mutually exclusive, e.g., if  $a$  occurs then  $b$  does not occur. To facilitate this, we can also consider the negative events  $\neg a$ , for all  $a \in \Sigma$ . Fig. 5 depicts the discovery of frequent items sets considering non-occurrence of events using the

candidate activity sets		frequent activity sets		candidate activity sets		frequent activity sets		candidate activity sets		frequent activity sets	
$\mathcal{C}_1$	supp	$\mathcal{A}_1$	supp	$\mathcal{C}_2$	supp	$\mathcal{A}_2$	supp	$\mathcal{C}_3$	supp	$\mathcal{A}_3$	supp
a	80	a	80	{a, b}	40	{a, c}	60	{a, b, c}	40	{a, c, e}	60
b	60	b	60	{a, c}	60	{a, e}	80	{a, b, e}	40		
c	80	c	80	{a, e}	80	{b, c}	60	{a, c, e}	60		
d	40	e	80	{a, -d}	40	{b, -d}	60	{a, b, -d}	40		
e	80	-d	60	{b, c}	60	{c, e}	60	{a, c, -d}	40		
-a	20			{b, e}	40	{c, -d}	60	{a, e, -d}	40		
-b	40			{b, -d}	60	{e, -d}	60	{b, c, e}	40		
-c	20			{c, e}	60			{b, c, -d}	60		
-d	60			{c, -d}	60			{b, e, -d}	40		
-e	20			{e, -d}	60			{c, e, -d}	40		

(a)
(b)
(c)

**Fig. 5.** Discovering frequent activity sets considering negative (non-occurrence) events using the Apriori algorithm in the event log  $\mathcal{L}$ . The support values are expressed in %.

**Table 2.** Association rule formulation for some Declare constraints

<i>template</i>	<i>LTL semantics</i>	<i>rule</i>	<i>antecedent</i>	<i>consequent</i>
responded existence	$\diamond a \rightarrow \diamond b$	If $a$ occurs then $b$ occurs	$a$	$b$
co-existence	$\diamond a \leftrightarrow \diamond b$	If $a$ occurs then $b$ occurs $\wedge$ If $b$ occurs then $a$ occurs	$(a \vee b)$	$(a \vee b)$
response	$\square(a \rightarrow \diamond b)$	If $a$ occurs then $b$ follows	$a$	$b$
precedence	$(\neg b \sqcup a) \vee \square(\neg b)$	If $b$ occurs then $a$ precedes	$b$	$a$
not succession	$\square(a \rightarrow \neg(\diamond b))$	If $a$ occurs then $b$ does not follow	$a$	$b$

apriori algorithm on the event log  $\mathcal{L}$ . Note that we now have additional frequent activity sets such as  $\{b, c, \neg d\}$  signifying that if  $b$  and  $c$  occurs then  $d$  does not occur.

## 4 Post-processing

The set of frequent activity sets helps in reducing the preliminary set of constraints that one uncovers. In particular, the candidate constraints generated from frequent sets all involve activities that occur frequently in the log. However, this does not mean that these constraints are also frequently (non-vacuously) satisfied in the log. Let us consider, for example,  $\{a, e\}$ , a frequent activity set for the event log  $\mathcal{L}$  mentioned earlier. One can define various Declare constraints involving these two activities, e.g., the response constraints  $\square(a \rightarrow \diamond e)$  and  $\square(e \rightarrow \diamond a)$ . However, only the former constraint holds for all cases in  $\mathcal{L}$  whereas the latter constraint is only satisfied in one of the five cases. Even more, according to the vacuity detection conditions illustrated in Table 1,  $\square(a \rightarrow \diamond e)$  is non-vacuously satisfied in four cases, whereas  $\square(e \rightarrow \diamond a)$  is never non-vacuously satisfied. Obviously, there is a need to further prune constraints that are less relevant, i.e., non-vacuously satisfied in a lower percentage of process instances than others. We can consider a Declare constraint as a rule or as a conjunction/disjunction of rules, e.g.,  $\square(a \rightarrow \diamond e)$  can be thought of as the rule “If  $a$  is executed, then  $e$  follows”. A rule is comprised of two components, the *antecedent* part and the *consequent* part. Table 2 depicts the interpretation of Declare constraints as association rules. We can adopt various metrics proposed in the association rule mining literature to evaluate the relevance of a rule and thereby the Declare constraints. We use four metrics, i.e., *support*, *confidence*, *interest factor*, and *CPJR*. The latter three metrics are defined on the primitive measure of *support* of the antecedent, consequent, and the rule.

**Definition 2 (Support (of a Declare constraint)).** *The support of a Declare constraint in an event log  $\mathcal{L}$  is defined as the fraction of process instances in which the constraint is non-vacuously satisfied, i.e., the percentage of interesting witnesses for that constraint in the log.*

**Definition 3 (Confidence).** *The confidence of a Declare constraint expressed as an association rule in an event log  $\mathcal{L}$  is the ratio between the support of the rule and the support of the antecedent*

$$\text{conf}(\text{rule}) = \frac{\text{supp}(\text{rule})}{\text{supp}(\text{antecedent})}$$

For the above event log  $\mathcal{L}$ , the support of the constraint  $\square(a \rightarrow \diamond e)$  is 0.8 and the confidence of the constraint is 1. One can use the support and confidence metrics to prune constraints, e.g., consider only those constraints whose support is above a minimum support threshold and/or whose confidence is above a minimum confidence threshold. Confidence measures might be misleading in scenarios where the support of either the antecedent or the consequent is 1. On a general note, frequent activity sets involving an activity whose support is 1 do not reflect the real correlation between the activities. Brin et al. [9] have proposed a measure called *interest factor* to deal with such scenarios. A stronger dependency between the antecedent and the consequent is associated with a value of this measure that is further from 1.

**Definition 4 (Interest Factor).** *The interest factor of a Declare constraint expressed as an association rule in an event log  $\mathcal{L}$  is the ratio between the support of the rule and the product of the support of the antecedent and the consequent.*

$$\text{InterestFactor}(\text{rule}) = \frac{\text{supp}(\text{rule})}{\text{supp}(\text{antecedent})\text{supp}(\text{consequent})}$$

Wu et al. [22] have proposed a *conditional-probability increment ratio* (CPIR) measure that assesses whether two entities (in our case activities)  $a$  and  $b$  are positively or negatively related.

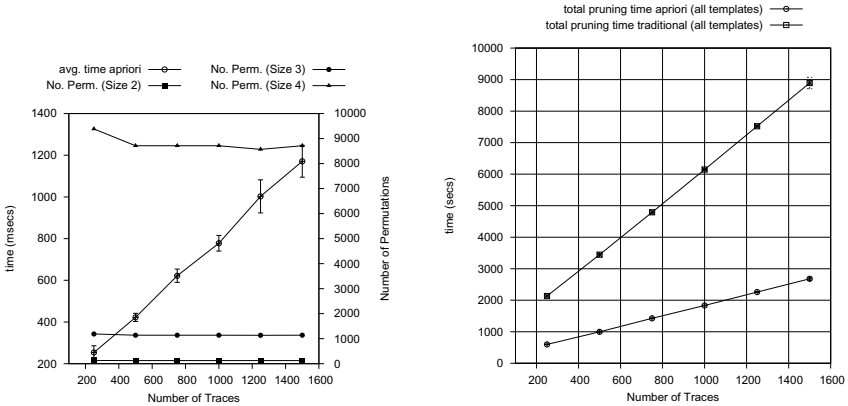
**Definition 5 (CPIR).** *The CPIR measure of a Declare constraint expressed as an association rule in an event log  $\mathcal{L}$  is defined as*

$$\text{CPIR}(\text{rule}) = \frac{\text{supp}(\text{rule}) - \text{supp}(\text{antecedent})\text{supp}(\text{consequent})}{\text{supp}(\text{antecedent})(1 - \text{supp}(\text{consequent}))}$$

CPIR can be used as a measure of confidence of a constraint and can be used to mine negative constraints as well. If  $\text{CPIR}(\text{rule})$  involving the activities  $a$  and  $b$  is greater than some threshold, then the constraint defined over  $a$  and  $b$  is a positive constraint of interest. If  $\text{CPIR}(\text{rule})$  involving the activities  $a$  and  $b$  is negative then  $a$  and  $b$  are negatively related or in other words  $a$  and  $\neg b$  are positively related.

## 5 Experiments and Discussion

To analyze the performance of our approach, we have simulated multiple event logs of the insurance claim example with varying numbers of cases/events.



(a) Average computation time needed for the generation of the frequent sets (with support of at least 0.4) and their permutations for varying sizes of the event log; numbers of permutations generated for frequent sets of sizes 2, 3 and 4

(b) Average computation time for pruning needed for the pruning phase in the naive approach and in the Apriori approach for varying sizes of the event log

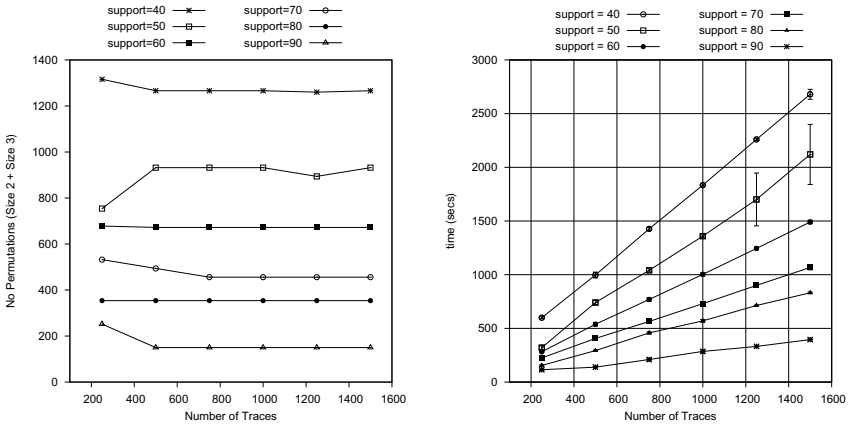
**Fig. 6.** Experimental results for the insurance claim process

In the *first phase* of our approach, we generate a set of candidate constraints. For a Declare template with  $k$  parameters, we use the Apriori algorithm to generate frequent sets of size  $k$ . Then, we generate all permutations of each frequent set. Fig. 6a shows the average computation time along with the 95% confidence interval (over five independent runs) required for this first phase<sup>1</sup>. We generated frequent sets of different sizes (2, 3 and 4) and with a support of (at least) 0.4. We can see that the time required varies linearly with the size of the log. Fig. 6a also depicts the number of permutations generated. As expected, the number of permutations is close to constant and is not significantly influenced by the size of the event log. Moreover, these results show that we need to generate (on average) 126 candidate constraints to discover constraints deriving from a Declare template with 2 parameters, (on average) 1,140 candidate constraints to discover constraints deriving from a Declare template with 3 parameters, and (on average) 8,800 candidate constraints to discover constraints deriving from a Declare template with 4 parameters. Considering that the number of activities in each of the considered logs is 15, these numbers are small compared to the naive approach where  $15^2$  (225),  $15^3$  (3,375) and  $15^4$  (50,625) candidate constraints would be generated for constraints with 2, 3 and 4 parameters respectively.

In the *second phase* of our approach, we prune the set of candidate constraints using the metrics described in Section 4. Fig. 6b shows the average computation

<sup>1</sup> All the computation times reported in this section are measured on a Core 2 Quad CPU @2.40 GHZ 3 GB RAM.





(a) Number of permutations generated for varying sizes of the event log and for different support values

(b) Average computation time for the pruning phase for varying sizes of the event log and for different support values

**Fig. 7.** Analysis of the Apriori-based approach

time required for pruning for the different event logs (with a 95% confidence interval computed over five independent runs). We compare the time needed for the pruning phase using the naive approach and the Apriori-based approach. In both cases, the time required varies linearly with respect to the size of the log. However, the Apriori-based approach clearly outperforms the naive approach. The computation time for the Apriori algorithm itself is negligible (less than 0.05% of the computation time needed for pruning).

Fig. 7a shows the number of permutations generated for varying sizes of the event log and for different support values for the Apriori algorithm (again with 95% confidence intervals over five independent runs). As expected, the number of frequent sets does not depend on the number of process instances in the log but on the support value used by the Apriori algorithm. The number of permutations clearly increases when the support value increases. Fig. 7b shows the average computation time in relation to the support and number of traces. The time required varies linearly with respect to the size of the log. However, the gradient of the lines increases when the support increases.

The Declare models obtained after pruning using the naive approach and the approach based on the Apriori algorithm are always the same but the latter is more efficient.

Table 3 shows a set of four constraints discovered from one of our synthetic logs (with 250 process instances). These constraints have been discovered using a minimum value for support (0.4), a minimum value for confidence (0.9), and a minimum value for CPIR (0.03). It is important to highlight that all these metrics are important when selecting relevant constraints. Consider, for instance, the two responded existence constraints in the table. Both constraints have a confidence

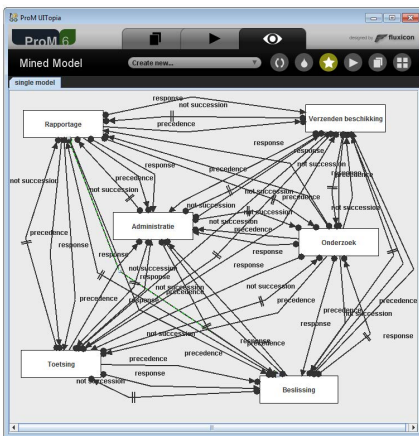
**Table 3.** Constraints discovered from a synthetic logs with 250 process instances

template	LTL semantics	parameter a	parameter b	support	confidence	interest factor	CPIR
response	$\Box(a \rightarrow \Diamond b)$	Contact Hospital	Receive Questionnaire Response	0.400	0.925	1.028	0.259
not succession	$\Box(a \rightarrow \neg(\Diamond b))$	Contact Hospital	High Insurance Check	0.432	1.000	1.633	1.000
responded existence	$\Diamond a \rightarrow \Diamond b$	Send Notification by Phone	Receive Questionnaire Response	0.712	0.903	1.000	0.035
responded existence	$\Diamond a \rightarrow \Diamond b$	High Medical History	High Insurance Check	0.496	1.000	1.633	0.999

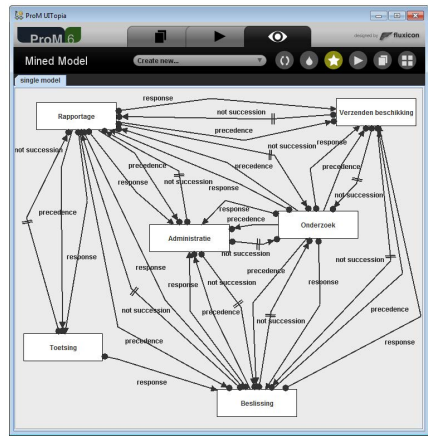
close to 1, but the responded existence between *Send Notification by Phone* and *Receive Questionnaire Response* has CPIR equal to 0.035, whereas the responded existence between *High Medical History* and *High Insurance Check* has CPIR equal to 0.999. This is due to the fact that the existence of *High Medical History* is strongly related to the existence of *High Insurance Check* (the former can only be executed if the latter has been executed), whereas in the other responded existence constraint the connection between *Send Notification by Phone* and *Receive Questionnaire Response* is less relevant.

## 6 Case Study

After showing experimental results focusing on the performance of our new approach to discover Declare models, we now demonstrate its applicability using an event log provided by a Dutch municipality. The log contains events related to requests for excerpt from the civil registration. The event log contains 3,760 cases and 19,060 events. There are 26 different activities.



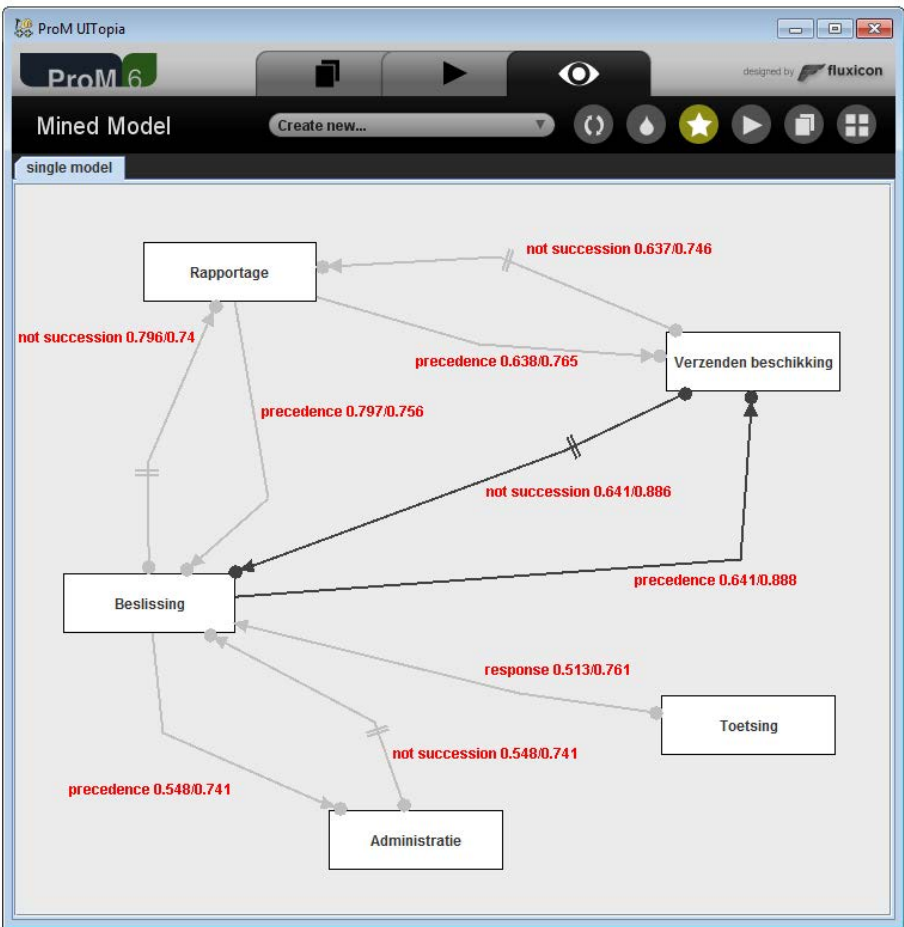
(a) 30% of interesting witnesses



(b) 50% of interesting witnesses

**Fig. 8.** Declare models discovered using the naive approach

Fig. 8a depicts the Declare model discovered using the naive approach showing all constraints with at least 30% of interesting witnesses in the log. The resulting Spaghetti-like model has 45 constraints. The computation time to generate this model is 1,711 seconds. This model can be improved by increasing the percentage of required interesting witnesses. Fig. 8b depicts the resulting Declare model using the naive approach including only constraints with at least 50% of interesting witnesses. This model has 33 constraints and the computation time needed to generate it is 1,743 seconds. Note that the computational time needed to discover the Declare models in Fig. 8 is approximately the same because in both cases 2,028 ( $3 \cdot 26^2$ ) constraints must be checked (we search for three types of constraints with 2 parameters, i.e., precedence, response and not succession).



**Fig. 9.** Declare model discovered using the new approach. Note that the most important constraints are highlighted.

Fig. 9 shows the results obtained using our new approach based on the Apriori algorithm and the pruning techniques described in this paper. This model is composed of 9 constraints and the computation time needed to generate it is 76 seconds. The model contains only constraints with a support greater than 0.5 and CPIR value greater than 0.7.

Moreover, the discovered model emphasizes the most important constraints, just like highways are highlighted on a roadmap. Constraints with a CPIR value of at least 0.85 (considered more relevant) are indicated in black, whereas the constraints with CPIR less than 0.85 (less relevant) are indicated in gray. Each constraint is annotated with support and CPIR values (in red). The graphical feedback facilitates the interpretation of the discovered Declare model.

We evaluate the support of a Declare constraint on the basis of the vacuity detection conditions in Table 1. These conditions guarantee that a process instance is an interesting witness for a constraint if no stronger constraint holds in the same instance. This means that if two constraints hold in the log and one of them is stronger than the other, our approach will discover the stronger one.

## 7 Conclusion

Although real-life processes discovered through process mining are often Spaghetti-like, lion's share of process discovery algorithms try to construct a procedural model (e.g., BPMN models, EPCs, Petri nets, or UML activity diagrams). The resulting models are often difficult to interpret. Therefore, it is interesting to discover declarative process models instead.

In this paper, we present an approach to efficiently discover Declare models that are understandable. Unlike earlier approaches we do not generate all possible constraints and only check the most promising ones using an Apriori algorithm. This results in dramatic performance improvements. We also defined several criteria to evaluate the relevance of a discovered constraint. The discovered model is pruned using these criteria and the most interesting constraints are highlighted. As demonstrated using a case study, this results in understandable process models.

## References

1. van der Aalst, W.M.P., Weijters, T., Maruster, L.: Workflow mining: discovering process models from event logs. *Knowledge and Data Engineering*, 1128–1142 (2004)
2. van der Aalst, W.M.P.: *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer (2011)
3. van der Aalst, W.M.P., de Beer, H.T., van Dongen, B.F.: Process Mining and Verification of Properties: An Approach Based on Temporal Logic. In: Meersman, R., Tari, Z. (eds.) *CoopIS/DOA/ODBASE 2005*. LNCS, vol. 3760, pp. 130–147. Springer, Heidelberg (2005)
4. van der Aalst, W.M.P., Pesic, M., Schonenberg, H.: Declarative Workflows: Balancing Between Flexibility and Support. *Computer Science - R&D*, 99–113 (2009)

5. van der Aalst, W.M.P., Reijers, H., Weijters, A., van Dongen, B., de Medeiros, A.A., Song, M., Verbeek, H.: Business Process Mining: An Industrial Application. *Information Systems*, 713–732 (2007)
6. Agrawal, R., Gunopulos, D., Leymann, F.: Mining Process Models from Workflow Logs. In: Schek, H.-J., Saltor, F., Ramos, I., Alonso, G. (eds.) *EDBT 1998*. LNCS, vol. 1377, pp. 469–483. Springer, Heidelberg (1998)
7. Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules. In: *VLDB 1994*, pp. 487–499 (1994)
8. Beer, I., Eisner, C.: Efficient detection of vacuity in temporal model checking. In: *Formal Methods in System Design*, pp. 200–201 (2001)
9. Brin, S., Motwani, R., Silverstein, C.: Beyond Market Baskets: Generalizing Association Rules to Correlations. In: *ACM SIGMOD 1997*, pp. 265–276 (1997)
10. Cook, J.E., Wolf, A.L.: Discovering models of software processes from event-based data. *ACM Trans. on Software Engineering and Methodology*, 215–249 (1998)
11. Damaggio, E., Deutsch, A., Hull, R., Vianu, V.: Automatic Verification of Data-Centric Business Processes. In: Rinderle-Ma, S., Toumani, F., Wolf, K. (eds.) *BPM 2011*. LNCS, vol. 6896, pp. 3–16. Springer, Heidelberg (2011)
12. Datta, A.: Automating the Discovery of As-Is Business Process Models: Probabilistic and Algorithmic Approaches. *Info. Systems Research*, 275–301 (1998)
13. Declare (2008), <http://declare.sf.net>
14. Günther, C.W., van der Aalst, W.M.P.: Fuzzy Mining – Adaptive Process Simplification Based on Multi-perspective Metrics. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) *BPM 2007*. LNCS, vol. 4714, pp. 328–343. Springer, Heidelberg (2007)
15. Kupferman, O., Vardi, M.Y.: Vacuity Detection in Temporal Model Checking. *International Journal on Software Tools for Technology Transfer*, 224–233 (2003)
16. Lamma, E., Mello, P., Montali, M., Riguzzi, F., Storari, S.: Inducing Declarative Logic-Based Models from Labeled Traces. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) *BPM 2007*. LNCS, vol. 4714, pp. 344–359. Springer, Heidelberg (2007)
17. Maggi, F.M., Mooij, A.J., van der Aalst, W.M.P.: User-guided discovery of declarative process models. In: *CIDM* (2011)
18. de Medeiros, A.A., Weijters, A., van de Aalst, W.M.P.: Genetic Process Mining: An Experimental Evaluation. *Data Mining and Knowledge Discovery*, 245–304 (2007)
19. Pesic, M.: Constraint-Based Workflow Management Systems: Shifting Controls to Users. Ph.D. thesis, Beta Research School for Operations Management and Logistics, Eindhoven (2008)
20. Pesic, M., Schonenberg, H., van der Aalst, W.M.P.: DECLARE: Full Support for Loosely-Structured Processes. In: *EDOC 2007*, pp. 287–298 (2007)
21. Schunselaar, D.M.M., Maggi, F.M., Sidorova, N.: Patterns for a Log-Based Strengthening of Declarative Compliance Models. In: *iFM 2012* (to appear, 2012)
22. Wu, X., Zhang, C., Zhang, S.: Efficient Mining of Both Positive and Negative Association Rules. *ACM Transactions on Information Systems*, 381–405 (2004)
23. Zugal, S., Pinggera, J., Weber, B.: The Impact of Testcases on the Maintainability of Declarative Process Models. In: Halpin, T., Nurcan, S., Krogstie, J., Soffer, P., Proper, E., Schmidt, R., Bider, I. (eds.) *BPMDs 2011 and EMMSAD 2011*. LNBI, vol. 81, pp. 163–177. Springer, Heidelberg (2011)

# OtO Matching System: A Multi-strategy Approach to Instance Matching

Evangelia Daskalaki and Dimitris Plexousakis

Foundation of Research and Technology Hellas, Institute of Computer Science, Crete, Greece  
{eva,dp}@ics.forth.gr

**Abstract.** In this paper we describe an ontology to ontology (OtO) matching system which implements a novel instance matching algorithm. The proposed multi-strategy matching system is domain independent and fully customizable at any level. It optimizes the instance matching process by leveraging (a) the rich semantic knowledge from the schema matching results, (b) the implicit knowledge of the domain expert by capturing the identification power of the properties and (c) the probability estimation of the result's validity, in order to accurately detect the ontology instances that represent the same real-world entity. Furthermore we evaluate the system with the ISLab Instance Matching Benchmark in the Ontology Alignment Evaluation Initiative 2009 campaign and report the results.

**Keywords:** Ontology, instance matching, record linkage, entity resolution.

## 1 Introduction

Ontologies, as the means to conceptualize domain knowledge, have become the enabler of the fulfillment of the semantic web vision. Their aim is to make data sharable in order to make them understood and processed by computers with minimum human interference.

Due to the fact that ontologies are heterogeneous and distributed, it is necessary to discover the alignment across ontologies but also the alignments across their instances. In other words, it is necessary to achieve semantic interoperability across ontologies. In [44] ontology alignment between two ontologies is described as a function which gives as output a relationship between the vocabularies of the ontologies.

Several frameworks for ontology integration have been proposed. Their research efforts are mainly focused on the integration at the ontology schema level. For those that consider integrating data at the ontology instance level only a few systems deal with this problem. At the same time, the demand for high-quality ontology instance matching is crucial in the context of identity recognition, ontology population and semantic integration.

We propose a new multi-strategy OtO matching system that focuses on the ontology instance matching, but also implements schema alignment. The OtO matching system is domain independent and fully customizable by the domain expert at any level. We argue that, unless a system is fully customizable, it can not claim to be

domain independent. The proposed system implements several different ontology matching processes, which can be roughly categorized into the processes that include schema matching algorithms and those that include instance matching algorithms.

The schema matching process of the proposed system leverages the lexical, semantic and syntactic similarities of the entities of the schema. As far as the instance matching process is concerned, it does not only rely on lexical similarity measures and pair-wise instance comparison of the instance properties. The system optimizes the instance matching process by leveraging (a) the rich semantic knowledge we gain from the output mappings of the schema matching process, (b) the implicit knowledge of the domain expert by capturing the identification power of the properties and (c) the probability calculation of the result's truth, in order to accurately and efficiently detect the ontology instances that represent the same real-world entity.

We evaluated the OtO matching system with the ISLab Instance Matching Benchmark (IIMB) where we achieved the best results in recall and the second best results in precision among all the systems that took part in the benchmark, with 100% recall and 97% precision.

The rest of this paper is organized as follows: In section 2 we give a formal definition of ontology instance matching. Furthermore, we analyze the ontology instance matching requirements. In section 3 we review the related scientific work done within the past years, as well as the related ontology instance matching frameworks. In section 4 we give a detailed description of the proposed OtO matching system, by focusing on the new instance matching algorithm. In section 4 we deal with the experiments and their results. We discuss the outcomes returned by running the test cases from the OAEI 2009 IIMB, in comparison with the other instance matching frameworks that took part in this benchmark. Finally, in section 6, we summarize the strengths and the weaknesses of the OtO matching systems and outline further research work.

## 2 Instance Matching

In contrast to ontology schema matching, instance matching deals with the actual individuals, not with the alignment of class structures or entity types. Instance matching has to deal with deciding whether two entity descriptions refer to the same individual. Thus, it crucially depends on measuring the similarity between sets of annotated instances.

As stated in [1] the application of instance matching is crucial in identity recognition, ontology population and semantic integration. Identity recognition refers to the capability of detecting whether two different resource descriptions refer to the same real-world entity, namely an individual. Ontology population refers to the need of supporting experts in managing ontology changes through advanced, and possibly automated, techniques [2,3]. It can be described as a part of ontology evolution, where ontology is evolved by acquiring new semantic descriptions of data extracted from heterogeneous data sources. For ontology population, instance matching plays a crucial role in order to correctly perform the insertion activity and discover the relationship between new incoming instances and the set of instances already existing in the

ontology. The matchings produced by instance matching are exploited for clustering instances that are recognized as referring to the same real-world entity.

Finally, due to the increasing popularity of Semantic Web technologies, a novel attention on semantic integration issues has risen. For semantic integration, advanced techniques for ontology instance matching are required to correctly combine data describing individuals in different sources and to improve the accuracy of the schema ontology alignment process. The notion behind this is that the more significant the overlap of common instances of two ontology concepts is the more related these concepts are.

In our work we focus on identity recognition. Identity recognition is formally described in [4] as the matching procedure of two instances that belong to the same ontology or different ontologies, where the result can either be {match} or {no match}. If the result is {no match} then this denotes the fact that the two instances do not refer to the same real object. On the other hand if the result is {match} then this denotes the fact that both instances refer to the same real object.

### 3 Related Work

The problem of instance matching has been a topic of research for many years. It is also known as the problem of record linkage [5,6], duplicate detection [7], entity resolution [8], merge/purge [9], and object reconciliation or identification [10]. Authors in [11] focus on the idea of establishing co-reference information and preserving it, in order to build large scale networks of knowledge. Data integration and methods are also discussed in [12] and [13].

While the majority of existing methods were developed for the task of matching database records, modern approaches mostly focus on graph-based data representations extended by additional schema information. This allows us to describe it within the well-established ontology matching framework [14].

The ontology instance matching problem is seen in the literature from two different perspectives. In the first perspective the instance matching results are the means to derive more accurate schema matching results [15-17]. The second perspective uses the outcomes of the ontology schema matching in order to improve the instance matching process [18]. The notion behind the latter perspective is that if two ontology concepts are semantically correlated then it is likely that their instances refer to the same individuals, but if the concepts are not related then their instances will also not be related to each other.

As far as the matching techniques are concerned, authors in [44] differentiate the instance matching systems into those that use linguistic-based approaches, using names and textual descriptions and those that use constraint-based approaches, using element constraints such as data types, domains and key characteristics.

Following the differentiation mentioned above DSSim [22], A-Flood [23], CODI [25], SERIMI [28], AgreementMaker [29] and Zhishi.links [30] are linguistic-based systems. On the other hand ASMOV [20], RiMOM [19], COMA++ [24], FBEM [21], H-Match2.0 [26], LN2R [31], ObjectCoref [27] and OtO system use constraint-based approaches.



In particular, the DSSim instance mapping process compares all the instance properties by using multi core processors and splitting up the large ontologies into smaller fragments. A-Flood uses firstly the schema matching results as input for the instance matching. Then, the alignment of the types of the instances is checked. CODI is a probabilistic –logical alignment system. It implements an approach which utilizes object-properties to determine the instances for which the similarity should be computed. In this system, the schemata are assumed to be the same. SERIMI extracts the instance labels and searches for instances in a target dataset that may have a similar label. Then it filters among the instances found in the target dataset, those that actually refer to the same entity in the real world as the source instance. The instance matching algorithm in AgreementMaker consists of doing a lookup using the label of the instance, the type, and querying against an index which returns a reasonable number of candidate target instances. The target instances are then ranked and eventually the best one is selected. Zhishi.links begins with the string similarity calculation and then with the semantic similarity calculation. Match candidates are sorted by their similarity scores.

The RiMOM system, ASMOV, FBEM, H-Match2.0, ObjectCoref and the OtO matching system are constraint-based systems, because they differentiate the significance of the instance properties with the use of the property weights (key-characteristics). The weights' calculation though, is done differently for every system. In ASMOV and RiMOM weights are fixed and depend on the occurrence of each instance property. ObjectCoref system's architecture follows a common self-training framework. For calculating the similarity between the instances it uses fixed weights proportional to the frequency of their property values. In H-Match2.0 and FBEM, weights are fixed and calculated by using a vocabulary of key properties. OtO system also uses a vocabulary of key properties, but property weights are not fixed. They can be customized from the domain expert. In COMA++ system constraints are compared. COMA++ distinguishes between the General constraints, the Numerical constraints, and the Pattern constraints. In the match process the constraint-based matcher determines the similarity of two elements by comparing their previously identified constraints. The content-based matcher determines the similarity of two elements by executing a pair-wise comparison of instance values using a similarity function. The result is a similarity matrix, with each dimension representing the instances of one element. Finally, the reference reconciliation system (LN2R) is a knowledge-based, unsupervised system, based on two methods, a logical one called L2R and a numerical one called N2R. The Logical method for Reference Reconciliation (L2R) is based on the translation in first order logic (Horn rules) of some of the schema semantics. In order to complement the partial results of L2R, a Numerical method for Reference Reconciliation (N2R) is created. It exploits the L2R results and allows computing similarity scores for each pair of references.

The advantage of the OtO system is that, apart from using the property weights to cluster the properties of high importance, moreover these weights can easily be customized from the domain expert, in order to achieve the best results. Another novel feature of the OtO system is that it does not filter the results only with the commonly used threshold, but it also uses the property weights to calculate the trust ability of the results truth. All these advantages are analyzed in the following sections.

## 4 The OtO Matching System

The proposed multi-strategy OtO matching system is domain independent and fully customizable at any level. It implements several different ontology matching strategies, which can be roughly categorized into the schema matching and the instance matching strategies. Our contribution in the OtO matching system is novel instance matching algorithm. In this section, we provide a detailed presentation of the matching process. The matching process consists of two process phases a) the schema matching, and b) instance matching.

The first phase of the matching process is the schema matching process. Depending on the domain expert's selections, the OtO matching system either proceeds with the schema matching process or not. If the user does not select to run a schema matching algorithm, the system assumes that the source schema and the target schema are the same and creates a "mirror mapping", which is stored into the database. In order to create the mirror mapping the OtO system reads the source schema and creates a mirror of it as the target schema.

If the domain expert has selected to proceed with the schema matching, then the OtO system follows the algorithm that is described in [33], which followed the work done in [34]. In [33], after storing the ontologies into the database as a directed acyclic graph, the schema matching algorithm follows a hybrid algorithm for matching the schemata. It is called hybrid algorithm because, depending on the domain expert's selections, the algorithm runs sequentially a lexicographic matching process, a semantic matching process and syntactic matching process between the schemata. In the lexicographic matching process, each pair of concepts is compared by using character-based metrics, token-based metrics, phonetic-based metrics and the WordNet® Thesaurus. Then it returns a similarity matrix of all the corresponding concepts. In the sequel, the semantic matching process calculates the semantic similarity for each pair of concepts, based on attributes/properties that characterize and describe the two concepts. Finally, both lexical and semantic similarities are taken into account for the next step, which is syntactic similarity. For each pair of concepts, the descendants' similarity is examined in order to produce the total similarity measurement. The final result is produced by combining lexical, semantic and syntactic similarity.

The schema matching results are presented to the end user in the form of a table with the source element, the target element and the similarity of each match. The domain expert can either accept the matches, or reject them, by unchecking the "accept" checkbox of each match. Furthermore, the domain expert can alter the automatically predefined weights of significance of each property match. The usage of these weights is described in detail in the next sections.

### 4.1 The Instance Matching Process

The second phase of the matching process is the ontology instance matching. This is the new feature of the OtO matching system which completes the OtO matching process by allowing the comparison of the instances. Through this procedure, the OtO matching system measures the similarity between sets of annotated instances. The goal of this process is to determine instances that represent the same real particulars.

The OtO matching system introduces a combination of methods and algorithms for better accuracy, efficiency and effectiveness. In particular the instance matching algorithm leverages (a) the rich semantic knowledge we gain from the output mappings of the schema matching process, (b) the implicit knowledge of domain expert capturing the identification power of the properties with the use of property-weights and (c) the 2-phase filtering method of the results, in order to accurately and efficiently detect the ontology instances that represent the same real-world entity.

As far as the first is concerned, the OtO matching instance algorithm uses the outputs of the schema matching process. As a result only the instances that belong to semantically related classes are included in the instance matching algorithm. The intuition behind this is that the schema mappings contain semantic knowledge which is important for the efficient instance matching. If for example the schema matching returns a mapping between the “person” entity from ontology A and the “sportsper-son” entity from Ontology B, then it is wise just to compare the instances of type “person” from ontology A, with the instances of type “sportsper-son” from ontology B, rather than comparing all the instances from ontology A with all the instances from ontology B.

Furthermore, the OtO system uses the keyword rules (vocabularies) in order to automatically detect the significant properties and assign them high property weights. By capturing the identification power of the properties, the OtO system can reduce the unnecessary comparisons of the property instances, and only compare the properties of the instances that can determine the identity recognition. An important feature of the system is that the domain expert can see the property weights and alter or correct them depending on his/her implicit knowledge on the ontologies.

As far as the third point is concerned, the OtO system uses two different filtering methods to determine whether a match is considered as confident or not. The first is the threshold filter and the second is the key-weight flag filter. If the weighted similarity of an instance pair is below the threshold then it is not considered as a match, otherwise it can be considered as match depending on the second filter. The second filter, the key-weight flag filter, is the probability of an instance pair to be a true match. Through the use of the property weights, the OtO matching system can calculate the probability of an instance pair to be true match or not.

## 4.2 Property Weights

The OtO instance matching algorithm uses the property weights in order to capture the identification power of the properties. The property weight is a real number in the range of [0.0, 1.0]. This number shows how indicative a property match is to determining whether two instances refer to the same real object or not. In other words, the property weight indicates the probability of a property match with the same value to be true particular match. This probability includes two kind of stochastic processes, as also described in [35]: the probability distribution of the property and the probability distribution of the knowledge of the property. The first refers to how many instances share the same property value in the real world and the second refers to how many instances share the same property in a specific knowledge base.

In order to have a better understanding of previously mentioned stochastic processes, let us see an example. Assumingly we are comparing two instances of the

ontology class “country”, in order to identify if these two instances refer to the same real country or not. This “country” class has two properties: (a) the “country-currency” and (b) the “country-name”. For both of the instances the value of the property “country-currency” is “EUR”. Given that 17 countries have as official currency the EUR, the probability of these two countries to refer to the same real country would be 5.89% (i.e., 1 out of 17). If the domain expert knew that only 10 out of the 17 are inserted in our knowledge base then the probability would be 10% that these two countries refer to the same real country.

As far as the property “country-name” is concerned the distribution of this property and the distribution of the knowledge of the property is the same; each country has a unique name. So the probability of two countries to refer to the same real country if they share the same name is 100%. Thus, the weight of the property “country-name” would be 1.0 and the weight of the property “country-currency” would be 0.1.

The domain expert is expected to alter the automatically inserted weights of each property-match or insert new ones in the matching process, if she considers it as necessary. A first attempt to insert weights to the properties is done automatically from the OtO system, by using keyword rules. These keyword rules can easily be customized by the domain expert. The property matches that include the specific keywords are given automatically the associated weight amount. Some examples of the keywords are: “name 1.0”, “homepage 1.0”, “coordinates 1.0”, “website 0.9”, etc. The other property matches are given zero weight. The domain expert is asked to correct the weights and insert more or delete the unnecessary ones according to her implicit knowledge.

As described above, through the use of weights the system is able to determine the matches that are crucial for the outcome and separate them from the matches that are not important. This is done in the matching results table, after the schema matching process. If the domain expert selects the weight of  $w_i=0$ , this means that the matching is not important for the result and thus does not match the values of the instance properties. On the other hand if he/she selects  $w_i=1.0$  this means that the similarity of the property match is of high importance and may indicate that the property values match the entire instance. Thus, the OtO matching system is able to reduce the number of comparisons conducted, without losing any important information.

### 4.3 The Key Weight Flag

The key weight flag is an important feature added into the instance matching algorithm in order to provide more accurate and trustworthy results. The key weight flag is a flag which allows instance matches with similarity above threshold to be added in the mapping result. In order to determine this, the key weight flag is turned into true (green flag to pass) only if the matched instances have at least one property match with weight above or equal 0.5 ( $w_i \geq 0.5$ ).

$$\text{Count} ([ w_1, w_2, w_3, \dots, w_p ] \geq 0.5) > 0$$

If the matched instances have similarity above the threshold but all their property annotation weights are less than 0.5 then the match is not considered as trustworthy and is not added in the instance matches result.

#### 4.4 Instance Matching Algorithm

The algorithm begins by loading the schema mappings. For each class mapping returned from the schema matching, the algorithm returns the source class instances and the target class instances.

For each of the instances of the source and the target mapping class, the algorithm returns the property annotations and their values. If there are no class mappings then the algorithm compares all the instances with each other. If the property annotations of the instances are among the property mappings, then the algorithm checks the property weights of the match. If the weight is over the low limit of 0.0, then the algorithm checks the property values. If the property values are references to other instances then it indicates the correct instance values. Once the values have been indicated the algorithm proceeds with the value matching algorithms.

The domain expert can select the string matching algorithms that will be applied on the instance values. There are many available instance matching algorithms. They contain character based metrics, token-based metrics, phonetic similarity metrics and semantic-based metrics. After applying the selected string matching algorithms, a weighted average of the property similarities is calculated. This process is followed for all the matching property annotations of each instance matching pair and the property similarities are stored in a property similarity matrix.

When there are no more properties to compare, a weighted average of the instance similarity is calculated this time for the instance pair with formula (1). The similarities  $\text{sim} = \{\text{sim}_1, \text{sim}_2, \text{sim}_3, \dots, \text{sim}_p\}$  represent the similarities of each property pair and the weights  $w = \{w_1, w_2, w_3, \dots, w_p\}$  represent the weights of each matching pair which are given by the domain expert.

$$\text{Instance sim} = \frac{\sum_{n=0}^p \text{sim}_p * w_p}{\sum_{n=0}^p w_p} \quad (1)$$

If the calculated instance similarity is above the given threshold then the instance pair is an accurate match and might be added in the matching results. If it will be added in the matching result or not, depends on the “key-weight flag”.

Unlike the schema matching algorithm, in the instance matching algorithm every source instance is allowed to match with no more than one target instance (1:1). That of course is based on the fact, that the aim of the instance matching algorithm is the identity recognition, so the system returns only the best target instance match for a given source instance. Figure 1 shows the instance matching algorithm in pseudocode.

```

for every class matching {
  return source instances and target instances
  for every instance pair{
    return source instances properties
    and target instances properties
    for every property pair{
      if property matching && accepted {
        if property weight>0.0{
          if property weight>=0.5
          { weightFlag=true}
          return weighted similarity
          from matching algorithms
        }
      }
      return weighted instance similarity
    if instance sim>=threshold && weightFlag=true{
      add instance match
    }
  }
}

```

Fig. 1. Instance matching algorithm in pseudo-code

## 5 Experiments and Results

In order to evaluate the matching system and test the instance matching algorithm we have used the ISLab Instance Matching Benchmark (IIMB) [36]. The IIMB is a benchmark that is generated automatically, starting from one data source that is automatically modified according to various criteria. The original data source contains OWL/RDF data about actors, sports, persons, and business firms provided by the OKKAM European project [37].

The benchmark is composed by 37 test cases. For each test case it is required to match the original data source (source ontology) against a new data source (target ontology). The original data source contains 222 different instances and the new data sources contain from 222 to 1700 different instances. Each test case contains a modified ABox (abox.owl + tbox.owl) and the corresponding mapping with the instances in the original ABox (Golden standard).

The IIMB benchmark was chosen because it contains many test sets which introduced different kinds of challenges as listed below:

- Test case 01: Contains an identical copy of the original data source (instance IDs are randomly changed)
- Test case 02 - Test case 10: Value transformations (i.e., typographical errors simulation, use of different standards for representing the same information). In order to simulate typographical errors, property values of each instance are randomly modified. Modifications are applied on different subsets of the instances property

values, with varying levels of difficulty (i.e., introducing a different number of errors).

- Test case 11 - Test case 19: Structural transformations (i.e., deletion of one or more values, transformation of datatype properties into object properties, separation of a single property into more properties).
- Test case 20 - Test case 29: Logical transformations (i.e., instantiation of identical individuals into different subclasses of the same class, instantiation of identical individuals into disjoint classes, instantiation of identical individuals into different classes of an explicitly declared class hierarchy).
- Test case 030 - Test case 037: Several combinations of the previous transformations.

Furthermore the IIMB Benchmark was firstly introduced in the Ontology Alignment Evaluation Initiative 2009 campaign [38]. Thus, apart from having a Golden Standard to compare our results we also have other systems' results to be compared against. The other systems that participated in the IIMB benchmark 2009 campaign are the Aflood system [39], the ASMOV system [40], the DSSim system [41], the HMatch system, the FBEM system [42] and the RiMOM system [43].

### 5.1 Overall Conclusions of the Test Cases

The results of the Harmonic mean of the test cases, as reported in [45], have shown that the OtO system is ranked in the first position in terms of recall among all systems with recall=100% and the second position in terms of precision with precision=97% as shown in Table 1.

**Table 1.** Harmonic mean of test cases

	<b>Prec.</b>	<b>Rec.</b>	<b>FMeas.</b>
<b>AFlood</b>	0.92	0.87	0.89
<b>ASMOV</b>	1.00	0.98	0.99
<b>DSSim</b>	0.92	0.48	0.63
<b>HMatch 2.0</b>	0.89	0.93	0.91
<b>FBEM</b>	0.16	0.75	0.27
<b>RiMOM</b>	0.96	0.98	0.97
<b>OtO</b>	0.97	1.00	0.98

Many of the test cases contained skewed data with typographical errors. An example of test case 10 can be seen in Figure 2. This example shows two instances from the source ontology and the target ontology respectively, that are reported to be a correct match according to the Golden Standard. The OtO matching system identified that the significant instance properties that should be compared are the “cogito-name” and the “cogito-first-sentence”. By applying character based metrics, token-based metrics and phonetic similarity metrics, the OtO matching system correctly indicated that these two instance where a true match.

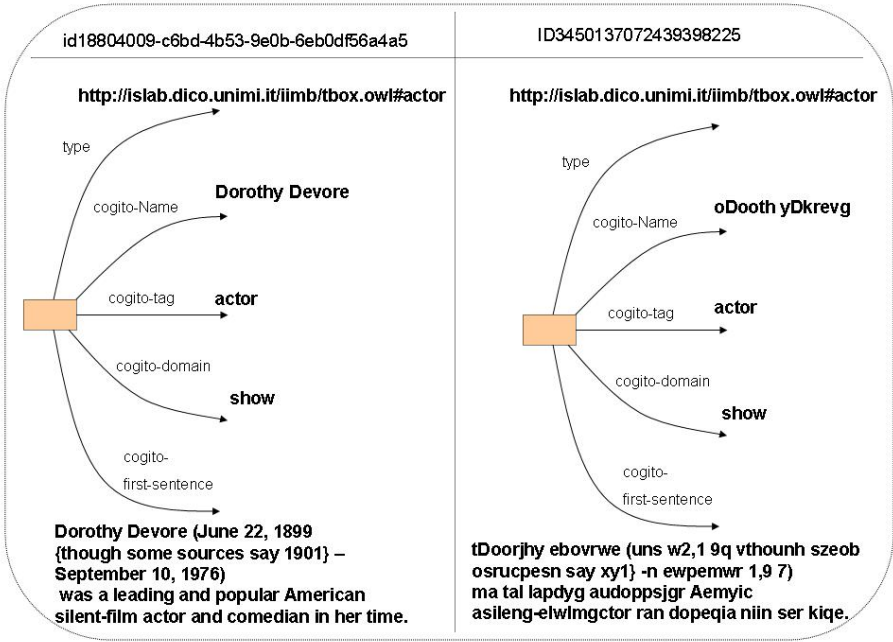


Fig. 2. Example of instances snapshot from test case 10

In some other test cases random properties have been removed, where some of these have been critical for the identification of the instances. Due to the 2-phase filtering method, the OtO matching system returned only those results where the confidence level of the results truth was high. Thus, when the instance properties were not significant enough, then the key weight flag prevented these results from being added in the OtO matching instances list. Figure 3 shows an example of test case 31.

Figure 3 shows two instances from the source ontology and the target ontology respectively. The Golden Standard reported that these two instances are a true match, but the OtO matching system did not return them as a correct match. The reason is that the key-weight flag filter indicated that the properties “cogito-domain” and “type” were not significant enough (the property weights were below 0.5) to determine if these two instances refer to the same real object.

Last but not least, the OtO matching system reported F-measure=1 in 84% of the test cases. This means that in eight out of ten cases the system returned all the correct answers and no false answer. What’s more, another success of the OtO system is that it was ranked in the first position in F-measure in 32 out of 37 test cases. For the results of all the test cases, please refer to: [http://www.ics.forth.gr/~eva/OtO\\_Matching\\_System\\_Results.pdf](http://www.ics.forth.gr/~eva/OtO_Matching_System_Results.pdf)



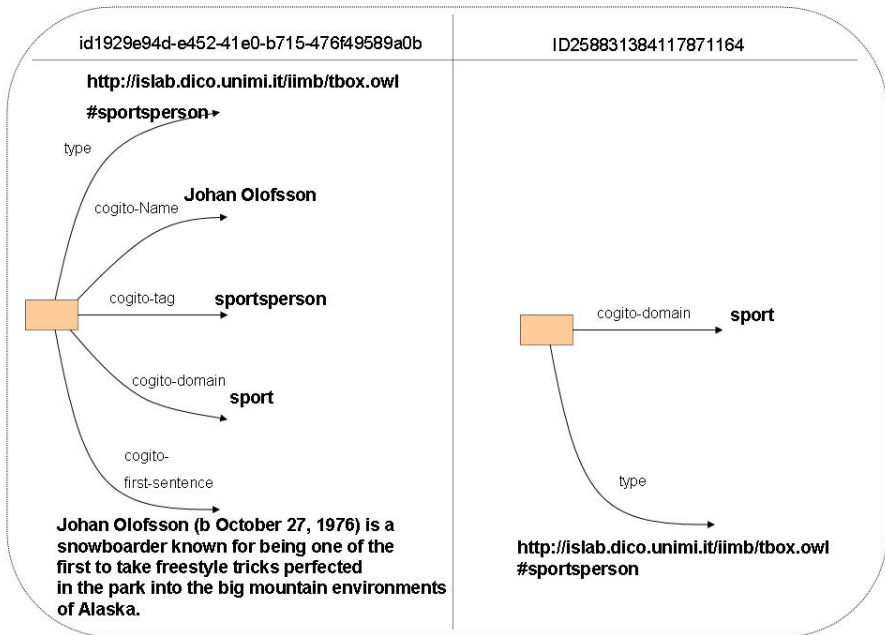


Fig. 3. Example of instances snapshot from test case 31

## 6 Conclusions and Future Work

In this paper, we presented the OtO matching system, which is a multi-strategy ontology to ontology matching system. It was shown that it is a domain independent and fully customizable matching system at any level, i.e. schema level or instance level.

In order to evaluate the matching system and test the instance matching algorithm we have used the ISLab Instance Matching Benchmark (IIMB) from the OAEI 2009 campaign. The IIMB is a benchmark that is generated automatically, starting from one data source that is automatically modified according to various criteria. The benchmark results have shown that the OtO matching system has returned one of the best results. In particular, it returned the best results in recall and the second best results in precision, among seven systems.

The success of the system is grounded on the carefully designed and analyzed instance matching process. Specifically speaking, the success is due to the leverage of (a) the rich semantic knowledge we gain from the output mappings of the schema matching process, (b) the implicit knowledge of the domain expert by capturing the identification power of the properties and (c) the probability calculation of the result's truth, in order to accurately and efficiently detect the ontology instances that represent the same real-world entity.

We aim to work on the OtO matching system in order to make it even more efficient when it comes to large ontologies with millions of instances. Our focus will be on the use of multi core processors and the splitting large ontologies into smaller.

**Acknowledgement.** This work has been supported by the eHealthMonitor project (<http://www.ehealthmonitor.eu>) and has been partly funded by the European Commission under contract FP7-287509.

## References

1. Castano, S., Ferrara, A., Montanelli, S., Lorusso, D.: Instance Matching for Ontology Population. In: Proceedings of the Sixteenth Italian Symposium on Advanced Database Systems, SEBD 2008, pp. 22–25 (2008)
2. Kondylakis, H., Plexousakis, D.: Exelixis: evolving ontology-based data integration system. In: Proceedings of SIGMOD Conference 2011, pp. 1283–1286 (2008)
3. Kondylakis, H., Flouris, G., Plexousakis, D.: Ontology and Schema Evolution in Data Integration: Review and Assessment. In: Meersman, R., Dillon, T., Herrero, P. (eds.) OTM 2009, Part II. LNCS, vol. 5871, pp. 932–947. Springer, Heidelberg (2009)
4. Ferrara, A., Lorusso, D., Montanelli, S., Varese, G.: Towards a Benchmark for Instance Matching. In OM (2008)
5. Fellegi, I., Sunter, A.: A theory for record linkage. *Journal of the American Statistical Association* 64(328), 1183–1210 (1969)
6. Newcombe, H.B., Kennedy, J.M., Axford, S.J., James, A.P.: Automatic Linkage of Vital Records. *Science* 130(3381), 954–959 (1959)
7. Elmagarmid, A.K., Ipeirotis, P.G., Verykios, V.S.: Duplicate Record Detection: A Survey. *IEEE Trans. Knowl. Data Eng.* 19(1), 1–16 (2007)
8. Bhattacharya, I., Getoor, L.: Entity resolution in graphs. In: *Mining Graph Data*. Wiley & Sons (2006)
9. Hernandez, M.A., Stolfo, S.J.: The merge/purge problem for large databases. *SIGMOD Rec.* 24(2), 127–138 (1995)
10. Noessner, J., Niepert, M., Meilicke, C., Stuckenschmidt, H.: Leveraging Terminological Structure for Object Reconciliation. In: Aroyo, L., Antoniou, G., Hyvönen, E., ten Teije, A., Stuckenschmidt, H., Cabral, L., Tudorache, T. (eds.) ESWC 2010, Part II. LNCS, vol. 6089, pp. 334–348. Springer, Heidelberg (2010)
11. Meghini, C., Doerr, M., Spyrtos, N.: Managing Co-reference Knowledge for Data Integration. In: *EJC 2008*, pp. 224–244 (2008)
12. Kondylakis, H., Doerr, M., Plexousakis, D.: Mapping Language for Information Integration. Technical Report 385, ICS-FORTH, pp. 1–10 (2006)
13. Kondylakis, H., Doerr, M., Plexousakis, D.: Empowering Provenance in Data Integration. In: Grundspenkis, J., Morzy, T., Vossen, G. (eds.) ADBIS 2009. LNCS, vol. 5739, pp. 270–285. Springer, Heidelberg (2009)
14. Euzenat, J., Shvaiko, P.: *Ontology Matching*. Springer, Heidelberg (2007)
15. Wang, C., Lu, J., Zhang, G.: Integration of Ontology Data through Learning Instance Matching. In: *Web Intelligence 2006*, pp. 536–539 (2006)
16. Isaac, A., van der Meij, L., Schlobach, S., Wang, S.: An Empirical Study of Instance-Based Ontology Matching. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L.J.B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825, pp. 253–266. Springer, Heidelberg (2007)

17. Schopman, B.A.C., Wang, S., Schlobach, S.: Deriving Concept Mappings through Instance Mappings. In: Domingue, J., Anutariya, C. (eds.) ASWC 2008. LNCS, vol. 5367, pp. 122–136. Springer, Heidelberg (2008)
18. Castano, S., Ferrara, A., Lorusso, D., Montanelli, S.: On the Ontology Instance Matching Problem. In: DEXA Workshops 2008, pp. 180–184. IEEE Computer Society (2008)
19. Tang, J., Li, J., Liang, B., Huang, X., Li, Y., Wang, K.: Using Bayesian Decision for Ontology Alignment. *J. Web Semantics* 4(4), 243–262 (2006)
20. Jean-Mary, Y.R., Kabuka, M.R.: ASMOV: Ontology Alignment with Semantic Validation. In: Joint SWDB-ODDIS Workshop, Vienna, Austria, pp. 15–20 (September 2007)
21. Stoermer, H., Rassadko, N., Vaidya, N.: Feature-Based Entity Matching: The FBEM Model, Implementation, Evaluation. In: Pernici, B. (ed.) CAiSE 2010. LNCS, vol. 6051, pp. 180–193. Springer, Heidelberg (2010)
22. Nagy, M., Vargas-Vera, M., Stolarski, P.: DSSim Results for OAEI. In: OM (2009)
23. Hanif, M.S., Aono, M.: MPEG-7 Based Multimedia Information Integration through Instance Matching. In: ICSC, pp. 618–623 (2009)
24. Engmann, D., Maßmann, S.: Instance Matching with COMA++. In: BTW Workshops, pp. 28–37 (2007)
25. Huber, J., Szytler, T., Noessner, J., Meilicke, C.: CODI: Combinatorial Optimization for Data Integration – Results for OAEI 2011. In: OM (2011)
26. Castano, S., Ferrara, A., Lorusso, D., Montanelli, S.: The HMatch 2.0 Suite for Ontology Matchmaking. In SWAP (2007)
27. Hu, W., Chen, J., Cheng, G., Qu, Y.: ObjectCoref & Falcon-AO: Results for OAEI 2010. In: OM (2010)
28. Araujo, S., Hidders, J., Schwabe, D., de Vries, A.P.: SERIMI – Resource Description Similarity, RDF Instance Matching and Interlinking. *CoRR*, Vol. abs/1107.1104 (2011).
29. Cruz, I.F., Antonelli, F.P., Stroe, C.: Efficient Selection of Mappings and Automatic Quality-driven Combination of Matching Methods. In OM (2009)
30. Niu, X., Rong, S., Zhang, Y., Wang, H.: Zhishi.links Results for OAEI 2011. In: OM (2011)
31. Sais, F., Niraula, N., Pernelle, N., Rousset, M.-C.: LN2R – A knowledge based reference reconciliation system: OAEI 2010 Results. In: OM (2010)
32. Jena Semantic Web Framework for Java, <http://jena.sourceforge.net/>
33. Kalaitzaki, M.: Design and implementation of a system for semantic schema matching. Master Thesis – University of Crete, Department of Computer Science (November 2009)
34. Manakanatas, D.: Design and Implementation of a tool for semi-automated semantic schema matching. Master Thesis – University of Crete, Department of Computer Science (April 2006)
35. Doerr, M., Papagelis, M.: A Method for Estimating the Precision of Placename Matching. *IEEE Trans. Knowl. Data Eng.*, 1089–1101 (2007)
36. The ISLab Instance Matching Benchmark, <http://islab.dico.unimi.it/iimb/>
37. Svatek, V., Svab-Zamazal, O.: Empirical knowledge discovery over ontology matching results. In: Proc. 1st ESWC International Workshop on Inductive Reasoning and Machine Learning on the Semantic Web, Heraklion, Greece (2009)
38. Euzenat, J., Ferrara, A., Hollink, L., Isaac, A., Joslyn, C., Malaisé, V., Meilicke, C., Nikolov, A., Pane, J., Sabou, M., Scharffe, F., Shvaiko, P., Spiliopoulos, V., Stuckenschmidt, H., Sváb-Zamazal, O., Svátek, V., Santos, C.T.D., Vouros, G.A., Wang, S.: Results of the Ontology Alignment Evaluation Initiative 2009. In: OM (2009)
39. Hanif, M.S., Aono, M.: Anchor-Flood: Results for OAEI 2009. In: OM (2009)
40. Jean-Mary, Y.R., Shironoshita, M., Kabuka, M.R.: ASMOV: Results for OAEI 2009. In: OM (2009)

41. Nagy, M., Vargas-Vera, M., Stolarski, P.: DSSim Results for OAEI. In: OM (2009)
42. Stoermer, H., Rassadko, N.: Results of OKKAM Feature based Entity Matching Algorithm for Instance Matching Contest of OAEI, In: OM (2009)
43. Zhang, X., Zhong, Q., Shi, F., Li, J., Tang, J.: RiMOM Results for OAEI, In: OM (2009)
44. Flouris, G., Manakanatas, D., Kondylakis, H., Plexousakis, D., Antoniou, G.: Ontology Change: Classification and Survey. Knowledge Engineering Review (KER) 23(2), 117–152 (2008)
45. Euzenat, J., Ferrara, A., Hollink, L., Isaac, A., Joslyn, C., Malaisé, V., Meilicke, C., Nikolov, A., Pane, J., Sabou, M., Scharffe, F., Shvaiko, P., Spiliopoulos, V., Stuckenschmidt, H., Sváb-Zamazal, O., Svátek, V., Santos, C.T.D., Vouros, G.A., Wang, S.: Results of the Ontology Alignment Evaluation Initiative. In: OM (2009)

# SCIMS: A Social Context Information Management System for Socially-Aware Applications\*

Muhammad Ashad Kabir, Jun Han, Jian Yu, and Alan Colman

Faculty of Information and Communication Technologies,  
Swinburne University of Technology,  
Melbourne, Australia  
{akabir, jhan, jianyu, acolman}@swin.edu.au

**Abstract.** Social Context Information has been used with encouraging results in developing socially-aware applications in different domains. However, users' social information is distributed over the web and managed by many different proprietary applications, which is a challenge for application developers as they must collect information from different sources and wade through a lot of irrelevant information to obtain the social context information of interest. Combining the social information from the diverse sources and incorporating richer semantics could greatly assist the developers and enrich the applications.

In this paper, we introduce *SCIMS*, a social context information management system. It includes the ability to acquire raw social data from multiple sources; an ontology based model for classifying, inferring and storing social context information, in particular, social relationships and status; an ontology based policy model and language for owners to control access to their information; a query interface for accessing and utilizing social context information. We evaluate the performance and scalability of *SCIMS* using real data from Facebook, LinkedIn, Twitter and Google calendar, and demonstrate its applicability through a socially-aware phone call application.

**Keywords:** Social context, Online Social Networks, Social relationship, Ontology, Access control, Information management.

## 1 Introduction

Recently we have witnessed an increasing number of efforts aimed at providing socially-aware applications from such domains as pervasive computing, semantic web and information retrieval. These applications exploit users' social context information such as relationship, interaction history, and so on, to provide "smart" services and capabilities [1]. For example, SmartObject considers the user's relationship information to turn on the audio player when friends are present [2].

---

\* This research was supported in part by the Commonwealth of Australia, through the Cooperative Research Centre for Advanced Automotive Technology (AutoCRC).

Review quality is quantified based on the interaction and social relationships of reviewers in [5].

On the other hand, the popularity of Online Social Networks (OSNs) such as Facebook, LinkedIn, Twitter and Google+ (being the prime examples) produce an unprecedented amount of Social Context Information (SCI) as people specify their relationships, update their status and share contents. This phenomenon offers a unique opportunity for this information to be leveraged in creating more intriguing socially-aware applications. As it is, however, the users social context information is distributed all over the web, emerging from and fragmented across many different proprietary applications. Combining this information from such diverse sources could provide a more accurate representation of the users' social world with semantically richer information, and enable a whole new set of socially-aware applications.

Most of the existing works collect and manage social information within the context of an application (as in the above examples), which has two major problems. First, it is a challenge for application developers as they must collect information from different sources and wade through a lot of irrelevant information to obtain the social context information of interest to the targeted functionality. Second, it makes it extremely hard for information owners to control how their information should be exposed to different users and applications. While early context-aware applications relied on ad hoc architectures and representations, it has already been recognized that separating the process of acquiring contextual information from actual applications is key to facilitating application development and maintenance [7]. Nevertheless, managing users' social context information for supporting the development of socially-aware applications is still a challenging task for several reasons:

- **Consistent Representation of SCI:** There are different types of social context information. One type is the “object-centric” relationship, identified between people who have shown common interest (e.g., like/tag in Facebook) or participated in common activities or become members of similar groups. This type of relationship has been used in applications to infer preferences [8] and incentives of resource sharing [3]. Another type is the “people-centric” relationship, which is formal and declarative. For example, a person identifies other persons as father, supervisor, school friend, etc. This type of relationship can be used in a socially-aware phone call application as identified in [9]. The different types of social relationships need to be represented consistently to facilitate application use.
- **Inferring Social Relationships with SCI Semantics:** An application may need social context information that is not directly available from the sources but can be derived from basic information. For instance, users may want to filter phone call based on relationship categories such as “family” and “best-friends” that are not provided directly but can be inferred from other available relationships. Thus, there is the need to define and obtain derived relationships (at different abstraction levels) based on the basic relationships

(e.g., mother and school-friend) and their semantics (e.g., mother being in family) and attributes (e.g., strength and trust).

- **Preserving Owner Privacy by Fine-Grained Access Control over SCI:** The user’s social context information is inherently sensitive and can be further used to infer sensitive information. The scenarios of emerging socially-aware applications require users to share their information for greater benefits but may also compromise their privacy. For example, allowing caller to know the status of callee before calling might reduce interruptions [9], but may also raise serious concerns regarding the privacy and access control over users’ status and other data. Thus, users should be able to retain control over who has access to their personal information under which conditions. In addition, a user may want to *fine-tune* the *granularity* of the answer provided to a given query, depending on the context of that query such as *who* is asking, *what* is asked for, user’s *current status*, etc. For instance, an employee may be happy that her boss knows her current status is “AtDesk”. However, she may not be happy to let the boss know her status is “chatting to friends via instant messaging”. Thus users should be able to control access to their information at different *levels of granularity*.

A number of existing works have attempted to gather and manage users’ social data. However, they address the above issues in a very limited manner. For instance, PocketSocial [10], Prometheus [11] and MobiSoC [12] each gather a particular type of relationship information but without considering its semantics, and as a consequence are not able to answer the relationship at different levels of abstraction. While Yarta [13] does consider semantics and uses ontology in their relationship representation, it does not consider the granularity in information access in their policy model.

In this paper we introduce *SCIMS*, a social context information management system for collecting, integrating, classifying, inferring and storing social context information from diverse sources. *SCIMS* allows efficient access to this information while respecting user privacy, in order to facilitate the development of socially-aware applications. This research has three major contributions. First, we propose an ontology based model for representing and storing both *people-* and *object-centric* social relationships, and users’ status information. Second, a rich set of social context information has been derived based on information acquired from disparate sources. Third, we propose a way to preserving owners’ privacy by allowing the owners to fine-tune the granularity of information access and to specify access control policies. In addition, we have developed a number of adapters to fetch information from various sources and implemented a set of query APIs for applications to access the context information. We have used the semantic web technologies to implement the overall system and also performed evaluations of *SCIMS*’s performance and scalability. A socially-aware phone call application has been developed on top of *SCIMS* to demonstrate its applicability.

The paper is organized as follows. Section 2 reviews related research. An overview of the proposed *SCIMS* is given in section 3. Section 4 introduces our approach to modeling and inferring social context information. An access control

model to protect context information is described in section 5. Section 6 presents a prototype implementation. Section 7 reports our experimental evaluation with a case study, while section 8 concludes the paper and highlights future work.

## 2 Related Work

In the field of general context-aware system, there has been significant amount of research effort for modeling and managing context of a physical nature such as location, time, activity, and so on. Comparatively, there have been only limited works concerning contexts of a social nature [2]. Some research efforts have attempted to adopt and extend the Friend of a Friend (FOAF) [4] ontology for representing social relationships [6][14]. They extend the *foaf:knows* object property, the only option offered by FOAF ontology, with sub-properties such as *colleagueOf*, and *friendOf*. However, representing relationships using such object property suffers from generality for two main reasons: (i) it does not allow the specification of different attributes such as strength and trust associated with a relationship; (ii) as a consequence more abstract and rich context information cannot be derived.

OSNs like Facebook, LinkedIn and so on, offer their native APIs [15][16] for accessing their simple, unprocessed social data. These APIs do not provide access to derived relationships like “best friend” or “Colleague of a Colleague”. Instead, an application must explicitly crawl through the graph to obtain them.

Policy-based access control has been the subject of extensive research over the past decade. Recent research efforts have tried to integrate semantic technologies both in access control model and policy specifications, thus enabling automated reasoning and policy enforcement over expressive access control specifications. In particular, the ROWLBAC [17] and ReBAC [18] models have provided us with useful insight for our socially-aware access control framework. Most policy models for social networking applications [19], and even those designed for pervasive computing applications [20] consider either *role* or *relationship* in their access control but not both. While the work [21] is very close to ours in protecting user social context information, it does not consider *role* in their access control model and therefore is not able to offer the advantages of role based access control. Moreover, they do not consider the obfuscation to support the access control at different granularity level which is an important aspect for access control [28].

Socially-aware applications are often designed from scratch by embedding all management functionalities into the application logic, providing an application-specific data representation models, and acquiring data from one or a few specific external sources (e.g., [2], [23]). In all of these cases, social knowledge has been mined in the context of a single application. Some efforts have already recognized the need to externalize the social context management functionalities and have taken steps towards systematically managing users’ social context information.

Prometheus [11] collects user’s social data from different OSNs represents it as multi-edged graphs, where vertices correspond to users and edges correspond to interactions between users. The interactions are described with a label



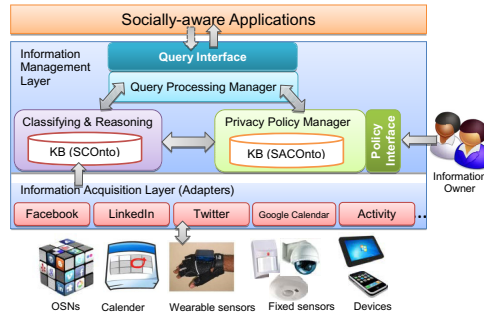


Fig. 1. Social context information management system architecture

(e.g., “football”, “music”) and a weight specifies the intensity of an interaction, and essentially represents an “object-centric” relationship. Prometheus implements a set of inference functions to answer queries like social strength, proximity, and so on, while enforcing user-defined access control policies. Like Prometheus, PocketSocial [10] also collects social data from different sources. But unlike Prometheus, it represents social data in JSON objects and supports only REST based APIs like Facebook, and does not provide any inference functions. Neither Prometheus nor PocketSocial represent both the *object-* and *people-centric* relationships with their semantics, and as a consequence they are not able to infer richer information or fine-tune the granularity of information access.

On the other hand, Yarta [13] adopts the FOAF ontology in their relationship representation, which has the drawbacks as discussed above. However, it only considers *people-centric* relationships and does not capture the *object-centric* ones. Even though it does consider social context in its policy model, its access control policy does not take into account granularity in information access. Moreover, its policy model is RDF-based which lacks in generality of OWL DL-based models.

Our work significantly differs from the previously noted approaches in that it not only collects users’ social relationship information from multiple sources and stores it in ontologies, but also considers the owners’ status information and their semantics, allowing information representation and derivation at different levels of abstraction and consequently facilitating access control and query processing.

### 3 Overview of the SCI Management System

We introduce an SCI management system, SCIMS, to address the challenges in developing socially-aware applications, as discussed in the introduction. Our proposed architecture for the SCIMS (see Figure 1) comprises two layers: (i) Information Acquisition and (ii) Information Management, and provides a platform for accumulating, storing and managing social context information.

The *information acquisition* layer is responsible for accumulating social context information from various sources such as OSNs, calendar entries, physical sensors, and so on. A common interface is provided so that different adapters can be built based on that interface to collect data from various sources.

For the *information management* layer, we propose an ontology based context model to store the processed social data being collected. The ontology based model provides the facilities of inferring and deducing more complex context information based on the processed data. We also propose a socially aware access control mechanism to allow information owners to protect their information by specifying privacy policy. Our policy model reflects the human way of thinking by considering social relationship, social role and status information when defining privacy preferences. Moreover, to allow different applications to access the social context information, we introduce a query interface so that application developers can build applications without the need to deal with the details of information representation schema and management. In the sections below, we introduce the detailed capabilities of the components in this SCIMS architecture.

## 4 Modeling and Inferring Social Context Information

We adopt an ontology based approach to modeling and representing social context information. Ontology based approaches have been evaluated as most promising for context modeling in pervasive computing. It has two main advantages: (i) it creates a common knowledge and understanding of context information, and (ii) it allows context reasoning [24]. We use OWL 2 DL [30], a sub-language of OWL 2, to model social context information. To date, OWL 2 DL has been the most practical choice for most ontological applications as it supports maximum expressiveness while retaining computational completeness and decidability [25].

Despite its intuitive anchoring in everyday life, social relationships or context information may be very challenging to represent in a formal model. However, at a certain level of abstraction, social relationships can be defined as a possible form of “relational ties”, as most of the existing works have considered (e.g., [13]). We also take this view to model social relationships. However, we model relationships as *first-class entities* rather than representing them as a generic *link* between people. This way of modeling allows us to benefit from Description Logic (DL) [26] in *classifying* and *reasoning* about relationships at different *levels of abstraction* based on the properties of these relationships.

In order to provide a suitable trade-off between formal modeling and application specific concepts, we define a set of basic concepts as building blocks for the creation of socially-aware applications. Towards this goal, we propose both upper and domain-specific (lower) ontologies. The *upper-level* ontology captures the basic concepts, abstracted from the analysis of: (i) real-world use case scenarios, like the socially-aware phone call [6]; (ii) different sources of context information, e.g., Facebook, LinkedIn, etc. Through a standard specification, this upper ontology can be shared, reused, and adapted to others systems. This

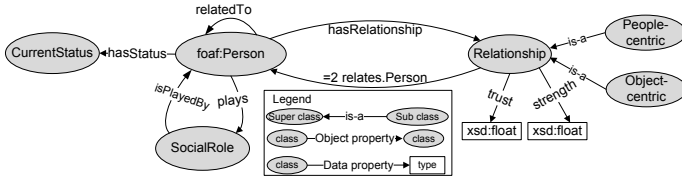


Fig. 2. Social context upper ontology – core classes and properties

can be customized to represent terms in different forms for different users. The upper ontology is also extensible to allow for the incorporation of new concepts and the specialization of concepts and constraints for a particular application, which we refer as *domain-specific* ontologies.

### 4.1 Core Upper Ontology

Our upper ontology model, called Social Context Ontology (*SCOnto*), defines four first-class entities, namely, *Person*, *SocialRole*, *Relationship*, and *CurrentStatus*. These entities can be organized into a hierarchy where the root of the hierarchy is the term *SCOnto*. To define *Person*, we adopt and extend the FOAF ontology. In the FOAF ontology, a person’s status is represented as literal. To support reasoning over a person’s status at different granularity levels we change the type of “status” from data property to object property (labeled “hasStatus”) which links the *Person* and *CurrentStatus* class. The FOAF model also includes a set of attributes that provide information about a person, such as *name*, *email*, *phone-number*, *gender* and so on. In real-world, a person may have different types of relationship as she plays various roles. To identify a person at such a fine grained level, we introduce the concept of *SocialRole* which *isPlayedBy* the *Person*. The core concept of our model is the (social) *Relationship* which is a subclass of *SCOnto* and *relates* exactly two persons that can be defined in OWL 2 using the DL [25] syntax as follows:

$$Relationship \sqsubseteq SCOnto \sqcap = 2 \text{ relates. } Person$$

where those persons should be different individuals (i.e., the *relatedTo* object property type should be set to *Irreflexive*).

Figure 2 shows the *SCOnto* model that can be read as follows. A *Person* may have *Relationship*(s); each *Relationship* can be classified as *object-* and *people-centric*, *relates* exactly two *Persons*; thus through a *Relationship* a *Person* is *relatedTo* another *Person* and *plays* a particular *SocialRole*; also a *Person* may have a *CurrentStatus*. The set of concepts included in our ontology is obviously non-exhaustive. However, we believe that this ontology can be profitably used to model many application scenarios.

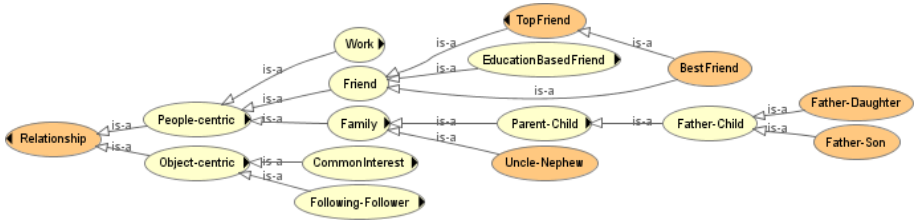


Fig. 3. Domain specific ontologies – an excerpt of different types of relationship

### 4.2 Domain-Specific Ontologies

Based on the core ontology we define some domain-specific ontologies (Figure 3). In particular, we propose a fine-grained relationship model for *Family*, *EducationBasedFriend*, *Work*, and *CommonInterest* relationships. Basically, we rely on users’ Facebook information to deduce *family* relationships. Facebook allows people to maintain a family list, where a user can annotate a person in his family list as *Brother*, *Father*, *Uncle* and so on, which ultimately indicates the role played by the person being annotated from the user’s point of view rather than the specific relationship between the user and that person. For instance, if a user annotates a person in his/her family list as “Father”, the relationship between them could be a “Father-Son” or “Father-Daughter” relationship, which can be inferred by utilizing the user’s gender information. In our ontology, we use DL for defining such derived relationships. Some examples are shown below:

$$\begin{aligned}
 \text{FatherSon} &\sqsubseteq \text{FatherChild} \sqcap \exists \text{relates}((\text{Person} \sqcap \exists \text{plays} \cdot \text{Father}) \sqcap (\text{Me} \sqcap \text{gender} \cdot \text{male})) \\
 \text{UncleNephew} &\sqsubseteq \text{Family} \sqcap \exists \text{relates}((\text{Person} \sqcap \exists \text{plays} \cdot \text{Uncle}) \sqcap (\text{Me} \sqcap \text{gender} \cdot \text{male}))
 \end{aligned}$$

Here, *Me* is a subclass of *Person* that specifies the user from whose perspective the relationship is computed. However, in some cases, we can deduce the relationship directly from the role name. For example, if a person is annotated as “husband”, we can directly deduce the relationship as “husband-wife”; the same applies for *cousin* and *partner* relationships.

Facebook is also a potential source of collecting *friend* information. A person is an education-based-friend if he/she studied with the user, so we categorize *EducationBasedFriend* as *PrimarySchoolFriend*, *HighSchoolFriend*, *CollegeFriend*, and *GraduateSchoolFriend*. Such basic friendship information can be used to further deduce and classify friendship relationships such as *TopFriend*, *BestFriend* and so on. For instance, a user can define that a person is a top friend if he/she is a *HighSchoolFriend*, *CollegeFriend* and *GraduateSchoolFriend*, as follows:

$$\text{TopFriend} \equiv \text{HighSchoolFriend} \sqcap \text{CollegeFriend} \sqcap \text{GraduateSchoolFriend}$$

Also, a person can be defined as a best friend if he/she is a top friend and has relationship *strength* greater than a certain value (e.g., 0.8)

$$\text{BestFriend} \equiv \text{TopFriend} \sqcap \geq 0.8 \text{ strength}$$

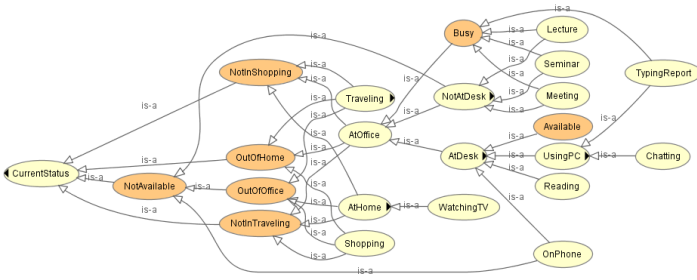


Fig. 4. An excerpt of current status ontology

Table 1. Granularity levels of classes in the *CurrentStatus* ontology

Granularity Level	Classes
1	NotAvailable, Available
2	Busy
3	OutOfHome, OutOfOffice, NotInTraveling, NotInShopping
4	Shopping, Traveling, AtOffice, AtHome
5	NotAtDesk, AtDesk, WatchingTV
6	Lecture, Meeting, Seminar, UsingPC, OnPhone
7	TypingReport, Chatting

This relationship *strength* information can be computed based on the user’s interaction activities in OSNs [26].

We classify the *work* relationship for an educational organization as *Student-Teacher*, *Student-Supervisor*, and *Colleague* relationships. From LinkedIn, we can get current and past positions of a user and all the persons connected with that user. Based on the institution *name* or *id* and the *positions* held by them, we can deduce their work relationships. For instance, if two persons hold any of the staff roles in an organization, we can deduce their relationship as being *Colleague*. Based on the specific types of staff roles, we can further classify the relationship as: *AcademicStaff-AdminStaff*, *AcademicStaff-AcademicStaff*, *AdminStaff-AdminStaff*.

From Twitter we can obtain *Following-Follower* relationship, where *Following* includes the person(s) whom the user is interested and following, and *Follower* includes person(s) who are following the user. *LivingAddress* based relationship can be acquired from Facebook and further categorized at *Country* and *City* levels. *CommonInterest* includes the relationships with person(s) who like the same *Digital-Content* (e.g., those people use “like” to express in Facebook), are members of the same *Group* (e.g., soccer, cricket, and so on), are interested in the same *Event* (e.g., conference), or have *CommonResearchInterest* which can be collected from LinkedIn.

Figure 4 shows part of the *CurrentStatus* ontology by considering some activities (both social activities (more than one person is involved) and individual

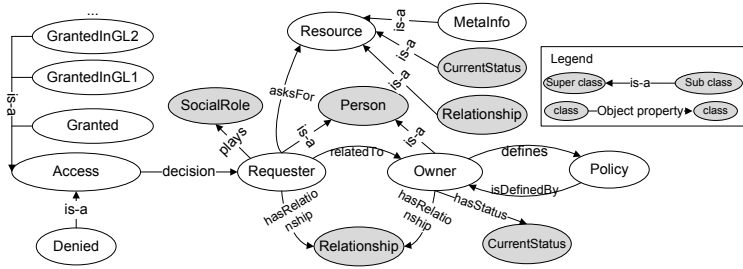


Fig. 5. Access control policy model

activities) in particular domains such as home, office, shopping and travel. Again, the set of information defined in this ontology is obviously non-exhaustive and can be further extended based on the need of the application or domain of interest. We assume that we can collect a user’s raw status information from the user’s diary, Facebook, LinkedIn, or Twitter status update. The user’s raw status information will be one of the leaf nodes in the ontology. We can deduce the status of being *Available* or *NotAvailable* for telephone call, or *Busy* based on the raw information, as follows:

$$\begin{aligned}
 \textit{Busy} &\equiv \textit{Lecture} \sqcup \textit{Meeting} \sqcup \textit{Seminar} \sqcup \textit{TypingReport} \\
 \textit{Available} &\equiv \textit{AtDesk} \sqcap \neg \textit{OnPhone} \\
 \textit{NotAvailable} &\equiv \textit{OnPhone} \sqcup \textit{NotAtDesk} \sqcup \textit{OutOfOffice}
 \end{aligned}$$

We have set the *granularity level* for each node/class in the ontology as an increasing number from root to leaf nodes, as depicted in Table 1. Thus for a given *status*, we can answer the *current status* of a user at different levels of granularity. For instance, for a status “Meeting” with granularity level 4, we can say that the user is “AtOffice”. Similarly, we also specify the granularity level for relationship ontologies.

## 5 Preserving Privacy

### 5.1 A Socially-Aware Access Control Policy Model

For controlling access to SCI, we have proposed an access control policy model to protect owners’ social context information based on the context information itself. Thus our policy model, called Socially-aware Access Control Ontology (*SACOnto*) (see Figure 5), reuses the concepts from core *SCOnto* model (shown in shaded ellipse) to define under which conditions (i.e., the social context) a given resource is accessible. The accessible resource could be any social context information (e.g., relationship, status) including the meta-information related to any relationship of a user such as the *numbers* of friends and colleagues.

Studies have revealed that users want to define policies that apply to all people with certain relationships or roles rather than explicitly name each person [20,

21]. Thus in access conditions, we consider both requester’s role and relationships with information owner to reflect the way users tend to group similar sets of people when deciding to share resources to other users. Therefore, our model is able to represent existing policy models, such as popular role-based and recently emerged relationship-based [16, 17], with enhanced expressive capabilities.

In addition, we consider the owner’s current status information which is another important aspect of defining policies [20]. Furthermore, our model can be easily extended to incorporate users’ physical context information such as *location*, *time*, and so on, in making access decision. For the sake of brevity, however, we do not consider it here. Users typically specify their policies using combinations of the main dimensions driving access control decisions. For example,

- Who is requesting access and what is her relationship with the owner?
- What role is the requester playing?
- The current status of the owner when the request comes?
- What type of resource is being requested and what are its characteristics?

Our ontological model enables a user (owner) to specify logical relations between the above fundamental elements. In the owners’ conceptual model, such who/what/when dimensions are typically interrelated. Our model (Figure 5) captures these dimensions which can be read as follows: A *Policy isDefinedBy* an *Owner* which specifies *Access decision* (*Denied* or *Granted* or *GrantedInGL1* (granted in granularity level 1) and so on) for *Requester(s)* who plays a *SocialRole*, *relatedTo Owner* and *hasRelationship* with owner named *Relationship*, *asksFor a Resource*, when the *Owner hasStatus CurrentStatus*. For instance, consider this policy, “Any of my colleagues can access my current status when I am in meeting” (policy #1). In such case, the owner’s access decision is based on the relationship with the requester (who), the resource being accessed (what), and the status of the owner at request time (when). While traditional approaches generally consider these dimensions orthogonal, our model reflects the owner’s way of thinking by supporting the cross dimensional definition of policies.

Another key feature of our model is the ability to specify access permissions at different levels of granularity, i.e., disclosing information at a certain level of abstraction by hiding the specific fact. For instance, “Upon being asked the current status by supervisor, a student may want to state her status as being ‘AtDesk’ (granularity 5) while she is actually chatting with friends” (policy#2).

## 5.2 Policy Specification and Enforcement

We use DL to specify access control policies, as one of its main advantages is that a DL reasoner can be used for automatic inconsistency detection both in policy specification and enforcement. A graphical user interface can be provided to the owner for specifying her privacy policies which can be easily transformed to the DL format. The template of our policy rule in the DL is defined as follows:

$$\begin{aligned} Denied / Granted / GrantedInGL?n \sqsubseteq Access \sqcap \exists decision.(Requester \sqcap \\ \exists hasRelationship.?Relationship \sqcap \exists asksFor.?Resource \sqcap \\ \exists plays.?SocialRole \sqcap \exists relatedTo.(Owner \sqcap \exists hasStatus.?CurrentStatus)) \end{aligned}$$

where bold words with a preceding ‘?’ mark are variables that will be filled based on the owner’s privacy preference statements. For example, we can represent the above policy #2 as follows:

$$\begin{aligned} \text{GrantedInGL5} \sqsubseteq & \text{Access} \sqcap \exists \text{decision}(\text{Requester} \sqcap \exists \text{asksFor}.\mathbf{CurrentStatus} \sqcap \\ & \exists \text{hasRelationship}.\mathbf{Student-Supervisor} \sqcap \\ & \exists \text{relatedTo}.\mathbf{(Owner} \sqcap \exists \text{hasStatus}.\mathbf{Chatting})) \end{aligned}$$

Once the policy is transformed to the DL format, it is asserted to the owner’s access control ontology. For instance, policy #2 will be asserted to the *SACOnto* as an equivalent class of *GrantedInGL5* class. When an access request or query comes which basically corresponds to invoking a function from the access APIs (see next section), some facts about the context of the query are temporarily asserted to the ontology – namely the requester *name* and *resource* being requested. In addition, an individual of *Access* class with a link to the requester using *decision* property is asserted. After that the reasoner is fired which classifies that individual of access class to one of its subclasses, i.e., Denied, GrantedInGL1, and so on, based on the defined policies. Query processing manager considers this result for answering the query.

## 6 Prototype Implementation

We have implemented a prototype system for SCI management, SCIMS, in Java 2 Platform Standard Edition (J2SE). For managing the knowledge base (*SCOnto* and *SACOnto*), we have used OWL API 3. We wrote *adapters* for Facebook, LinkedIn, Twitter, and Google calendar using their native APIs for fetching users’ social data, and follow the OAuth 2.0 protocol for authorization. For inferring and policy execution, we have used reasoners compliant with OWL 2 DL. In particular, for evaluation purposes, we have incorporated five different DL reasoners: Pellet 2.3.0, Hermit 1.3.5, TrOWL 0.8.1, Fact++ 1.5.2 and RacePro 2.0.

To aid developing socially-aware applications, we have implemented a set of APIs exposed as Web services. *Policy APIs* allow owners to add, update and delete their privacy preferences – *addPRule*, *deletePRule*, *getPRule*, *updatePRule*, and so on. *Management APIs* provide functionality (i) to manipulate ontologies, allowing inserting, editing and deleting both concepts and individuals, and (ii) to configure and execute information acquisition operations – *insertConcept*, *insertRelationship*, *updateCurrentStatus*, *startInfoAcqFromFB*, and so on. *Query APIs* for accessing both context and meta-context information – *getCurrentStatus*, *whatIsMyRelWithB*, *getRelNameByGLevel*, *isAHasRelRwithB*, *getNumberOfGraduateSchoolFriend*, *getColleagueOfAColleague*, and so on. Since most of the interface names are self explanatory, we do not provide description of those APIs. Also due to page limit, we cannot give the complete lists of the all types of APIs.

We adopt a DL based query language, named SPARQL-DL [28], for query processing and have used the derive 1.0 SPARQL-DL query engine with above mentioned DL reasoners. Recently, SPARQL-DL was introduced as a rich query



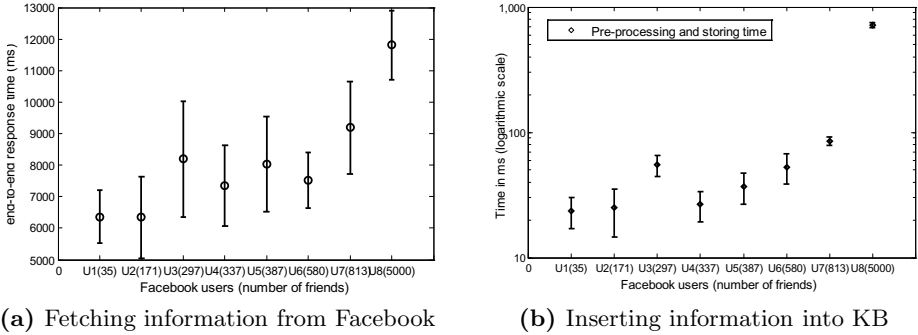


Fig. 6. Time of acquiring social context information from Facebook

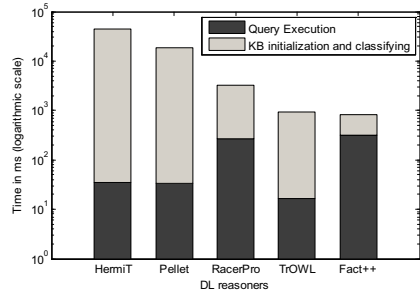
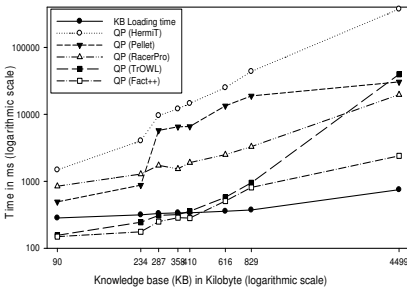
language for OWL 2, which is a distinct subset of SPARQL (a RDF based language), tailored to ontology specific queries. Therefore, different aspects of context such as granularity can be answered easily and in simpler fashion. For example, `getStatusByGLevel(A,gLevel)` can be realized in SPARQL-DL as follows:

```
PREFIX sc:<http://www.ict.swin.edu.au/Ontology/SCOnto#>
SELECT ?status {
  PropertyValue(A, sc:hasStatus, ?statIns)
  DirectType(?statusName, ?statIns)
  Annotation(strictSubClassOf(?statusName, ?status),
    sc:granularityLevel, gLevel)}
```

## 7 Experimental Evaluation and Case Study

The management system prototype is deployed and evaluated on a machine with Core 2 Duo E8400 3 GHz processor, 3 GB RAM, WinXP professional edition SP3, and Java 1.6. Our evaluation had three goals: (1) assess the systems scalability in acquiring social information using different sizes of real social data of multiple users; (2) measure the systems performance in executing a set of queries over a large data set using different existing reasoners; (3) validate the system by developing a case study social application on mobile phones.

**Scalability Evaluation for Information Acquisition.** To assess the scalability of the system regarding *information acquisition*, we evaluate the time cost of acquiring information using two metrics: (i) end-to-end response time – time required to fetch information from different sources – Facebook, LinkedIn, Twitter, and Google calendar – to quantify the network overhead; (ii) Preprocessing and storing time – time required to extract the data of interest from fetched objects, transform it to the suitable format, and insert it into the knowledge base (ontologies), to quantify the performance of inserting information into SCIMS.



(a) Loading and QP time over various size of KB using five different reasoners

(b) Details of QP time for a particular Knowledge-Base sized 829 kilobytes

Fig. 7. System performance on query processing (QP)

In these experiments, we choose eight different Facebook users who have increasing numbers of friends, including a user with 5000 friends which is the highest number Facebook allows. These users gave us permission to collect their social data. For each user, we run the experiment 50 times. Figure 6 shows the result where error bars depict standard deviation. For the highest number of friends (i.e., 5000), the average end-to-end response time was 12 sec (see Figure 6a) and the average time for preprocessing and inserting information into knowledge base (KB) was around 700 ms (see Figure 6b), which is an acceptable result, since this information acquisition is usually performed off-line and might not be required to update very frequently. Similarly, we have done experiments using LinkedIn, Google calendar and Twitter users; however, due to page limit those results are not included here.

**Performance Evaluation for Query Processing (QP).** For measuring the system performance on *query processing*, we use three metrics: (i) time required to load KB, (ii) time for classifying KB, and (iii) time to answer a set of queries – `isAFamilyOfB`, `getAllRelationships` and `getStatusByGLevel`. The KB that has been built in previous experiments using users’ social context information acquired from Facebook, LinkedIn, Twitter and Google calendar, is used for performance measurement. Figure 7a shows the result (average of 50 runs) of loading time which basically depends on the size of the knowledge base and not associated with any reasoners. Figure 7a also shows the result of query processing time (the sum of KB classification and query execution time) over various size of KB using five different reasoners. The result of QP shows that for a small sized KB, TrOWL performs very well. But as the size increases the computation cost increases dramatically. However, Fact++ outperforms consistently from small to large sized KBs. Figure 7b shows further details of query processing performance (separating KB classification and query execution) of different reasoners for a particular sized KB (i.e., 829 kilobytes – contains 953 relationship instances).

**Case Study – A Socially-Aware Phone Call Application.** To demonstrate the real-world applicability of our approach, we have developed a proof-of-concept application, called socially-aware phone call<sup>1</sup>. This application leverages the social context information in SCIMS to decide whether to ring, vibrate, reject, or reject and send status when a call comes. Thus for each incoming call, the application invokes `getMyCurrentStatus` and `whatIsMyRelwithB` functions to get the current status and the relationship information with the caller, respectively, and acts based on the preference defined by the callee. On the other hand, before calling, using the same application one can also obtain the status of the intended callee by invoking `getCurrentStatusOfB` function to judge the suitable time of making the call.

The application is written in Java for mobile devices running Android OS and was tested on LG Optimus One which communicates with SCIMS over a 3G network. We have tested meeting and seminar scenarios during a month where a user defines her *filtering preferences* using the application as “(1) If my status is *meeting* or *seminar* and a call comes from *friend*, action is reject; (2) For *family*, action is reject and forward my status at granularity level 2 (Busy); (3) for *colleague* action is reject and forward status at granularity level 6 (i.e., meeting or seminar); and (4) for *supervisor* action is vibrate”. To forward the status information of the callee (for case 2 and 3), application invokes the `getStatusByGLevel` function with particular granularity level as an attribute value (as specified in the filtering conditions). We also use policy #1 as user’s *privacy preference* regarding access to her status information by caller (to judge the suitable time of making the call) that SCIMS collects from user’s Google calendar. We observe that the application acts well in its real-time constraint, i.e., the application is able to acquire information from SCIMS and make decision before the call is forwarded to the voice-mail of the callee.

## 8 Conclusions and Future Work

In this paper, we have presented a novel social context information management system, SCIMS, to aid the development of complex socially-aware applications. Our system utilizes existing OSNs to accumulate user’s social context information; provides rich semantic support for representing and inferring SCI, particularly social relationships and status; offers flexible policies for controlling access to SCI based on owners’ preferences; provides a generic query interface for the use of the SCI. In particular, our semantic based approach to modeling and managing SCI provides the advantages of reusability and extensibility. The SCIMS implements a set of adapters to retrieve social data from different OSNs, stores information in an ontology-based knowledge base, and provides a number of interfaces for accessing and managing this information. We have demonstrated the efficacy and usability of the proposed system by conducting a performance and scalability evaluation and by developing a socially-aware phone call application running on Android phones.

<sup>1</sup> <http://www.ict.swin.edu.au/personal/akabir/sphone/spcall.htm>

We are currently extending the SCI management system along a number of directions, including support of continuous query over SCI and system realization in different platform settings (P2P, Cloud based and Mobile).

## References

1. Kabir, M.A., Han, J., Colman, A.: Modeling and Coordinating Social Interactions in Pervasive Environments. In: Proc. of the ICECCS, pp. 243–252 (2011)
2. Biamino, G.: Modeling social contexts for pervasive computing environments. In: PerCom CoMoRea Workshop, pp. 415–420 (2011)
3. Li, J., Dabek, F.: F2F: Reliable storage in open networks. In: P2P Workshop (2006)
4. Friend of a Friend (FOAF), <http://xmlns.com/foaf/spec/>
5. Lu, Y., Tsaparas, P., Ntoulas, A., Polanyi, L.: Exploiting social context for review quality prediction. In: Proc. of the 19th Intl. Conf. on WWW, pp. 691–700 (2010)
6. RELATIONSHIP, <http://vocab.org/relationship>
7. Dey, A.K., Abowd, G.D., Salber, D.: A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Hum.-Comput. Interact.* 16(2) (2001)
8. Mislove, A., Gummadi, K.P., Druschel, P.: Exploiting social networks for internet search. In: Proc. 5th Workshop on Hot Topics in Networks, pp. 79–84 (2006)
9. Toninelli, A., Khushraj, D., Lassila, O., Montanari, R.: Towards socially aware mobile phones. In: 1st Social Data on the Web Wrokshop. CEUR, Germany (2008)
10. Bo, X., Gronowski, K., Radia, N., Svensson, M., Ton, A.: PocketSocial: Your distributed social context now in your pocket. In: PerCom Workshops (2011)
11. Kourtellis, N., Finnis, J., Anderson, P., Blackburn, J., Borcea, C., Iamnitchi, A.: Prometheus: User-Controlled P2P Social Data Management for Socially-Aware Applications. In: Gupta, I., Mascolo, C. (eds.) *Middleware 2010*. LNCS, vol. 6452, pp. 212–231. Springer, Heidelberg (2010)
12. Gupta, A., Kalra, A., Boston, D., Borcea, C.: MobiSoC: a middleware for mobile social computing applications. *Mobile Netw. and Appl.* 14(1), 35–52 (2009)
13. Toninelli, A., Pathak, A., Issarny, V.: Yarta: A Middleware for Managing Mobile Social Ecosystems. In: Riekkki, J., Ylianttila, M., Guo, M. (eds.) *GPC 2011*. LNCS, vol. 6646, pp. 209–220. Springer, Heidelberg (2011)
14. Devlic, A., et al.: Context inference of users' social relationships and distributed policy management. In: PerCom Workshops, pp. 1–8 (2009)
15. Graph API – Facebook developers, <http://developers.facebook.com/>
16. LinkedIn APIs, <https://developer.linkedin.com/apis>
17. Finin, T., et al.: ROWLBAC: representing role based access control in OWL. In: Proc. of the 13th SACMAT, pp. 73–82 (2008)
18. Fong, P.W.L., Siahaan, I.: Relationship-based access control policies and their policy languages. In: Proc. of the 16th SACMAT, pp. 51–60. ACM (2011)
19. Carminati, B., Ferrari, E., Heatherly, R., Kantarcioglu, M., Thuraisingham, B.: A semantic web based framework for social network access control. In: Proc. of the 14th SACMAT, pp. 177–186 (2009)
20. Jagtap, P., Joshi, A., Finin, T., Zavala, L.: Preserving Privacy in Context-Aware Systems. In: Proc. of 5th Intl. Conf. on Semantic Computing, pp. 149–153 (2011)
21. Toninelli, A., Montanari, R., Lassila, O., Khushraj, D.: What's on Users' Minds? Toward a Usable Smart Phone Security Model. *Pervasive Computing* 8(2) (2009)

22. Khalil, A., Connelly, K.: Context-aware telephony: privacy preferences and sharing patterns. In: Proc. of the 20th CSCW Conf., pp. 469–478 (2006)
23. Beach, A., et al.: Fusing mobile, sensor, and social data to fully enable context-aware computing. In: Proc. of the Workshop on Mob. Comput. Sys. & Appl. (2010)
24. Bettini, C., et al.: A survey of context modelling and reasoning techniques. *Pervasive Mob. Comput.* 6(2), 161–180 (2010)
25. Riboni, D., Bettini, C.: OWL 2 modeling and reasoning with complex human activities. *Pervasive Mob. Comput.* 7(3), 379–395 (2011)
26. Baader, F.: *The description logic handbook: theory, implementation, and applications*. Cambridge Univ. Press (2003)
27. Xiang, R., Neville, J., Rogati, M.: Modeling relationship strength in online social networks. In: Proc. of the Intl. Conf. on World Wide Web, pp. 981–990 (2010)
28. Wishart, R., Henricksen, K., Indulska, J.: Context Privacy and Obfuscation Supported by Dynamic Context Source Discovery and Processing in a Context Management System. In: Proc. of the UIC, pp. 929–940 (2007)
29. Sirin, E., Parsia, B.: Sparql-dl: Sparql query for owl-dl. In: OWLED (2007)
30. OWL 2 Web Ontology Language, <http://www.w3.org/TR/owl2-overview/>

# Ontological Meta-properties of Derived Object Types

Giancarlo Guizzardi

Ontology and Conceptual Modeling Research Group (NEMO)  
Computer Science Department,  
Federal University of Espírito Santo (UFES), Brazil  
gguizzardi@inf.ufes.br

**Abstract.** In this paper, we revisit a number of classical formal meta-properties that have been used in the conceptual modeling and ontology engineering literature to provide finer-grained distinctions among the category of Object Types. These distinctions constitute an essential part of relevant existing approaches, in particular, the ontology-driven conceptual modeling language OntoUML, and the ontology and taxonomy evaluation methodology OntoClean. The idea in this paper is to investigate the interaction between these meta-properties and Derived Object Types, i.e., Object Types which extensions are dynamically inferred via Derivation Rules. The contributions here are two-fold: firstly, we revisit two classical Derivation Patterns and prove a number of results that can be used to infer the modal meta-properties of Derived Types from those of the types participating in the associated derivation rules; secondly, we demonstrate how these results can be applied in the automated support for model construction in OntoUML.

**Keywords:** Ontological Foundations, Meta-Properties, Derived Object Types.

## 1 Introduction

In recent years, there has been a growing interest in the use of Ontologically Well-Founded Conceptual Modeling languages to support the domain analysis phase in Information Systems Engineering. OntoUML is an example of a conceptual modeling language whose metamodel has been designed to comply with the ontological distinctions and axiomatic theories put forth by a theoretically well-grounded Foundational Ontology [1]. This language has been successfully employed in a number of industrial projects in several different domains, ranging from Petroleum and Gas [2] to News Information Management [3]. In fact, recently, it has been considered as a possible candidate for contributing to the OMG SIMF (Semantic Information Model Federation) standardization *request for proposal* [4] after a significant number of successful applications in real-world engineering settings [5,6].

One of the striking features of OntoUML is the fact that its modeling primitives are defined in terms of ontological meta-properties that not only give a precise semantics to the language's primitives but also serve as a methodological guideline for helping modelers to choose the most suitable category to model elements from the universe of

discourse [7]. OntoUML addresses all the classical primitives used in structural conceptual modeling, including Classes, Relations, Weak Entities, Attributes, Attribute Value Spaces (Datatypes) and different types of Part-Whole Relations. However, in this paper we will focus only on Object Types and their taxonomic relations, which are certainly among the most fundamental constructs in structural conceptual models.

In [1], we have shown that traditional conceptual modeling and knowledge representation languages (e.g., ER, UML, OWL) present a clear case of ontological incompleteness w.r.t. the representation of Object Types. In these languages, there is one single concept of object type and, hence, one single type of classification relation. In contrast, as demonstrated in [1,7], from an ontological and cognitive point of view, there are number of different categories of object types implying different types of classification relations among object types and their instances. As discussed in depth in [1,7], these categories provide a formally characterized, ontologically grounded and empirically supported version of modeling primitives which have been used and defined in an *ad hoc* manner in the history of conceptual modeling (e.g., Kind, Role, Mixin, Phase or State, etc.).

Among the most important meta-properties used to define the aforementioned distinctions, we have the meta-properties which have a modal nature, in particular, the so-called meta-properties of *Rigidity* and *Non-Rigidity* (including *Semi-Rigidity* and *Anti-Rigidity*). These meta-properties have also been used in the methodological guidelines of one of the most important and successfully employed Ontology Evaluation Methodologies, namely OntoClean [8,9] and have also been incorporated into ORM 2.0 [10].

In recent papers, we have demonstrated that the formal nature of these meta-properties, and the manner in which they constrain how the OntoUML modeling primitives can be combined, account for an important mechanism for building automated tools for formal verification [11] and conceptual model validation via visual simulation [12]. Moreover, as shown in [13], the constrained manner in which these primitives can be combined makes them converge to a number of *Ontological Design Patterns* and, hence, OntoUML can be characterized as a Pattern Language. In the same article, we present a computational tool that takes advantage of this characteristic of the language to help modelers to build ontologically sound conceptual models by assembling these models via the chained combination of these patterns.

However, up to this point, we have only analyzed the so-called *Base Types*, i.e., types which are explicitly defined in a conceptual model and which instance populations have to be explicitly asserted. In contrast, *Derived Types* are types which are derived from other (Base and Derived) Types via derivation rules. Again, in contrast to Base Types, the populations of Derived Types are dynamically inferred via their associated derivation rules. In [14], Olive discusses in depth the importance of derived types for conceptual modeling and presents a number of patterns for derived types with associated formal derivation rules.

In this paper, we take a step further in developing theoretical results for the foundational of conceptual modeling and ontology engineering, in general, and OntoUML and OntoClean (but also to some extent ORM), in particular. Firstly, we analyze how these modal meta-properties of Object Types interact with patterns for Derived types as defined in [14]. Moreover, we show how the modal properties of derived types can be inferred from those of the other types participating in the associated derivation rules (section 3). In particular, in the case of OntoUML, we

manage to show how the particular stereotypes (representing ontological categories) of these derived types can be inferred from those of the types participating in the derivation rules (section 4).

In section 2, we first present the background information for this paper. Firstly, we present two of Olive’s patterns for Derived Object types. Moreover, we present the modal meta-properties of Object Types (Rigidity, Non-Rigidity, Anti-Rigidity and Semi-Rigidity) underlying the theories of OntoUML and OntoClean, and briefly present the modeling primitives of OntoUML which are generated from the variations of these meta-properties.

Section 5 elaborates on final considerations and directions for future work.

## 2 Background: Derived Types, Ontological Modal Meta-properties and Object Type Distinctions in OntoUML

### 2.1 Derived Types and Derivation Patterns

Olive [14] presents and formally characterizes some properties of a number of categories of Derived Object types with their associated derivation rules. These include *Derivation by Participation*, *Derivation by Intersection*, *Derivation by Union* and *Derivation by Exclusion*. These different patterns are associated with different types of derivation rules which imply different types of constraints. Here, due to space limitations, we focus on *Derivation by Union* and *Derivation by Exclusion*.

**Derivation by Union:** The pattern of Derivation by Union takes place when we have a type T whose extension is necessarily defined as a union of the extension of a number of other types  $T_1 \dots T_n$ . As shown in [14], a Derivation by Union constraint implies a number of specialization relations between  $T_1 \dots T_n$  and the derived common supertype T (symbolized as /T, following UML), such that  $T_1 \dots T_n$  and T form together a complete *generalization set (GS)*. For instance, in figure 1.b, the type Student is defined by union of types Graduate Student and Undergraduate Student. In other words, all instances of Student are instances of either Graduate Student or Undergraduate Student. Formally, let W be a set of possible worlds and let  $ext(T,w)$  be function that maps a generic type T to its extension in world  $w \in W$ . In this case can state that: for all  $w \in W$ , we have that  $ext(Student,w) = ext(UndergraduateStudent,w) \cup ext(GraduateStudent,w)$ . In general, we can present the following formal definition:

**Definition 1 (Derivation by Union):** if the type T is derived by union from types  $T_1 \dots T_n$  such that T is a common supertype of  $T_1 \dots T_n$  forming a generalization set then for all worlds  $w \in W$ , we have that  $ext(T,w) = ext(T_1,w) \cup \dots \cup ext(T_n,w)$ . ■



Fig. 1. Pattern for *Derivation by Union* (a-left) and its exemplification (b)



The generalization set formed in the *derivation by union* pattern must be complete. However, in general, the types  $T_1 \dots T_n$  with a common supertype  $T$  do not have to be disjoint. However, as discussed by many authors in areas ranging from Conceptual modeling, Logic and Philosophy [15], defining generalization sets which are both disjoint and complete (i.e., partitions) of a common supertype is a design principle which should be pursued in taxonomic structures accounting for ontological, logical and methodological advantages. For this reason, when referring to the *derivation by union* pattern in this article, we mean a *derivation by disjoint union* pattern.

**Derivation by Exclusion:** The pattern of Derivation by Exclusion takes place when we have a type  $T_x$  whose extension is necessarily defined as the complement of the extension a number of other types  $T_1 \dots T_n$  w.r.t. a type  $T$ . As shown in [14], a Derivation by Exclusion constraint implies a number of specialization relations between  $T_1 \dots T_x \dots T_n$  and the common supertype  $T$ , forming a *disjoint* and *complete generalization set*. For instance, in figure 2.b, the type Deceased Person is defined by exclusion as the complement of type Living Person w.r.t. the type Person. In other words, all instances of Deceased Person are instances of Person which are not Living Persons. Formally, in this case can state that: for all  $w \in W$ , we have that  $ext(DeceasedPerson, w) = ext(Person, w) - ext(LivingPerson, w)$ . In general, we define the following constraint for this type of Derivation.

**Definition 2 (Derivation by Exclusion):** if the type  $T_x$  is derived by exclusion from types  $T_1 \dots T_n$  w.r.t. their common supertype  $T$  forming a Generalization Set then for all worlds  $w \in W$ , we have that  $ext(T_x, w) = ext(T, w) - (ext(T_1, w) \dots \cup \dots \cup ext(T_n, w))$ . ■



Fig. 2. Pattern for *Derivation by Exclusion* (a-left) and its exemplification (b)

## 2.2 The Ontological Meta-properties among the Categories of Object Types and Associations

OntoUML has been proposed as an extension of UML that incorporates in the UML 2.0 original metamodel a number of ontological distinctions and axioms put forth by the Unified Foundation Ontology (UFO) [1]. In this work, we focus our discussion on a small fragment of this metamodel. This fragment discusses an extension of the *Class* meta-construct in UML to capture a number of ontological distinctions among the Object Types categories proposed in UFO. Besides incorporating modeling primitives that represent these ontological distinctions, the extended OntoUML metamodel includes a number of logical constraints that govern how these primitives can be combined to form consistent conceptual models. In what follows, we briefly elaborate on the aforementioned distinctions and their related constraints for the fragment discussed in this article. For a fuller discussion and formal characterization of the language, the reader is referred to [1].

A fundamental distinction in OntoUML among the category of object types is the one between types which provide a uniform principle of identity, individuation and counting for their instances - termed *Sortal* types, and types which merely define characterizing properties for instances of multiple *Sortal* types – termed *Characterizing* or *Dispersive* types. For instance, contrast the Sortal types *Person* and *Car*, with the characterizing types *Insured Item*, *Colored Object* or *Spatially Extended Object*. Whilst the first categories of types provide a uniform principle of identity for their instances, the latter category of types merely define properties which are common to instances of different sortal types. For example, the type *Spatially Extended Object* describes properties which are common to all its instances but it cannot provide a uniform principle of identity for them. In other words, different instances of Spatially Extended Object (e.g., Cars, Houses, Persons, Watches, etc.) will obey different principles of identity which are provided by the respective sortal types they instantiate. As discussed in depth in [7], this distinction is one of the most supported ones in Philosophy of Language and there is a significant amount of empirical evidence in Cognitive Psychology supporting the claim that it is a fundamental cognitive distinction.

Orthogonally to this basis distinction, we can make another distinction among Object Types by considering a formal modal meta-property named *Rigidity* (henceforth symbolized  $R+$ ) [7,9]. In short, a type  $T$  is rigid iff for every instance  $x$  of that type,  $x$  is necessarily an instance of that type. Formally, we have that:

**Definition 3 (Rigidity  $R+$ ):** a type  $T$  is rigid if every instance of that type is necessarily (in a modal sense) an instance of that type. In other words, the extension of type  $T$  is world invariant, i.e., instances of  $T$  in any world  $w$  are its instances in all possible worlds. Hence, for all  $w, w' \in W$  we have that  $\text{ext}(T, w) = \text{ext}(T, w')$ . ■

In contrast, a type  $T'$  is anti-rigid (henceforth symbolized  $R\sim$ ) iff for every instance  $y$  of  $T'$ , there is always a possible world in which  $y$  is not an instance of  $T'$ . In other words, it is possible for every instance  $y$  of  $T'$  to cease to be so without ceasing to exist (without losing its identity) [7]. Formally,

**Definition 4 (Anti-rigidity  $R\sim$ ):** a type  $T$  is anti-rigid if every instance of that type can possibly (in a modal sense) cease to be an instance of that type. In other words, for all worlds  $w \in W$  and for all instances  $x$  such that  $x \in \text{ext}(T, w)$  we have that there is a world  $w' \in W$  such that  $x \notin \text{ext}(T, w')$ . ■

A stereotypical example of this distinction can be found by contrasting the rigid type *Person* with the anti-rigid type *Student*. Whilst every instance of *Person* cannot cease to be an instance of *Person* without ceasing to exist, instances of *Student* can move in and out the extension of that class and still exist preserving its identity, i.e., being the same *Person* [7]. One should notice, however, that Anti-Rigidity is not the logical negation of Rigidity, it is stronger than that. The logical negation of Rigidity is termed Non-Rigidity (henceforth symbolized  $R-$ ) and is defined as follows:

**Definition 5 (Non-rigidity  $R-$ ):** a type  $T$  is non-rigid if there is an instance of that type which can possibly (in a modal sense) cease to be an instance of that type. In other words, there is a world  $w \in W$  and an instance  $x$  such that  $x \in \text{ext}(T, w)$  such that for another world  $w' \in W$  we have that  $x \notin \text{ext}(T, w')$ . ■

From the above definition, it is clear that Anti-Rigidity implies Non-Rigidity, i.e., Anti-Rigidity is a strong case of Non-Rigidity in which the contingent classification in a type holds for all instances of that type  $T$ . A complement of Anti-Rigidity w.r.t. Non-Rigidity is named Semi-Rigidity (henceforth symbolized  $R\rightarrow$ ), which is a meta-properties much more common than Non-Rigidity in conceptual Models. Semi-Rigidity of type  $T$  means that the classification in  $T$  is necessary for some of its instances and contingent for others. In other words,  $T$  is *rigid for some of its subtypes and anti-rigid for others* [7,9]. Formally, we have that:

**Definition 6 (Semi-Rigidity  $R\rightarrow$ ):** a type  $T$  is semi-rigid if it is Non-Rigid but not Anti-Rigid. ■

Suppose a conceptualization in which People and Houses can optionally have an Insurance Policy but Cars are obliged to have one. In other words, instances of Person and House are contingently *Insured Items* but Cars are necessarily Insured Items. In this case, Insured Item will be an example of a Semi-Rigid type.

Sortal Rigid types can be related in a chain of taxonomic relations. For instance, the rigid types Man and Person are related such that the former is a specialization of the latter. The type in the root of a chain of specializations among sortal rigid types is termed a Kind (e.g., Person) and the remaining rigid types in this chain are named Subkinds (e.g., Man, Woman). As formally demonstrated in [1], we have the following constraints involving Kinds and Subkinds: (i) *every object in a conceptual model must be an instance of exactly one kind, which means that kinds cannot specialize other kinds*; (ii) *as consequence, we have that for every object type  $T$  in OntoUML,  $T$  is either a kind or it is the specialization of exactly one ultimate kind. In other words, all sortals in a conceptual model which are not kinds must specialize one unique kind*; (iii) *by definition, subkinds cannot specialize kinds*.

Among the anti-rigid Sortal types, we have again two subcategories: Phases and Roles. In both cases, we have cases of dynamic classification, i.e., the instances can move in and out of the extension of these types without any effect on their identity. However, while in the case of Phase these changes occur due to a change in the intrinsic properties of these instances, in the cases of Role, they occur due to a change in their relational properties. We contrast the types Child, Adolescent, Adult as phases of a Person with the Roles Student, Husband or Wife. In the former case, it is a change in the intrinsic property *age* of Person which causes an instance to move in and out of the extension of these phases. In contrast, a Student is a role that a Person plays when related to an Education Institution and it is the establishment (or termination) of this relation that alters the instantiation relation between an instance of Person and the type Student. Analogously, a Husband is a role played by a Person when married to (a Person playing the role of) Wife. *As formally proved in [7], rigid types cannot specialize (cannot be subtypes of) anti-rigid types*.

Sortal types are always Rigid or Anti-Rigid. In contrast, among the categories of Characterizing types, we have both Rigid, Anti-Rigid and Semi-Rigid types. Rigid Characterizing types describe essential (i.e., modally necessary) properties which are common to instances of different kinds. These are named *Category*. An example of a Category is Physical Object. Every Physical Object is necessarily a Physical Object. However, there is no uniform principle of identity obeyed by all types of Physical

Objects (e.g., Cars, Persons, Houses, Oranges). An example of an Anti-Rigid Characterizing type is Customer: every customer can cease to be a Customer. However, different instances of Customer can obey different principles of identity belonging to different kinds (e.g., People and Organizations). These sorts of characterizing types are named *RoleMixins* in OntoUML. Finally, we have already encountered an example of a Semi-Rigid Characterizing type in this article, namely, Insured Item. On one hand, it is a characterizing type classifying instances of different kinds. On the other hand, it is necessary for some of its instances (e.g., the instances of the subtype Car) and contingent to others (e.g., the instances of the subtype Person). These characterizing types are termed *Mixin* in OntoUML. *As also formally proved in [7], characterizing types cannot specialize (be subtypes) or sortals.* This is due to the fact that principles of identity are inherited in a specialization chain. Thus, if a characterizing type (e.g., Red Object) specializes a Sortal (e.g., Apple), it inherits the principle of identity of latter and becomes a Sortal (e.g., Red Apple).

### 3 Deriving Modal Meta-properties of Derived Object Types

In this section, we demonstrate how the modal meta-properties of derived types for two of the Derivation Patterns proposed in [14] can be inferred from the modal meta-properties of the types participating in the derivation. Firstly, we refrain from using directly the ontological categories of OntoUML so that the results can be made more general, i.e. can be re-used in other conceptual modeling and ontology engineering methodologies that make use of the same notions such as, for instance, OntoClean.

#### 3.1 Derivation by Union

Let us start with the case of *derivation by union* (exemplified in the pattern of figure 1.a). Let us first suppose that B in that figure is a rigid type. In the sequel we show that the rigidity value of A can be directly derived from the one of C. Let us begin by demonstrating that if C is a rigid type then so is A.

**Proof<sub>1</sub>:** Because the generalization set (GS) is complete, we have that  $\text{ext}(A,w) = \text{ext}(B,w) \cup \text{ext}(C,w)$  and that  $\text{ext}(B,w) \cap \text{ext}(C,w) = \emptyset$  for all  $w \in W$ . Thus, suppose an instance  $x$  of  $A$ . We have that  $x \in \text{ext}(A,w)$  and either  $x \in \text{ext}(B,w)$  or  $x \in \text{ext}(C,w)$ . Suppose that  $x \in \text{ext}(B,w)$  then the only manner that  $x$  can cease to be an instance of  $A$ , i.e.,  $x \notin \text{ext}(A,w')$  (for a  $w' \neq w$ ) is if  $x$  ceases to be an instance of B ( $x \notin \text{ext}(B,w')$ ). Now, since B is rigid then it is necessarily the case that  $x \in \text{ext}(B,w')$  (*mutatis mutandis* the same can be shown if  $x \in \text{ext}(C,w)$ ).  $\square$

Now, suppose that C is not rigid, i.e., that C is non-rigid (either anti-rigid or semi-rigid). In this case, we can show that A is necessarily semi-rigid.

**Proof<sub>2</sub>:** Because the GS is complete, we have that  $\text{ext}(A,w) = \text{ext}(B,w) \cup \text{ext}(C,w)$  and that  $\text{ext}(B,w) \cap \text{ext}(C,w) = \emptyset$  for all  $w \in W$ . Since C is non-rigid then there is an  $x$  such that  $x \in \text{ext}(C,w)$  and  $x \notin \text{ext}(C,w')$  for some arbitrary  $w, w' \in W$ . Because C is a subtype of  $A$  then we have that  $x \in \text{ext}(A,w)$ . Now, suppose that  $A$  is rigid. In this case, we have also that  $x \in \text{ext}(A,w')$ . Since GS is complete then either

$x \in \text{ext}(B,w')$  or  $x \in \text{ext}(C,w')$ . Since the latter is not the case, we must have that  $x \in \text{ext}(B,w')$ . However, since  $B$  is rigid then we also have that  $x \in \text{ext}(B,w)$  which is a contradiction since  $\text{ext}(B,w) \cap \text{ext}(C,w) = \emptyset$  for all  $w$ . We must then conclude that  $/A$  is necessarily non-rigid, which means that  $/A$  is either anti-rigid or semi-rigid;  $/A$  cannot be anti-rigid since  $B$  is rigid and an anti-rigid type cannot be a supertype of a rigid type, ergo, we have that  $/A$  is semi-rigid.  $\square$

The two patterns arising from the proofs 1 and 2 above are depicted in figures 3.a and 3.b below, respectively.

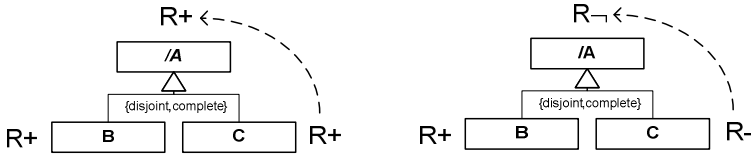


Fig. 3. Patterns for inferring the rigidity value of type  $/A$  following proofs 1 (a-left) and 2 (b)

Let us suppose now that  $B$  is anti-rigid. In this case, if  $C$  is rigid then we fall in the pattern of figure 3.b with classes  $B$  and  $C$  inverted (anti-rigidity is a special case of non-rigidity). If  $C$  instead is anti-rigid then  $A$  can be either rigid or anti-rigid. Suppose that  $/A$  is rigid then it means that its extension is invariant. Since  $B$  is anti-rigid then all instances it “acquires” or “loses” must come or go to the complement of  $B$  w.r.t.  $/A$ , namely,  $C$ . Finally, we know that  $/A$  cannot be semi-rigid. *By absurdum*, suppose the contrary, i.e., that  $/A$  is semi-rigid. In this case, by definition,  $/A$  must have a subtype which is rigid. Since neither  $B$  nor  $C$  are rigid then there must exist a rigid type  $D$  (which is a subtype of either  $B$  or  $C$ ). However, a rigid type cannot be a subtype of anti-rigid one. Therefore, we conclude that  $/A$  cannot be semi-rigid (**proof 3**).  $\square$

Finally, suppose that  $B$  is anti-rigid and  $C$  is semi-rigid. In this case,  $/A$  can be either rigid or itself semi-rigid. We can show, however, that  $/A$  cannot be anti-rigid: suppose that  $/A$  is anti-rigid. Since  $C$  is semi-rigid then there must be at least one rigid subtype  $D$  of  $C$ . Now, in this case,  $D$  would also be a subtype of  $/A$ . But since a rigid type cannot be a subtype of an anti-rigid one, we must conclude that  $/A$  cannot be semi-rigid (**proof 4**).  $\square$

The two patterns arising from the proofs 3 and 4 above are depicted in figures 4.a and 4.b below, respectively.

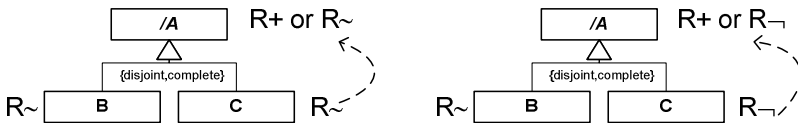
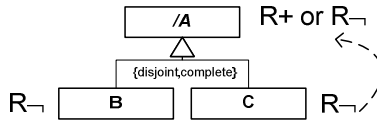


Fig. 4. Patterns for inferring the rigidity value of type  $/A$  following proofs 3 (a-left) and 4 (b)

Let us suppose now that B is semi-rigid. In this case, if C is rigid then we fall in the pattern of figure 3.b with classes B and C inverted (semi-rigidity is a special case of non-rigidity). If C is instead anti-rigid then we fall in the pattern of figure 4.b with classes B and C inverted. Finally, if C is semi-rigid then the only possibility we can rule out is /A being anti-rigid for the reason already discussed when arguing for pattern of figure 4.b (*proof 5*). □



**Fig. 5.** Pattern for inferring the rigidity value of type /A following proof 5

Table 1 below summarizes the results of this section. Table 2 presents a more compact version with the same information.

**Table 1.** Inferred rigidity value of derived type /A from the different rigidity values of types B and C (painted rows mark cases in which the value of /A is completely determined)

<b>B</b>	<b>C</b>	<b>/A (derived)</b>
R+	R+	R+
R+	R~	R¬
R+	R¬	R¬
R~	R+	R¬
R~	R~	R+, R~
R~	R¬	R+, R¬
R¬	R+	R¬
R¬	R~	R+, R¬
R¬	R¬	R+, R¬

**Table 2.** Summarized version of Table 1

<b>B</b>	<b>C</b>	<b>/A (derived)</b>
R+ (i.e., for whatever rigid type)	R+	R+
R+	R- (i.e., for whatever non-rigid type)	R¬
R¬ (i.e., for whatever semi-rigid type)	R-	R+, R¬
R~ (i.e., for whatever anti-rigid type)	R~	R+, R~

### 3.2 Derivation by Exclusion

We now analyze the case of *derivation by exclusion*, i.e., we analyze the case of the derived type /C (following the pattern of figure 2.a) such that the instances of /C are

exactly the instances of A which are not instances of B. Once more, here we can show that the rigidity value of /C can be derived from the ones of B and A.

We start by fixing the rigidity value of B as a rigid type. Firstly, let us show that if A is rigid then so is /C (figure 6.a).

**Proof<sub>6</sub>:** If A and B are rigid then theirs extension is invariant, i.e., for all  $w, w' \in W$ ,  $ext(A, w) = ext(A, w')$  and  $ext(B, w) = ext(B, w')$ . Now, since  $ext(A, w) = ext(B, w) \cup ext(/C, w)$ , for all  $w \in W$ , we must conclude that the extension of C is also world invariant, i.e.,  $ext(/C, w) = ext(/C, w')$ , for all  $w, w'$ . Alternatively, we can assume that /C is non-rigid. If this is the case there are  $x, w, w'$  so that  $x \in ext(/C, w)$  and  $x \notin ext(/C, w')$ . But since /C is a subtype of A then we know that  $x \in ext(A, w)$  and  $x \in ext(A, w')$ . Since this generalization set is complete, we have that  $x \in ext(B, w')$ . Now, since B is rigid then it is also the case that  $x \in ext(B, w)$ . But having that x is an instance of both B and /C in w, and that B and /C are disjoint, we have a contradiction, thus, concluding that /C cannot be non-rigid, i.e., it must be rigid. □

Let us now take the case that A is semi-rigid (notice that A cannot be anti-rigid). In this case, /C must be non-rigid (figure 6.b).

**Proof<sub>7</sub>:** Suppose that /C is rigid then we have a pattern which is analogous to the one in figure 3.a. Notice that the only assumption we make in proof 1 is that the generalization set involving A, B and C is a disjoint and complete one. So, here again we have if B and C are rigid so is A which is a contradiction (our initial assumption is that A is semi-rigid). Thus, /C must be non-rigid. □

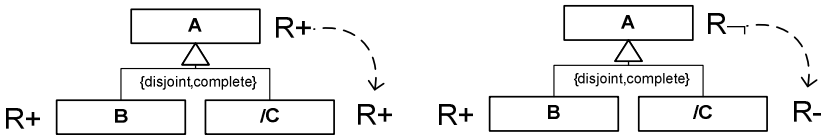


Fig. 6. Patterns for inferring the rigidity value of type /C following proofs 6 (a-left) and 7 (b)

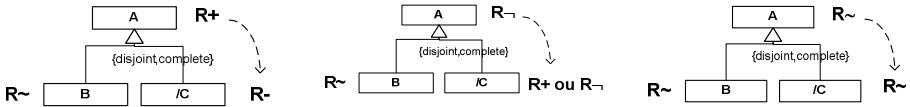
Let us now suppose that B is anti-rigid. Here, once more, the rigidity value of /C can be derived from the one of A: if A is rigid then /C must be non-rigid; if A is semi-rigid then /C must be either rigid or semi-rigid; if A is also anti-rigid then so is /C. In the latter case, however, the model must be deemed incomplete.

**Proof<sub>8</sub>:** Since B is anti-rigid then there is an x as well as  $w, w' \in W$  so that  $x \in ext(B, w)$  and  $x \notin ext(B, w')$ . Since B and /C are disjoint we know that  $x \notin ext(/C, w)$ , and since B is a subtype of A we know that  $x \in ext(A, w)$ . Now, since A is rigid then its extension is invariant and then we have that  $x \in ext(A, w')$ . Since this generalization set is complete, we have that  $ext(A, w') = ext(B, w') \cup ext(/C, w')$ . Given that  $x \notin ext(B, w')$  we must have that  $x \in ext(/C, w')$ . This demonstrates that /C's extension is not invariant and, thus, that it cannot be a rigid type. Ergo, /C is non-rigid. □

**Proof<sub>9</sub>:** If A is semi-rigid then (by definition) A must have as a subtype a rigid type. Since the Generalization set involving B and /C (and having A as common supertype) is complete then either B or one of its subtypes is rigid or /C or one of its subtypes is

rigid. B is anti-rigid and its subtypes cannot be rigid (an anti-rigid type cannot be a supertype of a rigid one). Thus, either /C is rigid or one its subtypes is rigid. If /C is not rigid then the one manner one of its subtypes can be rigid is if /C is semi-rigid. We conclude then that /C is either Rigid or semi-rigid.  $\square$

**Proof<sub>10</sub>:** If A is anti-rigid then /C cannot be rigid since an anti-rigid type cannot be a supertype of a rigid one. Moreover, /C cannot be a semi-rigid type because (by definition) /C would have as subtype a rigid type D. Now, in this case, D would be a (indirect) subtype of A and we would have an anti-rigid type as a supertype of a rigid one. Thus, we must conclude that C must be anti-rigid in this case.  $\square$

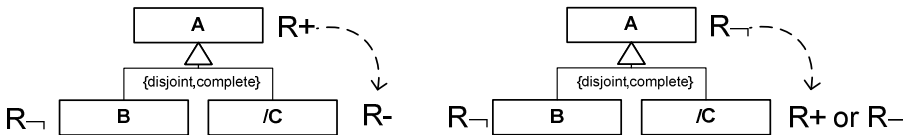


**Fig. 7.** Patterns for inferring the rigidity value of type /C following proofs 8 (a-left), 9 (b-center) and 10 (c-right)

Finally, let us now suppose that B is semi-rigid. Notice that, in this case, A cannot be anti-rigid. This is because by definition B must have a rigid subtype which would in turn be a (indirect) subtype of A. However, an anti-rigid type cannot be a supertype of a rigid one. If A is itself semi-rigid then there is nothing else at this point that we can infer regarding the rigidity value of /C, since it can be rigid, anti-rigid or semi-rigid (**proof 11**).  $\square$

Moreover, we can also prove now is that if A is rigid then /C must be non-rigid.

**Proof<sub>12</sub>:** Since B is semi-rigid then there is an  $x$  as well as  $w, w' \in W$  so that  $x \in \text{ext}(B, w)$  and  $x \notin \text{ext}(B, w')$ . Since B and /C are disjoint we know that  $x \notin \text{ext}(C, w)$ , and since B is a subtype of A we know that  $x \in \text{ext}(A, w)$ . Now, since A is rigid then its extension is invariant and then we have that  $x \in \text{ext}(A, w')$ . Since this generalization set is complete, we have that  $\text{ext}(A, w') = \text{ext}(B, w') \cup \text{ext}(C, w')$ . Given that  $x \notin \text{ext}(B, w')$  we must have that  $x \in \text{ext}(C, w')$ . This demonstrates that /C's extension is not invariant and, thus, that it cannot be a rigid type. Ergo, C is non-rigid.  $\square$



**Fig. 8.** Patterns for inferring the rigidity value of type /C following proofs 11 (a-left) and 12 (b)

The results of this section are summarized in table 3 below.



**Table 3.** Inferred rigidity value of the derived type /C from the different rigidity values of types B and A (painted rows mark cases in which the value of /C is completely determined)

	<b>B</b>	<b>A</b>	<b>/C (derived)</b>
1	R+	R+	R+
2	R+	R~	Logically Inconsistent
3	R+	R¬	R-
4	R~	R+	R-
5	R~	R~	R~
6	R~	R¬	R+, R¬
7	R¬	R+	R-
8	R¬	R~	Logically Inconsistent
9	R¬	R¬	R+, R-

#### 4 Deriving Ontological Categories of Object Types in OntoUML

In this section, by crossing the results of section 3 with the ontological categories underlying OntoUML, we demonstrate how the ontological category of Object Derived types in OntoUML can be inferred from the ontological categories of the types participating in the derivation.

##### 4.1 Derivation by Union

In table 4 below, we analyze the possibilities of ontological categories that can take the place of classes B, C and /A in the first row of table 2. In the sequel, we analyze the second row of table 2. For the sake of limitation, we will limit our analysis here to only these two cases.

**Table 4.** Inferred ontological category of derived type /A from those of types B and C

	<b>B</b>	<b>C</b>	<b>/A (derived)</b>
1	<b>Kind</b>	Kind	Category
2		Subkind	Category
3		Category	Category
4	<b>subkind</b>	Kind	Category
5		Category	Category
6		Subkind	kind, subkind, category
7	<b>category</b>	Kind	Category
8		Subkind	Category
9		Category	Category

As one can see, in eight out of nine valid possibilities for the configuration of types B and C we can directly infer the ontological category of /A. Since an object cannot be instance of more than one Kind, a Kind cannot have a supertype another Sortal.

Thus, whatever type which is a supertype of a Kind must be a Characterizing type. Since an anti-rigid type cannot be an instance of a rigid type, we have that a supertype of a Kind must either be a Category or a Mixin. Since in row 1 of table 2 the derived type is rigid, we conclude that the derived type /A must be a Category (rows 1-3 of table 4). For an analogous reason, in row 4, /A must be a Category (in table 4, row 4 is equivalent to row 2 with columns B and C inverted).

From the constraints that a characterizing type cannot be a subtype of a Sortal type, and that an anti-rigid type cannot be a supertype of rigid one, we have that a Category can only have as a supertype either a Category or a Mixin. From that, we conclude that in rows 5 but also 7-9 of table 4, the type /A must be a Category.

The only case that leaves open three possibilities for /A is the one in row (6) of this table. However, as previously discussed, a kind is a stereotypical case of a root *base type* since it is the type the supplies the principle of identities for its instances. In fact, the set of object instances of a model is exactly the union of the extension of all kinds [1]. As a consequence, the configuration in which /A is a kind *derived by union*, albeit logically consistent, is ontologically inconsistent. Thus, given that B and C are subkinds, the *derived by union* type /A can either be another subkind or a category. Notice that such a model is necessarily incomplete, i.e., the subkinds B and C must have as supertypes kinds  $B^K$  and  $C^K$  from which they should inherit their principle of identity. Now, the ontological category of /A can be determined by truth value of formula  $(B^K = C^K)$ . In other words, if  $(B^K = C^K)$  it means that /A is a subkind which is also a subtype of  $B^K$ (or  $C^K$ ); In contrast, if  $(B^K \neq C^K)$  then by definition /A is the union of entities that obey different principles of identity and is a category.

Let us now analyze the case of row (2) in table 2. Since the only ontological category which is semi-rigid is a *mixin* then we conclude that whatever combination we have of rigid type (kind, subkind or category) playing the role of B in figure 3.b with a non-rigid type (phase, role, roleMixin, phaseMixin, mixin) playing the role of C then the derived type /A (derived by union) must necessarily be a *mixin*.

These conclusions are summarized in table 5 below.

**Table 5.** A determinate version of Table 5 with additional constraints as model assumptions

	<b>B</b>	<b>C</b>	<b>/A (derived)</b>	<b>Constraints</b>
1	kind	R+	Category	
2	category	R+	Category	
3	subkind	subkind	Subkind	subtype(B, $B^K$ ), kind( $B^K$ ), subtype(C, $C^K$ ), kind( $C^K$ ) and ( $B^K = C^K$ )
4	subkind	subkind	Category	subtype(B, $B^K$ ), kind( $B^K$ ), subtype(C, $C^K$ ), kind( $C^K$ ) and ( $B^K \neq C^K$ )
5	R+	R-	Mixin	

As we have previously discussed, the results of table 5 can be directly implemented in an automated tool to assist the user in configuring generalization sets of derived types in ontologically correct design patterns. For instance, in figure 9, suppose the

modeler defines a common derived type (by union) MaleAnimal of subkinds MalePerson and MaleDog, then the tool could automatically infer that MaleAnimal is a Category (following row 4 of table 5). Moreover, if the modeler then defines the common derived type (by union) Animal, derived from MaleAnimal and FemaleAnimal, the tool could then automatically derive that Animal is a Category (not shown in the figure). Finally, if like in our previously mentioned example, we define InsuredItem as derived by union from the kind Car and the subtype of Person named InsuredPerson (a role), then the tool can derive that InsuredItem is a mixin (following row 5 of table 5).

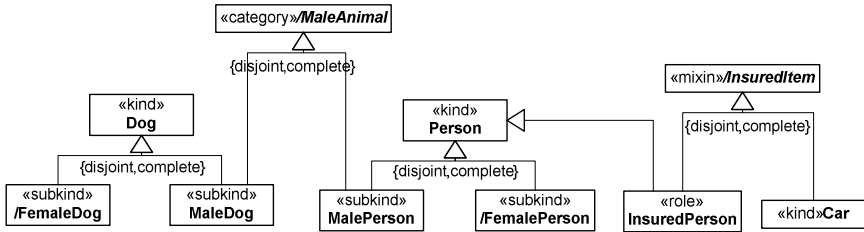


Fig. 9. Inferring the Ontological Categories of Derived Object types in OntoUML for the cases of Derivation by Union and Derivation by Exclusion

### 4.2 Derivation by Exclusion

Here, once more, due to the lack of space, we will focus only on the first row of table 3. In table 6 below, we analyze the possibilities of ontological categories that can take the place of classes A, B and /C in that row.

Table 6. Inferred ontological category of derived type /C from those of types A and B

	A	B	/C (derived)
1	kind	Subkind	Subkind
2	subkind	Subkind	Subkind
3	category	Kind	kind, subkind, category
4		Subkind	kind, subkind, category
5		Category	kind, subkind, category

As one can see, in the case that A is a kind, the only real possibility for B is to be a subkind since: a kind cannot be a supertype of another kind, and a kind cannot be a supertype a category. In that case, and for the same reasons, /C must also be a subkind. In a similar manner, if A is a subkind then B must also be a subkind, since: a subkind cannot be a supertype of a kind, and a subkind cannot be a supertype a category (a sortal cannot be a supertype of a characterizing type). For the same reasons, /C must be a subkind. An example of situation exemplifying row 1 of table 6 can be found in figure 9. If the modeler derives /FemalePerson by exclusion of subkind MalePerson w.r.t. to the kind Person then the tool can automatically infer that /FemalePerson must be a subkind.

In case A is a category, the situation is much less definitive. Since both B and /C could, in principle, assume any possibility among rigid types. In that case, we again would need to use information about other types in the model to draw a more conclusive inference. For instance, if we the modeler declares the model to be complete (an operation common to a particular mode of validation in OntoUML [11]), then we could infer that both B and /C would be kinds. This is because: every subkind in the model must be a subtype of a unique kind and every characterizing type in the model must be specialized (directly or indirectly) by a kind. In other words, if the model is assumed to be complete then neither B or /C could be either subkinds or categories.

## 5 Final Considerations and Future Work

In this paper, we have presented a theoretical contribution that can be of value both to traditional conceptual modeling and domain ontology engineering. On one side, it allows for employing ontological meta-properties which are well-known in the ontology engineering literature (namely, rigidity and its associated notions) to be used to define finer-grained distinctions among the categories of object types in conceptual modeling. On the other hand, it allows for employing patterns for object type derivation which are well-known in conceptual modeling to be used in the definition of taxonomic structures in ontology engineering.

In particular, the first contribution of this paper is to formally prove a number of patterns that demonstrate how the rigidity value of derived object types can be inferred from the rigidity value of other types participating in the associated derivation rules. As demonstrated in a number of works [7-9], the identification of the rigidity value of object types play an important role in evaluating the quality of taxonomic structures. These patterns, as presented in section 3, can be used in any approach in which these ontological meta-properties are recognized. Noticeable examples are OntoClean [8,9], ORM 2.0 [10] and OntoUML [1,7].

As a second contribution of this paper, we show the practical relevance of these inference patterns by demonstrating how they can be applied in the automated support for conceptual model construction in OntoUML. In section 4, we demonstrate how the ontological categories of derived object types in OntoUML can be inferred from the ontological categories of the types participating in the derivation.

Due to space limitations, we had to make a number of successive selections here. Firstly, we have only addressed two of Olive's derivation patterns. Secondly, in adapting the inference patterns of section 3 to OntoUML, we have merely addressed two cases for *Derivation by Union* and one case of *Derivation by Exclusion*. Finally, again due to space limitations, we refrained from showing that although exemplified by generalization sets with only two subtypes, the proofs presented here can be generalized to generalizations sets with any number of types. In an extended version of this paper, we intend to complete all the aforementioned gaps. Moreover, with regards to the adaptation of the full set of inference patterns to OntoUML, we intend to present there their implementation in the OntoUML editor [11].

**Acknowledgments.** This research is supported by FAPES (PRONEX grant #52272362) and CNPq (productivity grant #311578/2011-0).

## References

1. Guizzardi, G.: *Ontological Foundations for Structural Conceptual Models*. Universal Press, The Netherlands (2005) ISBN 1381-3617
2. Guizzardi, G., Lopes, M., Baião, F., Falbo, R.: On the importance of truly ontological representation languages. *International Journal of Information Systems Modeling and Design, IJISMD* (2010) ISSN: 1947-8186
3. Carolo, F., Burlamaqui, L.: Improving Web Content Management with Semantic Technologies. In: *Semantic Technology Conference (SemTech)*, San Francisco (2011)
4. Object Management Group, *Semantic Information Model Federation (SIMF): Candidates and Gaps*, <http://www.omgwiki.org/architecture-ecosystem/>
5. Bauman, B.T.: Prying Apart Semantics and Implementation: Generating XML Schemata directly from ontologically sound conceptual models. *Balisage Markup Conference* (2009)
6. U.S. Department of Defense, *Data Modeling Guide (DMG) for an Enterprise Logical Data Model (ELDM)*, [http://www.omgwiki.org/architecture-ecosystem/lib/exe/fetch.php?media=dmg\\_for\\_enterprise\\_ldm\\_v2\\_3.pdf](http://www.omgwiki.org/architecture-ecosystem/lib/exe/fetch.php?media=dmg_for_enterprise_ldm_v2_3.pdf)
7. Guizzardi, G., Wagner, G., Guarino, N., van Sinderen, M.: An Ontologically Well-Founded Profile for UML Conceptual Models. In: Persson, A., Stima, J. (eds.) *CAiSE 2004*. LNCS, vol. 3084, pp. 112–126. Springer, Heidelberg (2004)
8. Guarino, N., Welty, C.: Evaluating Ontological Decisions with OntoClean. *Communications of the ACM* 45(2), 61–65 (2002)
9. Guarino, N., Welty, C.: An Overview of OntoClean. In: Staab, S., Studer, R. (eds.) *Handbook on Ontologies*. Springer (2010)
10. Halpin, T., Morgan, T.: *Information Modeling and Relational Databases*. Morgan Kaufman (2008) ISBN 1558606726
11. Benevides, A.B., Guizzardi, G.: A Model-Based Tool for Conceptual Modeling and Domain Ontology Engineering in OntoUML. In: Filipe, J., Cordeiro, J. (eds.) *ICEIS 2009*. LNBIP, vol. 24, pp. 528–538. Springer, Heidelberg (2009)
12. Benevides, A.B., et al.: Validating modal aspects of OntoUML conceptual models using automatically generated visual world structures. *Journal of Universal Computer Science* 16/20, Special Issue on Evolving Theories of Conceptual Modeling (2010)
13. Guizzardi, G., das Graças, A., Guizzardi, R.S.S.: Design Patterns and Inductive Modeling Rules to Support the Construction of Ontologically Well-Founded Conceptual Models in OntoUML. In: *3rd Intl. Workshop on Ontology Driven Inf. Systems (ODISE 2011)*, London (2011)
14. Olive, A.: *Conceptual Modeling of Information Systems*. Springer (2007)
15. Wieringa, R.J., de Jonge, W., Spruit, P.A.: Using dynamic classes and role classes to model object migration. *Theory and Practice of Object Systems* 1(1), 61–83 (1995)

# An Automatic Approach for Mapping Product Taxonomies in E-Commerce Systems

Lennart J. Nederstigt, Steven S. Aanen, Damir Vandić, and Flavius Fräsinc̆ar

Erasmus Universiteit Rotterdam  
P.O. Box 1738, NL-3000 DR  
Rotterdam, The Netherlands  
len\_nederstigt@xs4all.nl, ssaanen@gmail.com,  
{vandic,frasinc̆ar}@ese.eur.nl

**Abstract.** The recent explosion of Web shops has made the user task of finding the desired products an increasingly difficult one. One way to solve this problem is to offer an integrated access to product information on the Web, for which an important component is the mapping of product taxonomies. In this paper, we introduce CMAP, an algorithm that can be used to map one product taxonomy to another product taxonomy. CMAP employs word sense disambiguation techniques and lexical and structural similarity measures in order to find the best matching categories. The performance on precision, recall, and the  $F_1$ -measure is compared favourably with state-of-the-art algorithms for taxonomy mapping.

**Keywords:** taxonomy mapping, e-commerce, lexical matching, word sense disambiguation.

## 1 Introduction

The rapid expansion and increased reach of the Web over the last twenty years has significantly changed the way in which people exchange information. According to a study by Zhang et al. [22], the Web is growing exponentially, doubling in size roughly every five years. This unprecedented growth also means that the Web plays an increasingly important role in the world economy. The revenue of online shopping in the USA grew from \$7.4 billion in 2000 to \$34.7 billion in 2007 [7], almost a fivefold increase.

With the ever-increasing amount of information available on the Web, it is important that users remain able to access and search all information, as well as have easy and intuitive navigation possibilities. Computers already index Web pages for the use in search engines, but what they often cannot do is understand the actual content found on these Web pages. A human mind is still needed to interpret the data. This is due to the fact that most Web pages are geared towards human-readability rather than machine-usage. In order to fully utilise the power of the Web, it is important that Web information becomes understandable for computers as well.

The Semantic Web would allow computers to grasp the meaning of information on the Web and be able to act independently upon this information. This is done by annotating data, so that it becomes understandable for all systems sharing the same ontology [5]. GoodRelations [6] is a good example of an ontology that can be used to annotate Web resources with e-commerce or product information. For instance, with GoodRelations it is possible to formally specify the warranty, the delivery options, the payment methods, the currency, etc. It is also possible to specify product specific properties, e.g., the screen size of a mobile phone or the CPU speed of laptop.

Unfortunately, very few websites have included a semantic description of their published information, although there are definitely advantages to be gained by doing so. For instance, search engines could provide much more relevant search results if they would actually understand what is published on every website they index. Furthermore, there is evidence that annotation is urgently needed in order to make the Web more useful. According to the aforementioned study on online shopping in the USA [7], more than half of the surveyed users have encountered various frustrations and frictions while shopping online. Information was often found to be lacking, confusing, or there simply was an overload of information.

Integrating the information found on multiple websites would help people in finding the information that is relevant to them. In the field of e-commerce this means that the information about products, as offered by different online retailers, should be integrated. An important part of the integration is the mapping of the product taxonomy of one online retailer to that of another retailer. Aggregating these taxonomy structures enables the presentation of product information in a unified way to the user, largely alleviating frustrations caused by scattered information and failing search results.

However, because of the heterogeneity of product taxonomies, used by different online retailers, the aggregation of product information is not a straightforward process. The heterogeneity is caused by the fact that the construction of product taxonomies is a loosely defined process and as such, there is no uniform way for creating one. This results in characteristics of a product taxonomy that are difficult to handle, like categories composed of multiple terms, or a loosely defined hierarchy for type-of relations. Furthermore, product taxonomies tend to have a large level of depth, which makes category mapping a complex and difficult process.

In this paper we propose the *Category Mapping Algorithm for Products* (CMAP). This algorithm can be used to map product taxonomies from multiple sources to each other. We employ *word sense disambiguation* techniques, using WordNet [15], to find synonyms, hypernyms, and possible senses of category names. Furthermore, we use similarity measures, such as *Jaccard* [8], to determine the most likely candidate category to map to. In order to evaluate the performance of our algorithm, we compare its precision, recall, and the  $F_1$ -measure with state-of-the-art algorithms for taxonomy mapping, i.e., the algorithm proposed by Park & Kim [18] and the PROMPT algorithm [17].

## 2 Related Work

Recently, the merging of taxonomies from heterogeneous sources has been extensively studied, which has resulted in various approaches that have been proposed and implemented. In general, we can distinguish between schema-based and ontology-based matching. Schema matching is usually performed with the help of techniques that are trying to guess the meaning encoded in the schemas, whereas ontology matching systems primarily try to exploit knowledge explicitly encoded in the ontologies [20]. In other words, ontology matching works with structured data that the computer can understand, while schema matching has to deal with unstructured data that needs to be interpreted.

Similar concepts from different taxonomies can be identified either by matching their corresponding terms or by examining their position in the structure of the taxonomy. Many algorithms employ term matching, which indicates how closely one term is related to another, either lexically or semantically. Various measures have been used for this purpose [8,9,11,19,21]. Some of the semantic similarity measures use a semantic lexicon, such as WordNet [15], or a shared upper ontology, such as DOLCE [4] or SUMO [16], to extract the meaning of a term. The path similarity, which indicates how closely two terms are related by analysing their context in the taxonomy structures, can be measured by using, for example, *neighbour matching* [23], *tree matching* [13,12], *path matching* [1,17], and *iterative fix-point computation* [14].

Besides algorithms that focus on automatic mapping, we find also semi-automatic approaches for taxonomy mapping in the literature. Chimaera [13] is an example of such an approach. Chimaera is a mapping tool that can be used either stand-alone or as part of the Ontolingua environment. It suggests, considering the structural similarity, potential mappings based on the lexical similarity between classes. LOM, which is an ontology mapping tool, uses different techniques in order to find the best match [10]. It analyses the lexical similarity between classes by using both exact and partial term matching. Furthermore, it refines the search for potential matches by using sets of synonyms and similar concepts. The synonym sets are found using WordNet and the similar concepts using the SUMO upper ontology.

Noy and Musen have created the PROMPT suite, which includes various tools that can be used for multiple-ontology management [17]. One of these tools is iPROMPT, which provides interactive ontology merging by letting the user decide on how to map a term. It assists the user by providing suggestions based on already defined mappings and the lexical similarity between terms. While the achieved results on recall and precision are quite good, it involves a lot of manual labour and is thus not suited for larger ontologies. AnchorPROMPT, an extension of iPROMPT and also part of the PROMPT suite, addresses this issue by exploiting the category position in the structure of a taxonomy for automatically defining the mappings. It automatically generates a larger set of identical concepts from a previously identified set. These pairs of related terms from the source ontologies, called *anchors*, can be created manually or generated by the system. This approach requires far less manual input than iPROMPT,



but it is not specifically targeted at matching product taxonomies and also the precision drops when longer category paths are used.

Park & Kim suggest to map product taxonomies using a *word sense disambiguation* technique [18]. Given a category, they first try to find the correct sense and the synonyms associated with that particular sense. Using these synonyms, all category paths that have one of the synonyms as a leaf category are retrieved, i.e., the candidate paths. Then the algorithm uses the similarity between the taxonomy paths in order to find the best match. The algorithm can perform better than ontology mapping algorithms that are not specifically designed for matching heterogeneous product taxonomies. However, it does not perform well on nodes close to the root node of a category hierarchy, since it needs more information content, which is only present along longer paths. Furthermore, the algorithm has a bias towards mapping to general categories, which does not work well when dealing with categories composed of multiple product concepts, e.g., mapping ‘Books’ to ‘Books, Movies, Music, Games’, while this last category also has a subcategory called ‘Books’.

### 3 CMAP

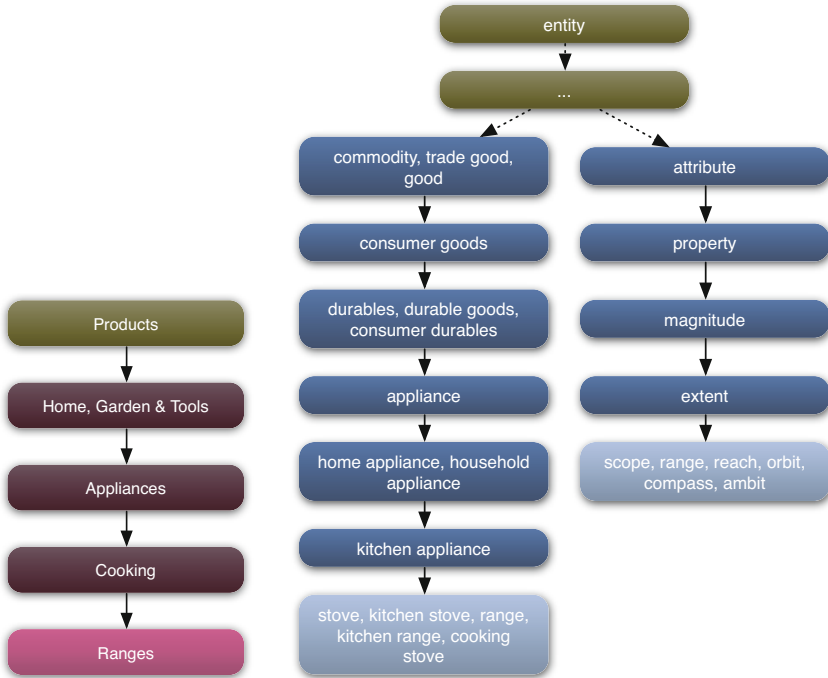
This section explains the CMAP algorithm. Although the CMAP algorithm is based on the algorithm proposed by Park & Kim, there are important differences. CMAP is designed to address the specific issues that arise from the structure of product taxonomies, rather than focusing on schema matching in general. Furthermore, we present a simple, but powerful procedure, which employs lexical term matching and path similarity in conjunction with word sense disambiguation techniques, to determine the best mapping.

Both CMAP and the algorithm from Park & Kim start with searching for the correct meaning of category names from the source taxonomy. Using the synonyms of the correct sense gathered from that process, candidate categories that match the current source category are selected in the target taxonomy. The best out of these candidates is selected using two measures called *co-occurrence* and *order-consistency*. We explain these steps in detail in the rest of this section. The described process is being repeated for every category path in the source taxonomy.

Our approach also handles root category nodes differently than Park & Kim. CMAP assumes that the root category term has no meaning and should therefore not be used in the matching process. For example, the root nodes from the datasets used in this research are ‘Online Shopping’, ‘Shopping’ and ‘Products’. For this reason, CMAP always maps the source root category to the target root category. Because the root category does not provide any information, it is skipped in computations that use category paths. Another important observation is, that all term matching within CMAP is case-insensitive, since the case of characters generally does not affect the meaning.

Many category names in existing datasets are actually composite categories, like ‘Home, Garden & Tools’, as shown in Fig. 1. This issue has not been investigated by Park & Kim. In CMAP, we address this issue by splitting the

category string on predefined words (e.g., ‘and’) or characters (e.g., ‘,’). The result is called the *split term set*. For each element from this set, the word sense disambiguation process discussed next is applied separately.



**Fig. 1.** Hierarchy for ‘Ranges’ in the Amazon taxonomy

**Fig. 2.** Two sense hierarchies for the term ‘ranges’ in WordNet

### 3.1 Word Sense Disambiguation

A core concept of both algorithms is finding the correct sense of a category term from the source taxonomy. WordNet [15], a semantic lexicon, is used for obtaining all possible senses of a category from the source taxonomy. In order to find the correct sense, we have to match the full path of a category to the hierarchy of hypernyms for each sense. For instance, if the path from the source taxonomy is ‘Computers/Notebook’, we can deduce that the correct sense of ‘Notebook’ would be a laptop in this case, rather than a notepad. By finding the correct sense of a term we can also use its synonyms, which enhances the ability to identify category matches, even when the category names are not lexically similar. This is particularly important for matching product taxonomies, because there is no standard for category names, so each taxonomy might be using different words to express the very same product category. For example, one taxonomy might

have a category called ‘Notebook’, while another might have ‘Laptop’. In both cases the meaning is the same, but they are impossible to match using just lexical similarity measures.

Using WordNet, different meanings for the category (split) term are acquired, in the form of their hypernym structure. We call this the sense hierarchy. Figure 2 shows two senses for the term ‘ranges’. The synonyms per sense for ‘ranges’ are at the bottom in light blue (light grey in black and white printing). The goal of the word sense disambiguation process is to end up with only one set of these synonyms, namely the one for the correct source category sense. Note that the word sense disambiguation process only uses the current source category, and the WordNet information about the possible meanings. The target taxonomy does not play a role in the algorithm yet.

In order to decide which meaning fits best to the source category that has to be mapped, a function is employed that finds matches between an upper category (an ancestor of the current node) and a sense hierarchy excluding the leaf node (denoted by list  $S$ ). Upper categories use dark purple (dark grey in black and white printing) nodes in Fig. 1. The sense hierarchy represents one of the meanings of the current category, like one out of the two paths in Fig. 2. The function is given by:

$$\text{computeSimilarity}(t, S) = \{x|x \in H, H \in S \text{ and } \text{baseform}(t) \in H\} \quad (1)$$

where  $t$  = an upper category to match

$H$  = the set of synonyms of one hypernym

$S$  = a sense hierarchy (one sense from WordNet)

where  $\text{baseform}()$  is a function which returns the lemma of a term using WordNet. The result of the function is a set of matches between an upper category, and a sense hierarchy (one meaning of the current category).

Using Equation (1), a measure can be defined that represents how well an ancestor of the source category fits to a certain sense hierarchy. In other words, how well does an upper category fit to one of the possible meanings of the source category? The match between an upper category and an element from the sense hierarchy that is closest to the sense leaf, is seen as most important match. Therefore,  $\text{hyperProximity}()$  uses the distance between the closest match and the leaf to compute a measure for the fit given by:

$$\text{hyperProximity}(t, S) = \begin{cases} \frac{1}{\min_{x \in C}(\text{dist}(x, b))} & \text{if } C \neq \emptyset \\ 0 & \text{if } C = \emptyset \end{cases} \quad (2)$$

where  $t$  = an upper category to match

$S$  = a sense hierarchy (one sense from WordNet)

$C$  =  $\text{computeSimilarity}(t, S)$

$b$  = base term (leaf) of the sense hierarchy  $S$

Function `min()` gives the minimum of a set of values, and `dist()` computes the distances between two nodes in the sense hierarchy. The distance can be explained as the number of arcs that are being traversed when navigating from node to node in a sense hierarchy. In the Ranges example, the hyperproximity for parent ‘Appliances’ and the left sense hierarchy in Fig. 2, would be  $\frac{1}{3}$ . This is because the node in the sense hierarchy closest to the base, is ‘appliance’.

Until now the measurements only used the relation between an upper category and a sense for the source category. However, we need to be able to measure how well a complete source category path, like the one depicted in Fig. 1 (denoted by  $P$ ), fits one particular sense (hierarchy)  $S$  of the category (split) term. The measure for this is given by:

$$\text{pathProximity}(P, S) = \sum_{x \in P} \frac{\text{hyperProximity}(x, S)}{|P| - 1} \quad (3)$$

where  $P$  = list of nodes from the source category path

$S$  = a sense hierarchy (one sense from WordNet)

$x \neq b$

$b$  = base term (leaf) of the sense hierarchy  $S$

It is the average hyperproximity for all upper categories with the current sense hierarchy. This differs from the Park & Kim algorithm, which divides by the total number of nodes (not only ancestors), which does not lead to an average.

Since ‘Appliances’ is the only upper category matching with something from the sense hierarchies, the path-proximity for the left sense hierarchy in Fig. 2 is  $\frac{1}{9}$ . This is because the denominator is 3 ( $|P| - 1$ ), i.e., CMAP does not use the root node in the calculation. The path-proximity for the sense hierarchy on the right in the figure is 0 since there are no matches.

**Selecting the Proper Sense.** Equation (3) measures the fit of the source category path to one possible sense of the category term. This must be extended to be able to deal with multiple possible senses. This is done by computing the path-proximity for all possible senses of the source category (split) term. The sense with the highest path-proximity is picked as the correct sense. In our example, the first (left in the figure) sense would be chosen, which is the correct conclusion.

Last, only the original source category term (or terms in case of a composite category) and the synonyms of the correct sense are used (or synonyms of the correct senses), i.e., the extended split term set. For our example depicted in Fig. 2, that would be the left node in light blue (light grey in black and white printing). According to Park & Kim’s algorithm, when none of the senses fit, or when the category (split) term is not known in WordNet, the process should be stopped and the source category should be mapped to nothing. However, while testing the algorithm of Park & Kim, we discovered that this would result in low

performance on recall. That is why, for evaluation purposes, we have implemented their algorithm in such a way, that the algorithm takes the source category name itself as the “extended” term set, and continue the process. CMAP does this as well, but then for the (possibly) split version of the (composite) category, i.e., the “extended” split term set.

### 3.2 Candidate Path Selection

Using the extended (split) term set that from the word sense disambiguation process, candidate target paths can be collected. This step only uses the target taxonomy. CMAP simply walks through every single category from the target taxonomy. If the split target category term is a superset of the extended split term set, the path of that category is being marked as a candidate. For example, source category ‘Home and Garden’ would fit into candidate category ‘Home, Garden & Tools’, but not the other way around.

Park & Kim, as they do not handle composite categories, check only if the source category term (or its synonyms) is contained in the target category. In addition, they claim that from all candidates, it is needed to remove all more specific ones. In other words, each candidate path of which an ancestor also exists as separate candidate, is removed. However, as explained in Sect. 2 this results in errors with composite categories. CMAP therefore gives every target category, despite its ancestors, the same chance of winning the match with the source category.

Now, there is a collection of candidate paths for the category mapping. To determine which candidate fits best, the co-occurrence and order-consistency will be measured for each of the target paths. The extended (split) term set is only being used for the candidate path selection. In the rest of the algorithm it will not be used again.

### 3.3 Co-occurrence

The co-occurrence is a measure that defines how well the current source category path fits to a candidate target path, while ignoring the order of the nodes in each path. It expresses the level of overlap between two category paths, i.e., the source taxonomy path and one of the candidate target paths.

The co-occurrence measure is based on the function `maxSim()`, which computes the similarity between one category name and another category path (either source or target). This similarity is defined as the highest value of the Jaccard index (for sets of characters), measured on each pair of the category name and a category name from the given category path. For the computation of the Jaccard index, we use a morphological processor in conjunction with WordNet to find the lemma of each category name. This improves the accuracy of the Jaccard index as it now ignores, for example, singular and plural forms of words. Using `maxSim()`, we define the co-occurrence as following:

$$\text{coOccurrence}(P_{\text{src}}, P_{\text{targ}}) = \left( \sum_{t \in P_{\text{targ}}} \frac{\text{maxSim}(t, P_{\text{src}})}{|P_{\text{targ}}|} \right) \cdot \left( \sum_{t \in P_{\text{src}}} \frac{\text{maxSim}(t, P_{\text{targ}})}{|P_{\text{src}}|} \right) \quad (4)$$

where  $P_{\text{src}}$  = list of nodes from the current source path  
 $P_{\text{targ}}$  = list of nodes from a candidate target path  
 $t$  = a category term

As we can notice from the above equation, the co-occurrence consists of a product of two terms that look similar at first sight. The difference between the two terms is the way of comparing, in the first term we compare each node from the target path to the source path and in the second term we compare each node from the source path to the target path. For each pair of a node and a path (either source or target), we compute the similarity using the `maxSim()` function. Both the left and right term in the co-occurrence equation are the average of this similarity for each node that is processed.

This process is best explained with an example. Consider a source path ‘Shopping/Books/Science’ from the ODP taxonomy and a candidate target path ‘Products/Books/Magazines/Science’ from the Amazon taxonomy. The first category in both paths is not used as we do not process the root nodes of taxonomies. This means that we start with ‘Books’, ‘Magazines’, and ‘Science’ from the Amazon path, i.e., the first term in the co-occurrence equation in this example. Because we have a full match for ‘Books’ and ‘Science’, we have `maxSim(‘Books’, {‘Books’, ‘Science’}) = 1` and `maxSim(‘Science’, {‘Books’, ‘Science’}) = 1`. For ‘Magazines’, we have `maxSim(‘Magazines’, {‘Books’, ‘Science’}) = 0.25` as the best match, because there are 3 characters in the intersection and 12 characters in the union of the two character sets ‘Magazines’ and ‘Science’. Thus, the first term of the co-occurrence measure from Equation 4 is in this example  $(1 + 1 + 0.25)/3 = 0.75$ .

The same process is then repeated except that we start with ‘Books’ and ‘Science’ from the ODP path, i.e., the second term in the co-occurrence equation in this example. From this it follows that the second part is  $(1 + 1)/2 = 1$ , as there is a perfect match for both the categories ‘Books’ and ‘Science’. Finally, we multiply both terms in order to obtain the co-occurrence measure, which is  $0.75 \times 1 = 0.75$ .

The Park & Kim algorithm has a similar definition for the co-occurrence measure, but instead of using the Jaccard index for the similarity, they use a function called `termMatch()`. If one of either words is contained within the other, `termMatch()` returns the value of the length of the smallest string divided by the length of the largest one. We find that this approach is too restricted to work well with a broad range of categories in product taxonomies.

### 3.4 Order-Consistency

The co-occurrence measure compares the similarity between nodes in source and candidate target paths. However, in order to find the best possible mapping it is also important that these nodes appear in the same order in both paths. The procedure that computes the order-consistency starts with the function `common()`, which searches for matching nodes between the source path and the current candidate target path. For every match, where also synonyms of the original term can be used, it adds the matching nodes to the common list.

Function `precedenceRelations()` uses the common list to find binary node associations of which the first node hierarchically comes before the second node in the source path. For every element in the common list, it creates pairs of node names from the source path. The first node in the pair should always occur before the other node in the source path. The order of the pairs themselves in the precedence relation set is not important.

Checking whether one precedence relation is applicable to the candidate target path as well is done by the function `consistent()`. If so, it returns the value 1, otherwise it returns 0. The order-consistency is given by:

$$\text{orderConsistency}(P_{\text{src}}, P_{\text{targ}}) = \sum_{r \in R} \frac{\text{consistent}(r, P_{\text{targ}})}{\binom{\text{length}(C)}{2}} \quad (5)$$

where  $P_{\text{src}}$  = list of nodes from the current source path

$P_{\text{targ}}$  = list of nodes from a candidate target path

$C$  = `common`( $P_{\text{src}}$ ,  $P_{\text{targ}}$ )

$R$  = `precedenceRelations`( $C$ ,  $P_{\text{src}}$ )

$r$  = one precedence relation

The denominator is the number of possible combinations to choose two out of the common nodes. Therefore, the order-consistency is the average number of precedence relations from the source path that are consistent with the candidate target path.

### 3.5 Overall Similarity

For every candidate target path, the co-occurrence and order-consistency are computed. The Park & Kim algorithm takes the candidate path that has the highest average of these measures as the measure for overall similarity. CMAP uses the following function to determine the overall similarity:

$$\text{overallSimilarity}(P_{\text{src}}, P_{\text{targ}}) = (\text{orderConsistency}(P_{\text{src}}, P_{\text{targ}}) + t) \cdot \text{coOccurrence}(P_{\text{src}}, P_{\text{targ}}) \quad (6)$$

where  $P_{\text{src}}$  = list of nodes from the current source path

$P_{\text{targ}}$  = list of nodes from a candidate target path

$t$  = the similarity threshold

The similarity threshold is a parameter for both algorithms that functions as a cut-off for the similarity measure for which the source path will not be mapped. CMAP uses this in the overall similarity measure to give candidate paths that have a co-occurrence of 1 and an order-consistency of 0 a chance to pass, since in that situation the overall similarity will not be below the cut-off value. The reason for this is that in the source taxonomy, children of the root node (e.g., ‘Shopping/Books’) will always have an order-consistency of 0 because there are no precedence relation pairs to be found, i.e., the root is not used in the process. However, it could well be that a candidate path like ‘Products/Books’ would fit, even though the order-consistency is 0. The order-consistency offset with the threshold is multiplied by the co-occurrence to make the two measures more dependent on each other. When one of them is very low, while the other is very high, the resulting measure would be smaller than when the average would be used. It ensures that candidate paths with both a fair order-consistency and co-occurrence will better score overall than candidate paths which only score well for one of the measures. This will prevent candidate paths that only have a small set of common categories (in the correct order) from being picked as the correct path based on their high score for order-consistency. In both algorithms, the candidate target path with the highest overall similarity, if passing the similarity threshold, will be chosen for the mapping.

When no good candidate target path has been found, CMAP, unlike the algorithm of Park & Kim, does try to provide a mapping. When the parent of the current source category could be mapped earlier, the source category is being mapped to the same target path. For instance, if ‘Shopping/Books/Braille’ can not be mapped, it will be mapped to ‘Products/Books’, since that is a more general category in which it also fits (i.e., the mapping of the parent ‘Books’). However, if the parent had the same problem with mapping, and is also mapped to its parent, the current source category will not be mapped. This is done in order to prevent too extreme generalisations from happening.

The algorithm basically follows the complete procedure from word sense disambiguation, to candidate selection, to finding the best of the candidate target paths using the two measures, for each source category path (from root category to leaf category) from the taxonomy, with a chosen similarity threshold.

## 4 Evaluation

We compare CMAP with the algorithm from Park & Kim and PROMPT, which are the most similar and relevant projects for our goals. We first discuss the design of the evaluation and what measures were considered during the experiments.

### 4.1 Evaluation Design

For the purpose of evaluating our algorithm, three real-life product taxonomies were collected. The first product taxonomy contains over 2,500 categories and is from Amazon.com, one of the largest online retailers in the world. The second dataset contains over 1,000 categories and is from Overstock.com (O.co), which



is a big retailer that has RDFa-tagged product pages for the GoodRelations [6] ontology. The third dataset is the largest one, containing 44,000 product categories, and is from the Open Directory Project (ODP). ODP is the largest human-edited directory of the Web. All data was collected using custom-built HTML DOM or RDFa crawlers. Using these three datasets we can run six different mappings, one for each combination of datasets. To be able to evaluate a mapping, it is necessary to do the mappings by hand as well. Since the datasets are too large for this, we have taken a random sample of five hundred nodes from each dataset. For every node, we ensure that its ancestors are also in the sample set. A mapping is always performed from a sampled dataset to a full dataset. In order to prevent any bias, the manual mappings were performed collectively by three independent individuals. The main experiment consisted of computing the mappings for all combinations of datasets (using different parameters) for each algorithm. For CMAP and Park & Kim a different overall similarity threshold was used, ranging from 0.0 to 1.0 in steps of 0.05. The optimal thresholds were obtained using a hill-climbing procedure with the  $F_1$ -measure as the optimization criterion. PROMPT has one important parameter, i.e., the maximum path length that is considered. Using the same procedure as for the algorithm of Park & Kim, we found that the optimal value for this parameter is 2. Finally, the evaluation results were obtained by comparing the generated mappings with the manual mappings.

We decided to use the well-established measures from information retrieval for our evaluation, i.e., precision, recall, accuracy, and specificity. For our evaluation, the common definitions cannot be used since we do not have one ‘positive’ class, but as many ‘positive’ classes as the number of correctly mapped paths in the target taxonomy. We do have only one ‘negative’ class, i.e., the action of mapping to nothing. The true and false positives (TP and FP) are therefore defined as the correct and incorrect mappings to a path, respectively, and the true and false negatives (TN and FN) as the correct and incorrect mappings to null (nothing), respectively. The precision, recall, accuracy, and specificity are then defined as:

$$\begin{aligned} \text{precision} &= \frac{\text{TP}}{\text{TP} + \text{FP}} & \text{recall} &= \frac{\text{TP}}{\text{TP} + \text{FN}} \\ \text{accuracy} &= \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} & \text{specificity} &= \frac{\text{TN}}{\text{FP} + \text{TN}} \end{aligned}$$

In order to have an overall score for each algorithm, we use the  $F_1$ -measure, which is the harmonic mean of the precision and recall.

## 4.2 Results

As shown in Table II, CMAP performs better on average for both precision and recall when compared with the algorithm of Park & Kim and PROMPT. Furthermore, this increase in performance does not come at the expense of a significant amount of additional computation time. The increase from 4.99 seconds to 5.82 seconds for the average computation time per mapping, on an Intel Core 2 Duo P8800 at 2.66GHz with 4GB of RAM, is mostly caused by a large

number of composite categories in the Amazon and Overstock datasets. These category names are split and processed individually by CMAP rather than processing the whole category name at once, which explains the slight increase in average computation time. Although we have not performed a formal computational complexity analysis due to the intricacies of the proposed algorithm, our results indicate the CMAP has linear time complexity w.r.t. the number of category nodes in the target path. CMAP maps 500 categories to ODP within 8 seconds, while the same 500 categories are mapped in approximately 0.2 seconds to Overstock.com. PROMPT is the fastest amongst all algorithms, at the expense of a lower precision and recall.

**Table 1.** Comparison of average results per algorithm

Algorithm	Precision	Recall	$F_1$ -measure	Computation time
PROMPT	19.82%	10.62%	0.1350	0.47 s
Park & Kim	37.89%	17.93%	0.2415	4.99 s
CMAP	41.82%	26.03%	0.3180	5.82 s

The recall is important for the matching of product taxonomies, because it is better to map many categories with some possible imprecision rather than only mapping a few categories with high precision in this field. The main objective of mapping product taxonomies is to reduce the number of search failures, when the user cannot find the products he is interested in, rather than making sure that no errors are being made. Although the recall performance is relatively low, we do improve over the method of Park & Kim and PROMPT.

The average precision has increased from 37.89% for the algorithm of Park & Kim to 41.82% for CMAP. Our increased precision is favoured by the new overall path similarity measure, which ensures that both order consistency and co-occurrence need to have high values for the proposed mappings. Also, the higher precision and recall can be attributed at least partially to the fact that CMAP will map a category to the mapping of its parent when no match can be found. This is intuitive for product taxonomies, because a very specific product category might not exist in the other taxonomy, but there might be a more general category to which the algorithm can map. Furthermore, splitting composite category names enables us to find more candidate paths to map to, which also greatly improves the recall. For the average recall, our algorithm outperforms the algorithm of Park & Kim with a difference of 8.10 percent point, i.e., our average recall is 45% higher than that of Park & Kim. Besides the ability to deal with composite categories, the high recall can be attributed also to the use of the Jaccard index for matching terms. The algorithm of Park & Kim uses an exact lexical match procedure for this purpose, which does not take the syntactic variations between terms into account.

**Table 2.** Best results for Park & Kim algorithm

Mapping	Precision	Accuracy	Specificity	Recall	$F_1$ -measure	Threshold
Amazon → ODP	29.10%	20.80%	40.63%	11.47%	0.1645	0.10
Amazon → O.co	47.15%	31.80%	60.84%	17.37%	0.2539	0.00
ODP → Amazon	27.07%	42.48%	64.47%	15.93%	0.2006	0.05
ODP → O.co	31.89%	27.66%	38.54%	20.07%	0.2464	0.00
O.co → Amazon	54.29%	32.20%	45.21%	26.84%	0.3592	0.00
O.co → ODP	37.86%	26.60%	47.90%	15.92%	0.2241	0.00
Average	37.89%	30.26%	49.60%	17.93%	0.2415	

**Table 3.** Best results for CMAP

Mapping	Precision	Accuracy	Specificity	Recall	$F_1$ -measure	Threshold
Amazon → ODP	31.44%	25.60%	26.29%	25.09%	0.2791	0.05
Amazon → O.co	69.94%	44.60%	66.23%	34.97%	0.4663	0.30
ODP → Amazon	27.06%	41.68%	56.64%	21.60%	0.2402	0.15
ODP → O.co	39.61%	31.26%	50.53%	19.61%	0.2624	0.10
O.co → Amazon	46.53%	32.00%	37.93%	28.83%	0.3561	0.20
O.co → ODP	36.32%	29.40%	34.15%	26.10%	0.3037	0.15
Average	41.82%	34.09%	45.30%	26.03%	0.3180	

**Table 4.** Best results for PROMPT

Mapping	Precision	Accuracy	Specificity	Recall	$F_1$ -measure
Amazon → ODP	7.74%	14.40%	29.56%	4.04%	0.0531
Amazon → O.co	35.59%	29.00%	57.54%	13.08%	0.1913
ODP → Amazon	8.08%	31.26%	43.48%	9.04%	0.0853
ODP → O.co	15.51%	20.84%	32.19%	10.90%	0.1280
O.co → Amazon	41.95%	27.80%	39.52%	21.92%	0.2879
O.co → ODP	10.07%	18.80%	39.02%	4.75%	0.0645
Average	19.82%	23.68%	40.22%	10.62%	0.1350

Tables 2, 3, and 4 show the detailed results for each combination of data sets, where in the Tables 2 and 3 the column ‘Threshold’ contains the different overall similarity thresholds. Compared to the algorithm of Park & Kim, the only mapping for which CMAP scores lower on the  $F_1$ -measure is the mapping from Overstock to Amazon. We can see in Tables 2 and 3 that the  $F_1$ -measure drops from 0.3592 to 0.3561. Table 4 shows that the results of PROMPT for the  $F_1$ -measure are inferior to both CMAP and the algorithm of Park & Kim.

## 5 Conclusions and Future Work

This paper proposes CMAP, an algorithm suitable for mapping product taxonomies. The main focus in the development of CMAP was to improve an existing mapping algorithm by accounting for issues that are specific for product taxonomies. It achieves this objective by enhancing the algorithm proposed by Park & Kim with composite category splitting, using Jaccard index as similarity measure instead of exact lexical matching, using a different measure for overall similarity, and by mapping to a more general target category when no match can be found for a source category. The algorithm was tested on three different datasets and its performance was compared with that of PROMPT and the algorithm of Park & Kim. When categories are mapped from one product taxonomy to another, the results show that a higher average precision, recall, and  $F_1$ -measure are obtained using CMAP than by using the other state-of-the-art algorithms.

The algorithm could be improved by not only analysing the ancestors of a category and the hypernyms of its sense, but also its children and hyponyms. This resolves the issue with short paths, where the word sense disambiguation process is often inconclusive because it simply cannot obtain enough information about the correct sense from the path. The word sense disambiguation process could also be further improved by splitting composite categories of the upper categories in a path as well, rather than just splitting the current category. Furthermore, the results from the word sense disambiguation process are now only used for selecting candidate paths, but they could also be used for the co-occurrence and order-consistency measures. Last, in future work we plan to distinguish between nouns and adjectives in category names. This would allow for more precise mapping, because nouns are generally more important for product similarity than adjectives.

**Acknowledgment.** This work is sponsored by the Netherlands Organisation for Scientific Research (NWO) Mozaiek project 017.007.142: Semantic Web Enhanced Product Search (SWEPS).

## References

1. Aumueller, D., Do, H., Massmann, S., Rahm, E.: Schema and Ontology Matching with COMA++. In: ACM SIGMOD International Conference on Management of Data 2005 (SIGMOD 2005), pp. 906–908. ACM (2005)
2. Castano, S., Ferrara, A., Montanelli, S., Zucchelli, D.: Helios: A General Framework for Ontology-based Knowledge Sharing and Evolution in P2P Systems. In: 14th International Workshop on Database and Expert Systems Applications (2003)
3. Ehrig, M., Staab, S.: QOM – Quick Ontology Mapping. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 683–697. Springer, Heidelberg (2004)
4. Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., Schneider, L.: Sweetening ontologies with DOLCE. In: Gómez-Pérez, A., Benjamins, V.R. (eds.) EKAW 2002. LNCS (LNAI), vol. 2473, pp. 166–181. Springer, Heidelberg (2002)

5. Gruber, T.: A Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition* 5(2), 199 (1993)
6. Hepp, M.: GoodRelations: An Ontology for Describing Products and Services Offers on the Web. In: Gangemi, A., Euzenat, J. (eds.) *EKAW 2008*. LNCS (LNAI), vol. 5268, pp. 329–346. Springer, Heidelberg (2008)
7. Horrigan, J.: Online Shopping. *Pew Internet & American Life Project Report* 36 (2008)
8. Jaccard, P.: The Distribution of the Flora in the Alpine Zone. *New Phytologist* 11(2), 37–50 (1912)
9. Levenshtein, V.: Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Soviet Physics Doklady* 10, 707–710 (1966)
10. Li, J.: LOM: A Lexicon-based Ontology Mapping Tool. In: *Performance Metrics for Intelligent Systems*, PerMIS 2004 (2004)
11. Lin, D.: An Information-Theoretic Definition of Similarity. In: *15th International Conference on Machine Learning (ICML 1998)*, pp. 296–304. Morgan Kaufmann Publishers Inc. (1998)
12. Madhavan, J., Bernstein, P., Rahm, E.: Generic Schema Matching with Cupid. In: *27th International Conference on Very Large Data Bases (VLDB 2001)*, pp. 49–58. Morgan Kaufmann Publishers Inc. (2001)
13. McGuinness, D., Fikes, R., Rice, J., Wilder, S.: The Chimaera Ontology Environment. In: *17th National Conference on Artificial Intelligence (AAAI 2000)*, pp. 1123–1124. AAAI Press (2000)
14. Melnik, S., Garcia-Molina, H., Rahm, E.: Similarity Flooding: A Versatile Graph Matching Algorithm and its Application to Schema Matching. In: *18th International Conference on Data Engineering (ICDE 2002)*, pp. 117–128. IEEE (2002)
15. Miller, G.: WordNet: A Lexical Database for English. *Communications of the ACM* 38(11), 39–41 (1995)
16. Niles, I., Pease, A.: Towards a Standard Upper Ontology. In: *International Conference on Formal Ontology in Information Systems 2001 (FOIS 2001)*, pp. 2–9. ACM (2001)
17. Noy, N., Musen, M.: The PROMPT Suite: Interactive Tools for Ontology Merging and Mapping. *International Journal of Human-Computer Studies* 59(6), 983–1024 (2003)
18. Park, S., Kim, W.: Ontology Mapping between Heterogeneous Product Taxonomies in an Electronic Commerce Environment. *International Journal of Electronic Commerce* 12(2), 69–87 (2007)
19. Resnik, P.: Using Information Content to Evaluate Semantic Similarity in a Taxonomy. In: *14th International Joint Conferences on Artificial Intelligence (IJCAI 1995)*, pp. 448–453 (1995)
20. Shvaiko, P., Euzenat, J.: A Survey of Schema-Based Matching Approaches. In: Spaccapietra, S. (ed.) *Journal on Data Semantics IV*. LNCS, vol. 3730, pp. 146–171. Springer, Heidelberg (2005)
21. Yang, Y., Liu, X.: A Re-examination of Text Categorization Methods. In: *22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1999)*, pp. 42–49. ACM (1999)
22. Zhang, G., Zhang, G., Yang, Q., Cheng, S., Zhou, T.: Evolution of the Internet and its Cores. *New Journal of Physics* 10, 123027 (2008)

# Requirements-Driven Root Cause Analysis Using Markov Logic Networks

Hamzeh Zawawy<sup>1</sup>, Kostas Kontogiannis<sup>2</sup>,  
John Mylopoulos<sup>3</sup>, and Serge Mankovskii<sup>4</sup>

<sup>1</sup> University of Waterloo, Ontario, Canada  
hzawawy@gmail.uwaterloo.ca

<sup>2</sup> National Technical University of Athens, Greece  
kkontog@softlab.ntua.gr

<sup>3</sup> University of Toronto, Ontario, Canada  
jm@cs.toronto.edu

<sup>4</sup> CA Labs, Ontario, Canada  
Serge.Mankovskii@ca.com

**Abstract.** Root cause analysis for software systems is a challenging diagnostic task, due to the complexity emanating from the interactions between system components and the sheer size of logged data. This diagnostic task is usually assisted by human experts who create mental models of the system-at-hand, in order to generate hypotheses and conduct the analysis. In this paper, we propose a root cause analysis framework based on requirement goal models. We consequently use these models to generate a Markov Logic Network that serves as a diagnostic knowledge repository. The network can be trained and used to provide inferences as to why and how a particular failure observation may be explained by collected logged data. The proposed framework improves over existing approaches by handling uncertainty in observations, using natively generated log data, and by providing ranked diagnoses. The framework is illustrated using a test environment based on commercial off-the-shelf software components.

**Keywords:** goal model, markov logic networks, root cause analysis.

## 1 Introduction

Software root cause analysis (RCA) is the process by which system administrators analyze symptoms in order to identify the faults that have led to system application failures. More specifically, for systems that comprise of a large number of components encompassing complex interactions, root cause analysis may require large amounts of system operation data to be logged, collected and analyzed. It is estimated that forty percent of large organizations generate more than one terabyte of log data per month, whereas eleven percent of them generate more than ten Terabytes of log data monthly [10].

In this context, and in order to maintain the required service quality levels, such IT systems must be constantly monitored and evaluated by analyzing complex logged data emanating from diverse components. However, the sheer size

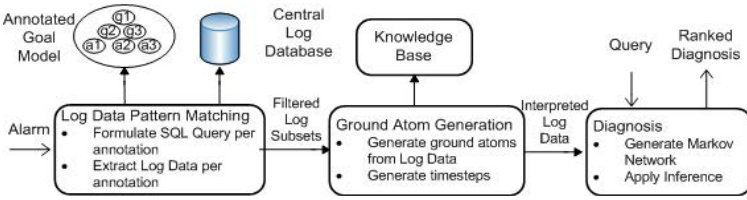


Fig. 1. Diagnostic Process

of such logged data often makes human analysis intractable and consequently, requires the use of algorithms and automated processes.

For this paper, we adopt a hybrid approach based on modeling the diagnostic knowledge as goal trees and on a probabilistic reasoning methodology based on Markov Logic Networks (MLNs). More specifically, the process is based on three main steps as illustrated in Fig. 1. In the first step, the diagnostic knowledge is denoted as a collection of goal models that represent how specific functional and non-functional system requirements can be achieved. Each goal model node is consequently annotated with *pre-condition*, *post-condition* and *occurrence* patterns. These patterns are used to filter and generate subsets of the logged data in order to increase the performance of the diagnostic process. The second step of the process is based on the use of a diagnostic rule knowledge base that is constructed by the goal models and the generation of ground atoms that are constructed by the logged data. The third and final step of the process is based on the use of the knowledge base and the ground atoms to reason on the validity of hypotheses (queries) that are generated as a result of the observed symptoms.

This paper is organized as follows. Section 2 covers the research baseline. Section 3 presents a motivating scenario. Section 4 describes the architecture and processes in the proposed framework. A case study is presented in section 5. Section 6 reviews related work. The conclusions are in section 7.

## 2 Research Baseline

### 2.1 Goal Models

Goal models have been for long proven to be effective in capturing large numbers of alternative sets of low-level tasks, operations, and configurations that can fulfill high-level stakeholder requirements [15]. Fig. 2 depicts a (simplified) goal model for a loan application. A goal model consists of goals and tasks. Goals—the squares in the diagram—are defined as states of affairs or conditions that one or more actors would like to achieve, e.g., *loan evaluation*. On the other hand, tasks—the oval shapes—describe activities that actors perform in order to fulfill their goals, e.g., *update loan database*.

Goals and tasks can impact each other’s satisfaction using contribution links:  $++S$ ,  $--S$ ,  $++D$ ,  $--D$ . More specifically, given two goals  $G_1$  and  $G_2$ , the link

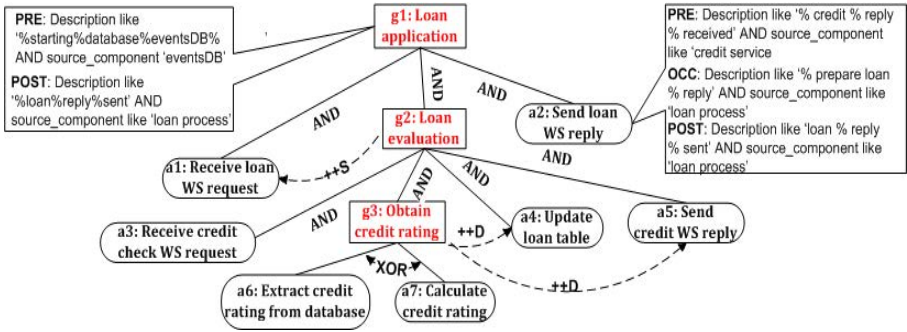


Fig. 2. Loan Application Business Process Goal Model

$G_1 \xrightarrow{++S} G_2$  (respectively  $G_1 \xrightarrow{--S} G_2$ ) means that if  $G_1$  is satisfied, then  $G_2$  is satisfied (respectively denied). The meaning of links  $++D$  and  $--D$  are dual w.r.t. to  $++S$  and  $--S$  respectively, by inverting satisfiability and deniability. The class of goal models used in our work has been originally formalized in [6], where appropriate algorithms have been proposed for inferring whether a set of root-level goals are satisfied or not.

For this paper, we use the loan application goal model originally presented in [16] as a running example to better illustrate the inner workings of the proposed approach (see Fig. 2). The example goal model contains three goals (rectangles) and seven tasks (circles). The root goal  $g_1$  (loan application) is AND-decomposed to goal  $g_2$  (loan evaluation) and tasks  $a_1$  (Receive loan web service request) and  $a_2$  (Send loan web service reply), indicating that goal  $g_1$  is satisfied if and only if goal  $g_2$ , tasks  $a_1$  and  $a_2$  are satisfied, and so on. Also, the contribution link  $++D$  from goal  $g_3$  to tasks  $a_4$  and  $a_5$  indicates that if  $g_3$  is denied then tasks  $a_4$  and  $a_5$  should be denied as well. Similarly, the contribution link  $++S$  from  $g_2$  to task  $a_1$  indicates that if  $g_2$  is satisfied then so must be  $a_1$ .

Furthermore, as illustrated in Fig. 2, we annotate goals and tasks with *precondition*, *occurrence* and *postcondition* expressions. A task/goal is satisfied if it has occurred and its preconditions (postconditions) are true before (respectively after) its occurrence. Occurrence as well as, preconditions and postconditions are evaluated by a pattern matching approach that is discussed in detail in [17]. In practice, finding the expressions used in the annotations is done by users that are familiar with the monitored systems and the contents of the generated log data, or can be generated by data analysis techniques such as data mining and generalised sequence pattern algorithms. The techniques have been presented in the literature and are outside the scope of this paper. Experimentally, we followed an iterative approach in finding the expressions for the annotations of the goal model in Fig. 2. This process consists of assigning expressions, generating and evaluating the observation. When false positives are identified, the corresponding expression(s) are updated accordingly, and the process is restarted.



## 2.2 Markov Logic Networks

MLNs have been recently proposed as a way of providing a framework that combines first order logic and probabilistic reasoning [5]. A knowledge base consisting of first-order logic formulae represents a set of hard constraints on the set of possible worlds that it describes. In probabilistic terms, if a world violates one formula, it has zero probability of being an interpretation of the knowledge base. Markov logic softens these constraints by making a world that violates a formula to be less probable but still, possible. The more formulas a world violates, the less probable it becomes. In MLNs, each logic formula  $F_i$  is associated with a positive real-valued weight  $w_i$ . Every grounding (instantiation) of  $F_i$  is given the same weight  $w_i$ . In this context, a Markov Network is an undirected graph that is built by an exhaustive grounding of the predicates and formulas as follows:

- Each node corresponds to a ground atom  $x_k$  (an instantiation of a predicate).
- If a subset of ground atoms  $x_{\{i\}} = \{x_k\}$  are related to each other by a formula  $F_i$  with weight  $w_i$ , then a clique (subset of the graph where each two vertices in the subset are connected by an edge)  $C_i$  over these variables is added to the network.  $C_i$  is associated with a weight  $w_i$  and a feature function  $f_i$  defined as follows,

$$f_i(x_{\{i\}}) = \begin{cases} 1 & F_i(x_{\{i\}}) = True, \\ 0 & otherwise \end{cases} \quad (1)$$

First-order logic formulae serve as templates to construct the Markov Network. Each ground atom,  $X$ , represents a binary variable in a Markov network. The overall network is then used to model the joint distribution of all ground atoms. The corresponding global energy function can be calculated as follows,

$$P(X = x) = \frac{1}{Z} \exp\left(\sum_i w_i f_i(x_{\{i\}})\right) \quad (2)$$

where  $Z$  is the normalizing factor calculated as,

$$Z = \sum_{x \in X} \exp\left(\sum_i w_i f_i(x_{\{i\}})\right) \quad (3)$$

where  $i$  denotes the subset of ground atoms  $x_{\{i\}}$  that are related to each other by a formula  $F_i$ . The Markov network can then be used to compute the marginal distribution of events and perform inference. Since inference in Markov networks is #P-complete, approximate inference is proposed to be performed using the Markov chain Monte Carlo (MCMC), and the Gibbs sampling. More details on the Markov Logic Networks and their use can be found in [11].

## 3 Motivating Scenario

We use as a running example the RCA for a failed execution of a loan evaluation business process implemented by a service oriented system. The normal

execution scenario for the system starts upon receiving a loan application in the form of a Web Service request. The loan applicant's information is extracted and used to build another request that is sent to a credit evaluation Web Service. The credit rating of the applicant is returned as Web Service reply. Based on the credit rating of the loan applicant, a decision is made on whether to accept or reject the loan application. This decision is stored in a table before a Web Service reply is sent back to the front end application.

The requirements of the *Loan application* process are modeled in the goal tree illustrated in Fig. 2. For our running example we consider that the operator observes the failure of the top goal  $g_1$  of the goal model. Surprisingly, even with this relatively simple scenario, the relationship between failures and their faults is not always obvious due to cascading errors and incomplete log data as well as due to intermittent connection errors, incorrect data entries that are hard to debug when large number of requests are processed during a short time.

## 4 Root Cause Analysis Framework

### 4.1 Building a Knowledge Base

In this section, we discuss the first component of the framework which consists of a process of building a diagnostic knowledge base from goal models.

**Goal Model Annotations.** We extend the goal models by annotating the goal model's nodes with additional information on the events pertaining to each of these nodes. In particular, tasks (leaf nodes) are associated with *pre-condition*, *occurrence* and *post-condition* patterns, while goals (non-leaf nodes) are associated with pre-condition and post-condition patterns only. These annotations are expressed using string pattern expressions of the form,

[not] *column\_name* [not] *LIKE* "*match\_string*"

where *column\_name* represents a field name in the log database and *match\_string* can contain the following pattern matching symbols:

- %: Matches strings of zero or many characters.
- Underscore (\_): Matches one character.
- [...]: enclose sets or ranges, such as [*abc*] or [*a – d*].

These symbols are based on the SQL Server 2008 R2 specifications [9] so that goal model annotations can readily be usable in an SQL query. Moreover, we adopt all the predicates from the SQL specifications such as *LIKE* and *CONTAINS*. With respect to our running example, an annotation is the precondition for goal  $g_1$  (Fig. 2) shown below:

*Pre(g<sub>1</sub>): Description LIKE '%starting%Database%eventsDB%' AND source\_component LIKE 'eventsDB'*

This annotation example is considered as a pre-condition pattern for goal  $g_1$  and succeeds when it matches event traces generated by the *eventsDB* database system where trace logs have a description text containing the keyword *starting*, followed by space, then followed by the keyword *Database*, then space then followed by the keyword *eventsDB* (see Fig. 2 for more annotation examples).

**Goal Model Predicates.** We represent the states and actions of the monitored system/service as first order logic predicates. A predicate is *intensional* if its truth value can only be inferred (i.e. cannot be directly observed). A predicate is *extensional* if its truth value can be directly observed. A predicate is *strictly extensional* if it can only be observed and not inferred for all its groundings [13]. We use the extensional predicates *ChildAND*(*parent\_node*, *child\_node*), *ChildOR*(*parent\_node*, *child\_node*) to denote the AND/OR goal decomposition. For instance, *ChildAND*(*parent*, *child*) is true when *child* is an AND-child of *parent* (similarly for *ChildOR*). Examples of AND goal decomposition are goals  $g_1$  and  $g_2$  (see Fig. 2). An example of OR decomposition is goal  $g_3$ . In Fig. 2, goal  $g_1$  is the AND parent of  $a_1$ ,  $g_2$  and  $a_2$ . Such parent-child relationships are represented by assigning truth values to the ground atoms: *ChildAND*( $g_1$ ,  $a_1$ ), *ChildAND*( $g_1$ ,  $g_2$ ) and *ChildAND*( $g_1$ ,  $a_2$ ). We use the extensional predicates *Pre*(*node*, *timestep*), *Occ*(*node*, *timestep*), and *Post*(*node*, *timestep*) to represent tasks' preconditions, occurrences and postconditions (respectively) at a certain timestep. For our work, we assume a total ordering of events according to their logical or physical timestamps [4]. Finally, we use the intensional predicates *G\_Occ*(*node*, *timestep*, *timestep*) and *Satisfied*(*node*, *timestep*) to represent the goals occurrences and the goals/tasks satisfaction. The predicate *Satisfied* is predominantly intensional except for the top goal which satisfaction is observable (i.e. the observed system failure that triggers the RCA process). If the overall service/transaction is successfully executed, then the top goal is considered to be satisfied, otherwise it is denied.

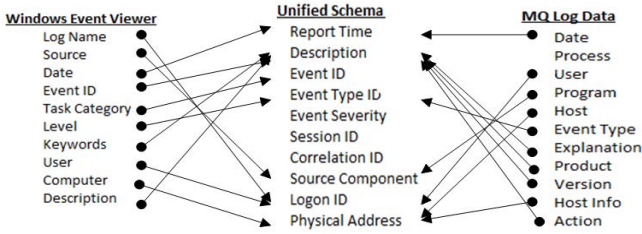
**Rules Generation.** The goal model relationships are used to generate a knowledge base consisting of a set of predicates, ground atoms and a set of rules in the form of first order logic expressions.

As presented above, the predicates *ChildAND*( $a, b$ ) and *ChildOR*( $c, d$ ) represent the AND and OR decomposition where  $a$  is the AND parent of  $b$ , and  $c$  is the OR parent of  $d$  (respectively).

A task  $a$  with a precondition *Pre*( $a, t - 1$ ) and a postcondition *Post*( $a, t + 1$ ) is satisfied at time  $t + 1$  if and only if  $\{Pre\}$  is true at physical or logical time  $t - 1$  that is before task  $a$  occurs at time  $t$ , and  $\{Post\}$  is true at time  $t + 1$  (see Equation 4 below).

$$Pre(a, t - 1) \wedge Occ(a, t) \wedge Post(a, t + 1) \Rightarrow Satisfied(a, t + 1) \quad (4)$$

Unlike tasks which occur on a specific moment of time, goal occurrences span over an interval  $[t_1, t_2]$  that encompasses the occurrences times of its children goals/tasks. We represent the occurrence of a goal  $g$  using the predicate *G\_Occ*( $g, t_1, t_2$ ) over the time interval  $[t_1, t_2]$ . Thus, a goal  $g$  with precondition



**Fig. 3.** Mappings from Windows Event Viewer and MQ Log into Unified Schema

$Pre(g, t_1)$  and postcondition  $Post(g, t_2)$  is satisfied at time  $t_2$  if and only if  $g$ 's occurrence completes at  $t_2$ , and its precondition is true before its occurrence started at  $t_1$  ( $t_1 < t_2$ ) and finally, if its postcondition is true when its occurrence is completed at  $t_2$  (see Equation 5 below).

$$Pre(g, t_1) \wedge G\_Occ(g, t_1, t_2) \wedge Post(g, t_2) \Rightarrow Satisfied(g, t_2) \tag{5}$$

The truth values of the intensional predicate  $G\_Occ(goal, t_1, t_2)$  (used in Equation 5) are inferred based on the satisfaction of all its children in the case of AND-decomposed goals (Equation 6) or at least one of its children in the case of OR-decomposed goals (Equation 7).

$$\forall a, Satisfied(a, t_1) \wedge ChildAND(g, a) \wedge (t_2 < t_1 < t_3) \Rightarrow G\_Occ(g, t_2, t_3) \tag{6}$$

$$\exists a, Satisfied(a, t_1) \wedge ChildOR(g, a) \wedge (t_2 < t_1 < t_3) \Rightarrow G\_Occ(g, t_2, t_3) \tag{7}$$

Contribution links of the form  $node_1 \xrightarrow{++S} node_2$  are represented in Equation 8. (Similarly for  $++D, --S, --D$ ).

$$Satisfied(node_1, t_1) \Rightarrow Satisfied(node_2, t_2) \tag{8}$$

## 4.2 Observation Generation

This section describes the second component of the framework as illustrated in Fig. 1.

**Goal Model Compilation.** This framework is built on the premise that the monitored system's requirements goal model is available by system analysts or can be reverse engineered from source code using techniques discussed in [15].

**Storing Log Data.** In a system such as the loan application system, each component generates log data using its own native schema. We consider mappings from the native schema of each logger into a common log schema as shown in Fig. 3. We consider a unified schema for this work containing ten fields classified into four categories: *general*, *event specific*, *session information* and *environment related*. This proposed schema represents a comprehensive list of data fields considered to be useful for diagnosis purposes. In practice, many commercial monitor

environments contain only a subset of this schema. The identification of the mappings between the native log schema of each monitor component and the unified schema is outside the scope of this paper. Such mappings can be compiled using semi-automated techniques discussed in detail in [2] or compiled manually by subject matter experts. For the purposes of this study, we have implemented the mappings as tables using the Java programming language.

**Log Data Interpretation.** Once logged data are stored in a unified format, the pattern expressions annotating the goal model nodes are used to generate SQL queries that are applied to collect a subset of the logged data pertaining to the analysis. This matching process is discussed in detail in [17]. The  $Pre(g_1)$  expression in section 4.1 can match the log data entry shown below:

Report_Time	Description	Physical_Address
2010-02-05 17:46:44.24	Starting database eventsDB...	DATABASE

In the case where  $Pre(g_1)$  returns log entries when applied to the log data store, we conclude that there is evidence that the event associated with this query (goal model annotation) has occurred. If this query does not return any log entries, then we can't conclude that the event did not occur but rather that we are uncertain about its occurrence.

**Ground Atoms Generation.** The truth assignment for the extensional predicates is done based on the pattern matched log data. We show below a subset of the ground atoms that could be generated from the goal model depicted in Fig. 2.

$pre(g_1,1), ?pre(a_1,1), !occ(a_1,2), post(a_1,3), \dots, ?post(g_1,13), !satisfied(g_1,13)$

The set of literals above, represents the observation of one failed loan application session. For some of the events corresponding to goals/tasks execution, there may be no evidence of their occurrence which can be interpreted as either they did not occur or they were missed from the observation set. We use this uncertainty by preceding the corresponding ground atoms with interrogation mark (?). In cases where there is evidence that an event did not occur, the corresponding ground atom is preceded with an exclamation mark (!). For example, in Fig. 2 the observation of the system failure is represented by  $!Satisfied(g_1,15)$  which indicates top goal  $g_1$  denial at timestep 15. Note in the above example, the precondition for task  $a_3$  was denied and the occurrence of  $a_1$  was not observed leading to the denial of task  $a_1$ , which led to goal  $g_1$  not to occur and thus be denied (see Fig. 2). In turn, the denial of goal  $g_2$  supports the observation goal  $g_1$  did not occur and consequently satisfied.

Note that goal models are independently analyzed. For example, the ground atoms shown above are generated with respect to goal model in Fig. 2. The same log data may be analyzed by another goal model leading to a different stream of ground atoms. The two streams could have common ground atoms if the two goal models have tasks (or even annotations) in common. In the case of large

number of goal models, the value of the atoms can be cached in order to optimize the framework's performance.

The filtering of the predicate with timesteps of interest is done using Algorithm 1. Algorithm 1 consists of two steps: first, a list of literals is generated (see example above) by depth-first traversing the goal model tree and generating a list of literals from the nodes annotations (precondition, occurrences and postconditions); second, sequentially go through that list and look for evidence in the log data for the occurrence of each literal within a certain time interval. This time interval is specified as sliding window that is centered around the time when the system is reported to have failed. The size of the window depends on the monitored applications and type of transactions.

---

### Algorithm 1. Observation Generation

---

**Input:** *goal\_model*: goal model for the monitored system  
*log\_data*: log data stored in central database

**Output:** *literals*: set of literals of the form  $[?,!]$  *literal*(*node*,*timestep*)

**Procedure** [*literals*] **generate\_literals**(*goal\_model*) {  
  Set *Curr\_Node* = top node in *goal\_model* //start at the top of the goal tree  
  Set *g\_counter* = 0 //set global counter to zero  
  [*literals*].addLast(*pre*(*Curr\_Node*, *node.precondition\_annotation*, *g\_counter*))  
  While *Curr\_Node* has children,  
    find leftmost not-visited *child* of *Curr\_Node*  
    recursively call *generate\_literals*(*child*)  
  If all children of *Curr\_Node* are visited return [*literals*];  
  If *Curr\_Node* has no children (task),  
    *g\_counter*++  
    [*literals*].addLast(*occ*(*Curr\_Node*, *node.occurrence\_annotation*, *g\_counter*))  
    *g\_counter*++  
    [*literals*].addLast(*post*(*Curr\_Node*, *node.postcondition\_annotation*, *g\_counter*))  
  return [*literals*];}}

**Procedure** [*literals*] **assign\_logical\_values**(*log\_data*, [*literals*]) {  
  for each variable *literal*(*node*, *annotation*, *counter*) in the set [*literals*]  
  filter the log data based on the pattern expression in the annotation;  
  if no log data found matching the pattern expression:  
    replace *literal*(...) by ( ? *literal*(*node*, *counter*) ) else skip  
  if system is reported to have failed:  
    *literals.addLast*( ! *satisfied*(*top*, *counter*) );  
  return [*literals*];}}

**main**(*goal\_model*, *log\_data*)  
  [*literals*] = *generate\_literals*(*goal\_model*);  
  *assign\_logical\_values*(*log\_data*, [*literals*])  
  return [*literals*];}

---

The following is an example of Algorithm 1 based on Fig. 2: the log entry (2010-02-05 17:46:44.24 Starting up database eventsDB ... DATABASE) matches the pattern of the precondition of goal  $g_1$  thus representing an evidence for the occurrence of this event (precondition in this case). For other logical literals that don't have evidence in the log data that they occurred, a (?) mark is assigned to these literals, such as the precondition of  $a_1$ . This process is also exemplified in Fig. 4 below where the log data is filtered and used to assign logical values to a set of ordered literals generated from the goal model.

**Uncertainty Representation.** This framework relies on log data as evidence for the diagnostic process. The process of selecting log data (described in the previous step) can potentially lead to false negatives and false positives which in turn lead to a decreased confidence in the observation.

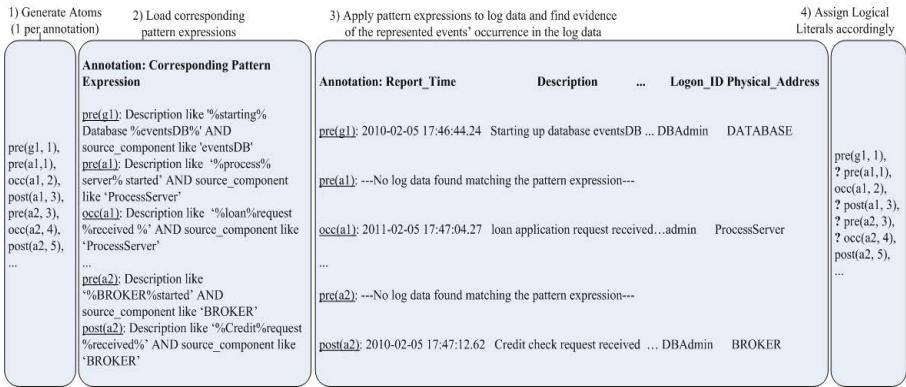


Fig. 4. Generating Observation from Log Data

We address uncertainty in observations using a combination of logic and probabilistic models: First, the domain knowledge representing the interdependencies between systems /services is modeled using weighted first order logic statements. The strength of each relationship is represented with a set of real-valued weights that are generated using a learning process and a training log data set. The weight of each rule represents our confidence relative to other rules in the knowledge base (weight learning is discussed in section 4.3). Consequently, the probability inferred for each atom (consequently the diagnosis) depends on the weight of the competing rules where this atom occurs. For instance, the probability of satisfaction of task  $a_4$  in Fig. 2 ( $Satisfied(a_4, t)$ ) is impacted by the weight of Equation 9 with weight  $w1$  and the weight of any other equation containing  $a_4$ ,

$$w1 \text{ Pre}(a4, t) \wedge \text{Occ}(a4, t + 1) \wedge \text{Post}(a4, t + 2) \Rightarrow \text{Satisfied}(a4, t + 2) \quad (9)$$

Second, uncertainty is also handled by applying an open world assumption to the observation where a lack of evidence does not necessarily negate an event's occurrence but rather weakens its probability.

### 4.3 Diagnosis

The third component in the framework generates a Markov Logic Network (MLN) based on the diagnostic knowledge base (Section 4.1) and then uses the generated observation (Section 4.2) to provide an inference on the root causes for the system failure.

**Markov Network Construction.** A Markov Logic Network is a graph constructed using an exhaustive list of rules and predicates as nodes as well as, grounding predicates with all possible values, and connecting them with a link if these predicates coexist in a grounded formula. The choice of possible values for grounding the predicates can lead to an explosion in the number of ground atoms and network connections if not carefully designed, in particular when modeling time. For this purpose, we represent time using timesteps (integers) that denote the time interval that one session of the service described by the goal model takes to execute.

**Weight Learning.** Learning the weight of the rules is performed using a generative learning algorithm with the use of a training set [8]. During automated weight learning, each formula is converted to CNF, and a weight is learned for each of its clauses. The weight of a clause is used as the mean of a Gaussian prior for the learned weight. On the other hand, the quality and completeness of the training set impact the set of learned weights. We measure the completeness of the training set used in our experiment with respect to the goal model representing the monitored application. In particular, the training set we considered in our experiments contains at least one pair of evidence/expected output for each node in the goal model. In addition, all the rules in the knowledge base are exercised at least once in the training set. The learnt weight can be further modified by the operator to reflect his or her confidence in the rules. For example, the rules that embed a fact such as when the system operator visually witness the system's failure (represented as the top goal being denied) should be given more weight than the rules where goal/tasks satisfaction are inferred based on "uncertain" log data.

**Inference.** Using the constructed Markov Network, we can infer the probability distribution for the ground atoms in the KB given the observations. Of particular interest are the ground atoms for the *Satisfied(node, timestep)* predicate which represents the satisfaction or denial of tasks and goals in the goal model at a certain timestep. Algorithm 2 below is used to produce a diagnosis for failure of a top goal at timestep T. MLN inference generates weights for all the ground atoms of *Satisfied(task, timestep)* for all tasks and at every timestep. Based on the



MLN rules listed in section 3, the contribution of a child node's satisfaction to its parent goal's occurrence depends on the timestep of when that child node was satisfied. Algorithm 2 recursively traverses the goal model starting at the top goal node. Using the MLN set of rules, the algorithm identifies the timestep  $t$  for each task's  $Satisfied(n,t)$  ground atom that contributes to its parent goal at a specific timestep ( $t'$ ). The  $Satisfied$  ground atoms of tasks with the identified timesteps are added to a secondary list and then ordered based on their timesteps. Finally, the algorithm inspects the weight of each ground atom in the secondary list (starting from the earliest timestep), and identifies the tasks with ground atoms that have a weight of less than 0.5 as the potential root cause for the top goal's failure. The tasks with ground atoms at earlier timesteps are identified as more likely to be the source of failure. A set of diagnosis scenarios based on the goal model in Fig. 2 is shown in Table 1 and discussed in section 5.

---

### Algorithm 2. Diagnosis Algorithm

---

**Input:**  $mln$ : weighted rules  $r$  for the goal model (diagnostic knowledge base)  
 $Satisfied(n,t)$ : ground atoms for predicate  $Satisfied$   
 $T$ : timestep when the top goal satisfaction is investigated

**Output:**  $\Gamma$ : ranked list of root causes

**Diagnose( $mln$ ,  $Satisfied(n,t)$ ,  $T$ )** {  
 initialize  $\Theta$  and  $\Gamma$  to be empty  
 add  $Satisfied(topgoal, T)$  to  $\Phi$   
 for each goal  $g$  corresponding to an atom in  $\Phi$  {  
   set  $t =$  timestep in the  $Satisfied(g,t)$  ground atom  
   for each task  $a$  child of goal  $g$  {  
     load rule  $r$  that shows the contribution of  $Satisfied(a,t1)$  to  $G\_Occ(g,t)$   
     identify the value of  $t1$   
     add ground atoms  $Satisfied(a,t1)$  and its weight to  $\Theta$   
   }  
 }  
 remove  $Satisfied(g1,t)$  and add  $Satisfied$  ground atoms of all sub-goals of  $g$  to  $\Phi$   
 }  
 next, order the ground atoms in  $\Theta$  based on timesteps  
 for each ground atom  $atom$  in  $\Theta$  {  
   if weight of  $Satisfied(atom,t) \leq 0.5$  {  
     if  $a$  is an AND child  $\Rightarrow$  add  $a$  to  $\Gamma$   
     if  $a$  is an OR child {  
       find siblings of  $a$  in goal model  
       if no sibling of  $a$  is satisfied in  $\Theta \Rightarrow$  add  $a$  to  $\Gamma$   
       if  $a$  has at least one satisfied sibling  $\Rightarrow$  CONTINUE.}}}  
 return  $\Gamma$ }

---

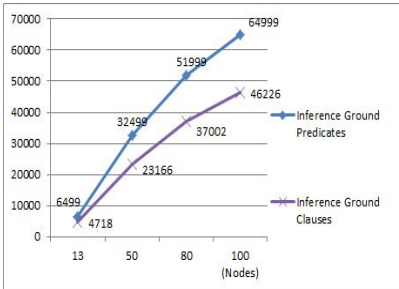
## 5 Case Study

The objective of this case study is to demonstrate the applicability of the proposed framework in detecting the root causes for failures in software systems that are composed of various components and services. The motivating scenario has been implemented as a proof of concept and includes 6 systems/services: a front end application (soapUI), a process server (IBM Process Server 6.1), a loan application business process, a message broker (IBM WebSphere Message Broker v7.0), a credit check Web Service and an SQL server (Microsoft SQL Server 2008). The case study consists of two scenarios. The two scenarios we present in this study include one success and one failure scenario (see Table 1). Scenario 1 represents a successful execution of the loan application process. Please note, that the denial of task  $a_7$  does not represent a failure in the process execution since goal  $g_3$  is exclusive OR-decomposed into  $a_6$  (*extract credit history for existing clients*) and  $a_7$  (*calculate credit history for new clients*), and the successful execution of either  $a_6$  or  $a_7$  is enough for  $g_3$ 's successful occurrence (see Fig. 2). During each loan evaluation and before the reply is sent back to the requesting application, a copy of the decision is stored in a local table ( $a_4$  (*Update loan table*)). Scenario 2 represents a failure to update the loan table leading to failure of top goal  $g_1$ . Using Algorithm 2, we identify  $a_4$  (Update loan table) as the root cause for failure. Note that although  $a_7$  was denied ahead of task  $a_4$ , it is not the root cause since it is an OR child of  $g_3$ , and its sibling task  $a_6$  was satisfied.

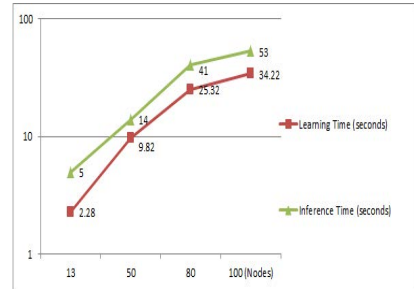
**Table 1.** Two scenarios for the Loan Application

Scenario	Observed (& Missing) Events(s)	Satisfied(task, timestep)						
		(a1,3)	(a3,5)	(a6,7)	(a7,7)	(a4,9)	(a5,11)	(a2,13)
Successful execution	Pre( $g_1,1$ ), Pre( $a_1,1$ ), Occ( $a_1,2$ ), Post( $a_1,3$ ), Pre( $g_2,3$ ), Pre( $a_3,3$ ), Occ( $a_3,4$ ), Post( $a_3,5$ ), Pre( $g_2,5$ ), Pre( $a_6,5$ ), Occ( $a_6,6$ ), Post( $a_6,7$ ), ?Pre( $a_7,5$ ), ?Occ( $a_7,6$ ), ?Post( $a_7,7$ ), Post( $g_3,7$ ), Pre( $a_4,7$ ), Occ( $a_4,8$ ), Post( $a_4,9$ ), Pre( $a_5,9$ ), Occ( $a_5,10$ ), Post( $a_5,11$ ), Pre( $a_2,11$ ), Occ( $a_2,12$ ), Post( $a_2,13$ ), Post( $g_1,13$ ), Satisfied( $g_1,13$ )	0.88	0.87	0.93	0.01	0.66	0.63	0.82
Failed to update loan database	Pre( $g_1,1$ ), Pre( $a_1,1$ ), Occ( $a_1,2$ ), Post( $a_1,3$ ), Pre( $g_2,3$ ), Pre( $a_3,3$ ), Occ( $a_3,4$ ), Post( $a_3,5$ ), Pre( $g_3,5$ ), Pre( $a_6,5$ ), Occ( $a_6,6$ ), Post( $a_6,7$ ), ?Pre( $a_7,5$ ), ?Occ( $a_7,6$ ), ?Post( $a_7,7$ ), Post( $g_3,7$ ), Pre( $a_4,7$ ), ?Occ( $a_4,8$ ), ?Post( $a_4,9$ ), ?Pre( $a_5,9$ ), ?Occ( $a_5,10$ ), ?Post( $a_5,11$ ), ?Pre( $a_2,11$ ), ?Occ( $a_2,12$ ), ?Post( $a_2,13$ ), ?Post( $g_1,13$ ), !Satisfied( $g_1,13$ )	0.88	0.87	0.90	0.01	0.01	0.01	0.01

The probability values (weights) of the ground atoms range  $0^+$  (highly denied) to 0.99 (highly satisfied). The framework was evaluated on a Ubuntu Linux running on Intel Pentium 2 Duo 2.2 GHz machine. We have used a set of extended goal models representing the loan application goal model to evaluate the performance of the framework when larger goal models are used. The 4 extended goal models contained 13, 50, 80 and 100 nodes respectively. Obtained results indicate that the matching and ground atom generation algorithms performance depends linearly on the size of the corresponding goal model and log data and thus lead us to believe that the process can easily scale for larger and more complex goal models. Our experiments have also suggested that the number of ground atoms/clauses, which directly impacts the size of the resulting Markov model, is linearly proportional to the goal model size, and that the number of ground atoms and clauses increases linearly with the size of the goal model (see Figure 5). Furthermore, results indicate that the learning and inference time ranged from 2.2 and 5 seconds for a goal model of 10 nodes, up to 34.2 and 53 seconds respectively for a model of 100 nodes (see Figure 6). As a result, these initial case studies suggest that our approach in its current implementation can be applied to industrial software applications with small to medium-sized requirement models (i.e. 100 nodes).



**Fig. 5.** Number of Ground Predicates/Clauses vs. Goal Model Size



**Fig. 6.** Learning and Inference Time vs. Goal Model Size

## 6 Related Work

Our current study is based in part on earlier work by Wang [14] and our previous work [16], [17]. Wang et al. [14] proposed annotated goal models to represent monitored systems and transformed the diagnostic problem into a propositional satisfiability (SAT) problem that can be solved using SAT solvers where evidence supporting or denying the truth of individual ground predicates is collected by instrumenting software systems. Zawawy et al. [16] enhanced the framework by Wang et al. by matching natively generated log data and using it as evidence which is less intrusive and more practical when analyzing off-the-shelf commercial products. The current study extends our previous work by introducing a diagnostic component based on MLNs allowing the handling of inaccuracies in

modeling the monitored systems dependencies as well as missing and inaccurate observations. Our approach uses weighted formulas instead of hard relationships allowing for conflicts such as when a goal/task is satisfied and denied at the same time. This is an advantage over [14] which depends on accurate observation. RCA approaches in the literature can be classified as based on probabilistic approaches such as Bayesian Belief Networks [12], or based on machine learning such as decision trees and data mining [1], [3], or based on rule sets and decision matrices [7]. Steinder et al. (2004) used Bayesian networks to represent dependencies between communication systems and diagnose failures while using dynamic, missing or inaccurate information about the system structure and state [12]. Our approach is similar in that we use Markov networks (undirected graphs) instead of Bayesian networks in order to fit the evidence. The advantage of our approach over [12] is that using first order logic; we are able to model complex relationships between the monitored systems and not just simple one to one causality relationships. A more recent work pertaining to log analysis and reduction is the work by Al-Mamory et al. where RCA is used to reduce the large number of alarms by finding the root causes of false alarms [1]. [1] uses data mining to group similar alarms into generalized alarms and then analyze these generalized alarms and classify them into true and false alarm. Later, the generalized alarms can be used as rules to filter out false positives. Chen et al. (2004) use statistical learning to build a decision tree which represents all the paths that lead to failures [3]. The decision tree is built by branching for each node based on the values of a specific feature and later on pruning subsumed branches. The advantages of our approach over [1], [3] is that it is resilient to missing or inaccurate log trace, adapts dynamically for new observations, and can be used to diagnose multiple simultaneous faults.

## 7 Conclusions

This paper presents a framework that assists operators perform RCA in software systems. The framework takes a goal driven approach whereby software system requirements are modeled as goal trees. Once the failure of a functional or non-functional requirement is observed as a symptom, the corresponding goal tree is analyzed. The analysis takes the form of first, selecting the events to be considered based on a pattern matching process, second on a rule and predicate generation process where goal models are denoted as Horn Clauses and third, a probabilistic reasoning process that is used to confirm or deny the goal model nodes. The most probable combinations of goal model nodes that can explain the failure of the top goal (i.e. the observed failure) is considered the most probable root cause. Goal models for large systems can be organized in a hierarchical fashion allowing for higher tractability and modularity in the diagnostic process. Initial results indicate that the approach is tractable and allows for multiple diagnoses to be achieved and ranked based on their probability of occurrence. This work is conducted in collaboration with CA Labs and is funded by CA Technologies and the Natural Sciences and Engineering Research Council of Canada.

## References

1. Al-Mamory, S.O., Zhang, H.: Intrusion detection alarms reduction using root cause analysis and clustering. *Comput. Commun.* 32(2), 419–430 (2009)
2. Alexe, B., Chiticariu, L., Miller, R.J., Tan, W.C.: Muse: Mapping understanding and design by example. In: *Proceedings of the 24th International Conference on Data Engineering, ICDE 2008, Cancun, Mexico, April 7-12*, pp. 10–19 (2008)
3. Chen, M., Zheng, A.X., Lloyd, J., Jordan, M.I., Brewe, E.: Failure diagnosis using decision trees. In: *Int'l. Conference on Autonomic Computing*, pp. 36–43 (2004)
4. Dollimore, J., Kindberg, T., Coulouris, G.: *Distributed Systems: Concepts and Design*, 4th edn. *Int'l Computer Science Series*. Addison Wesley (May 2005)
5. Domingos, P.: Real-World Learning with Markov Logic Networks. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) *ECML 2004. LNCS (LNAI)*, vol. 3201, p. 17. Springer, Heidelberg (2004)
6. Giorgini, P., Mylopoulos, J., Nicchiarelli, E., Sebastiani, R.: Reasoning with Goal Models. In: Spaccapietra, S., March, S.T., Kambayashi, Y. (eds.) *ER 2002. LNCS*, vol. 2503, pp. 167–181. Springer, Heidelberg (2002)
7. Hanemann, A.: A hybrid rule-based/case-based reasoning approach for service fault diagnosis. In: *AINA 2006: Proceedings of the 20th International Conference on Advanced Information Networking and Applications*, pp. 734–740. IEEE Computer Society, Washington, DC (2006)
8. Kok, S., Sumner, M., Richardson, M., Singla, P., Poon, H., Lowd, D., Domingos, P.: The alchemy system for statistical relational ai. technical report, university of washington, seattle, wa (2007), <http://alchemy.cs.washington.edu>
9. Library, M.M.: Transact-sql reference (2012), [http://msdn.microsoft.com/en-us/library/ms179859\(v=sql.100\).aspx#2](http://msdn.microsoft.com/en-us/library/ms179859(v=sql.100).aspx#2)
10. Oltsik, J.: The invisible log data explosion (2007), [http://news.cnet.com/8301-10784\\_3-9798165-7.html](http://news.cnet.com/8301-10784_3-9798165-7.html)
11. Richardson, M., Domingos, P.: Markov logic networks. *Mach. Learn.* 62, 107–136 (2006)
12. Steinder, M., Sethi, A.S.: Probabilistic fault diagnosis in communication systems through incremental hypothesis updating. *Comput. Netw.* 45(4), 537–562 (2004)
13. Tran, S.D., Davis, L.S.: Event Modeling and Recognition Using Markov Logic Networks. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) *ECCV 2008, Part II. LNCS*, vol. 5303, pp. 610–623. Springer, Heidelberg (2008)
14. Wang, Y., Mcilraith, S.A., Yu, Y., Mylopoulos, J.: Monitoring and diagnosing software requirements. *Automated Software Engg.* 16(1), 3–35 (2009)
15. Yu, Y., Lapouchnian, A., Liaskos, S., Mylopoulos, J., Leite, J.: From goals to high-variability software design, pp. 1–16 (2008)
16. Zawawy, H., Kontogiannis, K., Mylopoulos, J.: Log filtering and interpretation for root cause analysis. In: *ICSM 2010: Proceedings of the 26th IEEE International Conference on Software Maintenance* (2010)
17. Zawawy, H., Mylopoulos, J., Mankovski, S.: Requirements driven framework for root cause analysis in soa environments. In: *MESOA 2010: Proceedings of the 4th Int'l Workshop on Maintenance and Evolution of Service-Oriented Systems* (2010)

# Validation of User Intentions in Process Models

Gerd Gröner<sup>1</sup>, Mohsen Asadi<sup>2</sup>, Bardia Mohabbati<sup>2</sup>, Dragan Gašević<sup>3</sup>,  
Fernando Silva Parreiras<sup>4</sup>, and Marko Bošković<sup>3</sup>

<sup>1</sup> WeST Institute, University of Koblenz-Landau, Germany  
groener@uni-koblenz.de

<sup>2</sup> Simon Fraser University, Canada  
{masadi,mohabbati}@sfu.ca

<sup>3</sup> Athabasca University, Canada  
{dragang,marko.boskovic}@athabascau.ca

<sup>4</sup> FUMEC University, Brazil  
fernando.parreiras@fumec.br

**Abstract.** Goal models and business process models are complementary artifacts for capturing the requirements and their execution flow in software engineering. Usually, goal models serve as input for designing business process models, and it requires mappings between both types of models. Due to the large number of possible configurations of elements from both goal models and business process models, developers struggle with the challenge of maintaining consistent configurations of both models and their mappings. Managing these mappings manually is error-prone. In our work, we propose an automated solution that relies on Description Logics and automated reasoners for validating mappings that describe the realization of goals by activities in business process models. The results are the identification of two inconsistency patterns – strong inconsistency and potential inconsistency, and the development of the corresponding algorithms for detecting inconsistencies.

**Keywords:** requirement modeling, goal-oriented process engineering, inconsistency detection.

## 1 Introduction

With the growing importance of process-aware information systems (PAISs), business process modeling has gained a significant research attention [1]. A business process model is an operational representation of activities, their ordering and routing in achieving goals; that is, delivering services to customers. However, business process modeling is not an effective way to understand and elicit user requirements and intentions in the development life-cycle of PAISs.

Requirements engineering offers proven means for understanding user intentions. Goal-oriented modeling is a prominent formalism to describe requirements of a system in terms of goals and relationships (constraints) between goals. A goal describes a certain system functionality or property that should be achieved. It is

not surprising then that the recent research on PAISs has focused on integration of business process modeling with goal-oriented modeling (c.f. Sect. 7). Related research concentrates on basic mapping and alignment principles between goal models [2,3,4] or business models [5,6] and process models. Their main focus is either on enriching a process model with goals and relationships between goals or on transformations between goal models and process models. However, less attention has been paid to validation formalisms for the correctness of the mappings between goal-oriented models and business process models.

In this paper, we tackle the challenge of automated validation of the correctness of mappings between goal models and business process models. Such mappings define which parts of a business process model are responsible for the realization of a certain user goal. In that sense, an automated validation of the mappings needs to assure satisfaction of user goals in each execution configuration/path of business processes.

To address the challenge of the mapping validation, our first contribution (Sect. 3) is the definition of the types of inconsistencies between goal and process models based on the correspondence between intentional relationships and workflow patterns [7]. Next, based on these correspondences, we propose a modeling (Sect. 4) and validation (Sect. 5) approach in Description Logics. Our approach ensures realization equivalence between mapped goals and their corresponding activities; that is, there are no contradictions between relations of goals compared to their mapped activities.

## 2 Foundations

This section starts with background information on the requirement perspective and the design perspective of business process models. Furthermore, the notion of goal realization and the consequential problem statement is presented.

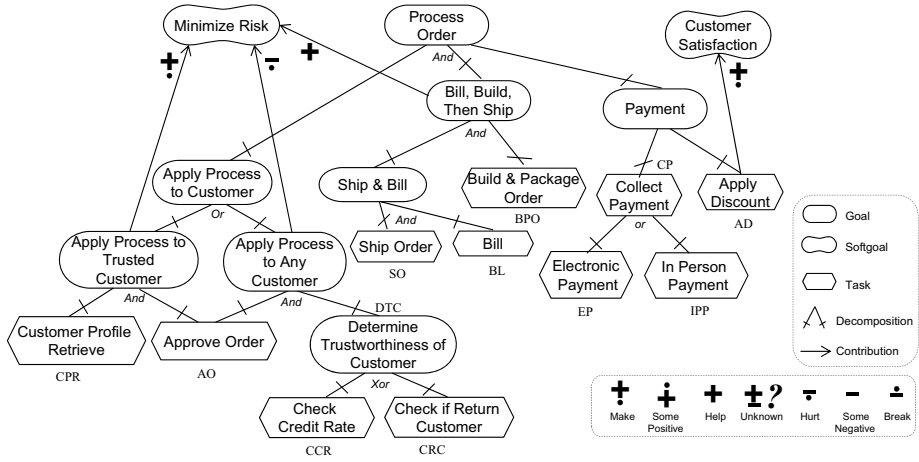
### 2.1 Goal Models

To date, several languages for goal models have been proposed, e.g., i\*/Tropos [8,9], NFR [10], KAOS [11] and Goal Requirements Language (GRL) [12]. We use the GRL, a part of the recent Recommendation of the International Telecommunications Union named User Requirements Notation (URN) [12]. GRL is a language that integrates core concepts of i\* and NFR [13].

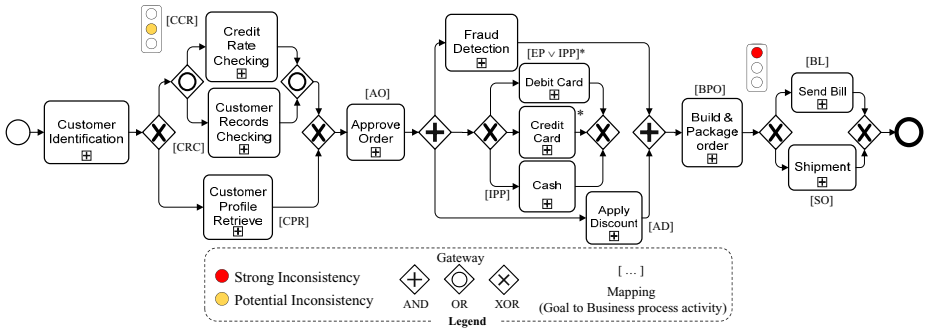
A goal model is a graph, consisting of *actors*, *intentional elements*, *links* and *decompositions* [12]. *Actors* are entities that can have intentions and carry out actions. Typically, they are stakeholders or systems.<sup>1</sup> *Intentional elements* used in this paper are (*hard*) *goals*, *soft goals* and *tasks*. *Soft goals* are similar to hard

---

<sup>1</sup> In this paper, however, we are interested in validation of user intentions in service orchestrations, typically achieved within one actor. Therefore, in the later text we do not consider satisfaction levels of actors nor dependencies.



(a) Goal Model of the E-Store



(b) Business Process Model of the E-Store

Fig. 1. Goal Realizations in Business Process Models

goals, but without a clear-cut criteria for whether the condition is achieved. They model non-functional requirements<sup>2</sup>. *Tasks* specify conceptual solutions.

A concrete goal model is depicted in Fig. 1(a). Links connect intentional elements. They can be either *decompositions* or *contributions*. *Decompositions* allow for a specification of what source intentional elements need to be satisfied in order to satisfy the target intentional elements. For instance, the goal Ship & Bill is achieved if both tasks Ship Order and Bill are fulfilled. GRL supports *AND*, *IOR* and *XOR* decompositions. An *AND* decomposition specifies that all source intentional elements need to be satisfied for the target intentional element to be satisfied. *IOR* is used to specify that the satisfaction of at least

<sup>2</sup> Some non-functional requirements, like security, can have clear cut criteria and can also be achieved with different operationalizations. The GRL standard does not specify how to model these situations. Nevertheless, the standard allows for specification of decompositions on soft goals.



one source satisfies the target, whereby *XOR* specifies that exactly one of the source elements is necessary to satisfy the target.

*Contribution* types can be seen in Fig. 1(a). The *Make* and *Break* contributions are respectively positive and negative, and sufficient for satisfaction of a target element. *Help* and *Hurt* are also respectively positive and negative, but insufficient. The extent of the contribution of *SomePositive* and *SomeNegative* is unknown. Finally, for the *Unknown* contribution link, both the extent and degree (positive or negative) of the contribution are unknown.

**Definition 1 (Goal Model).** *A goal model is a triple  $GM = (\mathcal{G}, \mathcal{C}, \mathcal{D})$ .  $\mathcal{G}$  is a set of goals (also called intentions or intentional elements). Intentions are (hard) goals ( $\mathcal{G}_g$ ), tasks ( $\mathcal{G}_t$ ) and soft goals ( $\mathcal{G}_s$ ).  $\mathcal{C}$  denotes positive and negative contributions ( $\mathcal{G} \times \{\ddagger, \ddagger, +, \pm?, \mp, -, \pm\} \times \mathcal{G}$ ).  $\mathcal{D}$  is a decomposition of intentional elements  $\mathcal{G} \times \{IOR, XOR, AND\} \times \mathcal{P}(\mathcal{G})$ , whereby an intention  $G \in \mathcal{G}$  is fulfilled either if at least one, exactly one or all source intentions are fulfilled.*

## 2.2 Business Process Models and Workflow Patterns

There is an emergence and proliferation of process-oriented software development methods for enterprises, where software is designed, built, executed by process engines, maintained and evolved on the basis of business process models [11, 14]. A business process can be considered as a set of ordered activities intended to realize and implement a goal [15] according to requirement models.

Business process models can be designed at different levels of abstraction. Currently, a number of graph-based business process modeling languages exists, e.g., BPMN, EPC, YAWL and UML Activity Diagram. Although the proposed solution in this paper is generic, we use here the Business Process Modeling Notation (BPMN). Despite of variance in expressiveness and modeling notations, all modeling languages share the common concepts of activities, gateways or routing nodes, artifacts and relations between them represented as control flow.

A business process model is illustrated in Fig. 1(b). Mappings between activities and tasks in the goal model are indicated by activity annotations. E.g., the activity *Credit Rate Checking* is mapped to goal *Check Credit Rate (CCR)*.

**Definition 2 (Business Process Model).** *A business process model is defined as a connected graph  $G_{PM} = (\mathcal{V}, \mathcal{E})$ , where  $\mathcal{V}$  depicts a set of vertices including a set of activities  $\mathcal{A}$  and gateways  $\mathcal{G}$ . A gateway has a type  $T(G)$  such that  $T(G) \in \{xor, or, and, disc\}$ .  $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$  denotes a set of edges between vertices.*

**Definition 3 (Materialized Business Process Model).** *Given a business process model  $G_{PM} = (\mathcal{V}, \mathcal{E})$ , a materialized process model  $PM$  is a quadruple  $(\mathcal{V}, \mathcal{E}, \mathcal{F}, \mathcal{E}_A)$ . The set  $\mathcal{F} \subseteq \mathcal{V} \times \mathcal{V} \times \mathcal{B}$  represents single-entry-single-exit (SESE) fragments, in which  $\mathcal{B}$  represents a set of branches between entry and exit vertex. A branch  $B \in \mathcal{B}$  is considered as a set of activities.  $\mathcal{E}_A$  is a set of sequence edges that are obtained from the process graph by treating gateways as transparent.*

SESE-fragments can be derived from a process model in linear time (cf. [16]). We consider structural well-formedness conditions for business process models [17][18][19]. Gateways have either exactly one incoming edge and multiple outgoing edges or multiple incoming edges and exactly one outgoing edge. The set  $\mathcal{E}_A$  is derived from  $\mathcal{E}$  by neglecting gateways and replacing them by their corresponding next predecessor or successor activity. Workflow patterns [20] describe structures and behavior of processes for the execution. They are defined in terms of how the process flow proceeds in sequences and splits into branches for executing the activities and how they converge. We select the main patterns of the Workflow Patterns framework<sup>3</sup> as a reference analysis framework.

### 3 Realization Inconsistencies

In goal-oriented requirements engineering, tasks are considered requirements if they are assigned into a system-to-be. To realize requirements (i.e., tasks in the goal model), activities (either atomic or composite, i.e., sub-processes) are defined in business process models. The execution relations between activities (i.e., workflow patterns) in business process models must be consistent with intentional relations between tasks and other intentional elements in goal models.

**Definition 4 (Realization Equivalence).** *Assume activities  $A_1, \dots, A_n \in \mathcal{A}$  are realizations of intentional elements  $G_1, \dots, G_m \in \mathcal{G}_t$ . A workflow pattern  $WF$  exists between activities  $A_1, \dots, A_n$ , and an intentional relation  $IR$  exists between intentional source elements  $G_1, \dots, G_m$  and an intentional target element  $G \in \mathcal{G}$ .  $WF$  is realization equivalent to  $IR$ , if all execution combinations of activities in  $WF$  lead to the satisfaction of target goal  $G$ .*

If  $WF$  is defined as a realization of  $IR$  in a business process model and there is no realization equivalence between these two relations, an inconsistency in the process model can occur with respect to the goal model. We define two types of inconsistencies: 1) *strong inconsistency* and 2) *potential inconsistency*.

**Definition 5 (Strong Inconsistency).** *Assume a workflow pattern  $WF$  is specified over activities  $A_1, \dots, A_n \in \mathcal{A}$  and an intentional relation  $IR$  with target element  $G \in \mathcal{G}$  is defined over intentional elements  $G_1, \dots, G_m \in \mathcal{G}_t$  (e.g., an AND decomposition  $G$  is a parent intentional element and the rest are children). Activities  $A_1, \dots, A_n$  are realizations of intentional elements  $G_1, \dots, G_m$ . A strong inconsistency between  $WF$  and  $IR$  occurs if there is no execution combination of activities that leads to the fulfillment of the target element  $G$ .*

**Definition 6 (Potential Inconsistency).** *Assume a workflow pattern  $WF$  is specified over activities  $A_1, \dots, A_n \in \mathcal{A}$  and an intentional relation  $IR$  with target element  $G \in \mathcal{G}$  is defined over intentional elements  $G_1, \dots, G_m \in \mathcal{G}_t$ . Activities  $A_1, \dots, A_n$  are realizations of intentional elements  $G_1, \dots, G_m$ . We define*

<sup>3</sup> <http://www.workflowpatterns.com>

a potential inconsistency between WF and IR if some execution combinations of activities lead to the fulfillment of intentional element G and some execution combinations of activities do not lead to the fulfillment of G.

The example in Fig. 1 contains a strong and a potential inconsistency. The strong inconsistency is due to the activities Send Bill and Shipment that are mapped to tasks Bill (BL) and Ship Order (SO), whereby these tasks are AND-siblings. Thus, each satisfaction of the target element Ship & Bill requires that both tasks Bill (BL) and Ship Order (SO) are fulfilled simultaneously, while each process execution allows either the execution of Send Bill or Shipment.

**Table 1.** Correspondence between Intentional Relations and Workflow Patterns

Workflow Patterns	Intentional Relations				
	AND	IOR	XOR	±	±
AND-AND Parallel split - Synchronization	✓	✓	↯	✓	↯
AND-OR Parallel split - Multi merge	✓	✓	↯	✓	↯
AND-DISC Parallel split - Discriminator	✓	✓	↯	✓	↯
AND-XOR Parallel split - Simple merge	✓	✓	↯	✓	↯
OR-OR Multi choice - Synchronizing merge	±	✓	±	±	±
OR-DISC Multi choice - Discriminator	±	✓	±	±	±
OR-XOR Multi choice - Simple merge	±	✓	±	±	±
XOR-XOR Exclusive - Simple merge	↯	✓	✓	±	✓
Sequence	✓	✓	↯	✓	↯

A potential inconsistency is caused by the mapping of activities Credit Rate Checking and Customer Records Checking to the exclusive sibling tasks Check Credit Rate (CCR) and Check if Return Customer (CRC). There are executions where both activities are executed, but this would contradict to the exclusiveness of the tasks Check Credit Rate (CCR) and Check if Return Customer (CRC) to fulfill their target goal DTC.

Combinations of intentional relations (IR) and workflow patterns (WF) are shown in Table 1. If a workflow pattern WF is realization equivalent to an intentional relation IR, the corresponding cell is marked with ✓. Strong and potential inconsistencies are shown with (↯) and (±), respectively. If a set of workflow patterns  $WF_1, WF_2, \dots, WF_n$  are realizations of an intentional relation IR, then all combinations  $(WF_1, IR), \dots, (WF_n, IR)$  should be realization equivalent.

Regarding the mappings between the goal models and business process models, we consider two assumptions: i) Only tasks in the goal model are mapped to atomic activities or composite activities (sub-processes) in the business process models, since only tasks are operationalizations and realizations; thus, hard goals and soft goals are not mapped to activities. ii) If there are unmapped activities within a business process model, then those activities do not contribute to any realization.

## 4 Knowledge Base for Realization Validation

We use Description Logics (DL) [21] to model workflow patterns, intentional relations and mappings. Based on this representation, DL reasoning services are used to validate realizations of intentional elements by workflow patterns.

### 4.1 Foundations of Description Logics

DL is a decidable subset of first-order logic (FOL). A DL knowledge base consists of a TBox (Terminological Box) and an ABox (Assertional Box). The TBox is used to specify concepts, which denote sets of individuals and roles defining binary relations between individuals. The main syntactic constructs are depicted in Table 2, supplemented by the corresponding FOL expressions. Concept inclusion axioms  $C \sqsubseteq D$  mean that each individual of the concept  $C$  is also an individual of  $D$ . There are two special concepts in Description Logics, namely the universal concept (top concept)  $\top$  and the bottom concept  $\perp$ . The top concept  $\top$  is the superconcept of all concepts, i.e.,  $C \sqsubseteq \top$  holds for each concept  $C$ .  $\perp$  is an unsatisfiable concept. A concept equivalence (or definition)  $C \equiv D$  is an abbreviation for two concept inclusion axioms  $C \sqsubseteq D$  and  $D \sqsubseteq C$ .

Inference services of DL rely on the well defined Tarski-style semantics. The subset of DL constructs we use in our models ( $\mathcal{ALC}$  expressiveness) in combination with existing highly optimized reasoning algorithms and systems allow for practical efficient reasoning support. In the remainder of this paper, we use subsumption checking and concept classification as basic reasoning services.

### 4.2 Representation of Models and Realizations

The key part of our modeling formalism contains the relations of both models, combined with mappings between them. The goal model describes *intentional relations* between goals and the business process model specify *control flow relations* on activities, which refer to basic *workflow patterns* [7,20].

**Representation of Intentional Relations.** The *intentional relations* of a goal model  $GM$  are described in a DL knowledge base  $\Sigma_{GM}$ . Algorithm 1 depicts the representation of intentional relations of a goal model  $GM = (\mathcal{G}, \mathcal{C}, \mathcal{D})$  in a DL knowledge base  $\Sigma_{GM}$ . Only tasks are realized by activities. Thus, the algorithm introduces for each task  $G$  ( $G \in \mathcal{G}_t$ ) a concept  $Rel_G$  to capture its intentional relations, which are either decompositions or contributions. Intentional elements  $G$  are also concepts in DL.

Lines 2-4 treat IOR-decompositions of an intention into subgoals  $G_i$ . Thus, intentions  $G_i$  are disjunctively related to each other. This is represented in DL by a concept union over all intentions  $G_j$  that are members of the IOR-decomposition. For each intention  $G_i$  that is part of the IOR-decomposition, we introduce a concept  $Rel_{G_i}$  to describe the relations of each intention. In this vein, a conjunctive decomposition (lines 5-7) is described by a concept intersection in DL. As in the

**Table 2.** Constructs and Notations in DL and FOL Syntax

Construct Name	DL Syntax	FOL Syntax
atomic concept, atomic role	$C, R$	$C(x), R(x, y)$
concept inclusion axiom	$C \sqsubseteq D$	$\forall x. C(x) \rightarrow D(x)$
concept union	$C_1 \sqcup \dots \sqcup C_n$	$C_1(x) \vee \dots \vee C_n(x)$
concept intersection	$C_1 \sqcap \dots \sqcap C_n$	$C_1(x) \wedge \dots \wedge C_n(x)$
concept negation	$\neg C$	$\neg C(x)$
universal quantification	$\forall P.C$	$\forall y.(P(x, y) \rightarrow C(y))$
existential quantification	$\exists P.C$	$\exists y.(P(x, y) \wedge C(y))$

**Algorithm 1.** Representation of the Intentional Relations  $\Sigma_{GM}$ 


---

```

1: Input: Goal model  $GM = (\mathcal{G}, \mathcal{C}, \mathcal{D})$ 
2: if  $(G, IOR, \{G_1, \dots, G_n\}) \in \mathcal{D}$  then
3:    $Rel_{G_i} \equiv \bigsqcup_{j=1, \dots, n} \exists requires.G_j$  (for  $i = 1, \dots, n$ )
4: end if
5: if  $(G, AND, \{G_1, \dots, G_n\}) \in \mathcal{D}$  then
6:    $Rel_{G_i} \equiv \prod_{j=1, \dots, n} \exists requires.G_j$  (for  $i = 1, \dots, n$ )
7: end if
8: if  $(G, XOR, \{G_1, \dots, G_n\}) \in \mathcal{D}$  then
9:    $Rel_{G_i} \equiv (\bigsqcup_{G' \in \{G_1, \dots, G_n\}} \exists requires.G'$ 
       $\sqcap \neg(\bigsqcup_{G'', G''' \in \{G_1, \dots, G_n\}} (\exists requires.G'' \sqcap \exists requires.G''')))$ 
10: end if
11: for all  $(G', \dagger, G) \in \mathcal{C}$  do
12:    $Rel_G := Rel_G \sqcap \exists requires.G'$ 
13: end for
14: for all  $(G', \blacktriangleright, G) \in \mathcal{C}$  do
15:    $Rel_G := Rel_G \sqcap \neg \exists requires.G'$ 
16: end for

```

---

previous case, we introduce relation concepts  $Rel_{G_i}$  to capture conjunction for all members of the decomposition.

The representation of exclusive decompositions is straightforward (lines 8-10). The concept union describes that at least one intentional element  $G'$  has to be satisfied in order to satisfy  $G$ , like in inclusive or decompositions. The second part of the expression excludes the case that more than one intention ( $G''$  and  $G'''$ ) is satisfied. This is expressed by the concept negation ( $\neg$ ) that precedes the second part of the concept expression. An intention can only be the source of one decomposition, i.e., either IOR, XOR or AND.

Sufficient positive contributions (lines 11-13) specify that the fulfillment of  $G$  requires the fulfillment of  $G'$ . Thus, we add the expression  $\exists requires.G'$  to the definition of the relation concept  $Rel_G$ . Sufficient negative contributions (lines 14-16) use concept negation in order to represent that the fulfillment of  $G$  can not be achieved if  $G'$  is fulfilled. A task might be involved in multiple (positive and negative) contributions simultaneously.

**Algorithm 2.** Representation of the Workflow Patterns  $\Sigma_{PM}$ 


---

```

1: Input: Materialized process model  $PM = (\mathcal{V}, \mathcal{E}, \mathcal{F}, \mathcal{E}_A)$ 
2: for all  $A \in \mathcal{A}$  ( $\mathcal{A} \subseteq \mathcal{V}$ ) do
3:    $Rel_A \equiv \top$ 
4: end for
5: for all  $E \in \mathcal{E}_A$  do
6:   if  $\langle A_1, A_2 \rangle = E$  then
7:      $Rel_{A_1} := Rel_{A_1} \sqcap \exists requires.A_2$  and  $Rel_{A_2} := Rel_{A_2} \sqcap \exists requires.A_1$ 
8:   end if
9: end for
10: for all  $F \in \mathcal{F}$  do
11:   if  $F = (and, and, \mathcal{B}) \vee F = (and, or, \mathcal{B}) \vee F = (and, xor, \mathcal{B}) \vee F = (and, disc, \mathcal{B})$ 
   then
12:      $Rel_{A_i} := Rel_{A_i} \sqcap \sqcap_{B_j \in \mathcal{B}} \exists requires.(\sqcap_{A_k \in B_j} A_k)$ 
13:   end if
14:   if  $F = (ior, ior, \mathcal{B}) \vee F = (ior, disc, \mathcal{B}) \vee F = (ior, xor, \mathcal{B})$  then
15:      $Rel_{A_i} := Rel_{A_i} \sqcap \sqcup_{B_j \in \mathcal{B}} \exists requires.(\sqcap_{A_k \in B_j} A_k)$ 
16:   end if
17:   if  $F = (xor, xor, \mathcal{B})$  then
18:      $Rel_{A_i} := Rel_{A_i} \sqcap (\sqcup_{B_j \in \mathcal{B}} \exists requires.(\sqcap_{A_k \in B_j} A_k))$ 
      $\sqcap \neg(\sqcap_{B_j \in \mathcal{B}} \exists requires.(\sqcap_{A_k \in B_j} A_k))$ 
19:   end if
20: end for

```

---

$$Rel_{ApproveOrder} \equiv \exists requires.ApproveOrder \sqcap \exists requires.CustomerProfileRetrieve \quad (1)$$

$$Rel_{ElectronicPayment} \equiv \exists requires.ElectronicPayment \sqcup \exists requires.InPersonPayment \quad (2)$$

Axiom **1** describes an AND-relation of the sibling intentional elements *ApproveOrder* and *CustomerProfileRetrieve*. The intentional relation of *CustomerProfileRetrieve* is defined equally. The AND-relation is expressed by the concept intersection ( $\sqcap$ ) of the concept expressions  $\exists requires.ApproveOrder$  and  $\exists requires.CustomerProfileRetrieve$ .

An inclusive OR-relation between *ElectronicPayment* or *InPersonPayment* is exemplified in Axiom **2**, in which the fulfillment relationship of both tasks is reflected, while both tasks are inclusive siblings within an OR-decomposition. The relation of intention *InPersonPayment* is defined equally.

**Workflow Relations.** We represent business process models in terms of control flow relations between activities (cf. Sect. **2**). Algorithm **2** describes how the corresponding knowledge base  $\Sigma_{PM}$  is built.

There might be an overlapping of activity relations. For instance, the activity Cash in Fig. **1(b)** is part of an exclusive branching fragment (internal fragment in Fig. **1(b)**) and also within a parallel branching fragment, i.e., the activity Cash is conjunctively and exclusively related to other activities. Accordingly, we build relations of activities ( $Rel_A$ ) as a conjunction (intersection in DL) of activities

from the different control flow patterns. Initially, each relation concept  $Rel_A$  is defined as equivalent to the universal concept  $\top$  (line 3).

In lines 5-9, the sequential control flow relations (in both directions) of an activity  $A$  are covered by restricting the concept  $Rel_A$ . Since gateways do not realize goals, they are transparent in the representation (cf. Def. 3). Afterwards, relations within fragments  $\mathcal{F}$  are considered, whereby only those fragments that start and end with a gateway are relevant. Each branch  $B \in \mathcal{B}$  of a fragment  $F \in \mathcal{F}$  is a set of activities. In lines 11-13, the algorithm restricts the concept definitions  $Rel_{A_i}$ , in which  $A_i$  are activities in parallel branches. We use an intersection between activity sets of sibling branches, indicating the conjunctive relationship between sibling activities in parallel branches. From a logical point of view, we treat different exit gateways (multiple merge, synchronization and discriminator) equally. Branching relations do not impose restrictions on activities within the same branch in a fragment (in contrast to the sequence pattern). Thus, we describe all activities of the same branch by a concept intersection, independent of the kind of branching.

Multi choices are treated in lines 14-16, including synchronizing merge, simple merge and discriminator. Logically, activities of sibling branches are connected by a concept union. In case of exclusive branchings (lines 17-19), the concept definitions  $Rel_{A_i}$  contain a further restriction that allows only the execution of one branch. Like in goal models, this is expressed by a concept negation ( $\neg$ ) in the DL concept definition. In both cases, activities of the same branch are treated like in the parallel case, i.e., they are represented by a concept intersection since IOR and XOR relations refer to activities of sibling branches, but not to activities of the same branch.

Axiom 3 depicts a part of the control flow relation of activity *CreditRateChecking* (cf. Fig. 1(b)). The activity is part of a choice fragment, i.e., either activity *CreditRateChecking* or *CustomerRecordsChecking* can be executed, or even both. This relation is represented by a concept union in the first line of the axiom. The second line of the axiom covers sequential relations of the activity *CreditRateChecking* to its predecessor (*CustomerIdentification*) and its successor (*ApplyDiscount*). Axiom 4 describes the relation of activity *FraudDetection*, as member of a parallel branch, in which activity *ApplyDiscount* and the internal fragment with activities *DebitCard*, *CreditCard* and *Cash*. The relation concept also covers predecessor (*SelectPaymentMethod*) and successor (*BuildAndPackageOrder*) activities.

$$\begin{aligned}
 Rel_{CreditRateChecking} &\equiv (\exists \text{ requires. } CreditRateChecking \\
 &\quad \sqcup \exists \text{ requires. } CustomerRecordsChecking) \\
 &\quad \sqcap \exists \text{ requires. } CustomerIdentification \sqcap \exists \text{ requires. } ApproveOrder \quad (3) \\
 Rel_{FraudDetection} &\equiv (\exists \text{ requires. } FraudDetection \sqcap \exists \text{ requires. } ApplyDiscount \\
 &\quad \sqcap \exists \text{ requires. } (DebitCard \sqcap CreditCard \sqcap Cash)) \\
 &\quad \sqcap \exists \text{ requires. } SelectPaymentMethod \\
 &\quad \sqcap \exists \text{ requires. } BuildAndPackageOrder \quad (4)
 \end{aligned}$$

**Mapping between Goals and Activities.** Besides intentional relations and workflow patterns, we have to represent the realization of tasks by the corresponding activities in terms of mappings in the knowledge base  $\Sigma_M$ . A mapping is described as a concept equivalence in the knowledge base. If there is a mapping  $m(G, A)$  from a task  $G$  to an activity  $A$ , we represent the mapping by an axiom  $A \equiv G$ . In both models, we use the same role *requires* in order to allow for a comparison of relations of both models.

## 5 Validation of Realization Equivalence

Mappings describe the realization of an intentional element by an activity. As a consequence, it is expected that the relations of an intentional element and its corresponding activity are not contradicting.

For a given mapping  $m(G, A)$ , we compare the corresponding workflow patterns WF of activity  $A$  and the intentional relations IR of intentional element  $G$  in order to test whether they are realization equivalent or not. Their relations are represented by concepts  $Rel_G$  and  $Rel_A$ . In case they are not realization equivalent, we want to know whether the reason is a strong inconsistency or a potential inconsistency, regarding to the distinction of Table 1 (Sect. 3).

From a logical point of view, concepts  $Rel_G$  and  $Rel_A$  represent formulas. The three different cases of inconsistencies and realization equivalence are reflected by the concepts  $Rel_G$  and  $Rel_A$  as follows: (i) A strong inconsistency means that there can not be any execution combination of activities that fulfills the intentional relations of the corresponding intentional elements. In the DL sense, the intersection of both concepts  $Rel_G \sqcap Rel_A$  is unsatisfiable, i.e., the intersection  $Rel_G \sqcap Rel_A$  cannot have a common individual. (ii) A potential inconsistency indicates that there might be execution combinations of activities, in which the corresponding intentional relations are not fulfilled. In this case, the intersection  $Rel_G \sqcap Rel_A$  is satisfiable, i.e., there are common individuals of both concepts. (iii) The intentional relations of  $G$  and the workflow patterns of activity  $A$  are realization equivalent if all execution combinations that involve activity  $A$  lead to a fulfillment of the intentional relations of  $G$ . This is true if the subsumption  $Rel_A \sqsubseteq Rel_G$  holds. Logically, this means that  $Rel_A$  implies  $Rel_G$ .

In order to check these three different cases, we introduce the following validation concepts. The concept  $Valid^\vee$  is defined as  $\neg Rel_A \sqcup Rel_G$  to encode the subsumption test  $Rel_A \sqsubseteq Rel_G$ . Thus,  $Rel_A$  is subsumed by  $Rel_G$  if  $Valid^\vee \equiv \neg Rel_A \sqcup Rel_G$  is equivalent to the universal concept  $\top$ . This indicates the realization equivalence. A concept  $Valid^\pm$  is defined as the intersection  $Rel_A \sqcap Rel_G$  to test whether there is a potential or a strong inconsistency, i.e., if  $Valid^\pm$  is satisfiable there is a potential inconsistency, otherwise a strong inconsistency. Formally, the knowledge base  $\Sigma$  is obtained as described in Def. 7.

**Definition 7 (Final Knowledge Base  $\Sigma$ ).** *The knowledge base  $\Sigma := \Sigma_{GM} \cup \Sigma_{PM} \cup \Sigma_M$  is extended as follows: For each mapping  $m(G, A)$*



from an intentional element  $G$  to an activity  $A$ , which is represented in  $\Sigma_M$  by an axiom  $G \equiv A$ , we insert the following axioms into  $\Sigma$ :

(1)  $Valid_{A,G}^{\checkmark} \equiv \neg Rel_A \sqcup Rel_G$  and (2)  $Valid_{A,G}^{\pm} \equiv Rel_A \sqcap Rel_G$

Given the final knowledge base, we get the validation result *en passant*. Classifying the validation concepts  $Valid_{A,G}^{\checkmark}$  and  $Valid_{A,G}^{\pm}$  of the knowledge base  $\Sigma$  for each mapping  $m(G, A)$  leads to the following results:

1. If  $Valid_{A,G}^{\checkmark}$  is classified equal to the universal concept  $\top$ , we can guarantee the realization equivalence of the workflow patterns over activity  $A$  and the intentional relations over intentional element  $G$ .
2. In the other case, there is either a strong inconsistency or a potential inconsistency. This is indicated by the classification of the concept  $Valid_{A,G}^{\pm}$ . If  $Valid_{A,G}^{\pm}$  is classified as a subconcept of the bottom concept  $\perp$  (i.e.,  $Valid_{A,G}^{\pm} \sqsubseteq \perp$ ), there is a strong inconsistency, otherwise (i.e.,  $Valid_{A,G}^{\pm} \not\sqsubseteq \perp$ ) there is a potential inconsistency.

## 6 Proof-of Concept and Discussion

We conduct an evaluation by providing a proof-of concept, in which transformations of workflow patterns from BPMN models and intentional relations from GRL goal models into an OWL DL knowledge base are implemented.

**Setting and Data Set.** To calibrate our approach, we have used 20 goal models and 20 BPMN models, derived in different variants from the e-store case study [22]. The goal models have on average 24 goals and about 60 % of the goals are tasks, which can be realized by activities. The average size of the business process models is about 21 activities, the maximum number is 34 activities and the maximum depth of nested fragments is 4. In each setting, about 80 % of the activities are mapped to at least one task of the goal model.

Given a goal model, a business process model and mappings between both models, our tool creates the knowledge base  $\Sigma$  in an average time of 2480 msec. The DL expressiveness of  $\Sigma$  is  $\mathcal{ALC}$ . After reasoning on the knowledge base  $\Sigma$ , our tool produces a list of realization equivalent mappings ( $Valid^{\checkmark} \equiv \top$ ) and a list of strong violations  $Valid^{\pm} \sqsubseteq \perp$ , the remaining are known as potential violations (i.e.,  $Valid^{\pm} \not\sqsubseteq \perp$ ). The reasoning time for the classification is on average 3480 msec. The ontology creation is implemented with the OWL-API [4]. For reasoning, we used the Pellet reasoner [5]. Our test system is a Notebook with an Intel Core 2 Duo 8700 CPU (2.5 GHz, 4 MB cache and 2GB DDR2 RAM).

**Validation Exemplified.** We demonstrate the validation for an excerpt of Fig. 11. Consider the strong inconsistency for the activities *SendBill* and *Shipment*. They are siblings in exclusive branches of the same fragment. In

<sup>4</sup> OWL-API site: <http://owlapi.sourceforge.net/>

<sup>5</sup> Pellet reasoner site: <http://clarkparsia.com/pellet/>

$\Sigma_{PM}$ , there are the definitions of this relation for both activities. An excerpt of the relation definition of activity *SendBill* is shown in Axiom 5.

$$\begin{aligned} Rel_{SendBill} &\equiv (\exists \text{requires.SendBill} \sqcup \exists \text{requires.Shipment}) \\ &\quad \sqcap \neg(\exists \text{requires.SendBill} \sqcap \exists \text{requires.Shipping}) \end{aligned} \quad (5)$$

$$Rel_{Bill} \equiv \exists \text{requires.ShipOrder} \sqcap \exists \text{requires.Bill} \quad (6)$$

Since *SendBill* is mapped to the intentional element Bill (BL), *SendBill* is defined as equivalent to the concept *Bill* and *Shipping* is equivalent to *ShipOrder* (mapping knowledge base  $\Sigma_M$ ). From the goal model, there is a relation definition of *Bill* in  $\Sigma_{GM}$  as depicted in Axiom 6.

The validation concept  $Valid^\vee \equiv \neg Rel_{SendBill} \sqcup Rel_{Bill}$  is classified by the reasoner as different from the universal concept  $\top$ , i.e., we know that the realization equivalence does not hold between *SendBill* and *Bill*. The other validation concept  $Valid^\pm \equiv Rel_{SendBill} \sqcap Rel_{Bill}$  is classified equivalent to the bottom concept  $\perp$ , i.e., indicating a strong inconsistency. The same holds for *Shipment*.

**Lessons Learned and Limitations.** As mentioned in Sect. 3, we restrict mappings to atomic tasks in the goal model since they operationalize stakeholders' goals, and therefore, they can be directly realized by activities in a process. Discussions whether hard and especially soft goals could be mapped to activities are outside of the scope of this paper. The proposed modeling framework is based on these assumptions. However, the modeling and validation approach is quite generic and DL is expressive enough to incorporate further aspects like mappings of other intentional elements like soft goals to activities.

The purpose of the validation is to detect inconsistencies between mapped elements with respect to their relationships, i.e., intentional relations and relations of activities. This is based on a comparison how these elements are logically related to each other, e.g., whether their relations (or constraints) coincide or contradict each other. However, an analysis whether or to which degree the execution of an activity satisfies a particular goal from a qualitative perspective leads to further interesting research questions.

## 7 Related Work

The first group of related work considers relations between goal models and business process models, but not validation as it is done in our work. Transforming goal models into business process model has been one of the major research areas in business process management. Lapouchnian et al. [23] use goal models for the configuration of business processes. Goal models are annotated with control flow information, they are transformed into BPEL processes. Similarly, Decreus and Poels [24] annotate goal-oriented models in the so-called B-SCP framework with control flow information and transform them into BPMN skeletons. Furthermore, Frankova et al. [25] transform SI\*/Secure Tropos models into skeletons

of process models in BPMN, from which they generate executable processes in BPEL. On the other hand, Santos et al. [4] derive goal models from existing process models. Such goal models are then used to control variability and configuration of processes. Finally, Koliadis et al. [2] annotate activities of process with effects, whereby these effects serve for a comparison of a process with goals, i.e., effects of activities are compared with goal fulfillments. The basic principle is to reflect changes from an  $i^*$  model to a BPMN model and vice versa.

The second group of related research is more related to our work, where some kind of verification between requirements and business process models is investigated. In this line of related research, Kazhamiakin et al. [26] propose a methodology that provides a set of high-level mapping rules to produce process models in BPEL from Tropos models. In order to enable requirements driven verification of process models, they employ the formal Tropos language and temporal constraints. While their work and tools support several types of formal analyses, they do not focus on validation of process models with respect to the satisfaction of goal models, as it is done in our work. Soffer and Wand [27,28] propose a generic theory-based process modeling framework based on Bunge's ontology for a goal-driven analysis of process models to check goal reachability in process models. They define a goal as stable state, which indicates the termination of the process. Their framework defines a set assumptions and parameters (based on goal relations and workflow relations), which are used to check if a set of workflow patterns ensure that processes can always reach to their goals. Hence, using these parameters it is possible to identify set of valid and invalid design decision with respect to business goals. They also suggest appropriate redesign actions to eliminate these invalid designs. Our approach follows the similar objective, but we use DL based reasoning to identify inconsistencies automatically.

Liaskos et al. [29] also introduce an approach for goal based customization of workflows. A family of processes is extended with the notation for partial temporal ordering of goals. A particular member of the family is specified by constraints with the means of linear temporal logic operators. However, this approach does not take into consideration business process logic embedded in the realization of business processes, which is the focus of our validation. La Rosa et al. [30] propose a questionnaire based customization of configurable business processes represented in C-YAWL [31]. They use a Petri-net based reasoner [32] to preserve the correctness during process configuration. Variability patterns of La Rosa et al.'s work are narrower than patterns introduced in this work.

## 8 Conclusion

In this paper, we have presented a novel approach for handling inconsistencies resulting from mapping goal models and business process models. By automatically identifying inconsistencies, we are able to detect executable processes that did not meet user requirements or lead to undesired executions. Additionally, we allow for goal models and process models to evolve independently. Our contribution extends the body of knowledge in the field by considering these mapping as first-class citizens along with goal models and business process models. We plan

to extend this approach in combination with our existing work on configuration of business process families [33] to provide a complete solution for software product lines that use goal models, feature models and process models as main artifacts.

## References

1. Dumas, M., Recker, J., Weske, M.: Management and engineering of process-aware information systems: Introduction to the special issue. *Information Systems* 37(2), 77–79 (2012)
2. Koliadis, G., Vranesevic, A., Bhuiyan, M.A., Krishna, A., Ghose, A.K.: Combining  $i^*$  and BPMN for Business Process Model Lifecycle Management. In: Eder, J., Dustdar, S. (eds.) *BPM Workshops 2006*. LNCS, vol. 4103, pp. 416–427. Springer, Heidelberg (2006)
3. Kueng, P., Kawalek, P.: Goal-based Business Process Models: Creation and evaluation. *Business Process Management Journal*, 17–38 (1997)
4. Santos, E., Castro, J., Sánchez, J., Pastor, O.: A Goal-Oriented Approach for Variability in BPMN. In: *Workshop em Engenharia de Requisitos, WER* (2010)
5. Andersson, B., Bergholtz, M., Edirisuriya, A., Ilayperuma, T., Johannesson, P.: A Declarative Foundation of Process Models. In: Pastor, Ó., Falcão e Cunha, J. (eds.) *CAiSE 2005*. LNCS, vol. 3520, pp. 233–247. Springer, Heidelberg (2005)
6. Bergholtz, M., Jayaweera, P., Johannesson, P., Wohed, P.: A Pattern and Dependency Based Approach to the Design of Process Models. In: Atzeni, P., Chu, W., Lu, H., Zhou, S., Ling, T.-W. (eds.) *ER 2004*. LNCS, vol. 3288, pp. 724–739. Springer, Heidelberg (2004)
7. Russel, N., ter Hofstede, A., van der Aalst, W., Mulyar, N.: *Workflow Control-Flow Patterns: A Revised View*. Technical report, BPM Center Report BPM-06-22, BPMcenter.org (2006)
8. Yu, E., Giorgin, P., Maiden, N., Mylopoulos, J. (eds.): *Social Modeling for Requirements Engineering*. MIT (2011)
9. Giorgini, P., Mylopoulos, J., Sebastiani, R.: Goal-oriented requirements analysis and reasoning in the TROPOS methodology. *Engineering Applications of Artificial Intelligence* 18, 159–171 (2005)
10. Chung, L., Nixon, B.A., Yu, E., Mylopoulos, J.: *Non-Functional Requirements in Software Engineering*, 1st edn. Springer (1999)
11. van Lamsweerde, A.: *Requirements Engineering: From System Goals to UML Models to Software Specifications*. Wiley (2009)
12. ITU-T: Recommendation Z.151 (09/08): User Requirements Notation (URN) Language Definition (2008)
13. Amyot, D., Ghanavati, S., Horkoff, J., Mussbacher, G., Peyton, L., Yu, E.: Evaluating Goal Models within the Goal-Oriented Requirement Language. *International Journal on Intelligent Systems* 25, 841–877 (2010)
14. Ouyang, C., Dumas, M., Aalst, W.M.P.V.D., Hofstede, A.H.M.T., Mendling, J.: From business process models to process-oriented software systems. *ACM Trans. Softw. Eng. Methodol.* 19, 2:1–2:37 (2009)
15. Feiler, P.H., Humphrey, W.S.: Software process development and enactment: concepts and definitions. In: *Second International Conference on the Software Process, Continuous Software Process Improvement*, pp. 28–40 (February 1993)
16. Johnson, R., Pearson, D., Pingali, K.: The Program Structure Tree: Computing Control Regions in Linear Time. In: *PLDI*, pp. 171–185 (1994)

17. Küster, J.M., Gerth, C., Förster, A., Engels, G.: Detecting and Resolving Process Model Differences in the Absence of a Change Log. In: Dumas, M., Reichert, M., Shan, M.-C. (eds.) BPM 2008. LNCS, vol. 5240, pp. 244–260. Springer, Heidelberg (2008)
18. Vanhatalo, J., Völzer, H., Leymann, F.: Faster and More Focused Control-Flow Analysis for Business Process Models Through SESE Decomposition. In: Krämer, B.J., Lin, K.-J., Narasimhan, P. (eds.) ICSOC 2007. LNCS, vol. 4749, pp. 43–55. Springer, Heidelberg (2007)
19. Kiepuszewski, B., ter Hofstede, A.H.M., Bussler, C.J.: On Structured Workflow Modelling. In: Wangler, B., Bergman, L.D. (eds.) CAiSE 2000. LNCS, vol. 1789, pp. 431–445. Springer, Heidelberg (2000)
20. van der Aalst, W., ter Hofstede, A., Kiepuszewski, B., Barros, A.: Workflow Patterns. In: Distributed and Parallel Databases (2003)
21. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook, 2nd edn. Cambridge University Press (2007)
22. Lau, S.Q.: Domain Analysis of E-Commerce Systems Using Feature-Based Model Templates. Master's thesis, University of Waterloo, Waterloo (2006)
23. Lapouchnian, A., Yu, Y., Mylopoulos, J.: Requirements-Driven Design and Configuration Management of Business Processes. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 246–261. Springer, Heidelberg (2007)
24. Decreus, K., Poels, G.: A Goal-Oriented Requirements Engineering Method for Business Processes. In: Soffer, P., Proper, E. (eds.) CAiSE Forum 2010. LNBIP, vol. 72, pp. 29–43. Springer, Heidelberg (2011)
25. Frankova, G., Frankova, G., Massacci, F., Massacci, F., Seguran, M., Seguran, M.: From early requirements analysis towards secure workflows. Technical Report TR DIT-07-036, University of Trento (2007)
26. Kazhamiakin, R., Pistore, M., Roveri, M.: A framework for integrating business processes and business requirements. In: Proc. IEEE DOC 2004 Conf., pp. 9–20 (2004)
27. Soffer, P., Wand, Y.: Goal-Driven Analysis of Process Model Validity. In: Persson, A., Stirna, J. (eds.) CAiSE 2004. LNCS, vol. 3084, pp. 521–535. Springer, Heidelberg (2004)
28. Soffer, P., Wand, Y., Kaner, M.: Semantic Analysis of Flow Patterns in Business Process Modeling. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 400–407. Springer, Heidelberg (2007)
29. Liaskos, S., Litoiu, M., Jungblut, M.D., Mylopoulos, J.: Goal-Based Behavioral Customization of Information Systems. In: Mouratidis, H., Rolland, C. (eds.) CAiSE 2011. LNCS, vol. 6741, pp. 77–92. Springer, Heidelberg (2011)
30. La Rosa, M., van der Aalst, W., Dumas, M., ter Hofstede, A.: Questionnaire-based Variability Modeling for System Configuration. *Software and Systems Modeling* 8, 251–274 (2009)
31. Gottschalk, F., van der Aalst, W.M.P., Jansen-Vullers, M.H., Rosa, M.L.: Configurable workflow models. *International Journal on Cooperative Information Systems* 17(2), 177–221 (2008)
32. van der Aalst, W., Dumas, M., Gottschalk, F., ter Hofstede, A., La Rosa, M., Mendling, J.: Preserving correctness during business process model configuration. *Formal Aspects of Computing* 22, 459–482 (2010)
33. Gröner, G., Wende, C., Bošković, M., Silva Parreiras, F., Walter, T., Heidenreich, F., Gašević, D., Staab, S.: Validation of Families of Business Processes. In: Mouratidis, H., Rolland, C. (eds.) CAiSE 2011. LNCS, vol. 6741, pp. 551–565. Springer, Heidelberg (2011)

# Agile Requirements Evolution via Paraconsistent Reasoning

Neil A. Ernst<sup>1</sup>, Alexander Borgida<sup>2</sup>, John Mylopoulos<sup>3</sup>, and Ivan J. Jureta<sup>4</sup>

<sup>1</sup> Department of Computer Science, University of British Columbia  
nernst@cs.ubc.ca

<sup>2</sup> Department of Computer Science, Rutgers University  
borgida@cs.rutgers.edu

<sup>3</sup> Dipartimento di Ingegneria e Scienza dell'Informazione, University of Trento  
jm@disi.unitn.it

<sup>4</sup> FNRS & Louvain School of Management, University of Namur  
ijureta@fundp.ac.be

**Abstract.** Innovative companies need an agile approach for the engineering of their product requirements, to rapidly respond to and exploit changing conditions. The agile approach to requirements must nonetheless be systematic, especially with respect to accommodating legal and nonfunctional requirements. This paper examines how to support a combination of lightweight, agile requirements which can still be systematically modeled, analyzed and changed. We propose a framework, RE-KOMBINE, which is based on a propositional language for requirements modeling called *Techne*. We define operations on *Techne* models which tolerate the presence of inconsistencies in the requirements. This paraconsistent reasoning is vital for supporting delayed commitment to particular design solutions. We evaluate these operations with an industry case study using two well-known formal analysis tools. Our evaluations show that the proposed framework scales to industry-sized requirements models, while still retaining (via propositional logic) the informality that is so useful during early requirements analysis.

**Keywords:** paraconsistency, agility, requirements, evolution.

## 1 Introduction

It is increasingly uncommon for software systems to be fully specified before implementation begins. This is because uncertainty about the right requirements is inescapable. Furthermore, it is highly desirable to avoid premature commitment by being able to change/revise requirements throughout the development lifecycle. Being flexible in this fashion is a source of competitive advantage for a business; for example, by delivering the correct product before competitors. The notion that one should engage in what has been called “big design up front” as part of the design activity is no longer defensible [1], since inevitably the plan must be abandoned, or at best revised. A variety of studies and experience reports (most recently [2]) have shown that requirements changes are very

expensive to accommodate and constitute the most frequent cause of project failures.

There is a shift, instead, to models of software development which avoid premature commitment to decisions. The central tenet of these models, including most Agile methodologies, is that requirements are discussed iteratively. These requirements are often manifested as very brief *user stories*, which serve as conversation starters with business representatives. A major concern with such lightweight requirements “engineering” is that non-functional requirements, such as security, are often neglected since system functionality is the focus [3].

While this lightweight approach to Requirements Engineering (RE) has become popular in many segments of industry, the IEEE standard for software requirements [4] uses words like “correct” and “unambiguous” to describe its recommended practice for RE. Thus, many previous approaches to the problem of system specification have methodological constraints insisting that conflicts and obstacles be resolved before solutions are identified. The past decade has revealed that in most cases these criteria are rarely, if ever, possible to achieve. In this paper we argue that the above shift demands a much more flexible approach to requirements modeling and analysis.

To illustrate the usefulness of deferring conflict resolution, consider the requirements fragment represented in Fig. 1. The figure represents the requirements (shown as ovals) for the business (“optimize sales”) and imposed requirements from an applicable security standard (PCI-DSS, which we discuss fully in Section 4.1). The red, X-headed relation between goals “..WEP” and “4.1.1...” represents a conflict. In this case, the conflict is between the business use of the Wireless Encryption Protocol (WEP) and the security problems with WEP. Existing approaches to requirements analysis either (i) insist that the conflict is resolved before proceeding with further reasoning (e.g., KAOS [5]), or (ii) represent the conflict as tradeoffs for higher-level goals (such as those in [6]). By contrast, our approach supports two possible courses of action. Because our framework is paraconsistent, it can (1) isolate and ignore the conflict for the time being. This is useful if, for example, the need for compliance is not immediate,

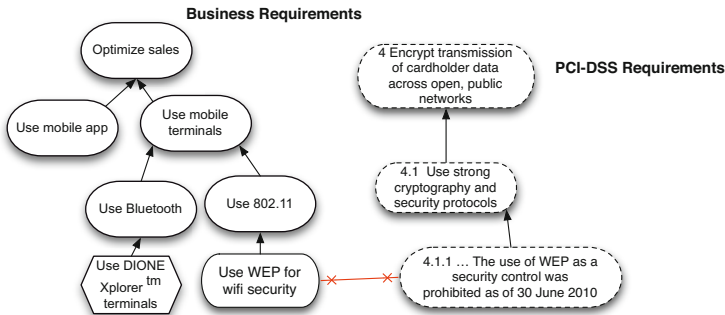


Fig. 1. Fragment of the requirements model from the payment card case. (Section 4.1)

and time and money can be better spent elsewhere. The other course of action is to (2) proceed with one of the conflicting choices, and insist that the model be later revised to support that choice. In this case, we would ask for alternative solutions for optimizing sales. In the example, we might abandon mobile payment terminals in favour of mobile apps, or use Bluetooth rather than WEP for wireless transmission. In this way the stakeholders are presented with a range of options for proceeding.

In this paper we introduce a framework, RE-KOMBINE, which supports this shift to flexibility. The framework represents possibly contradictory requirements as assertions about the current state of the requirements model, allowing us to reason paraconsistently about requirements problems<sup>1</sup>, thus helping to accommodate flexible, agile decision-making. The chief advantage of our approach is that it permits derivation of useful knowledge about problems of interest in the moment, while postponing decisions about currently inconsistent states of the problem until a decision must be made.

This paper makes the following contributions:

- identifies the importance of accommodating variability by supporting paraconsistency in software development, and modifying the way in which the requirements are queried.;
- proposes a framework, RE-KOMBINE, for finding solutions to possibly inconsistent requirements problems in a goal-oriented framework;
- explicitly introduces paraconsistent reasoning into a prototype tool;
- evaluates the framework with an industrial case study.

In [7], we introduced the notion of a Requirements Engineering Knowledge Base REKB for maintaining a requirements model. In this paper, we build on the notion of an REKB to handle the case where problems are changing and possibly inconsistent (i.e., contain contradictory assertions).

## 2 Background

### 2.1 The Requirements Problem

We start from Zave & Jackson's [8] definition of the *requirements problem*: given requirements  $R$ , and domain assumptions  $D$ , find specification  $S$ , satisfying  $D \cup S \vdash R$  under the condition that  $D \cup S$  is consistent.

In our work on the *Techne* language [9,7], requirements problems are structured, representing notions ranging from high-level requirements ("sell more products") to low-level tasks ("use Moneris payment terminals"). A key part of solving requirements problems is therefore to find ways to refine requirements so they are eventually reduced into tasks, and to record conflicts between requirements. We therefore re-state the requirements problem as the search for tasks  $T$  and refinements/realizations/constraints  $R$  that can be added to the

---

<sup>1</sup> In other words, local inconsistencies are not propagated globally, "polluting" all inferences, as in standard logic.



world knowledge/domain assumptions  $D$ , such that requirements, captured as goals  $G$ , are satisfied, i.e.,

$$D \cup T \cup R \vdash G \tag{1}$$

Techne proposes to identify all *candidate solutions* to the problem — all sets  $T$  which achieve the goals<sup>2</sup>

Techne’s REKB accepts the following as well-formed formulae:

$$formula ::= atom \mid \left( \bigwedge_{i=1}^n atom_i \right) \rightarrow atom \mid \left( \bigwedge_{i=1}^n atom_i \right) \rightarrow \perp \tag{2}$$

where atoms are propositions (un-analyzed natural language sentences). Implications of the form  $\beta \rightarrow \perp$  represent conflicts between the atoms in  $\beta$ , while those of the form  $\beta \rightarrow b$  encode refinements/realizations (where  $b$  is an atom).

### 2.2 Inconsistency and Conflict

The ability to represent conflicts between requirements is an essential part of any requirements modeling language. In formal logic, a theory  $\mathcal{T}$  is said to be *inconsistent* if one can derive *False*/ $\perp$ , from  $\mathcal{T}$ . Classical logic trivializes in the sense that anything can be derived from an inconsistent  $\mathcal{T}$  (*ex falso quodlibet*). This makes classic, inconsistent REKB useless for solving requirements problems, and is the reason why the premise of the sequent  $\vdash$  in (1) must be consistent.

There are several ways to interpret the existence of a *conflict* relation between requirements  $A$  and  $B$ , recorded as  $A \wedge B \rightarrow \perp$ . The conflict might mean that neither requirement can be satisfied. The conflict could also mean that at most one, but not both can be satisfied. Finally, it could be more drastic, and suggest that the entire model must be resolved to remove the conflict. Techne adopts the second attitude. Part of searching for solutions to Techne requirements problems is to find ways to ensure at most one of A or B, where A and B are in conflict, is satisfied.

In the requirements engineering research community, the term “conflict” has typically been used to denote social disagreement over the nature of the system requirements. Robinson et al. [10] define it as “requirements held by two or more stakeholders that cause an inconsistency”. The term “inconsistency” denotes the technical, formal existence of a “broken rule” [11]. Zowghi and Gervasi [12] show that “consistency” is causally related to requirements “completeness”: a more complete requirements document is often less consistent (since more competing requirements are introduced). In Techne, the conflict relation is formally between two or more requirements, and not between stakeholders. Any conflicts between stakeholders, such as a disagreement over terminology, are the purview of other techniques (e.g., model merging). In this paper, conflict is the presence of  $\perp$ , while inconsistency is when false ( $\perp$ ) is derived. Ultimately, we not only want to permit conflict (and possibly inconsistency) to be represented; we also want to specify what we ought to do when inconsistency is detected.

---

<sup>2</sup> The second step, of selecting among the candidates a solution using some techniques to rank alternatives, is not discussed in this paper.

Classical languages, such as propositional logic and first-order logic, cannot tolerate inconsistency, in the sense that no useful reasoning can be done in its presence, and yet, in the requirements engineering domain, tolerating inconsistency is important. Nuseibeh *et al* [13] give a few important reasons:

- to facilitate distributed collaborative working,
- to prevent premature commitment to design decisions,
- to ensure all stakeholder views are taken into account,
- to focus attention on problem areas [of the specification].

Perhaps the most useful reason for the case of evolving requirements problems is the second one. Avoiding premature commitment, in the sense of Thimbleby [14], means to wait until the “last responsible moment” to make decisions regarding the system. Not only does this apply to deciding *how* to satisfy our goals, but also in the choice of those goals themselves. Tolerating inconsistency therefore allows us to continue to make progress on design (and even implementation) while fire-walling the conflicting parts of the system. Section 4.1 will show how this becomes crucial in our case study.

In our case, part of tolerating inconsistency in the REKB involves *paraconsistent reasoning*. A paraconsistent logic, broadly, is one which does not trivialize in the presence of inconsistency. Section 3.2 will show how we define operators on the knowledge base that continue to give meaningful answers even when inconsistency is present.

### 3 RE-KOMBINE

We now define a framework for managing the inconsistency in requirements problems that arise due to variability and evolution. We will do so by defining below two fundamental operators that can be applied to an REKB in order to find solutions to the requirements problem, even in the presence of inconsistency. But before doing so, we need some formal machinery and discussion for its motivation.

#### 3.1 Paraconsistent Reasoning on Requirements Problems

We first present a general approach to defining what it means to draw conclusions from a possibly inconsistent set of assumptions, denoted by the  $\sim$  symbol. To define  $\sim$ , we go back to one of the early attempts to deal with paraconsistent reasoning, that of Rescher and Manor [15]. Given a theory  $\Delta$ , define  $MC(\Delta)$  as the set of maximal consistent subsets of  $\Delta$ , and then consider “weak” (a.k.a. “credulous”) consequences those that follow from one element of  $MC(\Delta)$ , while “inevitable” (a.k.a. “cautious, skeptical”) ones hold in all such maximal subsets.

We propose the following definition

**Definition 1.**  $\Delta \sim S$  iff there exists  $\Pi \subseteq \Delta$  such that

1.  $\Pi \in MC(\Delta)$ ,
2.  $\Pi$  contains all implications in  $\Delta$ , (written  $Implications(\Delta)$ ),
3.  $\Pi \vdash S$

Given domain theory  $D$  and a specific set of (high-level) goals  $G_0$  which we are trying to achieve, a solution to the requirements problem will then be said to consist of a set of refinements/conflicts  $R_0$  and a set of task atoms  $T_0$  such that

$$D \cup R_0 \cup T_0 \sim G_0 \quad (3)$$

which replaces the original equation (1).

The motivation for condition 2. in Definition 1, which is the one addition to the original “weak” entailment in [15], is specific to Requirements Engineering, particularly the *Techne* family of languages and their methodology: In any specific situations we are looking for a consistent set of tasks and goals which solve the requirements problem. If we allow implications to be excluded, then we might miss inconsistencies between these atoms. Moreover, a set of Horn clauses is always satisfiable/consistent, so requiring condition 2. does not affect the existence of maximal consistent subsets.

The above definition is “credulous” since it depends only on the existence of *some* set  $\Pi$ , from which  $S$  can be derived. In contrast, much of non-monotonic reasoning and database reasoning with inconsistent data deals with the “skeptical” mode:  $S$  must be derivable from *all* maximally consistent  $\Pi$  of the above form. This distinction is less significant in requirements problems: since we are considering possible future states of the world, we are making assumptions about which tasks to implement. In the paraconsistent case, we are identifying individual sets of tasks which solve the requirements problem. Thus, the skeptical approach is overly constraining, since the presence of a single solution is all we need for the implementation phase. Implicitly, what the sceptical semantics for paraconsistency in requirements engineering do is restrict the nature of the eventual system we build. In RE-KOMBINE, the only constraint imposed is that implications may not be discarded.

It remains to consider whether we want to be able to obtain solutions under all conditions, or whether there are additional criteria for solutions to make sense. Consider the following criteria:

$$D \cup R_0 \not\vdash \perp \quad (4)$$

$$D \cup R_0 \cup G_0 \not\vdash \perp \quad (5)$$

$$D \cup R_0 \cup T_0 \not\vdash \perp \quad (6)$$

Violating criterion (4) indicates the presence of something we call ‘blockers’: since domain assumptions hold, then no “reasonable” solution can ever exist if criterion (4) does not hold. We also want to insist that the goals we are trying to achieve (i.e.,  $\bigwedge G_0$ ) are mutually consistent in view of the background theory, criterion (5). Finally, we expect that not only can we achieve those goals, but that there are consistent sets of tasks which will do so, criterion (6).

In order to achieve the above, in RE-KOMBINE we restrict the asserted members of the knowledge base theory to be elements of  $D$  and  $R$ . Additional formulas that are implications can be added to  $R$  and  $D$ . We must include domain assumptions because, at least for now, such atoms are universally true, and thus always relevant. We include refinements and conflicts because they form the set

$R$ , mentioned above, and presumably reflect some domain knowledge about how requirements interact.<sup>3</sup>

In some requirements problems we may wish to speculate about certain low-level goals being true or tasks having to be carried out; e.g., “suppose goal  $g_1$  were achieved; what else is necessary to achieve top-level goal  $g_0$ ?”. In that case we may “*hypothetically*” assert these atoms by including them in  $R$  (“r: goal  $g_1$  is achieved”), which makes it appear that we *assume* that the corresponding goal/task has been achieved.

The operator specifications below follow the above discussion, by alerting the user if condition 5 (and hence 4) is violated. Condition 6 will not be violated by virtue of Definition 1, and equation (3).

### 3.2 Operators for Paraconsistent Requirements Problems

RE-KOMBINE is defined in a functional style, specifying update and query operators on the requirements knowledge base (REKB). In 7 we introduced operators for consistent requirements problems, but the need for consistency ruled out the flexible approach we describe in this work. In particular, the crucial operators which help users select and decide on solutions to their requirements problems are specified using paraconsistent consequence ( $\vdash$ ) introduced above. If the arguments to the operation are themselves (internally) inconsistent, the reasoner will generate an exception. We describe the operators in the style of Javadoc by naming the parameters and their types, etc. We use  $\wp(S)$  to represent the set of all subsets of  $S$  (powerset). Examples from a case study are shown in Section 4.

---

#### Operation 1 — PARACONSIST-MIN-GOAL-ACHIEVEMENT

**@param** wantedG :  $\wp(\text{GOALS})$

**@return** TaskSets :  $\wp(\wp(\text{TASKS}))$  consisting of all sets  $S$  of tasks such that:

**@effect** REKB  $\cup S \vdash \bigwedge$  wantedG, and no subset of  $S$  has this property.

**@throws** exception if wantedG  $\cup$  Implications(REKB)  $\vdash \perp$ .

---

The PARACONSIST-MIN-GOAL-ACHIEVEMENT operation supports what has been called “backward reasoning” in the RE literature (e.g., 6). Backward reasoning sets some high-level goals as desiderata, and determines which tasks can accomplish those goals. PARACONSIST-MIN-GOAL-ACHIEVEMENT is an abductive search. Abduction only works from consistent theories, so the classical version of this operation generates an exception if the theory REKB is inconsistent. Since Implications(REKB) is always consistent, the above modified version excludes aspects of REKB that may be inconsistent *on their own* – hence the paraconsistent behaviour of our operator. Note that the minimality of  $S$  in the above specification also prevents  $\vdash$  from choosing  $S$  that have conflicting tasks, and gives preference to tasks specified as obligatorily occurring in REKB.

---

<sup>3</sup> Although arguably such refinements are not domain knowledge, but also part of the desired state of affairs. However, for the purposes of this paper we consider them assumptions.

*Example.* Consider the requirements problem which can be represented as  $\mathbf{R} = \{D \wedge E \rightarrow B, F \wedge H \rightarrow C, C \rightarrow A, B \rightarrow A, E \wedge F \rightarrow \perp\}$ ,  $\mathbf{D} = \{E, F\}$ . If we then define  $\mathbf{G} = \text{wantedG} = \{A\}$ , the problem is *classically* inconsistent, since there is a conflict when we identify potential solution sets which must contain the domain assumptions  $E$  and  $F$ . Paraconsistently, however, we can identify two separate answers to the operation:  $\mathbf{S}_1 = \{D, E\}$ ,  $\mathbf{S}_2 = \{F, H\}$ . This supports our desire to continuing to reason despite a conflict.

In the requirements problem, we are interested in optimality with respect to the people communicating the requirements for the new system (stakeholders). In that context, the stakeholder may not be content with a subset-minimal implementation that satisfies the mandatory requirements (as returned by PARACONSIST-MIN-GOAL-ACHIEVEMENT). Rather, he or she is interested in implementations which also satisfy other, non-mandatory goals. Furthermore, while still subset-minimal with respect to tasks, we add the constraint that the set of goals achieved is maximized. This answers the question, “*If I wish to accomplish the following extra goals, in addition to certain mandatory requirements, what are the minimal sets of tasks I must perform?*”

---

**Operation 2 — PARACONSIST-GET-CANDIDATE-SOLUTIONS**

**@param** mandG :  $\wp(\text{GOALS})$

**@param** wishG :  $\wp(\text{GOALS})$

**@return** set of pairs  $\langle \text{solnT}, \text{satG} \rangle$ , where solnT is a set of tasks, and satG is a set of goals such that

**@effect** 1)  $\text{REKB} \cup \text{solnT} \vdash \text{satG}$ ; 2)  $\text{satG} = \text{mandG}$  (the mandatory goals)  $\cup$  wishG<sub>0</sub> (a subset of the optional goals wishG), such that  $\text{satG} \cup \text{Implications}(\text{REKB})$  is consistent; 3) satG is maximal with respect to the above properties; 4) solnT is subset-minimal to achieve the above.

**@throws** an exception if mandG is inconsistent with  $\text{Implications}(\text{REKB})$ .

---

Operation PARACONSIST-GET-CANDIDATE-SOLUTIONS requires that the set solnT paraconsistently derive satG, i.e., there is a consistent subset of REKB, which includes implications, from which one can classically prove satG, using solnT.

The above operators illustrate how paraconsistency is a fairly natural concept in solving the requirements problem. Our focus remains on the appropriate sets to return, but we adopt a credulous approach in which a single consistent subset of the larger requirements problem can be used to derive our mandatory requirements. This model of operation also maps nicely to our choice of the ATMS for implementation, as we discuss in the following subsection.

Note that there are more operators than the ones we described here, although we believe these are the two most relevant to variability in requirements. In

particular, operators to calculate so-called “forward-reasoning” [6], using input tasks and a set of high-level goals, are useful (and more tractable).

### 3.3 Tool-Supported RE-KOMBINE

We have implemented the RE-KOMBINE framework using an Assumption-based Truth Maintenance System (ATMS) [16]. An ATMS naturally supports our simple definition of well-formed formulae, and support paraconsistent reasoning. However, other choices are possible, including the use of weighted SATisfiability solvers, as used in Sebastiani et al. [6].

De Kleer [16] proposed the ATMS. In an ATMS each node has associated a set of possible *explanations*, in which that node is :IN (interpreted as true). Explanations are sets of assumptions which ultimately justify that node (i.e., from which that node can be derived from assumptions via definite Horn-rules called justifications.) The *label* for a given node  $N$  will typically be labelled with explanations: all sets of assumptions for which it can be derived :IN.

Most importantly, these sets are minimal – no nodes can be removed from such an explanation without losing the full justifications, and the sets are consistent in the sense that no contradictions ( $\perp$ ) can be derived from them. We encode atoms in RE-KOMBINE as ATMS nodes; atoms with sort TASK become assumptions, and implications or contradictions become justifications and contradictions, respectively, with a special CONTRADICTION node added as necessary. Cycles are supported in the ATMS because of short-circuit evaluation. If a node has a label which is the same as a possible new label, evaluation terminates.

Listing [4.1] gives an example of the domain-specific language (DSL) for capturing requirements problems in *Techne*. The operation `declare-atomic` introduces (but does not assert) goals in the model, while `assert-formula` defines inference or conflict relations, in this case between `g0` and its antecedents. The tool supports a graphical front-end (as seen in the images below) with a translation to the DSL, and output of the reasoning is likewise textual or graphical. In agile software development, in particular, it is important that the process artifacts be perceived as nearly invisible (hence the frequent use of index cards and whiteboards). RE-KOMBINE, while clearly more onerous in terms of ease of use, makes the tradeoff that this flexibility nonetheless needs longer-term support, particularly in more complex domains.

We are currently integrating RE-KOMBINE with a commercial requirements tool, where the intention is to permit requirements to be captured easily and managed using RE-KOMBINE. The workflow is for requirements elicitation to proceed as usual, with the sole exception being that the analyst or developer enters the requirements as *Techne* statements (which are simple propositional statements with formal relations). Then, during the prioritization phase at the beginning of a development iteration, the RE-KOMBINE tool can provide answers regarding which requirements/features/user stories to work on.

```
(defvar
g0 (declare-atomic nil "Comply with PCI DSS" :GOAL *rekb*)
; ...
(assert-formula gc1.2.1.2.1 (list g0) :DA *rekb*))
```

**Listing 1.1.** DSL for introducing *Techne* formulae into RE-KOMBINE.

Another approach is using Qualitative Goal Models, introduced in [17]. They support qualitative (strong, weak) evidence both in favor and against propositional goals. The solution of Giorgini et al. [18] is to formalize this by replacing each proposition  $g$ , standing for a goal, by four propositions ( $FS_g, PS_g, PD_g, FD_g$ ) representing that there is full (resp. partial) evidence for the satisfaction (resp. denial) of  $g$ . A traditional implication such as  $p \wedge q \rightarrow r$  is then translated into a series of implications connecting these new symbols, including  $FS_p \wedge FS_q \rightarrow FS_r$ ,  $PS_p \wedge PS_q \rightarrow PS_r$ , as well as  $FD_p \rightarrow FD_r$ ,  $FD_q \rightarrow FD_r$ , etc. The important point is that, first, the result is a classical propositional theory, but one where there is *never any inconsistency* that would cause everything to be inferred, since conflicts between  $a$  and  $b$  are recorded by axioms of the form  $FS_a \rightarrow FD_b$ , and it is quite consistent to have both  $FS_x$  and  $FD_x$  be true at the same time (there is strong evidence both for and against  $x$ ). In fact, the above can be viewed as a many-valued logic, where symbol  $g$  can be assigned a *subset* of the truth values  $\{FS, PS, PD, FD\}$ .

Giorgini et al.'s approach is extended by Sebastiani et al. [6], including axioms for avoiding conflicts of the form  $FS_a \wedge FD_a$ . If we represent (for example) the *Techne* conflict relation  $A \wedge B \rightarrow \perp$  as their  $(FS_A \rightarrow FD_B) \wedge (FS_B \rightarrow FD_A)$ , this supports the detection of conflicts of the form “the solution may not contain both goal A and goal B together”. By using a MinWeight SAT solver which can minimize the number of asserted atoms involved in the solution, and a model which avoids the use of partial contributions, one can simulate portions of the RE-KOMBINE framework. We use this tool to compare with the ATMS implementation in Section 4.2.

## 4 Evaluating RE-KOMBINE

We begin with a description of our case study of variability and evolution in requirements found in the Data Security Standard (henceforth PCI-DSS) [19], an industry standard which regulates security of credit card transactions. We use this case study to first illustrate how paraconsistency is essential for solving the requirements problem in a specific example. We then discuss how our framework scales to industrially relevant sizes.

### 4.1 Case Study: Payment Card Standards and Requirements Variability

PCI-DSS version 2.0 was released in October, 2010, and is currently in force. There is a two-year cycle between major revisions, with a three month announcement window immediately prior to the new standard coming into force. This

provides organizations time to achieve compliance. PCI-DSS has the following sub-goals for compliance: (i) Build and maintain a secure network, (ii) Protect cardholder data, (iii) Maintain a vulnerability management program, (iv) Implement strong access control measures, (v) Regularly monitor and test networks, (vi) Maintain an information security policy.

PCI-DSS is well-suited for representation as a requirements problem. We map requirements as goals, and constraints as domain assumptions. Tasks are used to represent compliance tests in the PCI-DSS. A solution to the requirements problem is a series of tasks which can pass the compliance audit, a *compliance strategy*. For this paper, the relevance of this case study is in describing how the changes to the standard are realized in individual organizations, which also have entirely separate sets of organization-specific requirements. This was shown in Fig. 1 earlier: the organization-specific goals must be related to the standard's compliance goals. We then translate this to a domain-specific language (DSL) which can be run against the ATMS or MinWeightSAT solvers.

**Solving the PCI-DSS Requirements Problem.** We now illustrate how changes in an external standard, such as PCI-DSS, might lead to inconsistency in an organizational requirements model. We focus in particular on how the RE-KOMBINE tool can use paraconsistent reasoning to support strategic, change-tolerant decisions.

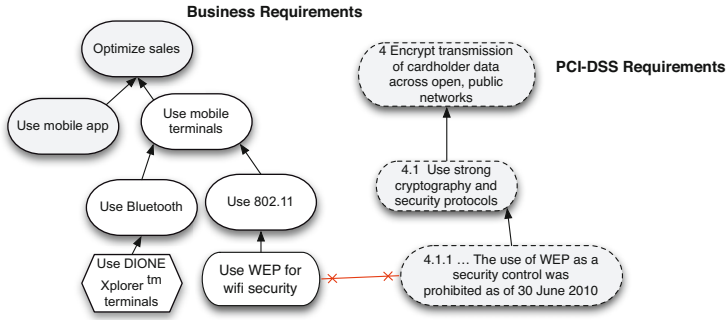
Let us return to the example from Section 1, shown again in Fig. 2. We showed that the mere presence of the conflict relation would cause classical reasoning to fail. In RE-KOMBINE, we are able to continue to search for solutions in spite of the conflict. Recall that the requirements problem has been defined as  $D \cup RUT \vdash G$ . The state of the REKB, that is, the nature of the requirements problem of Fig. 2, is as follows.

Set  $R$  contains all implications and conflicts, among others the refinement of Optimize Sales by Use Mobile App and the conflict between Use WEP and WEP Prohibited. Set  $G$  contains the twin goals of Optimize Sales and Encrypt Transmission, since in this case study the business would like to achieve both compliance and business success. There is only a single task identified in  $T$ , the use of Dione XPlorer terminals. For this fragment there are no domain assumptions in  $D$ . Finally, we add to  $R$  the assumptions that goal Use WEP is already satisfied, i.e., currently the state of affairs for the business network, and that we must abide by the PCI-DSS requirements, i.e., that goal WEP Prohibited is also satisfied. This would be the case if the business was compliant prior to the June 2010 enforcement of the restriction on WEP.

In this fragment of the model we are seeking to find tasks  $T$  or assumptions for  $R$  such that a subset of goals in  $G$  are satisfied. One way to do this is with the following operation. Call PARACONSIST-MIN-GOAL-ACHIEVEMENT with the argument wantedG = {Optimize Sales, Encrypt Transmission}. RE-KOMBINE performs an abductive search to find minimal sets of tasks or assumptions which will satisfy the conjunction of these goals. In this case, possible answers, returned as TaskSets, include {Use Mobile App, WEP Prohibited}, {WEP Prohibited, DioneXplorer}. Solutions which include both assumptions which lead to



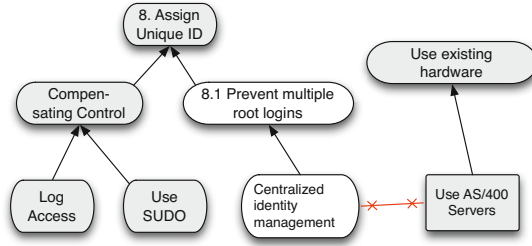
conflict are excluded. In this case, since the PCI-DSS is external and presumably compliance is essential, the business would choose to phase out WEP, using one of these strategies instead. Again, paraconsistency in RE-KOMBINE allows us to maintain the complete model, adding or retracting tasks as required.



**Fig. 2.** WEP fragment of the model from the payment card case showing a possible configuration. Grey indicates satisfied goals.

In the PCI-DSS there are multiple types of changes we must accommodate, including variations in adopting best practices the standard identifies (such as application security practices). There is also provision for variation *within* an organization in *proving* compliance (i.e., selecting a compliance strategy). In PCI-DSS these are known as **compensating controls**, and they define solutions for proving compliance where domain assumptions in the standard are invalid. For example (numbers in parentheses refer to the PCI-DSS standard, v2), in environments that cannot prevent multiple root logins (requirement 8.1), the organization is permitted to use SUDO (a command to give an ordinary user full but temporary privileges), just as long as the system carefully logs each access. The reason to prevent root login is that the use of a super-user account is opaque, without this control: it is not clear what physical access is behind the root account.

Fig. 3 captures this fragment of the requirements problem. Assume there is a call to PARACONSIST-GET-CANDIDATE-SOLUTIONS with  $wishG = \{Use\ existing\ hardware\}$  and  $mandG = \{Assign\ Unique\ ID\}$ . Then the result is the tuple  $\langle solnT:\{Log\ Access,Use\ SUDO,Use\ AS/400\ Servers\}, satG:\{Use\ existing\ hardware,Assign\ Unique\ ID\}\rangle$ . This reflects the (simplistic) result that the best way to satisfy the wished-for goal to use existing hardware is to apply for the compensating control of logging access. Again, we have placed Use AS/400 Servers into  $R$  as an assumption, since it describes the current state of affairs. In this case a conflict exists if we also assume that the use of Centralized identity management is satisfied. The paraconsistent operator has allowed us to find alternative solutions to this inconsistent state.



**Fig. 3.** The graphical representation of the compensating controls example

## 4.2 Demonstrating Scalability

In [7] we showed that the ATMS reasoner was scalable. It can return decisions in less than 100 seconds on random models that were as large as six hundred requirements and two hundred relations. In this paper, we have extended our tool to introduce some useful pre-processing steps, including the elision of and-subtrees, which greatly reduces the number of assumptions.<sup>4</sup> We then ran our reasoner on the PCI-DSS case study model. We also evaluated the case study using the tool from [6], as described in Section 3.3. Applying the tool to industrial problems is predicated on a useful set of requirements propositions being generated during requirements elicitation activities, whether geared to lightweight user-story gathering or more formal use case or IEEE 930 approaches.

The examples above were derived from a complete RE-KOMBINE model of the case study. It consisted of two connected components. One was the representation of the PCI-DSS model, which has 254 requirements and 65 relationships. This is the most basic model, in which every requirements must be adhered to (thus, all relationships are AND-style). This is trivially easy to reason on. We then added subcomponents representing scenarios for variation (i.e., nodes with multiple justifications in the PCI model, representing alternatives for compliance), consisting of 41 nodes and 18 relations, and scenarios for evolution (changes between version 1.1 and version 2). Finally, we created components that reflected the business objectives of a soccer stadium, based on the case study described in [20]. The final model was 342 nodes and 127 relations in size.

To compile the model in ATMS took 614 seconds, on a Macbook Pro 2.4Ghz with 8Gb RAM. This reflects the amount of time needed to generate the abductively minimal solutions for *all* possible goals in the model. Querying a set of these goals then requires a polynomial amount of comparisons to generate the answer which is trivial relative to the exponential abduction problem. This size of model compares in size with industrial examples of design requirements described in the literature (e.g., van Lamsweerde [21] listed KAOS model sizes that were, on average, 540 goals and requirements). We have found that the

<sup>4</sup> The complete model and source code is available at <http://github.com/neilernst/Techne-TMS>.

benefits of the ATMS are more apparent when performing incremental computations, e.g., when the model is evolved. ATMS is inherently incremental and so evolutionary changes are relatively painless.

A single call to the MinWeight SAT solver from [6], using weak conflict avoiding, did not find a single minimal solution after 30 minutes. Admittedly, the MinWeight tool is not state of the art for SAT solvers. A call to a non-minSAT solver, zChaff 2007.3.12, by comparison, returned a solution (a single satisfying instance that is not minimal) in a few milliseconds.

## 5 Related Work

We described the work of Giorgini et al. [18] and Sebastiani et al. [6] in Section 3.3. Their qualitative approach can simulate some of the capability of RE-KOMBINE. A major difference in philosophy is the omission, in RE-KOMBINE, of qualitative, partial satisfaction/denial relations. RE-KOMBINE deliberately omits this notion of partial satisfaction, because in practice, this bipartite approach leads to frequent occurrences of conflicting information about a given requirement. In a dynamic environment, partial satisfaction of goals results in lack of actionable information. For example, consider the case where we know that the goal Comply with PCI-DSS is both partially satisfied and partially denied. This type of conflict can lead to analysis paralysis and a substantial cost of delay. RE-KOMBINE is tailored for automatic, binary answers over conflicting goals. Qualitative reasoning is better suited to up-front problem exploration. RE-KOMBINE's systematic, lightweight approach is more suitable when we are doing an iterative problem exploration by committing to small increments of the model.

Zowghi and Offen [22] and Ghose [23] both use a default logic approach to requirements modeling. Zowghi and Offen approach things from a verification perspective. Their central concern is to ensure that the requirements specification is complete and consistent following change. To evolve a specification, Zowghi and Offen define a partial order over the requirements in order to select the requirements that should be removed to maintain consistency. Like us, Ghose is concerned with avoiding premature commitment. However, Ghose insists on obtaining from an oracle the possible critical states of system behaviour, which he calls a trajectory. With this predictive oracle, the solutions to the requirements problem captured in his language can then be optimized. The oracle is capturing significant contextual variation in the assumptions. We do not insist on anticipating future change in this fashion.

Hunter and Nuseibeh [24] described one of the early approaches to inconsistent requirements specifications. They focused primarily on up-front requirements specification, in particular the possibility that different stakeholders may model the problem differently. They used labeled Quasi-Classical logic to permit the resolution of inconsistency during specification development. This work prefigures ours in using a paraconsistent language for continuing to reason in the presence of inconsistency, although different inferences are drawn. Our innovation is the introduction of operations on requirements problems, including

the notion of minimal solutions, as a way to support design decision-making. We also feel that the simpler propositional language of *Techne* is more suited to the light-weight analysis common in industry.

The concepts of specification, requirements and domain assumptions also exist in formal methods research. For example, Poppleton and Groves [25] discuss the notions of *refinement* and *retrenchment*, which are used to model the transformation of a program as the specification changes. The distinction is primarily in the degree of formality that the tools demand. We feel that only formalizing high-level relationships between elements in the requirements problem is more likely to support the wide range of scenarios one might see in an agile setting.

## 6 Conclusion

This paper has operated from a premise that establishing the entirety of a project's requirements up-front is unrealistic and even undesirable. It proposed a systematic approach to agile requirements evolution where it is easy to change requirements and automatically evaluate the consequences of these changes. We showed that this reconciliation also makes it possible to delay decisions about conflicting requirements until more information becomes available. In order to support these trends, we described a framework, **RE-KOMBINE**, for expressing requirements formally yet sufficiently flexibly as to enable deferred commitment. The paper introduced a new definition of paraconsistency in requirements specifications using *Techne* as the underlying propositional language. It then described properties for defining a paraconsistent consequence operator,  $\vdash$ . Using that operator, we introduced two operations for reasoning paraconsistently on requirements models, searching for minimal solutions to the requirements problem despite the existence of contradictory or missing information. We introduced an industrial case study of payment card requirements, and showed that the operations can be scaled to typical industrial design problems. In future, we intend to continue investigating how the approach can be used in even more complex, real-world problems.

## References

1. Jarke, M., Loucopoulos, P., Lyytinen, K., Mylopoulos, J., Robinson, W.: The Brave New World of Design Requirements: Four Key Principles. In: Pernici, B. (ed.) CAiSE 2010. LNCS, vol. 6051, pp. 470–482. Springer, Heidelberg (2010)
2. Mcgee, S., Greer, D.: Software Requirements Change Taxonomy: Evaluation by Case Study. In: Int. Conf. on Req. Engineering, Trento, Italy (September 2011)
3. Ramesh, B., Cao, L., Baskerville, R.: Agile requirements engineering practices and challenges: an empirical study. *Information Systems Journal* 20(5), 449–480 (2010)
4. IEEE Software Engineering Standards Committee: IEEE Recommended Practice for Software Requirements Specifications. Technical report, IEEE (1998)
5. Dardenne, A., van Lamsweerde, A., Fickas, S.: Goal-directed requirements acquisition. *Science of Computer Programming* 20(1-2), 3–50 (1993)

6. Sebastiani, R., Giorgini, P., Mylopoulos, J.: Simple and Minimum-Cost Satisfiability for Goal Models. In: Persson, A., Stirna, J. (eds.) CAiSE 2004. LNCS, vol. 3084, pp. 20–35. Springer, Heidelberg (2004)
7. Ernst, N.A., Borgida, A., Jureta, I.: Finding Incremental Solutions for Evolving Requirements. In: Int. Conf. on Req. Engineering, Trento, Italy (September 2011)
8. Zave, P., Jackson, M.: Four Dark Corners of Requirements Engineering. *ACM Transactions on Software Engineering and Methodology* 6, 1–30 (1997)
9. Jureta, I.J., Borgida, A., Ernst, N.A., Mylopoulos, J.: Techne: Towards a New Generation of Requirements Modeling Languages with Goals, Preferences, and Inconsistency Handling. In: Int. Conf. on Req. Engineering, Sydney, Australia (September 2010)
10. Robinson, W.N., Pawlowski, S.D., Volkov, V.: Requirements interaction management. *ACM Computing Surveys* 35(2), 132 (2003)
11. Easterbrook, S.M., Nuseibeh, B.: Managing inconsistencies in an evolving specification. In: Int. Conf. on Req. Engineering, York, England, pp. 48–55 (1995)
12. Zowghi, D., Gervasi, V.: On the interplay between consistency, completeness, and correctness in requirements evolution. *Information and Software Technology* 45(14), 993–1009 (2003)
13. Nuseibeh, B., Easterbrook, S.M., Russo, A.: Making inconsistency respectable in software development. *Journal of Systems and Software* 58(2), 171–180 (2001)
14. Thimbleby, H.: Delaying commitment. *IEEE Software* 5(3), 78–86 (1988)
15. Rescher, N., Manor, R.: On inference from inconsistent premisses. *Theory and Decision* 1(2), 179–217 (1970)
16. de Kleer, J.: An assumption-based TMS. *Artificial Intelligence* 28(2), 127–162 (1986)
17. Chung, L., Mylopoulos, J., Nixon, B.: Representing and Using Nonfunctional Requirements: A Process-Oriented Approach. *Trans. Soft. Eng.* 18, 483–497 (1992)
18. Giorgini, P., Mylopoulos, J., Nicchiarelli, E., Sebastiani, R.: Formal Reasoning Techniques for Goal Models. In: Spaccapietra, S., March, S., Aberer, K. (eds.) *Journal on Data Semantics I*. LNCS, vol. 2800, pp. 1–20. Springer, Heidelberg (2003)
19. PCI Security Standards Council: PCI DSS Requirements and Security Assessment Procedures, Version 2.0. Technical report, PCI, Boston (October 2010)
20. O’Callaghan, R.: Fixing the payment system at Alvalade XXI: a case on IT project risk management. *Journal of Information Technology* 22(4), 399–409 (2007)
21. van Lamsweerde, A.: Goal-oriented requirements engineering: a roundtrip from research to practice. In: Int. Conf. on Req. Engineering, pp. 4–7 (2004)
22. Zowghi, D., Offen, R.: A logical framework for modeling and reasoning about the evolution of requirements. In: Int. Conf. on Req. Engineering, pp. 247–257 (1997)
23. Ghose, A.: Formal tools for managing inconsistency and change in RE. In: International Workshop on Software Specification and Design, pp. 171–181 (2000)
24. Hunter, A., Nuseibeh, B.: Managing inconsistent specifications: reasoning, analysis, and action. *ACM Transactions on Software Engineering and Methodology* 7(4) (1998)
25. Poppleton, M., Groves, L.: Software evolution with refinement and retrenchment. In: International Workshop on Refinement of Critical Systems: Methods, Tools and Developments, Turku, Finland (2003)

# On Analyzing Process Compliance in Skin Cancer Treatment: An Experience Report from the Evidence-Based Medical Compliance Cluster (EBMC<sup>2</sup>)<sup>\*</sup>

Michael Binder<sup>1</sup>, Wolfgang Dorda<sup>2</sup>, Georg Duftschmid<sup>2</sup>, Reinhold Dunkl<sup>3</sup>, Karl Anton Fröschl<sup>3</sup>, Walter Gall<sup>2</sup>, Wilfried Grossmann<sup>3</sup>, Kaan Harmankaya<sup>1</sup>, Milan Hronsky<sup>2</sup>, Stefanie Rinderle-Ma<sup>3</sup>, Christoph Rinner<sup>2</sup>, and Stefanie Weber<sup>1</sup>

<sup>1</sup> Department of Dermatology, Medical University of Vienna, Austria

`firstname.lastname@meduniwien.ac.at`

<sup>2</sup> Center for Medical Statistics, Informatics and Intelligent Systems, Medical University of Vienna, Austria

`firstname.lastname@meduniwien.ac.at`

<sup>3</sup> Faculty of Computer Science, University of Vienna, Austria

`firstname.[midname.]lastname@univie.ac.at`

**Abstract.** Process mining has proven itself as a promising analysis technique for processes in the health care domain. The goal of the EBMC<sup>2</sup> project is to analyze skin cancer treatment processes regarding their compliance with relevant guidelines. For this, first of all, the actual treatment processes have to be discovered from the available data sources. In general, the L\* life cycle model has been suggested as structured methodology for process mining projects. In this experience paper, we describe the challenges and lessons learned when realizing the L\* life cycle model in the EBMC<sup>2</sup> context. Specifically, we provide and discuss different approaches to empower data of low maturity levels, i.e., data that is not already available in temporally ordered event logs, including a prototype for structured data acquisition. Further, first results on how process mining techniques can be utilized for data screening are presented.

**Keywords:** Data Quality, Healthcare Processes, Process Modeling, Process Mining.

## 1 Introduction

The L\* life cycle model – defined in the Process Mining Manifesto [2] – presents a structured modeling approach for monitoring and analyzing real world processes in general. Application of this proposal to the domain of health care is a challenging task due to the following reasons. First of all, because of quality

---

<sup>\*</sup> The work presented in this paper was conducted in the context of EBMC<sup>2</sup> that is co-funded by the University of Vienna and the Medical University of Vienna.

concerns, event-logs in medical treatment do not fit easily into the hierarchy of maturity levels provided in the Manifesto. Rather, activities in the management of electronic health records, including data interchange, indicate that current practice still is a long way from transactional data management in, for example, business applications featuring already a fairly automatized and standardized tracking of business processes. Moreover, the specification of medical treatment process models is generally expressed in terms of high-level *medical guidelines* which is to say that, again, the accomplished level of standardization and formalization is fairly weak as compared to typical business applications. Many and even critical process details lack concise specification, much room is left for tacit, or uncodified, medical expert knowledge, and often important diagnostic or therapeutic decisions rest on patient conditions stated informally only. Since many treatment aspects and medical interventions are still questions of vivid research, this is no different in the particular case of skin cancer (melanoma) treatment focused upon in the subsequent discussion.

In spite of the various sources of imprecision and formal under-specification of treatment processes, this domain of consideration has been chosen deliberately because, on the one hand, there is a lot of current interest in assessing and improving clinical melanoma treatment routines while, on the other hand, growing incidence rates – combined with steeply rising costs of ever more efficient therapy options – suggest a significantly increasing relevance of effective as possible skin cancer treatment regimes. Thus, in addition to improvements in individual treatment episodes, treatment process analyses are also of particular interest in terms of population health and health economy, that is: resource allocation.

Methodologically, the issue obviously provides a fertile field of interdisciplinary research in medicine, data management, and epidemiology, seeking to improve the practice of medical decision making by use particularly of *evidence-based analysis* of data originating from empirical treatment processes. In this experience report, we describe first steps and results of an on-going effort to model medical care processes in the domain of skin cancer treatment as a prerequisite to the application of process mining techniques. Accordingly, Section 2 introduces the methodological set-up envisaged, trying to interrelate formal process analysis with relevant problem perspectives, notably evidence-based medicine and epidemiology. Next, Section 3 describes and evaluates data sources at hand. Section 4, then, presents a *temporal* data model in favor of integration of different data sources. On top of this, Section 5 provides preliminary evidence on how results of process mining may effectively support medical treatment practices. Finally, Section 6 summarizes the state of development, and concludes with an outlook on further activities of the project consortium.

## 2 The Treatment Compliance Analysis Framework

The ambition of analyzing empirical conformity of medical care processes to treatment standards defined by medical guidelines requires a linkage of expertise in fields of science as diverse as business process engineering, formal process modeling, medical information management, data analytics and statistical modeling,

and, last but not the least, medical knowledge and experience in skin cancer diagnosis and therapy. Apparently, such an integrated effort of analyzing evidence-based medical practice towards a valid interpretation according to statistically defined health determinants over whole patient populations calls for a cooperation of quite a range of disciplines. To this end, the Evidence Based Medicine Compliance Cluster (EBMC<sup>2</sup>, <http://ebmc2.univie.ac.at/>) has been instigated as a co-funded collaborative project of the Medical University of Vienna together with the University of Vienna, where the former contributes expertise predominantly in medical informatics and dermatology while the latter is in charge of process management knowledge and statistical expertise.

To lay the foundation for further discussion, this section maps, in brief, how formal process modeling techniques may help in assessing medical treatment compliance by way of identifying and representing, respectively, different empirical variants of treatment processes – some compliant to defined guidelines, others possibly not – from meshed-up medical data of available sources.

## 2.1 The Basic Analytic Setting and Related Work

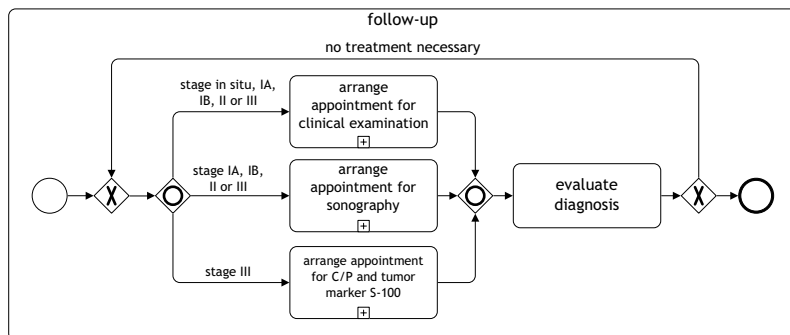
The basic setting of medical treatment compliance assessment rests on research in computer science, medicine, statistics, and the interplay between these areas. From a computer science viewpoint, the most important topics to be mentioned certainly are: process modelling, process mining, and information systems. In particular, the activities of the IEEE task force on process mining (notably, its Process Mining Manifesto [2]), recent developments in the area of standardization of event-log formats (cf. [www.xes-standard.org](http://www.xes-standard.org)) as well as the seminal process mining book [1] constitute basic theoretical pillars for project work in EBMC<sup>2</sup>.

As to medicine, the project considers medical guidelines (cf. GRADE [11]) as its most important source of input. Medical guidelines integrate bits and pieces of medical evidence studies into coherent treatment processes and, in so doing, provide assistance to health professionals towards effective, good-quality medical practice. However, one has to keep in mind that besides such codified medical knowledge there exists a considerable amount of *informal*, and thus hard to come by, knowledge based on the long-term expertise of physicians. Through observation of actual treatment paths the inferred physicians' diagnostic and therapeutic choices are practically filling in the top-level advice of guidelines.

As a first use case in the project, the guidelines for treating patients with cutaneous melanoma have been selected. Skin malignancies are generally recognized as global major health problem among Caucasian populations, all the more so as incidence rates keep rising.

Currently, already a number of proposals combines ideas from workflow management with medical guidelines; for example, modeling languages such as GLIF, Asbru, EON/SAGE, PROforma, GUIDE, or PRODIGY, all of which are fairly close to traditional workflow languages [14]. From the field of business process engineering, BPMN (Business Process Modeling Notation) represents the state-of-the-art, and a direct comparison between BPMN and PROforma confirms





**Fig. 1.** Guideline Stage: Follow-up Subprocess

that BPMN can indeed cope quite well with requirements of the health care domain [9]. Hence, EBMC<sup>2</sup> decided to favor BPMN as modeling language for re-expressing formally the mostly verbal medical guideline.

For the sake of illustration, Figure 1 shows part of the European guideline on melanoma treatment [10] modeled in BPMN. The exhibit depicts abstractly the aftercare process highlighting follow-up examinations of melanoma patients, that is, patients are coerced to arrange appointments for regularly spaced follow-up examinations depending on their respective *stage* of melanoma status as defined in the cited guideline. In the example shown, stage I to III melanoma patients are covered, while stage IV patients – who already suffer from distant metastases – are not included in this scheme as, because of the severe progress of their medical condition, all due examinations are arranged individually in order to locate any further metastases as quickly as possible.

It is little surprise that empirical data about treatment processes are often structured simply according to the specific purpose at hand. Thus, many times the guiding principle of data organization involves barely more than a crude template for diagnostic findings. Certainly, those ad hoc data approaches badly cover any unprecedented later data usage, encouraging diverse efforts in medical informatics to remedy this obvious shortcoming: (i) a terminology-oriented approach, giving rise to medical ontologies like SNOMED [18]; (ii) a data exchange-oriented approach like the HL7 model [12]; and (iii) an archetype-oriented approach, most typical of which probably being the Open-EHR initiative [19]. To EBMC<sup>2</sup>, all of these ideas are of utmost importance as a means of formally re-expressing (medical) information and (treatment) knowledge: based on such formal representations, models for the integration of patient data of different origin can be arranged to carry data transformations bearing data formats compatible with process mining tools. The latter provide the machinery for the identification of *typical* empirical realizations of treatment processes as defined theoretically by medical guidelines and, in doing so, deliver the main (statistical) input for all successive compliance analyses. In terms of methodology, statistics mainly contributes to the interpretation of empirical process structures, based on which the

re-design of process models commences with respect to a perspective of population health. In this regard, cancer registries and cause-of-death registries, respectively, provide the most valuable analytical contributions of official statistics as these sources of data allow the calculation of both incidence and prevalence rates (using established principles of epidemiology); however, this issue will not be pursued any further in what follows.

### 2.2 Comparing Guidelines with Empirical Data

Analyzing the relationship between the process model defined by guidelines on the one hand and empirical process data represented in formal event-logs on the other hand constitutes one of the main tasks in the stages 2 and 3 of the  $L^*$  model. In EBMC<sup>2</sup> context, an overall methodology is being developed that admits the transformation of both, real world medical data and (mostly informal) medical guidelines, to a (formal) level of representation amenable to a rigorous analytical comparison. Furthermore, this transformation is supposed to allow compliance checking to feed back derived evidence-based process models to adaptations of the guidelines themselves (Fig. 2).

This model conceives the treatment process depending, in all medical applications, on two different types of attributes, viz. (i) patient specific attributes (denoted by  $X$  in Figure 2), and (ii) institutional attributes,  $\theta$ , capturing a variety of peculiarities of health care units possibly affecting treatments in a relevant way; the latter attributes are termed “parameters” further on. Usually, medical guidelines do not include all patient attributes which may conceivably influence the treatment process. For example, the guideline for skin cancer treatment accounts for the staging of the tumor only, but omits any reference to personal characteristics such as age or gender. Empirically, of course, such attributes may account significantly for a treatment progression. Hence, Figure 2 splits the personal attributes,  $X = (X_d, X_p)$ , where  $X_d$  denotes the set of diagnostic attributes

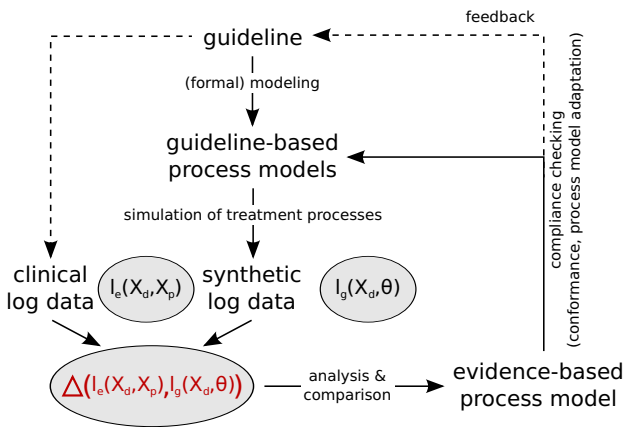


Fig. 2. Basic Methodology (cf. [5])

used explicitly in the guideline, while  $X_p$  refers to patient attributes not at all mentioned there.

The left-hand part of Figure 2 indicates the analytical path producing real world clinical data from applying treatments to (melanoma) patients supposedly implementing the pertinent medical guideline. Accordingly, the emanating clinical log data reflect both structure and terminology used by the guideline, albeit originating from different sources (e.g., hospitals, or clinical information systems etc.). Contrary to the medical processes bringing forth event-log data, the middle section of Figure 2 visualizes the path from guideline via formal process model (derived from the guideline) to synthetic log data, obtained from conducting simulated patient treatment processes. In other words, in a first step the guideline gets transformed into a guideline-based process model based on the verbal guideline as well as additional knowledge of domain experts, enriching the guideline-based process model. This enriched model, taking account of also diagnostic attributes as well as the institutional parameters (the statistical distributions of which chosen to reflect real patient demographics), feeds into a process simulation tool that produces synthetic log data as its output.

Both clinical and synthetic log data can now be analyzed with process mining tools to sift out empirical process representatives. The first step in analyzing log data is to generate a process model which allows further statistical techniques to be applied like clustering [1] or decision analysis [17]. In this respect, it is of course particularly interesting to inquire whether, and to which degree, there is consensus with medical experts about the actually mined treatment processes compared to their guideline-based reference process: process mining thus is used as a means of checking consistency of guideline specifications (cf. Section 5 for an example). A pivotal *delta* analysis [1] consists in comparing clinical and synthetic log data to assess actual concordance. Since, in general, treatment processes may depend on personal patient attributes, some diversity amongst – still compliant – treatment processes is to be expected (for a preliminary description cf. [5]). For data with irregular time structure methods well known in the area of longitudinal data analysis [7,8] can be applied.

Finally, the right-hand part of Figure 2 emphasizes the feedback of analysis results as condensed analytically in the derived evidence-based process model to the formal treatment process model (to better fit observed clinical practice), or – eventually – to the adaptation of the guideline itself. Feedback with respect to the population health as mentioned in the previous section will not be pursued any further in the present paper.

### 3 Available Data Sources: Problems and Challenges

For the time being, work in EBMC<sup>2</sup> draws upon two main sources of melanoma-related data, viz. (i) a detailed data collection of clinical *Cutaneous Melanoma* (CM) stage IV protocols (Stage IV Melanoma Database, S4MDB, for short), and (ii) administrative data of the Main Association of Austrian Social Security Institutions comprising a billing-oriented view of medical patient treatments

(GAP-DRG). In neither case, data sources conform well to the L\* life cycle model (cf. [2]). This is true also of another relevant source of data, the Austrian Cancer Registry, which, however, is not used in EBMC<sup>2</sup> as yet.

### 3.1 The Stage IV Melanoma Database

*Cutaneous Melanoma* (CM) is a malignant tumor arising from melanocytic cells and primarily involving the skin. It is the deadliest form of skin cancer causing approximately 90 % of skin cancer mortality [10]. CMs are staged according to the classification of the American Joint Committee on Cancer (AJCC) [3]. In a retrospective data analysis, 389 patients diagnosed with *Metastatic Melanoma* (MM), who underwent therapeutic treatment between 2000 and 2010 at the Department of Dermatology, Medical University of Vienna, were included in an electronic database. Only patients with a histologically verified melanoma (AJCC stage IV) were included in this study. The data were retrospectively assessed by 2 diploma students under the supervision of an expert in dermatology. The progress of patients with MM, the median survival rate and the influence of different treatment options on survival were the major objectives of this project. Furthermore, the median survival and long-term survival rate were evaluated with reference to different prognostic criteria. The study was approved by the local ethics committee. The data acquisition was based on 3 different sources: the hospital information system (HIS) as well as digital archived and physical patient records. The patient data were anonymized, using a unique patient-specific identification number. Relevant parameters included melanoma type, localization of the primary tumor, date of the primary excision, histological diagnosis, and the AJCC tumor stage at time of excision. For all metastatic events, diagnostic procedures to verify the metastatic tumor stage, localization of metastases, and treatment regimes, including the number of treatment cycles and the treatment duration, were evaluated.

In view of process mining purposes, evaluation of available stage IV data made it apparent quickly that data could not be used as is; rather, a structured data re-acquisition was due. While available data met the requirements for medical evaluation of treatment cycles, durations, and outcomes of patients with MM, they often typically lacked appropriate *temporal* references which, for obvious reasons, are of utmost importance to process mining. Additionally, causative relationships of different clinical pathways are of pivotal interest. Thus, a re-design of the original S4MDB structure became inevitable, including particularly time-related clinical pathways based on the analysis of individual patients.

### 3.2 Austrian Social Security Institution Data

The Main Association of Austrian Social Security Institutions maintain a huge data repository gathering patient and treatment related medical data, for accounting and billing purposes, of all (domestic as well as foreign) patients treated in Austria. A subset of pseudonymized data covering the period from January

2006 to December 2007 is available in an accessible GAP-DRG database including hospitalization data [4], patient treatments received from resident physicians, administered medications dispensed at pharmacies as well as sick certificates. GAP-DRG harmonizes some 15 data bodies of different Austrian Social Security Institutions, making use of established taxonomies such as (i) the ICD10 (International Statistical Classification of Diseases and Related Health Problems) to document primary and ancillary diagnoses, (ii) a MEL catalog [4] to describe medical treatments in hospitals, (iii) a national “Metahonorar” catalog coding medical treatments of resident physicians, and (iv) the ATC (Anatomical Therapeutic Chemical Classification System) codes to categorize the active agents of medications dispensed at pharmacies. GAP-DRG relates data entries to a rough temporal grid of billing periods determined by (legal) accounting requirements. Although entailing some effort, data entries can be linked to pseudonymized individual patients, so that the data subset of skin cancer (melanoma) patients can be selected effectively from the database.

### 3.3 Data Source Integration and Assessment

The joint analysis of CM patient treatment data of different provenance necessitates a preliminary data integration step. Accordingly, EBMC<sup>2</sup> explores and applies Extract-Transform-Load (ETL) methodologies [13] to wrap data sources, highlighting particularly

- *time-related data* ([1]: 113) indispensable for ordering medical activities and deriving / mining treatment processes, and
- *routing data* flowing into decision point analysis for expressing process branchings.

Identification and allocation of *time-related data* are acknowledged challenges in extracting event-logs from data sources, as are the different levels of granularity ([1]: 114). Compared to the level hierarchy established in the Process Mining Manifesto [2], the data sources introduced in Subsections 3.1 and 3.2, respectively, roughly fare as exhibited in Table 1.

First, and preliminary, process mining experience in analyzing patient data from both of the data sources considered can be summarized as follows:

- S4MDB data: throughout, a process comprising a sequence of three activities, appearing in identical order, was mined for all of the patients. Due to the specific purpose of the S4MDB, no system log was maintained for data recording, conveying only limited information to process mining. However, the database offers quite detailed information potentially useful as *routing data* for analyzing decision points within treatment processes.
- GAP-DRG data: After transformation, this dataset results in fairly meaningful process models due to the temporal indexing of the provided data. Conversely, it became apparent that *routing data* are lacking by and large, compromising decision point analysis severely.

**Table 1.** Evaluation of Data from Different Data Sources

	Level	Characteristics
Medical Records	*	<ul style="list-style-type: none"> <li>● recorded by hand</li> </ul>
S4MDB	* / **	<ul style="list-style-type: none"> <li>● recorded by hand</li> <li>● no systematic approach</li> <li>● trustworthy</li> <li>● correct</li> </ul>
GAP-DRG	**	<ul style="list-style-type: none"> <li>● recorded by hand</li> <li>● recorded systematically</li> <li>● trustworthy</li> <li>● correct</li> </ul>

- A *guided* post-collection of additional, or enriching, data is indispensable for obtaining satisfactory process mining results: since, in general, existing data sources have been designed quite unaware of later use in process mining, linking them to add-on data providing temporal references or further explanatory patient attributes, whenever possible, helps to increase their analytical value for process mining significantly.

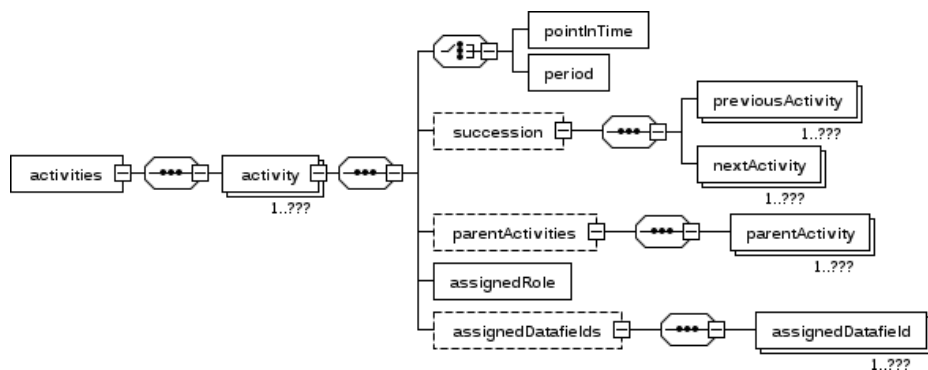
Unfortunately, this is a recommendation hard to achieve most of the time because, as in the medical domain of interest, available data often is structured weakly only (e.g., in terms of digital scans of documents) and prepared ad hoc for particular analyses; furthermore, the diversity and locality of data sources tends to inhibit effective data integration, a problem additionally exacerbated by problems of measurement incommensurability and taxonomy mismatches enforcing, e.g., a laborious recoding or disadvantageous aggregation of data entries.

## 4 Empowering the Data Sources

Each data source presents its own challenges, rendering a *universal* approach of data management quite elusive; therefore, also in this process/data mining project an indispensable preparatory step concerns the consolidation of data sources [6]. Accordingly, the following subsections describe crucial steps of data preparation by data source. This experience report of data consolidation illustrates the importance of this central task to build a basis for the following process mining application.

### 4.1 Meta Data Enrichment of the Stage IV Melanoma Database

As already mentioned, the S4MDB lacks critical *time-related data*. While retaining the flat S4MDB structure, a set of meta data is introduced to add *structural* information. Figure 3 depicts an XML schema of this ensuing enrichment file. Now, a medical activity can either correspond to a *pointInTime* (with a single date) or a *period* (with both, a start and an end date). XML element attributes



**Fig. 3.** XSD for Meta Data Enrichment of the Stage IV Melanoma Database

are used for referencing the S4MDB column headings comprising the respective data. If not yet existing, further columns are simply added to the S4MDB structure.

The *succession* element of the XML enrichment file enables the interpolation of calendar dates, whenever missing, for activity entries in S4MDB. To accomplish the interpolation, all *previousActivity* and *nextActivity* elements can be searched jointly in the meta XML file and S4MDB; detecting the latest calendar date of all previous activities and the earliest calendar date of all successive activities, respectively, allows derivation of the missing calendar date.

Likewise, the element *parentActivities* helps to resolve different data granularities (cf. Subsection 3.3). For instance, S4MDB, comprises a lot of subordinate follow up treatments not even mentioned in the guideline. Therefore, it is reasonable to group such treatments to the granularity level of the guideline in order to accommodate for comparisons without changing the original database.

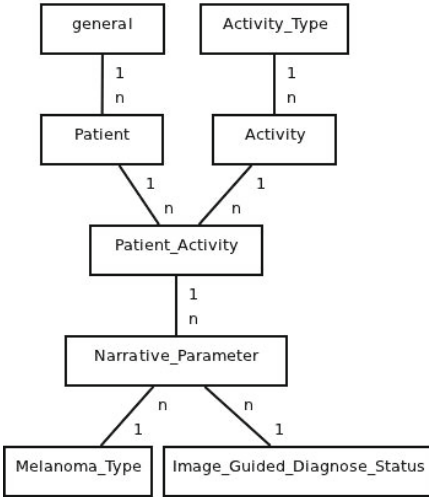
The element *assignedRole* admits inclusion of organizational information about which institution performed the recorded activity. While not of specific interest for the time being, this information may become relevant later on. Additional data different from genuine activities to record – the bulk of S4MDB data, indeed – can be added using the *assignedDatafields* element; this information is of use in, e.g., decision point analysis.

The meta data enrichment XML file for the existing S4MDB, comprising a test sample of ten patients, was created by medical practitioners of EBMC<sup>2</sup> for evaluation purposes reported on in Section 5.

## 4.2 Restating the Data Model

Meta data enrichment relieves, to some degree, the rigidity of the flat S4MDB structure. To gain yet more versatility in representing treatment process data, the data model depicted on the left side of Figure 4 was proposed in favor of the requirements declared in Section 2. Similar to meta data enrichment, it defines activities (*Activity*) with assigned data fields (*Narrative\_Parameter*).

A) Original Model



B) Enhanced Model

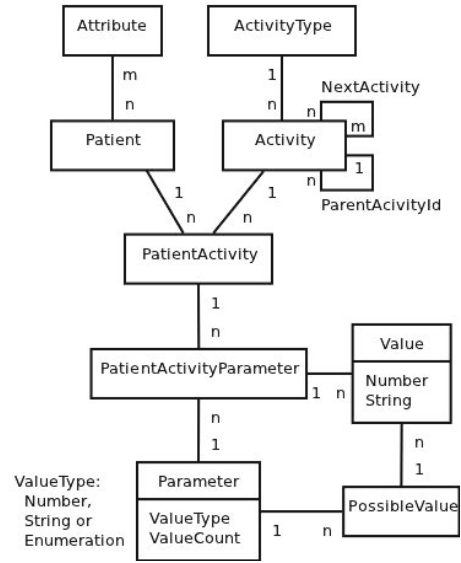


Fig. 4. Initial Melanoma Data Structure and Adapted Extensible Data Structure

This model does not yet comply with the Process Mining Manifesto [2], and thus does not apply generally to all conceivable clinical applications. Hence, a further generalization of the data model is depicted on the right side of Figure 4, allowing for the addition of – typed – data fields as separate entities, this way avoiding changes in the data structure while keeping the data model consistent and normalized. In turn, this more generic approach demands a higher discipline of data collectors as it allows the *ad hoc* definition of possible values and data types. The model also includes (i) information about the sequencing of activities (*NextActivity*), useful, e.g., in interpolating missing activity dates, and (ii) hierarchical information (*ParentActivity*) for handling varying value granularities of data entries.

The devised data structure has been implemented as a web application (named *PTDoc*, Patient Treatment Documentation), using the ZK Java Framework [16] on top of MySQL [15] as data store. The user interface, cf. Figure 5, splits into three panes, to be walked through from left to right:

- Pane 1: select either an existing patient, or add a new one
- Pane 2: assign either an existing activity or a new one to this patient
- Pane 3: add new parameters to that activity of this patient

The application has an *administration section* where new activities as well as their parameters and default values can be registered, modified or deleted. The *PTDoc* application is currently evaluated at the Department of Dermatology at the Medical University of Vienna.



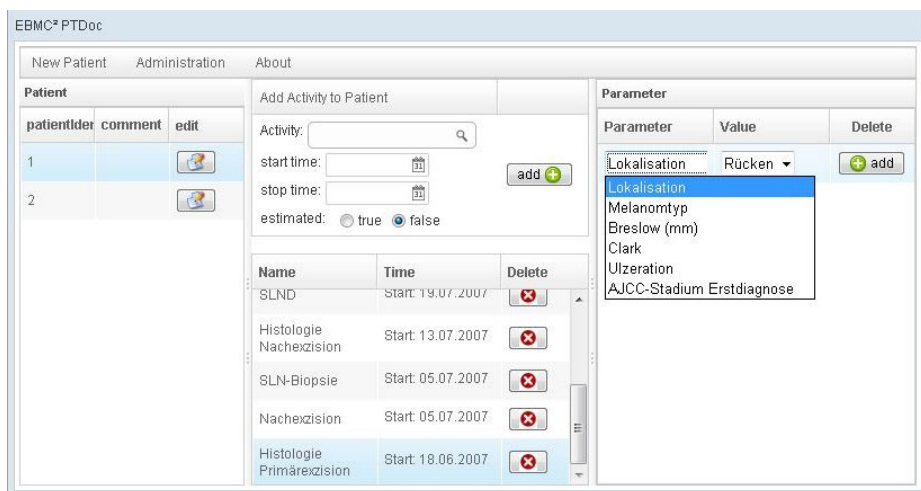


Fig. 5. Screenshot: Prototype - Patient Treatment Data

### 4.3 GAP-DRG Data Transformation

As described, this data source provides rather good time-related data. However, when it comes to routing data, little information is offered only. Hence, again, in spite of its analytical usefulness, GAP-DRG data needs a preprocessing.

Essentially, GAP-DRG data is organized by hospitalizations, each carrying calendar dates of patient admissions and dismissals but no direct reference to treatments (activities). Rather, diagnosis codes are attached to hospitalizations from which the relevance of the data for subsequent analysis can be deduced. Diagnosis codes also carry further activity-related information such as, for example, an ICD-10 code C43-1480 entry telling implicitly that the patient had a malign melanoma excised on the recorded day of hospitalization; additionally, the code also discloses the localization of the melanoma (in this case, the ear or the acoustic meatus). To achieve a canonical representation of this information, such super-code data entries need to be split into elementary PTDoc components.

Likewise, GAP-DRG data does not provide direct information about administered treatments as defined in the medical guideline; rather, medications picked up at the pharmacy are recorded, from which underlying treatments may be reconstructed.

## 5 Evaluation Based on Selected S4MDB Cases

In a pilot trial of process mining comparing actual clinical treatment processes with specific peer-reviewed medical guidelines, data of 10 patients selected from S4MDB (cf. Subsection 3.1) were entered in PTDoc. In this first attempt of deriving an evidence-based melanoma treatment process model, the heuristic miner (HM, for short) of the ProM framework [14] was used to generate a process

model using the provided data. The ProM framework hosts a set of modules implementing various process mining algorithms processing input log data. After transforming S4MDB data into the ProM input log file format, it was possible to generate a first process model. This model was used for evaluation with domain experts to verify its compliance with medical reality.

During discussion of the resulting process model, generated with the HM, with dermatological experts, the following three conspicuous features became apparent; cf. excerpts of this process model shown in Figures 6 and 7, respectively:

- starting activities are mis-identified as concurrent (Figure 6);
- an additional treatment commencing during another treatment already applied, properly reflecting reality (Figure 7 A);
- (two) treatments applied in succession, contradicting medical reality (Figure 7 B).

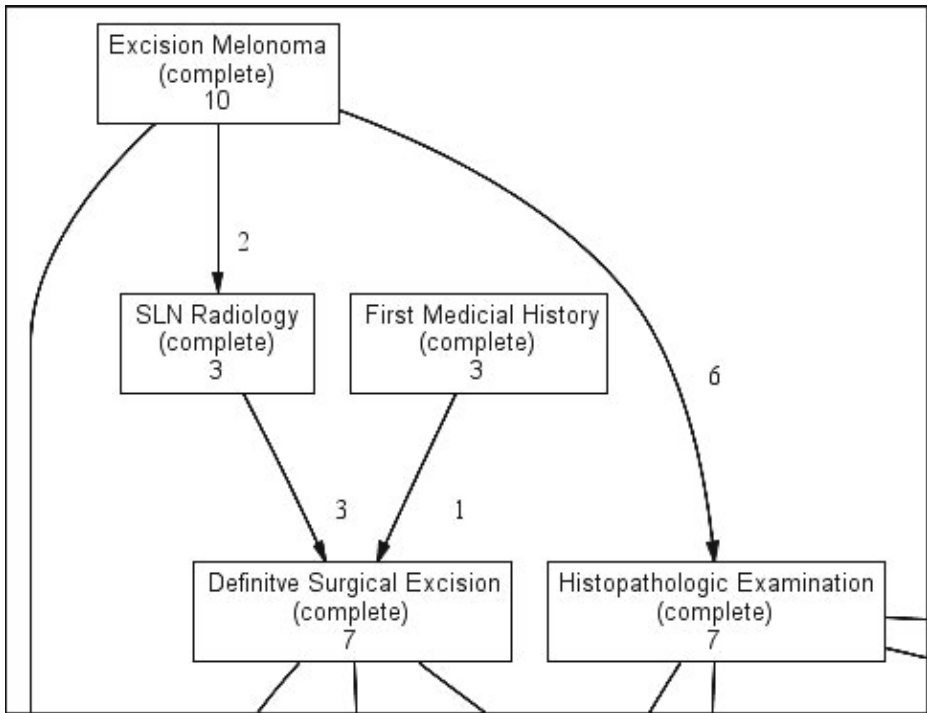
In both figures, each activity bears either a *start* or a *complete* label indicating the starting and ending times of activities, respectively (instantaneous activities record an ending time only). Within the figures, numbers represent (absolute) frequencies of activities or activity transitions.

Figure 6 depicts the case of activities *First Medical History* and *Excision Melanoma* identified erroneously as feasible starting activities *pari passu* while *First Medical History* – recorded in only 3 out of 10 observed cases – is supposed to precede *Excision Melanoma* throughout. Checking the data set for these three cases immediately explains why the HM assumes these activities as parallel: in only one case *First Medical History* actually preceded *Excision Melanoma*. In another case both activities recorded the same day, thus inhibiting the inference of any activity precedence by the algorithm, while in the third case the sequence of the activities had been recorded in reverse order – probably a previously unrecognized data glitch. Based on just three cases like these, the HM simply cannot detect the correct activity precedence relation.

Conversely, Figure 7 A) demonstrates a treatment modality for MM patients where, during treatment with the chemotherapeutic drug Fotemustine, also an IL2sc treatment – a proinflammatory cytokine – commenced. This treatment pathway is plausible and represents clinical reality.

In contrast to that, Figure 7 B) again demonstrates a treatment modality for MM patients where the application of Gamma Knife (used in radiosurgery of the head to treat brain metastases) follows the treatment with Thalidomide (a drug also used in oncology). Both treatments are applied for patients with late-stage melanoma. However, even though the therapeutic setting implies a thematic proximity of these treatments, this time the induced activity precedence conflicts clinical reality, representing rather an artifact probably caused by another data error.

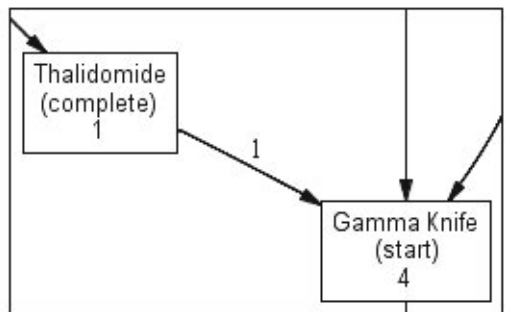
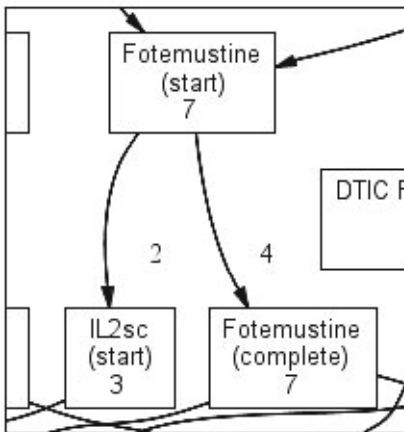
Altogether, these examples of a first evaluation of the methodology show the strengths of the process view as a data-screening instrument. Clearly, a sample of just 10 observation cases falls short of admitting any far-reaching conclusions, but still can pinpoint salient data (and data quality) requirements as well as the advantages of a *process-centered* view of patient treatment data, shedding light on guideline adherence of medical practice.



**Fig. 6.** Excerpt from the Process Model minded by the Heuristic Miner showing the identified starting activities

A) Reflecting Reality

B) Contradicting Reality



**Fig. 7.** Excerpt from the Process Model minded by the Heuristic Miner showing an activity sequence A) reflecting and B) contradicting reality

## 6 Summary and Outlook

In this report, we presented first experiences with the process-driven analysis of skin cancer treatment processes in the EMBC<sup>2</sup> project, specifically analysis of their guideline compliance. Focus was put on the transformation and integration of the available data sources, i.e., clinical Cutaneous Melanoma stage IV protocols as well as billing data of the Main Association of Austrian Social Security Institutions. The challenge was to extract and integrate the data in a process-oriented way in order to apply process mining techniques in the sequel. Due to the characteristics of the different data, techniques such as meta data enrichment and extended data models are proposed. A glimpse into process mining results in the form of data screening are presented. In future work, we will extend and enrich the available data in order to analyze the differences between the actual treatment processes and the corresponding guidelines and will use melanoma-relevant excerpts of the Austrian Cancer Registry for interpretation.

## References

1. van der Aalst, W.: *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer, Heidelberg (2011)
2. van der Aalst, W., et al.: *Process Mining Manifesto*. In: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) *BPM Workshops 2011, Part I. LNBIP*, vol. 99, pp. 169–194. Springer, Heidelberg (2012)
3. Balch, C., et al.: *Final version of 2009 ajcc melanoma staging and classification*. *Journal of Clinical Oncology: Official Journal of the American Society of Clinical Oncology* 27(36), 199–206 (2009)
4. Bundesministerium für Gesundheit, *Leistungsorientierte Krankenanstaltenfinanzierung - L K F - Medizinische Dokumentation*, [http://www.bmg.gv.at/cms/home/attachments/0/4/1/CH1166/CMS1128332936305/medizinische\\_dokumentation\\_2012.pdf](http://www.bmg.gv.at/cms/home/attachments/0/4/1/CH1166/CMS1128332936305/medizinische_dokumentation_2012.pdf) (accessed November 20, 2011)
5. Dunkl, R., Fröschl, K.A., Grossmann, W., Rinderle-Ma, S.: *Assessing Medical Treatment Compliance Based on Formal Process Modeling*. In: Holzinger, A., Simonic, K.-M. (eds.) *USAB 2011. LNCS*, vol. 7058, pp. 533–546. Springer, Heidelberg (2011)
6. Džeroski, S.: *Towards a General Framework for Data Mining*. In: Džeroski, S., Struyf, J. (eds.) *KDID 2006. LNCS*, vol. 4747, pp. 259–300. Springer, Heidelberg (2007)
7. Everitt, B., Hothorn, T.: *A Handbook of Statistical Analyses Using R*. Chapman & Hall-CRC Press, Boca Raton (2006)
8. Fitzmaurice, G., et al.: *Longitudinal Data Analysis (Chapman & Hall/CRC Handbooks of Modern Statistical Methods)*. Chapman & Hall-CRC Press, Boca Raton (2009)
9. Fox, J., Black, E., Chronakis, I., Dunlop, R., Petkar, V., South, M., Thomson, R.: *From guidelines to careflows: Modelling and supporting complex clinical processes*. In: *Computer-based Medical Guidelines and Protocols: a Primer and Current Trends*, pp. 44–61. IOS Press, Netherlands (2008)

10. Garbe, C., Peris, K., Hauschild, A., Saiag, P., Middleton, M., Spatz, A., Grob, J., Malvey, J., Newton-Bishop, J., Stratigos, A., Pehamberger, H., Eggermont, A.: Diagnosis and treatment of melanoma: European consensus-based interdisciplinary guideline. *European Journal of Cancer* 46(2), 270–283 (2010)
11. Guyatt, G., Oxman, A., Schünemann, H., Tugwell, P., Knottnerus, A.: Grade guidelines: A new series of articles in the journal of clinical epidemiology. *Journal of Clinical Epidemiology* 64(4), 380–382 (2011)
12. Health Level Seven International, <http://www.hl7.org/> (accessed November 29, 2011)
13. Jarke, M., Lenzerini, M., Vassiliou, Y., Vassiliadis, P.: *Fundamentals of Data Warehouses*. Springer, Heidelberg (2010)
14. Mans, R.: *Workflow Support for the Healthcare Domain*. Proefschriftmaken.nl, Netherlands (2011)
15. Oracle Corporation, <http://www.mysql.com/> (accessed November 28, 2011)
16. Potix Corporation, <http://www.zkoss.org/> (accessed November 28, 2011)
17. Rozinat, A., van der Aalst, W.M.P.: Decision Mining in ProM. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) *BPM 2006*. LNCS, vol. 4102, pp. 420–425. Springer, Heidelberg (2006)
18. The International Health Terminology Standards Development Organisation, <http://www.ihtsdo.org/> (accessed November 29, 2011)
19. The openEHR Foundation, <http://www.openehr.org/> (accessed November 28, 2011)

# Compliance Evaluation Featuring Heat Maps (CE-HM): A Meta-Modeling-Based Approach

Dimitris Karagiannis<sup>1</sup>, Christoph Moser<sup>1</sup>, and Arash Mostashari<sup>2</sup>

<sup>1</sup> University of Vienna, Research Group Knowledge Engineering Bruennerstraße 72,  
1210 Vienna, Austria

{dk, c.moser}@dke.univie.ac.at

<sup>2</sup> BOC IS GmbH, Wipplingerstr. 1, 1010 Vienna, Austria  
arash.mostashari@boc-eu.com

**Abstract.** The recent global economic crisis and the growing reliance on information technology pile pressure on organizations to comply with regulations, legal rules and laws. Organizations tend to struggle with paper work for the mere purpose of proving compliance. Compliance Management should be subject to existing management frameworks. Otherwise, ineffective procedures will evolve, and the organization will fail to combine Compliance Management with continuous improvement measures. We advocate for a generic compliance evaluation method, which builds upon existing enterprise modeling frameworks, for three reasons: First, synergies in data acquisition will arise. Second, reusing organization-wide accepted viewpoints will create trust among stakeholders and ease communication of the compliance status. Third, taking steps to improve compliance will be part of daily operations based on institutionalized processes geared to the established management frameworks. In addition, the prototypical implementation of the ‘Compliance Evaluation Featuring Heat Maps (CE-HM)’ method based on a meta-modeling platform is presented.

**Keywords:** Compliance, Evaluation, Heat Maps, Meta-Modeling.

## 1 Introduction

Leveraging internal auditing is the bedrock of Compliance Management, regardless of whether compliance audit refers to internal procedures in organizations, external regulatory requirements, such as Sarbanes Oxley, Basel II/III, and Solvency II, or industry codes of conduct, such as ISO 14001 for Environmental Management.

While the definition of the term Compliance and Compliance Management varies among different sources, one can point out a number of similarities between these definitions. For this paper we define *Compliance* in a broad sense, namely as the set of laws, regulations, policies, or best practices an organization needs to fulfill, regardless of whether they are self-imposed or forced by governmental regulations. We define *Compliance Management* akin to [24], who refers to it as the processes for accomplishing, monitoring and managing compliance of an organization.

Organizations often pursue Compliance Management merely to comply with legal rules and laws, without any context to the organization's continuous improvement measures. Our method is based on enterprise modeling methods, such as Business Process Management and Enterprise Architecture Management. The advantages that accrue from the synergy between enterprise modeling methods and Compliance Management are as follows: first, the existing enterprise models provide an ideal starting base increasing the productivity by sharing the efforts for creating and maintaining the underlying information base. Second, synergies arise by reusing established and well-known viewpoints, such as process maps or application architecture diagrams. Third, as Compliance Management activities are based on established BPM/EAM processes, resolving compliance issues will be part of day-to-day work activities. However, it must be indicated, that although the presented method supports both, a green-field approach as well as the reuse of an existing information base and established viewpoints, the latter can only be achieved if the existing content is held in a meta-modeling platform.

An intuitive and coherent visualization of the organization's compliance state will support internal audits and help the organization to pass external audits [24]. In our method, we utilize heat maps to visualize the degree of conformance with regulations. The term heat map originates from the field of cartography. Cartographers use choropleth maps to highlight specific areas on maps by coloring or shading them. However, in enterprise modeling literature, the preferred term for this type of maps is referred to as heat map (see e.g. [14] and [17]). We see heat maps as a powerful tool quantifying compliance, enabling organizations to visualize their compliance gaps and to take steps to improve compliance. By applying to Shneiderman's visualization mantra [25]: 'overview first', 'zoom and filter', and 'details on demand', our method combines a variety of viewpoints, such as diagrams and matrices [27]. We implemented our CE-HM method (according to the definitions established by the Open Model Initiative [13]) on the meta-modeling platform ADOxx. This method is designed for tools that enable direct and simple access to the meta-model of the provided modeling method.

The remainder of the paper is organized as follows: In section 2 related approaches to Compliance Management, some of them laying the groundwork of our method, are summarized. Furthermore, a number of general requirements for compliance management frameworks are briefly discussed. Subsequently, section 3 discusses the theoretical grounding we apply, namely the definition of the modeling method, followed by our main contribution, which emphasizes a compliance evaluation method based on meta-modeling and heat maps (CE-HM). Section 4 briefly presents a CE-HM prototype utilizing the meta-modeling platform ADOxx. Finally, section 5 gives a summary followed by an outlook on future work.

## 2 Related Work

Prominent standards and frameworks for Compliance Management, such as COSO, CobiT, and SAC (for a broad overview see for example [4]), typically provide a rich

foundation of guidelines in the form of control objectives or principles. However, none of those give detailed methodical advice to measure the degree of compliance. Typically, they do not provide tool support for assessing and communicating compliance. Thus, usually spreadsheets are used for assessing the compliance status, an approach [9] referred to as document-based approach, being the most common approach for Compliance Management at the present time. [9] states as one of the major drawbacks of this manual applied approach, that reusability is significantly impacted and the approach is typically associated with high efforts for ongoing maintenance activities, status tracking and communication of compliance issues. In the following, we focus on related work in the area of conceptual modeling, as it is one of the main pillars of our method.

As early as the end of the 1970s, [18] recognizes the potentials of supporting Compliance Management through conceptual models. [26] explicates that most present approaches focus on Business Process Management such as the Business Process Management Notation (BPMN) and Event-driven Process Chain (EPC), and advocates enterprise modeling approaches such as MEMO [8] and ArchiMate [16], usually covering the entire organization as an ideal foundation for Compliance Management. By giving advice on enriching enterprise modeling approaches in order to support internal control modeling, the authors constitute the groundwork for the compliance evaluation method at hand. However, they put a strong focus on language design. Evaluation mechanisms and specific visualizations for analyzing, visualizing and communicating the degree of compliance are not central.

According to [1], architecture principles (classified into inherent laws, imposed laws, and guidelines) are mainly imposed to limit design freedom regarding enterprise architectures. Hence, in the broader sense, architecture principles address the subject of compliance with regulations, laws and standards. EA frameworks, such as TOGAF [27], FEAF [6] and TEAF [5] exemplarily state architecture principles and give advice on how to define an appropriate set of principles. According to [7], one important quality for every principle is to clearly define how to measure its fulfillment. However, none of the aforementioned EA frameworks specify a method for measuring the fulfillment level consistently, nor suggest how to visualize and communicate the compliance posture.

[14] and [17] introduce heat maps in the fields of EA Management for visualizing business capabilities. Although, merely discussing the approach in the context of Business Capability Management, the sketched heat mapping mechanisms can be applied to any modeling domain. According to [14] *'representation of heat maps allows a high variability in itself'*. It is stated that any set of attributes (of the underlying meta-model) can be evaluated and combined using any evaluation function, assigning colors to the graphical representation of architecture artifacts represented in a diagram. However, concrete evaluation functions are not discussed.

An example of an advanced evaluation function for qualitative enterprise architecture management, based on Probabilistic Relations Models (PRM), is presented in [2]. The quality 'availability' is exemplarily used for illustration purposes although the authors state that the method is meant to be applicable more generally. The discussed method as a prerequisite requires a strong foundation of input parameters. This might



restrain organizational-wide acceptance and use. Furthermore, as the method is not explicitly intended for ongoing Compliance Management activities, mechanisms for putting the underlying architecture artifacts and their relations under control (e.g. time-related views, versioning) are not elaborated.

In the following, a short overview on requirements for compliance evaluation methods – identified for the CE-HM - is given. The requirements have been distilled from the above discussed literature as well as from [30], and above all from [29]:

*Requirement 1 – Change Management:* Regulations underlie continual change. The method needs to deal with this permanent change. This requirement will be covered by the time-view mechanisms, discussed in section 3.2, allowing to represent different points in time (as-is, intermediate and target architectures) in one repository.

*Requirement 2 – Traceability and Accountability:* This requirement is covered by the ADOxx platform providing a detailed changelog for any changes made to the models and evaluations.

*Requirement 3 – Complexity:* This refers to the complexity of the involved modeling effort. According to [29] ‘a generic compliance management framework should not be custom-tailored to a specific purpose or a specific domain’. This requirement is covered, as the CE-HM method works with any meta-model.

*Requirement 4 – Efficiency:* This requirement focuses on the efficiency of the underlying principles. The CE-HM method advocates the usage of principles derived from best practice frameworks, such as CobiT [11], but does not incorporate mechanisms to assess the efficiency of the chosen principles as such. Therefore, this requirement is only covered indirectly by the presented method in this paper.

*Requirement 5 – Cost:* The method must allow for economical and resource-conserving processes. CE-HM resolves this by reusing existing model base of established modeling methods, thus saving time and efforts.

*Requirement 6 – Enforceability:* The method should support the enforceability of defined principles. This is done implicitly by combining CE-HM with established management methods such as TOGAF [27], which typically provide the required management structures.

*Requirement 7 – Scalability:* The method must allow for growth (and scoping). Typically the number of principles and models will grow over time. The method needs to consider this and remain manageable.

*Requirement 8 – Multi-dimensional Evaluation and Drill-Down Mechanisms:* According to [30], means for ‘evaluation along structural, temporal and functional aggregation dimensions of an organization’ must be possible. Furthermore, propagation of evaluations needs to be supported. Hence, it must be possible to aggregate low-level evaluations (e.g. individual organizational units) to more abstract levels (e.g. the entire organization). This requirement is covered by the discussed evaluation configurations and the corresponding propagation mechanisms of the CE-HM method (see section 3.2), which work with any established meta-model.

### 3 The Compliance Evaluation Method ‘CE-HM’

The CE-HM method – and according to [15], any structured modeling task – is based on modeling methods. We apply the definition of [15] regarding modeling methods as follows. Modeling methods are composed of two major components: (1) a modeling technique, divided into a modeling language and a modeling procedure, and (2) mechanisms and algorithms working on the described models in conformity with the modeling language.

The presented method can be seen as a ‘method plug-in’, enriching enterprise modeling frameworks to allow for compliance evaluation based on heat maps. Like a modeling method itself, the method plug-in consists of the main building blocks: modeling language, modeling technique, and mechanisms and algorithms, all of which are seamlessly integrated into a (possibly) established enterprise modeling method. In the following, the method is structured into a static component - the modeling language - and a dynamic component - the modeling technique and the required mechanisms/algorithms.

#### 3.1 Static Component

The language for representing the relevant architecture artifacts, their interdependencies, and their relations to principles is described by its meta-model, i.e. ‘the meta-model is a model of its corresponding modeling language’ [15]. The method (plug-in) can be applied to any existing meta-model in the fields of enterprise modeling. The only requirement is to extend the meta-model with the following meta-model extensions (if not already part of the given method).

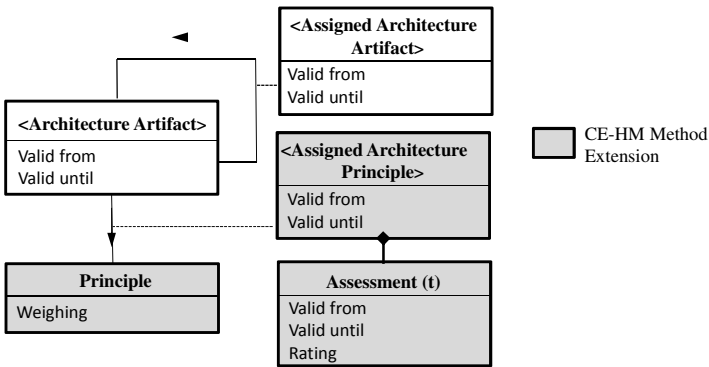


Fig. 1. Minimum Requirements on the Meta-Model

Besides the given modeling classes (which are part of the underlying modeling method we apply) for instantiating architecture artifacts (such as business processes, applications, and technology components) a modeling class representing principles is required. With *principles* we refer to control objectives, architecture principles or any

other concept constituting regulation codes, claiming compliance with regulations, laws or standards.

Principles are assigned to architecture artifacts via the relation type *Assigned Principle*. All modeling and relation classes have a time context, meaning that instances of these (relation) classes are valid only within a time frame (namely between *valid from* and *valid until*). Furthermore, principles are weighted regarding their importance. The rating-per-time interval of an architecture artifact in the context of a principle is defined via the construct *Assessment* of the relation class *Assigned Principle*. This allows for rating the compliance of an architecture artifact (considering the assigned principles) per evaluation cycle.

Fig. 2 exemplarily depicts an extract of TOGAF’s core content meta-model [27] enriched with the CE-HM methods meta-model extension.

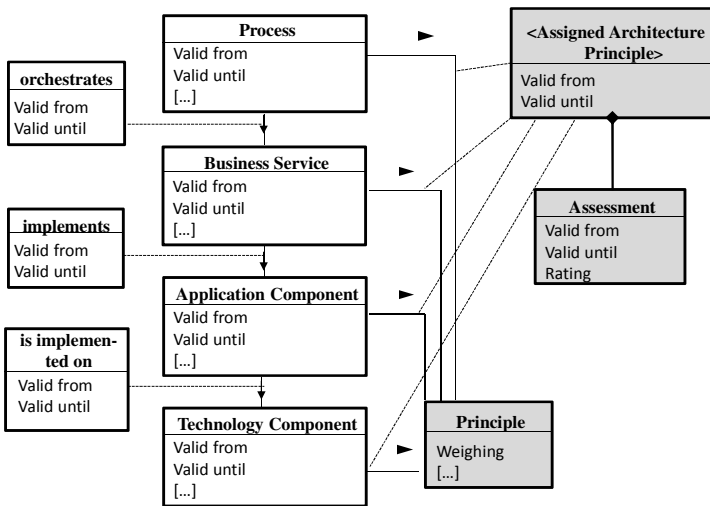


Fig. 2. Excerpt of the TOGAF9 Meta-Model

### 3.2 Dynamic Component

The (modeling) procedure – as one part of the dynamical component of the CE-HM method - defines the compliance evaluation processes, composed of the tasks *Define Scope and Configure Assessment*, *Conduct Principle Definition and Weighing*, *Conduct Data Acquisition*, *Assess Architecture Artifacts*, and *Conduct Analysis*.

Ideally, these tasks are integrated into the management processes of the organization. Responsibilities and stakeholders will typically differ from organization to organization. According to [12], ‘*the body responsible for governance will also normally be responsible for approving the architecture principles, and for resolving architecture issues*’. This body is here referred to ‘Compliance Team’. The ‘Architecture Artifact Owner’ (e.g. application or process owners) are responsible for maintaining their architecture artifacts within an architecture repository. For a more detailed discussion on roles and stakeholders that are typically involved, we refer to [28].

**Phase 1 - Define Scope/Configure Assessment.** In this phase, the scope of the Compliance Evaluation effort is defined and the principles to comply with are determined. The compliance team may choose CobiT's control objectives [11] as the prime regulation code to comply with. In a next step the meta-models of the established enterprise modeling approaches are analyzed in respect of their relevance for the Compliance Management endeavor. If relevant architecture artifacts (and underlying modeling classes) are missing, existing meta-models and approaches need to be adapted.

Subsequently, *evaluation configurations*, representing the part of the meta-model (the perspectives) stakeholders are interested in, are defined. Each of these evaluation configurations conform to a directed acyclic path  $EC = (M, R)$ , comprising a set  $M$  of modeling classes (consisting of architecture artifacts and principles) and a set  $R$  of relations of the underlying meta-model. Principles equate to the end nodes of this directed path. In other words, the modeling classes of the underlying meta-model are put into hierarchy. Each modeling class (architecture artifact type) represents one level in the hierarchy and the modeling class *principle* represents the bottom level. Referring to our running example (see Fig. 2) the modeling class *business service* might be on meta-model level  $n$  and the subordinated modeling class *application component* is on meta-model level  $n-1$ .

**Phase 2 - Conduct Principle Definition and Weighing.** The second phase serves a twofold purpose: First, relevant principles are defined and integrated into the model. Consider for example CobiT's control objectives for Change Management, such as *AI6.1 Change Standards and Procedures*, *AI6.2 Impact Assessment, Prioritization, and Authorization* and *AI6.3 Emergency Changes*. Together these control objectives might form the principle *Apply Change Management* which is assigned to architecture artifacts of the type *application component* or *technology component*.

Second, the defined and agreed principles are weighted according to their importance for the organization. Weighing can be done e.g. by pairwise comparison of principles. Other more sophisticated methods are the Delphi method [10] or the Analytic Hierarchy Process (AHP) [22] – both widely accepted approaches for group decision-making. In the presented method each principle is weighted from 1 (minor importance) to 5 (high importance). As there is no limit on the relevant number of principles, indifference of assigned weights is allowed. This leads to the following definition:

$$\omega(p) \in \{1, 2, 3, 4, 5\},$$

with  $\omega(p)$  being the function which returns the weight which has been assigned to a principle.

**Phase 3 - Conduct Data Acquisition.** In this phase the model, i.e. the architecture artifacts and their relations, is defined. Ideally data acquisition and maintenance is done on a regular basis and at least once for each compliance evaluation cycle. Process patterns on how to organize the data acquisition and maintenance can be found in [20]. Referring to these patterns we advocate a decentralized approach, where architecture artifact owners are responsible for data currency and completeness,

and the compliance team conducts periodic ‘verification and audit’ on the enterprise model. However, as we relate to institutionalized enterprise modeling initiatives of an organization, this certainly depends on the given enterprise modeling approaches.

**Phase 4 – Assess Architecture Artifacts.** In phase 4, the principles are assigned to architecture artifacts and each architecture artifact is assessed respectively in the context of the related principle. Again, we use a scale from 1 to 5, where 1 stands for being *non-compliant* and 5 for being *fully compliant* to the particular principle. This leads to the following definition:

$$\rho(a, p) \in \{0,1,2,3,4,5\},$$

with  $\rho()$  being the function which returns the rating of an architecture artifact ‘a’ in the context of a principle ‘p’. The rating value null (0) implies that the relation between a and p has not been evaluated so far. There is no standardized way for defining the compliance levels. However, to ensure enterprise-wide consistency in the ratings it is recommended to (1) have a small and concerted team of reviewers (members of the compliance team), and (2) provide detailed instructions for assessing a specific principle to ensure traceability and organizational-wide reproducibility of conducted compliance assessments.

Take the above example of the principle *Apply Change Management* which might be related to architecture artifacts of the modeling class *application component* in respect of a particular evaluation configuration defined in phase 1. Changes to an application component (where the principle has been assigned to) need to be managed in conformity with this principle. Thus, these application components are assessed against the principle, respectively against the control objectives the principle consists of. The given assessment value indicates, whether changes to a concrete application component are always made in compliance with the principle. Hence, it is evaluated whether, upfront to changes to the application component, impact assessment and prioritization is conducted, and whether the changes are authorized in accordance with the organization’s stipulations. The control objectives the principle consists of can be understood as the detailed instructions necessary for ensuring traceability and reproducibility of the assessments.

**Phase 5 – Conduct Analysis.** Subsequently, in phase 5, the compliance evaluation of the organization is performed, delivering the compliance level (CL) of architecture artifacts on any level of the evaluation configuration. Compliance levels of bottom-level architecture artifacts are propagated to their superior architecture artifacts (in accordance with the evaluation configuration, defined in phase 1). Take the above example: The compliance level related to the principle *Apply Change Management* is assessed for a set of application components. As those application components are assigned to a higher-level business service, the fulfillment level of the very principle on business service level is determined by propagating the compliance levels to the higher-level business services. With the same approach compliance level of business processes can be determined (by propagation of the compliance levels of their subordinated business services). Hence, it can be stated, that a certain business process uses

business services, and the underlying application components of this business service comply with the principle *Apply Change Management* to a certain degree.

Based on the calculated compliance levels, views are generated and enriched with color coding (heat maps) to support decision-making related to compliance issues. The heat map approach can be applied to any type of viewpoint and to any architecture artifact (which is part of the evaluation configuration). We distinguish the following heat map types:

- **Status heat maps** represent the compliance status of architecture artifacts to identify weak spots and non-conformant parts of the organization at any point in time.
- **Alteration heat maps** illustrate significant improvement and worsening of compliance of architecture artifacts between two points in time.
- **Objective deviation heat maps** illustrate significant divergence between current and target compliance state of architecture artifacts. In many cases achieving compliance will be dependent on limited budgets and resources. Thus, organizations will define target values which will rise over time until reaching the required level. In exceptional cases the organization may even decide to accept non-compliance. The compliance team will be highly interested in seeing progress concerning the planned compliance level. Note: For this evaluation type, in phase 4, target values ( $\rho_{\text{target}}(a, p, t)$ ) for compliance values need to be defined for each pair of architecture artifact and related principle. The meta-model must be extended accordingly.

Based on these heat maps the compliance team studies the enterprise in order to identify weak spots. Analysis takes place on any level of the enterprise, respectively on any meta-model level of the determined evaluation configurations. The compliance team uses drill-down functionality to clearly identify the weak spots and to set adequate measures.

Each of the three mentioned heat map types is based on mechanisms and algorithms for compliance evaluation and visualization. Each of these *compliance evaluation mechanisms* in turn builds on *basic mechanisms* which are primarily needed for putting the architecture artifacts and the principles under control. [19] discusses some of those basic mechanisms with focus on EA modeling, meant to be more generally applicable. Amongst others, mechanisms to organize version control of architecture artifacts, to establish and to maintain time-related views, and mechanisms for status management (e.g. release workflows) are defined. These mechanisms fully apply to the method at hand.

Mechanisms supporting time-related views are paramount to compliance evaluation, as compliance evaluations are performed repeatedly; this basic mechanism is briefly discussed, before we focus on the *compliance evaluation mechanisms*. Time-related views provide the capability for describing different states of architecture artifacts and their relations at different times. Hence current as well as former versions of the entire model (incl. architecture artifacts, principles and performed ratings) are traceable at any point in time. While an application component might be valid at one point in time (status ‘productive’), at a later date it might become invalid (status ‘retired’). This is not only true for architecture artifacts and principles, but also for relations between those architecture artifacts, as well as for their relations to principles.

[23] differentiates between two types of time versions: logical time vs. physical time. Time values used in the CE-HM method are considered as logical time – the time at which changes to architecture artifacts take place in the real world and the time a compliance rating refers to. The time at which the changes took place in the repository – physical time – will not be discussed in detail, although it is certainly relevant to keep compliance evaluations traceable. The meta-model (extension) discussed in section 3.1 considers and incorporates time-related views. Modeling classes and relation classes include the attributes ‘valid from’ and ‘valid until’ representing the time-frame during which an architecture artifact (or relation) exists (e.g. state ‘productive’). The point of time at which the enterprise model is viewed, affects the visibility of architecture artifacts and their relations. It is obvious that the change of architecture artifacts comes along with the necessity for new appraisal of its conformity with the related principles. Hence, the assessment value needs to be time-related as well.

**Evaluation Algorithms.** In this section domain-specific evaluation algorithms for Compliance Evaluation are formalized. On modeling level, architecture artifacts are put into hierarchy in conformity with the above evaluation configurations on meta-model level, forming a directed acyclic graph with principles being the leaves (see phase 1).

For calculating the compliance levels for the status heat maps two alternative calculation methods are presented, namely the *Worst Case Rating* and the *Weighted Average Rating*.

**Alternative 1 - Worst Case Rating:** The values of worst rated principle are propagated to an architecture artifact (and from there to the higher-level architecture artifact).

The compliance level of an architecture artifact, which is directly related to one or more principles (architecture artifact on the bottom level of an evaluation configuration), is defined as follows:

$$CL_{min}(a, t) := \begin{cases} \min_{p \in \mathbb{P}} \rho(a, p, t), & \text{if } \mathbb{P} \neq \emptyset \text{ and } \rho(a, p, t) \neq 0 \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

where  $\mathbb{P}$  is the set of chosen principles relevant for the compliance evaluation run, and with  $\rho()$  being the function which returns the assessment value of a particular architecture artifact in the context of a particular architecture principle for the time (t) the compliance level evaluation is conducted. The compliance level for propagating the CL to the higher-level architecture artifacts is calculated as follows:

$$CL_{min}(a^n, t) := \begin{cases} \min_{a^{n-1} \in \{sub(a^n)\}} CL_{min}(a^{n-1}, t), & \text{if } sub(a^n) \neq \emptyset \text{ and } CL_{min}(a^{n-1}, t) \neq 0 \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

with  $a^n$  representing an architecture artifact on level n and with  $sub()$  being the function which returns all subordinated architecture artifacts of a particular architecture artifact.

**Alternative 2 - Weighted Average Rating:** The ratings on each level of the evaluation method are averaged and propagated to the particular higher-level architecture artifact.

The compliance level of an architecture artifact, which is directly related to a principle (architecture artifact on the bottom level of an evaluation configuration), is defined as follows:

$$CL_{avg}(a, t) := \begin{cases} \frac{\sum_{p \in \mathbb{P}} \rho(a, p, t) \times \omega(p)}{\sum_{p \in \mathbb{P}} \omega(p)}, & \text{if } \mathbb{P} \neq \emptyset \text{ and } \rho(a, p, t) \neq 0 \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

with  $\omega()$  being the function which returns the weight assigned to the principle, and  $\mathbb{P}$  and  $\rho()$  defined as above. The formula for propagating the CL to higher-level architecture artifacts in case of weighted average rating is defined as follows:

$$CL_{avg}(a^n, t) := \begin{cases} \frac{\sum_{a^{n-1} \in \{\text{sub}(a^n)\}} CL_{avg}(a^{n-1}, t)}{\sigma(a^n, t)}, & \text{if } \sigma(a^n, t) \neq 0 \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

where  $\sigma(a^n, t) := \#\{a^{n-1} \in \text{sub}(a^n) \mid CL_{avg}(a^{n-1}, t) \neq 0\}$  with  $a^n$  representing an architecture artifact on level  $n$  of a particular evaluation configuration.

Based on the calculations for status heat maps, evaluation values for *alteration heat maps* and *objective deviation heat maps* (introduced in phase 5) are calculated.

**Alteration heat maps** are calculated as follows:

$$CL_{\Delta \text{Alteration}}(a, t_1, t_2) := CL(a, t_1) - CL(a, t_2) \quad (5)$$

For calculating the evaluation values for **objective deviation heat maps** compliance levels of the architecture artifacts at some point in time need to be compared with the actual planned target values. The following function applies:

$$CL_{\Delta \text{DEVIATION}}(a, t) := CL_{plan}(a, t) - CL_{as-is}(a, t) \quad (6)$$

where  $CL_{as-is}(a, t)$  represents the actual compliance level at some point in time and  $CL_{plan}(a, t)$  represents the planned compliance level at some point in time, and the calculation of  $CL_{plan}(a, t)$  is done analogous to  $CL_{as-is}(a, t)$ .

**Algorithms for Color Coding.** In order to visualize the calculated compliance levels (status, alteration and deviation) the given shapes of the architecture artifacts (defined in the modeling language) are colored using specific color progressions. We use bipolar color progressions, which are, according to [21], typically used for mapping temperatures. A color progression from dark red (indicating non-compliant architecture artifacts) to dark green (indicating fully compliant architecture artifacts) with yellow in the center is used. Following the recommendation of [21] five hue categories are used, as the authors state that the human eye can easily distinguish at maximum seven hues.



The algorithms for applying the colors considers this recommendation by putting evaluation results into intervals, with each interval assigned to a color of the color progression from red to green. In the following status heat maps including color codes evaluated via weighted average method are defined exemplarily. In this case the possible range for compliance level values is defined from 1 to 5. With the span width of  $5 - 1 = 4$ , and with five planned intervals the required interval size is  $4/5 (= 0.8)$ . The following function specifies the color of the graphical representation of the architecture artifacts at a given point in time:

$$\text{colour}(a, t) := \begin{cases} \text{red, if } 1 \leq \text{CL}(a, t) < 1.8 \\ \text{orange, if } 1.8 \leq \text{CL}(a, t) < 2.6 \\ \text{yellow, if } 2.6 \leq \text{CL}(a, t) < 3.4 \\ \text{lightgreen, if } 3.4 \leq \text{CL}(a, t) < 4.2 \\ \text{green, if } 4.2 \leq \text{CL}(a, t) \leq 5 \\ \text{grey, if } \text{CL}(a, t) = 0 \end{cases} \quad (7)$$

**View Generation.** For illustration purposes, we apply the heat maps to any viewpoint available in the method at hand. For automatic generation of views layout algorithms (see specific mechanisms and algorithms) are used. In conformance to the running example at hand heat maps are assigned to the three main viewpoints defined in TOGAF: (1) diagrams (supporting presentation of architecture artifacts and relations in a visual way suitable for stakeholder communication), (2) matrices (supporting presentation of relationships between architecture artifacts), and (3) catalogues (lists of architecture artifacts). An illustrative example is given in **Fig. 3**. However, the discussed mechanisms work for any other viewpoint like those defined in [3].

## 4 A CE-HM Prototype Based on ADOxx

This section presents the prototypical implementation of the CE-HM method based on the meta-modeling platform ADOxx. Based on its meta-modeling capabilities, ADOxx allows for implementing any enterprise modeling framework. It incorporates collaboration features for data acquisition and on-going data maintenance, a central repository, and mechanisms for controlling relevant architecture artifacts. By utilizing the platform's open APIs the discussed evaluation mechanisms, and heat maps for compliance evaluation have been prototypically implemented.

Fig. 3 depicts a screenshot of the tool. The user interface is subdivided into three areas: (1) The modeling surface presenting graphical viewpoints, such as diagrams and matrices (diagram and matrix viewpoints), (2) the object catalogue (catalogue viewpoints), and the panels for choosing (3a) evaluation configurations (including relevant principles) as well as (3b) a time-slider for choosing the relevant point in time. On the modeling surface an automatically generated cluster map – as one example of a viewpoint in ADOxx – is shown. The visualized clusters (rectangles) are colored automatically in accordance with the underlying evaluation mechanism.

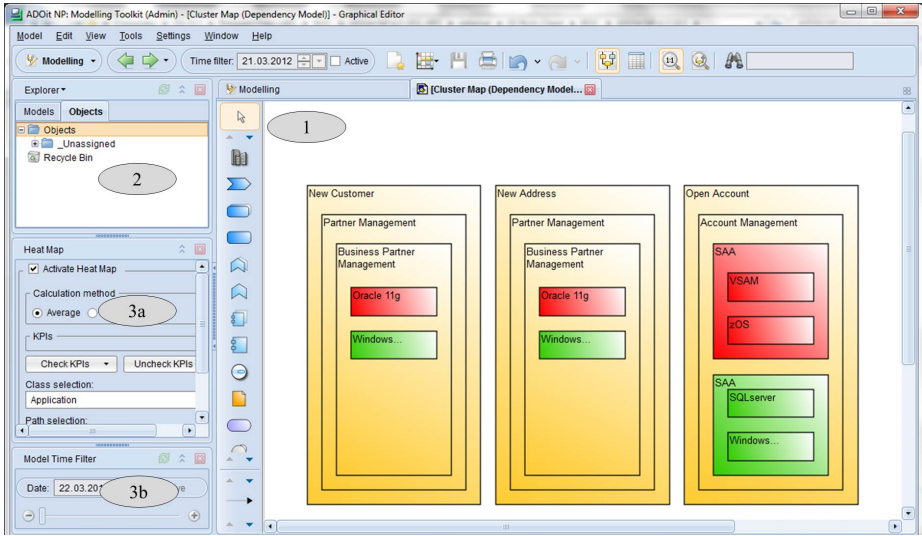


Fig. 3. Reference Implementation based on ADOxx

## 5 Conclusion

The CE-HM method and its prototypical implementation based on a meta-modeling platform are presented. The method supports compliance evaluation in an effective and efficient manner, by building on established enterprise modeling initiatives of an organization. For illustration of the compliance degree of architecture artifacts a sophisticated heat mapping approach is applied. The presented mechanisms evolved step by step during compliance evaluation and enterprise modeling projects.

The suitability of the presented method certainly needs to be proven in a structured manner. We are working on the evaluation of the CE-HM method in an application domain, namely the banking area. The method will be used for measuring and planning compliance of enterprise architectures based on self-imposed architecture principles comparable with those of TOGAF [27]. First evaluation results show the importance of a consistently maintained architecture repository. An approach planned to be evaluated in the next step is the integration of further information bases, e.g. of the organization's configuration management system (CMS). Maintenance procedures of the entire architecture repository and the Compliance Management need to be aligned. This might be followed by resistance of the different stakeholders, not willing to provide transparency in their domains. It needs to be evaluated, whether the presented scoring approach and the propagation rules prove reasonable in terms of (1) expressiveness of the gained heat maps, (2) efforts related, e.g. for weighing principles and grading compliance degrees, and (3) most importantly, acceptance within the organization. Our evaluation approach will define and evaluate key performance indicators (KPIs), e.g. the 'reduction in time/effort for performing compliance evaluation', and indicators evaluating the reduction in workload regarding compliance

evaluation efforts. Soft indicators, such as improved communication means by reusing viewpoints familiar to the stakeholders, as they are part of existing enterprise modeling initiatives, will be collected and evaluated via structured interviews.

**Acknowledgements.** Many thanks to the reviewers for providing us with constructive criticism, all of which have been carefully considered.

## References

1. van Bommel, P., et al.: Architecture Principles - A Regulative Perspective on Enterprise Architecture, vol. P-119 GI, pp. 47–60 (2007)
2. Buckl, S., et al.: A Pattern based Approach for constructing Enterprise Architecture Management Information Models, Universitaetsverlag Karlsruhe, pp. 145–162 (2007)
3. Buckl, S., et al.: Enterprise Architecture Management Pattern Catalog. Release 1.0, Garching b. München, Germany (2008), <http://srvmatthes8.informatik.tu-muenchen.de:8083/file/EAMPatternCatalogV1.0.pdf> (access: October 15, 2008)
4. Campbell, P.L.: An Introduction to Information Control Models. Sandia National Laboratories report SAND2002-0131, Albuquerque, New Mexico (2003)
5. US Department of the Treasury - Chief Information Officer Council: Treasury Enterprise Architecture Framework (TEAF). Version 1 (2000)
6. FEA Working Group: E-Gov Enterprise Architecture Guidance (Common Reference Model) Version 2.0 (2002)
7. Fischer, C., et al.: What Is an Enterprise Architecture Design Principle? - Towards a Consolidated Definition. In: Proceedings of the 2nd International Workshop on Enterprise Architecture Challenges and Responses, Yonezawa, Japan (2010)
8. Frank, U.: The MEMO Meta Modelling Language (MML) and Language Architecture. In: ICB Research Report 24, Institute for Computer Science and Business Information Systems (ICB), Germany (2008), [http://www.icb.unidue.de/fileadmin/ICB/research/research\\_reports/ICBReport24.pdf](http://www.icb.unidue.de/fileadmin/ICB/research/research_reports/ICBReport24.pdf) (access: November 12, 2010)
9. Ghanavati, S., et al.: Comparative Analysis between Document-based and Model-based Compliance Management Approaches. In: RELAW 2008, Barcelona, Catalunya, pp. 35–39 (2008)
10. Helmer, I., Rescher, N.: On the Epistemology of the Inexact Sciences. *Management Science* 6(1) (1959)
11. IT Governance Institute: COBIT. Version 4.1, IT Governance Institute, <http://www.isaca.org/downloads> (access: December 31, 2010)
12. IT Governance Institute: Mapping of Togaf 8.1 with CobiT 4.0, <http://www.isaca.org/Knowledge-Center/Research/ResearchDeliverables/Pages/COBIT-Mapping-Mapping-of-TOGAF-8-1-With-COBIT-4-0.aspx> (access: August 15, 2011)
13. Karagiannis, D., et al.: Open Models Initiative Feasibility Study, [http://cms.dke.univie.ac.at/uploads/media/Open\\_Models\\_Feasibility\\_Study\\_SEPT\\_2008.pdf](http://cms.dke.univie.ac.at/uploads/media/Open_Models_Feasibility_Study_SEPT_2008.pdf) (access: January 23, 2011)
14. Keller, W.: Using Capabilities in Enterprise Architecture Management, Version of 2009-12-18, <http://www.objectarchitects.biz/ResourcesDontDelete/CapabilityBasedEAMWhitepaper.pdf> (access: April 20, 2010)

15. Karagiannis, D., Kühn, H.: Metamodelling Platforms. In: Bauknecht, K., Tjoa, A.M., Quirchmayr, G. (eds.) EC-Web 2002. LNCS, vol. 2455, p. 182. Springer, Heidelberg (2002)
16. Lankhorst, M.: Enterprise Architecture at Work: Modelling, Communication and Analysis. Springer, Berlin (2005)
17. Microsoft Services: Microsoft Motion Heat Mapping Tool, <http://blogs.microsoft.co.il/files/folders/2034/download.aspx> (access: December 23, 2010)
18. McCarthy, W.E.: An entity-relationship view of accounting models. *The Accounting Review* 54(4), 667–686 (1979)
19. Moser, C., et al.: Business Objectives Compliance Framework. Mechanisms for Controlling Enterprise Artifacts, vol. 127, pp. 73–88. GI (2008)
20. Moser, C., et al.: Some Process Patterns for Enterprise Architecture Management, vol. 150, pp. 19–30. GI (2009)
21. Robinson, et al.: Elements of Cartography, 6th edn. John Wiley & Sons, New York (1995)
22. Saaty, T.L.: Decision Making for Leaders – The Analytic Hierarchy Process for Decisions in a Complex World, 3rd edn. RWS Publishing, Pittsburgh (2001)
23. Sciore, E.: Versioning and Configuration Management in an Object-Oriented Data Model. *VLDB Journal* (3), 77–106 (1994)
24. Silveira, P., et al.: On the design of compliance governance dashboards for effective compliance and audit management. In: Proc. of the 3rd Workshop on Non-Functional Properties and SLA Management in SOC, NFPSLAM-SOC 2009, pp. 208–217 (2009)
25. Shneiderman, B.: The eyes have it: A task by data type taxonomy for information visualizations. In: Proc. IEEE Symposium on Visual Languages, Boulder, Colorado, pp. 336–343 (1996)
26. Strecker, S.: Toward modeling constructs for audit risk assessment: Reflections on internal controls modeling, vol. 171, pp. 131–148. GI (2010)
27. TOGAF: The Open Group Architecture Framework, Enterprise Edition. Version 8.1.1, <http://www.opengroup.org/architecture/togaf8-doc/arch/> (access: August 15, 2007)
28. Vicente, P., Mira da Silva, M.: A Business Viewpoint for Integrated IT Governance, Risk and Compliance. In: Services, 2011 IEEE World Congress on Services, Washington, DC, USA, pp. 422–428 (2011)
29. El Kharbili, M., et al.: Towards a Framework for Semantic Business Process Compliance Management. In: The Impact of Governance, Risk, and Compliance on Information Systems (GRCIS), Montpellier, France. CEUR Workshop Proceedings, vol. 339, pp. 1–15 (2008)
30. Popova, V., Sharpanskykh, A.: Formal Modelling of Organisational Goals Based on Performance Indicators. *Data & Knowledge Engineering* 70(4), 335–364 (2011)

# A Compliance Management Ontology: Developing Shared Understanding through Models

Norris Syed Abdullah<sup>1</sup>, Shazia Sadiq<sup>1</sup>, and Marta Indulska<sup>2</sup>

<sup>1</sup> School of ITEE, The University of Queensland, Brisbane, Australia  
{norris, shazia}@itee.uq.edu.au

<sup>2</sup> UQ Business School, The University of Queensland, Brisbane, Australia  
m.indulska@business.uq.edu.au

**Abstract.** Managing regulatory compliance is increasingly challenging and costly for organizations world-wide. Due to the diversity of stakeholders in compliance management initiatives, any effort towards providing compliance management solutions demands a common understanding of compliance management concepts and practice. This paper reports on research undertaken to develop an ontology to create a shared conceptualization of the compliance management domain, namely CoMOn (Compliance Management Ontology). The ontology concepts are extracted from interviews and surveys of compliance management experts and practitioners, and refined through synthesis with leading academic literature related to compliance management. A semiotic framework was utilized to conduct a rigorous evaluation of CoMOn through a series of eight case studies spanning a number of industry sectors. The consensus achieved through the evaluation has positioned CoMOn as a comprehensive domain ontology for Compliance Management.

**Keywords:** domain ontology, compliance management, compliance vocabulary, semiotics.

## 1 Introduction

Compliance refers to ensuring that business processes, operations, and practice are in accordance with a prescribed and/or agreed set of norms [1]. Compliance requirements are associated with regulations that may be introduced either externally to an organization or internally by the organization itself. They may stem from legislature and regulatory bodies (e.g. Sarbanes-Oxley, Basel II, HIPAA), standards and codes of practice (e.g. SCOR, ISO9000) and business partner contracts. Accordingly, compliance management is referred to as the coordinated set of activities designed to assure that all elements of the business (processes, employees, partners, and assets) strictly follow any established regulatory requirements.

Regulatory compliance has attracted much concern by organizations across the globe over the last decade. Introduction of regulations such as the Sarbanes-Oxley Act of 2002 (SOX), Health Insurance Portability and Accountability Act of 1996

(HIPAA), and Anti-Money Laundering and Counter-Terrorism Financing Act 2006 has made regulatory compliance a focal point of many organizations. Even though compliance activities are predominantly viewed as a burden by organizations [2], failing to comply is no longer an option [3]. Breaches of compliance may result in serious, and sometimes even disastrous, situations for the organizations concerned. For example, several high profile, corporate scandals - Enron, WorldCom (USA), Parmalat (Italy), HIH (Australia), and Tyco International (France) - were associated with significant market and reputational damage.

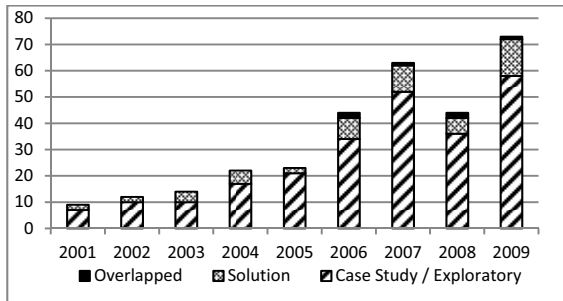
Compliance management spans across many, if not all, industry sectors and applications, such as financial services, information security, environment, healthcare services, and manufacturing [4]. Managing compliance is highly challenging, given that compliance engages interests from a wide variety of stakeholders, such as compliance professionals, auditors, customers (clients), business/contract partners, suppliers, and regulatory/authoritative bodies. The variety of stakeholders, together with the variety of sources of compliance requirements and their frequently changing nature, leads to increasing challenges for compliance management, in part due to the lack of a common vocabulary and shared understanding of related concepts and artifacts for use in compliance management initiatives. In particular, difficulties in establishing the relationship of compliance and risk functions of an organization with its various (line of) business functions has been highlighted by the compliance expert and practitioner community [5]. We posit that the vast body of knowledge that exists within conceptual modeling research, and its established role in facilitating shared understanding, can contribute significantly in this regard. For example, Gruber [6] defines ontology, in the context of computer and information sciences, as a set of representational primitives with which one can model a domain of knowledge or discourse. The primitives are typically classes (or sets), attributes (or properties), and relationships (or relations among class members) [6]. Gruniger and Lee [7] argue that the use of ontology benefits an organization in three ways. First, it serves as a communication medium between computational systems and humans. Second, it is useful as a computational reference. Third, it facilitates the reuse of knowledge for structuring or organizing libraries or repositories of plans. Not surprisingly, ontology has been widely used to represent many real world cases [8, 9].

Accordingly, we embarked upon the development of a shared conceptualization of compliance management landscape within organizations in the form of a Compliance Management Ontology – named CoMON. CoMON is derived from interviews and surveys of compliance management experts and practitioners, and further synthesized with industry and scholarly articles. It is developed and evaluated following the ENTERPRISE methodology [10, 11], and can be represented formally (e.g. through web ontology language - OWL) or informally (see section 4.4). CoMON has also been subject to a rigorous validation process based on a well established semiotic framework [12]. The validation is conducted through a series of eight case studies that span several industry sectors in both public and private arenas. This paper presents the results of the study that led to the development of CoMON, its subsequent evaluation and final refinement.

The remainder of the paper unfolds as follows. In section 2, we introduce the related work associated with compliance management and ontology development. Following this discussion, section 3 presents the methodology employed for CoMOn development, including a detailed report on the evaluation phase. Section 4 presents the actual ontology building development process, Section 5 details of the (post-evaluation and refinement) CoMOn concepts and levels. Finally, a summary of the methodological and empirical lessons learnt from this work, together with future research, is presented in section 6.

## 2 Related Work

Over the past decade, compliance management has received relatively scarce yet increasing attention from the academic Information Systems community [4]. A recent review of compliance management research [4] provides a comprehensive snapshot of articles that address relevant compliance management topics categorized as exploratory or case studies, solution papers, and papers that have elements of both. Figure 1 presents the yearly distribution of articles relevant to compliance management that are published in Information Systems (IS) publication outlets. The study includes 374 papers (identified as relevant from 22227 papers) from premium Information Systems journals (as promoted by the Association for Information Systems), reputed Information Systems conferences and some additional popular journals in the discipline.



**Fig. 1.** Distribution of Compliance Management related Articles per Type per Year

Among these articles, several offer compliance management solutions. To name a few, Banker et al.[13], for example, introduce a model which assists in identifying intensity of terms required in contractual agreement. Kim et al. [14] describe the development and application of an evaluative data model for ISO 9000 compliance, while Weitzner et al. [15] propose a technical architecture required to support information systems accountability. Several contributions also offer frameworks. For example, in the context of HIPAA, Davis and Hikmet [16] introduce a framework that facilitates identifying and analyzing the training needs of organizations. Meanwhile, Mishra and Weistroffer [17] offer a framework that can be applied as a practical guide by systems development managers in planning early compliance needs.

Despite the apparent increase in number of published papers, there are gaps between these research contributions and industry needs [5]. One area particularly highlighted by industry experts is the absence of comprehensive shared vocabulary on compliance, and conceptualization of compliance management requirements in general [5]. Without an agreement on required compliance management concepts and vocabulary, not only is the uptake of leading research difficult, but so is the communication of compliance requirements within an organization. The latter in particular has the potential to result in severe penalties for organizations and the responsible personnel.

Addressing the above need is the focus of our research and the following sections present the various aspects of the development of a comprehensive domain ontology for compliance management.

### **3 Research Methodology**

Our study is governed by the over-arching Design Science paradigm [18] given our focus on the development and evaluation of an artifact – an ontology. Ontology development is a difficult and time-consuming process [6] and, thus, requires a structured approach. A review of methods and techniques for ontology development indicates that several methodologies are available [7, 10, 11, 19-22].

#### **3.1 Ontology Development Approach**

While comparisons of methodologies and proposals of new methodologies do exist [23], our experience with ontology development indicates that there is limited guidance within existing ontology development methodologies on how to identify, gather, and use input in ontology development. One of the few works on this topic is that of Velardi et al. [24], who describe a text mining technique that aids an ontology engineer in identifying the important concepts in a domain ontology. Similarly, Brusa et al. [8] discuss their experience in developing a government budgetary ontology based on inputs from the provincial budgetary application, its related documentations, and a group of experts within an organization. While both works utilize inputs from a particular domain, they do not discuss how the relevant inputs were identified and prepared prior to the concept capturing process, and how the concepts were coded. In contrast, these aspects are considered in much detail and rigor in the ontology development approach adopted for CoMON. Further, considering that industry relevance is an important factor that contributes to the usability and acceptability of a particular ontology, we address this need directly through our methodology.

Given that there are no related existing ontologies that can be applied for compliance management, the development of CoMON is, by necessity, from the ground up. Accordingly, we adopt the ENTERPRISE [10, 11] methodology to guide the development. The choice of ENTERPRISE was largely based on its wide spread



utilization [25, 26] as a credible approach. Secondly, ENTERPRISE provides a good compromise between development guidance and freedom of representation of the domain, thus providing clear direction on the development front, while allowing the most appropriate choice of formal representation [21].

### 3.2 Ontology Evaluation Approach

The ontology evaluation approach requires a special mention from a methodological perspective. All methodologies for building ontologies recognize the importance of evaluation [21], as does the Design Science approach inherent to our research [18]. The evaluation of ontology includes examining how the ontology fits the particular domain it is designed to serve. For this purpose, we select a case study approach with a targeted set of questions based on the semiotic framework [12], as described below.

For the conduct of the evaluation case studies, the research team consisted of three experienced empirical researchers, one with the role of the main interviewer and two with a support role of note taking, related document analysis, and further probing. The case studies utilized data gathering instruments that facilitated feedback from practitioners in both quantitative and qualitative forms. A semiotic framework [12] was utilized to evaluate the compliance management ontology *quality*. In [12], Burton-Jones *et al.* introduce four metrics to evaluate the quality of ontology; namely, Syntactic Quality, Semantic Quality, Pragmatic Quality and Social Quality. Syntactic Quality is measured through Lawfulness (correctness of syntax) and Richness (breadth of syntax used). The second metric, Semantic Quality is measured through Interpretability (meaningfulness of terms), Consistency (consistency of meaning of terms) and Clarity (average number of word senses). The third metric, Pragmatic Quality includes Comprehensiveness (number of classes and properties), Accuracy (accuracy of information), and Relevance (relevance of information for a task). Finally, the fourth metric Social Quality includes Authority (extent to which other ontologies rely on this ontology) and History (the number of times the ontology has been used). As our ontology is new, we exclude Social Quality in this study. We argue that an ontology can only be evaluated on its social quality after it has been in use for a period of time and thus leave this component for future work.

For the purpose of the evaluation, we develop an interview protocol aimed at capturing participant feedback on the quality of the ontology with respect to clarity (C), interpretability (I), comprehensiveness (M), accuracy (A), and relevance (R) of an individual concept in ontology (as derived from the semiotic framework [12]). Three remaining criteria *viz.* consistency, lawfulness, richness were excluded for this stage of the study. While these three criteria remain significant to our overall evaluation of the ontology, we leave these for our future work. We argue that evaluation on those criteria needs (1) familiarity with ontological terminology, for example, richness requires assessment of the types of terms (e.g. class, subclass, type, property, and relationships types); and (2) exemplification of concepts and their properties in order

to faithfully provide evaluation feedback, for example lawfulness requires consideration on the correctness of syntax which becomes more evident when reasoning through examples. More details on the planned future work are provided in section 6.

The interview protocol is structured through three supporting documents: The first document, Core Concept Evaluation that contains instructions to the participant and definition for each core concept. The participants specify their perception on the five criteria for quality evaluation, namely clarity (C), interpretability (I), comprehensiveness (M), and accuracy (A) for each of the concept on a 7-point Likert scale representing the level of their agreement for a particular criteria associated with a particular concept. Following this, relevance is captured by requiring participants to state whether a particular concept is relevant or not. The second document, Overall CoMON Diagram is a document that provides the participants with the structure of concepts that represent CoMON. This diagram includes Core concepts and its sub-concepts and also how these concepts are associated with each other. Finally, Catalogue of CoMON's Concepts provides participants with the definitions of all detailed concepts, which were used particularly to evaluate the Comprehensiveness criteria. In addition, a number of open ended questions were included at the end to gather deeper insights and to ensure that any missing concepts were probed and identified.

In the following, we discuss the four phases, based on ENTERPRISE methodology, that have been adopted for the development of CoMON.

## **4 Developing the Ontology**

### **4.1 Identify Purpose and Scope**

Identifying the purpose and scope of an ontology is critical to ensure a clear understanding of its intended use(s) and users [10]. In our study, the purpose of CoMON is to provide practitioners, as well as the research community, with a shared conceptualization of compliance management domain. The expected users of this ontology are compliance management professionals, businesses that have compliance obligations, regulators, and researchers. The intended use is as a reference in describing, communicating, and implementing compliance related tasks.

Development of CoMON is based on a synthesis of qualitative data from compliance management experts [5] and practitioners [27] (both being important in reflecting the real world aspects of compliance management) and leading research identified as relevant through a comprehensive literature study [4].

This collection of sources, with a stronger emphasis on qualitative data, provides a well-informed and industry-relevant ontology and reduces the risk of missing concepts. While the qualitative data stems from interviews with compliance management experts and surveys with compliance management professionals, the scholarly articles include a collection of highly cited compliance related articles published in workshops, conferences, and journals, as well as relevant industry sources such as Gartner Research, KPMG, and Open Compliance and Ethics Group (OCEG).

## 4.2 Ontology Building

ENTERPRISE [10, 11] recommends three stages in building an ontology *viz.* capturing, coding and integrating existing ontologies, as discussed below.

**Capture.** Ontology capture includes identifying key concepts and relationships in the domain, producing precise and unambiguous text definitions for such concepts and identifying terms to refer to such concepts [10]. Concept identification is a challenging task in the development of any ontology.

The data sources used for ontology development are qualitative in nature. Therefore, concept identification for CoMOn involved coding and analysis of all main sources of data, facilitated by a qualitative analysis tool (NVivo)<sup>1</sup>. NVivo was used to facilitate coding of the initial concepts and relationships that make up the ontology. The process started with an exhaustive analysis of corpus by the research team. Using a dual-coder approach, the researchers coded a fragment in the data sources when it represented a concept related to compliance management. The identification and selection of the fragment was based on whether the concept was directly mentioned in the fragment, or contained a phrase or statement which implied the concept. A node was created in NVivo to represent a group of fragments from the data sources that are relevant to a particular concept. The number of fragments supporting a particular node indicates how many times a particular concept was mentioned in the data sources. This process continued until all data were coded and resulted in 254 initial concept nodes.

After the initial capture of the 254 concept nodes, the concept identification was followed by a review process with the view to remove redundancy. Where synonyms were found, either one of the terms was selected due to wider usage, or a new, more accurate, term was defined to represent the concept. This process resulted in duplicate free and more generic concepts, reducing the number to 64 concepts. After a second round of coding and validation, the 64 concepts were hierarchically categorized based on 10 most prominent (or core) and generalized concepts. These provide us with 10 concepts as tier 1 and 54 concepts structured in tiers 2 through to 4. The set of 10 core concepts includes concepts of: **Business Process, Culture, Cost, Program, Requirements, Regulatee, Regulator, Risk Management, Service Provider, and Solutions.**

During the concept identification process, a simultaneous analysis on the data sources was also performed to identify any relationships that were indicated between the identified compliance management concepts. A node was also created in NVivo to represent a group of fragments from the data sources that provided evidence of a particular relationship between concepts.

The relationship nodes were then further classified by referring to the category of relationship i.e. specialization, aggregation and association [28]. Specialization and

---

<sup>1</sup> A qualitative data analysis software package that is used to code and analyse qualitative data gathered from surveys, interviews, observations, document analysis, or other text-based data. [www.qsrinternational.com](http://www.qsrinternational.com).

aggregation were used to structure the hierarchy of the identified concepts. Whereas association was used to depict non-hierarchical relationships e.g. dependency or impact. These three relationship types are referred to as type-of (specialization), part-of (aggregation), and assoc. (association) in the remainder of this paper.

**Coding.** Coding involves an explicit formal representation of the conceptualization captured in the earlier stage. Formal representation is required to restrict the possible misinterpretations of a particular concept. Furthermore, concepts are usually hierarchically organized through a structuring relation, such as is-a (class-superclass, instance-class) or part-of [21]. Typical representations that are available for ontology documentation are Web Ontology Language (OWL), KIF, Cyc, Ontolingua, and FLogic [29].

In our work, we employ the use of OWL (Web Ontology Language) [30] - a de facto standard for ontology representation on the web - to provide a formal representation of compliance management ontology (complete specification omitted due to space). The result is a formal representation of the concepts and their associated relationships and attributes in OWL. The OWL coding serves the multiple purposes of further disambiguation/checking, relationship structuring, version management and a foundation for future studies and tool support.

**Integration with Existing Ontologies.** Integration of ontologies is the process of reusing and synthesizing one or more relevant ontologies from different domains to develop a new ontology [31]. Ontology integration typically involves aggregating, combining, assembling together the source ontologies to form the resulting ontology, possibly after reused ontologies have suffered some changes, such as, extension, specialization, or adaptation [32]. At present, we found no existing ontology that is fit to be integrated with this ontology. Accordingly, at this stage we do not consider integration with existing ontologies due to this lack of relevance. This may change in future as relevant ontologies emerge.

### 4.3 Ontology Evaluation and Refinement

The methodological approach followed for evaluation of CoMOn is detailed in section 3.2. Following the aforementioned approach, eight case studies were conducted through interview sessions arranged with eight compliance management practitioners and professionals, and any additional documents being analyzed where relevant (e.g. policy documents). Participation was voluntary. We were motivated to ensure that participants include practitioners and professionals who are directly involved in compliance management practice in their organization and/or those who provide compliance management advisory services to clients. To that end, we enlisted the help of the Australasian Compliance Institute (ACI) and obtained a selection of experienced participants with insight into compliance management in organizations. The participants come from different industry domains such as legal, financial and

insurance services, gaming, transportation and public utilities. The size of organizations involved in the case studies includes two organizations with between 51-200 employees, one organization with between 201-500 employees and five organizations with 1001-5000 employees. Typical roles interviewed included: head of compliance, compliance managers, regulators and consultants. Each case study session started with an overview and description of CoMON and provided the participants with Core Concepts Evaluation document and explaining how to use the Core Concept Evaluation document. At the same time, the overall CoMON Diagram and Catalogue of CoMON Concepts was introduced to facilitate participant understanding of CoMON concepts.

**Table 1.** Mean scores for quality of core concepts

Concepts	C	I	A	M	R (Yes =1, No=0)
Business Process	6	6.4	6.1	5.8	1
Cost	5.5	5.8	4.6	4.5	1
Culture	5	5.6	5	5.5	1
Program	6	6.1	5.4	5.5	1
Regulator	5.3	6	4.8	5.6	1
Regulatee	6.3	6.5	5.9	6.1	1
Requirements	5.5	6.1	5.5	5.6	1
Risk Management	5.8	5.8	5.5	5.6	1
Solutions	5.6	5.9	5.6	5.9	1
Service Provider	5.5	5.6	5	5.5	1

In Table 1, we provide the results from the overall quality evaluation of core CoMON concepts, in the form of mean scores of the five quality criteria for each of the individual core concepts. These scores were used to identify the concepts that may need refinement. For example, low accuracy and low comprehensiveness scores for the Cost concept (A=4.6 and M=4.5) may suggest that the definition of Cost as a concept needs a review to improve its accuracy and comprehensiveness in representing cost related concepts in compliance management context. We note that agreement is reached in terms of the relevancy of all core CoMON concepts.

Overall, the feedback received throughout the evaluation case studies exhibited a high level of consistency, particularly in qualitative feedback. This feedback was utilized to drive the ontology refinement process and, hence, all changes made can be traced back to participants' feedback in most cases consensus. While quantitative data is based on scores on the quality of the concepts, qualitative data is based on interview recording and notes made during the interview sessions. Accordingly, we examine and assess each concept to decide whether it needs changes, will remain unchanged, or will be abandoned. The following paragraphs provide details of refinement process associated with CoMON's core concepts.

*Business Process.* Business Process concept received high scores generally, except for Comprehensiveness (5.8), as participants highlighted the need for monitoring

mechanisms for processes and measurement to assess process success or failure. Therefore, the **Business Process concept was expanded to Business Process Management** which includes Business Process, Process Monitoring, and Process Improvement as its sub concepts.

*Cost.* Lower scores for the Cost concept particularly on Accuracy (4.6) and Comprehensiveness (4.5), as well as the reasons highlighted by the participants indicated that the “cost concepts need to incorporate more than monetary value” and “to include resources as part of non-compliance cost e.g. due to extra time allocated to persuade people to comply”. Hence, the **Cost concept was renamed to Resources**, which include monetary and non-monetary costs (both are now the sub-concepts for Resources). The **definition of the Resources concept and the definition of its associated sub-concepts were also revised.**

*Culture.* Culture concept gained fair scores on all criteria indicating a need for improvement as highlighted by the participants e.g. “suggest to remove term ‘system’ from definition of culture”, “to include ‘strategic focus’ to improve the definition”, and “to include culture measurement...”. Therefore, the **Culture concept was expanded to Culture Management**, which includes Culture, Organizational Commitment, and Culture Measurement; **the definitions were revised** accordingly.

*Program.* In conjunction with only fair scores for Accuracy and Comprehensiveness, the participants highlighted the need to align Program concept to AS 3806-2006 Standard on Compliance Programs<sup>2</sup>. Following the feedback, **the definition of the Program concept was aligned to Compliance Programs definition** as per AS 3806-2006 Standard, and also **requisite changes were made at the lower level concepts** to align the overall Program sub-concepts to AS 3806-2006 Standard. Interestingly we note that the initial Program concept and its sub-concepts were actually quite close to the standard. In particular, only 6% (one concept) of the total Program concepts (16 concepts) were removed, 2 new concepts have been added, and another 19% (3 concepts) were renamed and/or rearranged.

*Requirements.* Lower scores for Clarity, Accuracy and Comprehensiveness compared to Interpretability hints the need to improve the definition of the concept and also the coverage of Requirements concept. This is also supported by participants’ feedback i.e., “obligations is a better term to be used to replace ‘requirements’ term”, “the concept must include mandatory and voluntary obligations” and “the concept must include code of practice as its sub-concept”, and “the lack of comprehensiveness due restricting the definition to only mandated requirements”. Thus, the **Requirements concept was renamed to Obligations**. The **definition for this concept was also**

---

<sup>2</sup> AS 3806-2006, published on 9 March 2006, provides principles for the development, implementation and maintenance of effective compliance programs within both public and private organizations.

**revised** to include **mandatory and voluntary compliance obligations** and **Code of Practice was added** as a sub-concept of Voluntary Obligations.

*Regulator.* Regulator gained mixed scores on all criteria which we believe are a result of incorrect usage of term in the definition that makes the concept restricted. This is highlighted by participants as “the definition is too narrow and not applicable to its sub-concepts particularly Standard Organization and Contracting Party.” and “need to cover ‘self-regulated’ industry, what kind of regulator suites them”. Hence, the **Regulator concept was moved to be an attribute of Obligations** (previously Requirements).

*Regulatee.* Although Regulatee concept received reasonable scores, participant feedback indicated lack of need for it: “This concept may not necessarily be required as this would mean the organization itself.” and “I have never heard of this term”. Considering the need for ontology parsimony, the **Regulatee concept was removed from CoMON.**

*Risk Management.* Risk Management concept received fair scores overall, suggesting that Risk Management concept needs improvement. Therefore, as suggested by all the participants the concept was aligned to the AS/NZS ISO 31000-2009 Standard on Risk Management.<sup>3</sup> Accordingly, **the definition for the Risk Management concept as well as some of the sub-concepts, was aligned to the Risk Management definition as per the standard.**

*Solutions.* In line with the scores, the participants highlighted that the concept can be improved by “to include explicit connection with Program concepts”, “the definition can be broadened to include other services”, and “to include audit in as a concept”. Therefore, **the definition of the Solutions concept was revised to include services. Some changes were also made to the lower level concepts** with inclusion of Services and its sub-concepts namely Audit, Assurance and Advisory.

*Service Provider.* Service Provider concept received fair scores only and the participants stressed that Service Provider concept can be improved by “expanding the coverage to include service providers that also help managing compliance obligation”, “including external party engagement to facilitate understanding compliance obligations” and “broadening the definition to include such as outsource internal audit function”. Considering Service Provider is closely tied with Solutions concept (service providers provide compliance related solutions), we found that it is more accurate to explicitly link Service Provider to Solutions concept. At the same time, this will help us to minimize the number of concepts in CoMON. This concept **became an attribute of Services** (under the Solutions concept).

---

<sup>3</sup> AS/NZS ISO 31000-2009 is identical to, and has been reproduced from, ISO 31000:2009, Risk management—Principles and guidelines. Minor changes have been made to the Introduction to address the application of the standard in Australia and New Zealand.

**4.4 Document**

To facilitate future utility of the ontology, the ontology is documented in a way that can be easily referred to by the target users. In addition to the formal OWL representation, informal visual representations, concept catalog as well as a user friendly ‘reference manual’ has been prepared which facilitated the evaluation case studies discussed above, but will also be used as a reference document in future ontology realization and deployment studies with the intended users i.e. compliance management professionals, businesses (regulated entities), and regulators.

**5 Compliance Management Ontology (Refined)**

The refined ontology consists of 81 concepts. These concepts are structured into four main tiers, representing different levels of detail derived through progressive decomposition of higher tier concepts. For example, the Program concept has been detailed into Obligations Identification and Assessment, Competency and Training, Controls and Monitoring, Record Keeping and Reporting, Review, and Structure in Tier 2, and so on. Each concept is equipped with a definition, attributes, and examples of realization where available. Figure 2 shows CoMOn with its first, second and third tiers concepts after the evaluation and refinement phases. Due to space limitations, we are unable to provide a full list of definitions for all concepts. However, in Table 2, we provide a list of contains definitions for core CoMOn concepts.

**Table 2.** Definitions of core concepts

Concepts	Definitions
Business Process Management	A holistic management approach focused on aligning all aspects of an organization with the wants and needs of clients. It promotes business effectiveness and efficiency while striving for innovation, flexibility, and integration with technology.
Culture Management	The way the organization cultivates compliance culture.
Obligations	The prescribed and/or agreed set of norms that are mandated or voluntarily adopted by an organization or individual.
Program	A series of activities that when combined are intended to achieve the desired level of compliance.
Resources	Monetary and non-monetary resources allocated to meet compliance obligations.
Risk Management	Coordinated activities to direct and control an organization with regard to risk.
Solutions	A particular method, tool or service that provides assistance to the regulated organization in meeting their compliance obligations.



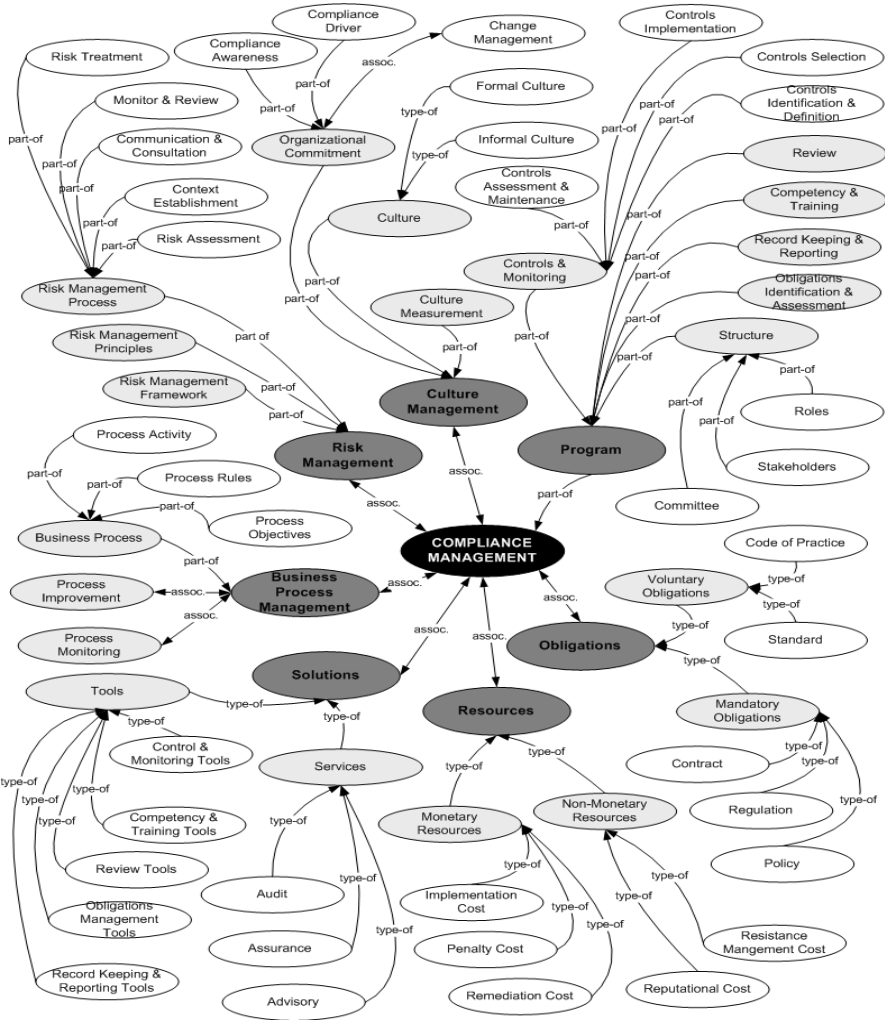


Fig. 2. Refined CoMOn with its first, second and third tier concepts

## 6 Concluding Discussion

In this paper, we have presented CoMOn - A Compliance Management Ontology, which was developed to address an evident need within the compliance management professional and research communities to have a common understanding of the various concepts that define the compliance management landscape. CoMOn is the result of a study that has spanned across the various phases of ontology development, evaluation and refinement. In particular, the ontological consensus achieved through the evaluation has positioned CoMOn as a comprehensive domain ontology for Compliance Management. Further, its role as a tool for communication and facilitation of shared understanding between various organizational functions has been notably recognized

within the Asia Pacific compliance management practitioner community, evidenced through interest from a number of industry groups to engage in future studies. The professional community has indicated that they also see this as a tool that they can use to assess the thoroughness of their compliance regimens and identify aspects of their compliance initiatives and programs that may be missing or lacking.

We are committed to ensuring that CoMoN is well evaluated from a variety of perspectives. Therefore, we are currently conducting an evaluation for the relationships within CoMoN, with the aim to further validate the existing relationships as well as identify further or missing relationships. As a next phase of this research, CoMoN will also be deployed in 7-10 organizations in early 2012 to conduct a longitudinal ontology realization study to thoroughly evaluate CoMoN's usability.

The development of this domain ontology has provided us with a number of insights into the methodological and empirical approaches adopted, which are important to share. A detailed report on the experience is presented in [33], however below we provide a brief summary. First, ensuring the diversity of the sources that constitute the initial corpus for ontology capture is critical. We relied on four sources in this regard, namely industry experts or thought leaders, practitioners with more operational knowledge, industry reports and publications, and research literature. One weakness in this regard is the restriction of research literature to information systems venues only. Although to some extent the limitation does not seem to have compromised the overall quality of the ontology as presented in the results above. Second, we also note that use of support tools (e.g. NVivo) to conduct systematic analysis of the large body of text generated from multiple sources is essential.

Further it is evident in the compliance management industry, as with any other, that the related communities and stakeholders develop a number of conventions (or vocabulary favourites) that are hard to break. The introduction of two standards in the compliance (and risk) management industry, namely AS 3806-2006 Standard on Compliance Programs and AS/NZS ISO 31000-2009 Standard on Risk Management have assisted in overcoming this problem to some extent. Thus, we observe that where as achieving a full consensus is not practical, the ontology evaluation and refinement has to be carefully and systematically undertaken to ensure that the changes made can be fully justified. The use of quantitative scores as well as the systematic analysis of qualitative feedback and informal discussions has proved immensely useful in achieving justifiable changes and extensions to the ontology.

Finally, the empirical data collected through various phases of the study has interestingly revealed a number of other roles of such a domain ontology that were not listed in the original statement of purpose which was primarily to provide a shared understanding of the related concepts.

Overall, the ontology has been seen by the compliance professionals to potentially contribute towards: (1) facilitation of communication, not just for compliance obligations but also for organizational change that may stem from such obligations; (2) training and awareness raising related to organization's compliance and risk functions, and for improving employee competencies in this regard; (3) benchmarking or assessing the current state of the compliance management practice *viz.* what they have now, what is the best or typical practice, and what they possibly missed; (4) and lastly and rather obviously as a record keeping tool, given that the ontology provides a comprehensive information model for compliance related concepts.

## References

1. Sadiq, W., Governatori, G., Namiri, K.: Modeling Control Objectives for Business Process Compliance. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) *BPM 2007*. LNCS, vol. 4714, pp. 149–164. Springer, Heidelberg (2007)
2. Lu, R., Sadiq, S., Governatori, G.: Compliance Aware Business Process Design. In: 3rd International Workshop on Business Process Design (BPD 2007) (2007); In: ter Hofstede, A.H.M., Benatallah, B., Paik, H.-Y. (eds.) *BPM Workshops 2007*. LNCS, vol. 4928, pp. 120–131. Springer, Heidelberg (2008)
3. Louis Anon, J., Filowitz, H., Kovatch, J.M.: Integrating Sarbanes-Oxley Controls into an Investment Firm Governance Framework. *The Journal of Investment Compliance* 8, 40–43 (2007)
4. Syed Abdullah, N., Indulska, M., Sadiq, S.: A Study of Compliance Management in Information Systems Research. In: *The 17th European Conference on Information Systems (ECIS 2009)*, Verona, Italy (2009)
5. Syed Abdullah, N., Sadiq, S., Indulska, M.: Emerging Challenges in Information Systems Research for Regulatory Compliance Management. In: Pernici, B. (ed.) *CAiSE 2010*. LNCS, vol. 6051, pp. 251–265. Springer, Heidelberg (2010)
6. Gruber, T.: Ontology. In: Liu, L., Özsu, M.T. (eds.) *Encyclopedia of Database Systems*. Springer, U.S (2009)
7. Grüninger, M., Lee, J.: Ontology Application and Design: Introduction. *Communication of the ACM* 45, 39–41 (2002)
8. Blomqvist, E., Öhgren, A.: Constructing an Enterprise Ontology for an Automotive Supplier. *Engineering Applications of Artificial Intelligence* 21, 386–397 (2008)
9. Moreira, E.D.S., Andréia, L., Martimiano, F., Brandão, A.J.D.S., Bernardes, M.C.: Ontologies for Information Security Management and Governance. *Information Management & Computer Security* 16, 150–165 (2008)
10. Uschold, M., King, M.: Towards a Methodology for Building Ontologies. In: *Workshop on Basic Ontological Issues in Knowledge Sharing, Held in Conduction with IJCAI 1995*, Montreal, Canada (1995)
11. Uschold, M.: Building Ontologies: Towards a Unified Methodology. In: 16th Annual Conf. of the British Computer Society Specialist Group on Expert Systems (1996)
12. Burton-Jones, A., Storey, V.C., Sugumaran, V., Ahluwalia, P.: A Semiotic Metrics Suite for Assessing the Quality of Ontologies. *Data & Knowledge Engineering* 55, 84–102 (2005)
13. Banker, R.D., Kalvenes, J., Patterson, R.A.: Information Technology, Contract Completeness, and Buyer-Supplier Relationships. *Information Systems Research* 17, 180–193 (2007)
14. Kim, H.M., Fox, M.S., Sengupta, A.: How to Build Enterprise Data Models to Achieve Compliance to Standards or Regulatory Requirements (and share data). *Journal of the Association for Information Systems* 8, 105–128 (2007)
15. Weitzner, D.J., Abelson, H., Berners-Lee, T., Feigenbaum, J., Hendler, J., Sussman, G.J.: Information Accountability. *Communication of the ACM* 51, 82–87 (2008)
16. Davis, C.J., Hikmet, N.: Training as Regulation and Development: An Exploration of the Needs of Enterprise Systems Users. *Information & Management* 45, 341–348 (2008)
17. Mishra, S., Weistroffer, H.R.: A Framework for Integrating Sarbanes-Oxley Compliance into The Systems Development Process. *Communication of the Association for Information Systems* 20, 712–727 (2007)

18. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design Science in Information Systems Research. *MIS Quarterly* 28, 75–105 (2004)
19. Fernández, M., Gómez-Pérez, A., Juristo, N.: METHONTOLOGY: From Ontological Art towards Ontological Engineering. In: *AAAI 1997 Spring Symposium Series*, pp. 33–40. AAAI Press (1997)
20. Fernández-López, M., Gómez-Pérez, A.: Overview and Analysis of Methodologies for Building Ontologies. *The Knowledge Engineering Review* 17, 129–156 (2002)
21. Pinto, H.S., Martins, J.P.: Ontologies: How can They be Built? *Knowledge and Information Systems* 6, 441–464 (2004)
22. Spyns, P., Tang, Y., Meersman, R.: An Ontology Engineering Methodology for DOGMA. *Applied Ontology* 3, 13–39 (2008)
23. Syed Abdullah, N., Sadiq, S., Indulska, M.: A Framework for Industry-Relevant Ontology Development. In: *Proceedings of the 22nd Australasian Conference on Information Systems (ACIS 2011)*. AIS Electronic Library (AISeL), Sydney, Australia (2011)
24. Denaux, R., Dolbear, C., Hart, G., Dimitrova, V., Cohn, A.G.: Supporting Domain Experts to Construct Conceptual Ontologies: A Holistic Approach. *Web Semantics: Science, Services and Agents on the World Wide Web* 9, 113–127 (2011)
25. Paredes-Moreno, A., Martínez-López, F.J., Schwartz, D.G.: A Methodology for the Semi-automatic Creation of Data-driven Detailed Business Ontologies. *Information Systems* 35, 758–773 (2010)
26. Lim, S.C.J., Liu, Y., Lee, W.B.: A Methodology for Building a Semantically Annotated Multi-faceted Ontology for Product Family Modelling. *Advanced Engineering Informatics* 25, 147–161 (2011)
27. Syed Abdullah, N., Sadiq, S., Indulska, M.: Information Systems Research: Aligning to Industry Challenges in Management of Regulatory Compliance. In: *Proceedings of the Pacific Asia Conference on Information Systems Engineering (PACIS 2010)*, Taipei, Taiwan (2010)
28. Booch, G., Rumbaugh, J., Jacobson, I.: *Unified Modeling Language User Guide*. Addison-Wesley Professional (2005)
29. Pulido, J.R.G., Ruiz, M.A.G., Herrera, R., Cabello, E., Legrand, S., Elliman, D.: Ontology Languages for the Semantic Web: A Never Completely Updated Review. *Knowledge-Based Systems* 19, 489–497 (2006)
30. W3C OWL Working Group: *OWL 2 Web Ontology Language - Document Overview*. World Wide Web Consortium (2009)
31. Pinto, H.S., Gómez-Pérez, A., Martins, J.P.: Some Issues on Ontology Integration. In: *Proceedings of the IJCAI 1999 Workshop on Ontologies and Problem-Solving Methods (1999)*
32. Pinto, H.S., Martins, J.P.: A Methodology for Ontology Integration. In: *Proceedings of the 1st International Conference on Knowledge Capture*, pp. 131–138. ACM, Victoria (2001)
33. Syed Abdullah, N., Sadiq, S., Indulska, M.: A Study of Ontology Construction: The Case of a Compliance Management Ontology. In: Ahmad, M.N., Colomb, R.M., Abdullah, M.S. (eds.) *Ontology-Based Applications for Enterprise Systems and Knowledge Management*. IGI Global (in press)

# Patterns to Enable Mass-Customized Business Process Monitoring

Marco Comuzzi<sup>1</sup>, Samuil Angelov<sup>2</sup>, and Jochem Vonk<sup>3</sup>

<sup>1</sup>Eindhoven University of Technology, Eindhoven, The Netherlands  
m.comuzzi@tue.nl

<sup>2</sup>Fontys University of Applied Sciences, Eindhoven, The Netherlands  
s.angelov@fontys.nl

<sup>3</sup>Logica Nederland BV, Eindhoven, The Netherlands  
jochem.vonk@logica.com

**Abstract.** Mass-customization challenges the one-size-fits-all assumption of mass production, allowing customers to specify the options that best fit their requirements when choosing a product or a service. In business process management, to achieve mass-customization, providers offer to their customers the opportunity to customize the way in which a process will be enacted. We focus on monitoring as a specific customization aspect. We propose a multi-dimensional classification of modeling patterns for customized monitoring infrastructures. Patterns enable the provider to offer a set of customizable options to customers and design a monitoring infrastructure that fits the preferences specified by customers on such options. An example in the online advertising industry demonstrates how our framework can improve the services currently offered by providers.

**Keywords:** monitoring framework, monitoring patterns, business process monitoring, process customization.

## 1 Introduction

Mass-customization is defined as “developing, producing, marketing and delivering affordable goods and services with enough variety and customization that nearly everyone finds exactly what they want” [20]. The aim of mass customization is to challenge the one-size-fits-all assumption of mass production, allowing customers choosing a product or a service to also specify the options that best fit their requirements. As such, it has been successfully achieved by many organizations particularly in the manufacturing industry, with examples ranging from the BMW’s online product configurator to Dell’s hardware configuration services.

In this paper, we consider mass-customization in business process management and, specifically, how to support process providers in implementing mass-customized business processes. In B2B, provider companies perform a set of activities, i.e. a business process, to satisfy a customer company. Mass-customization, in this context, should result in the customers’ ability to customize the way in which providers will

execute a process on their behalf. Given a standard version of a process, customization ranges from process control [18], [21], e.g. the opportunity for the customer to skip, redo, or cancel activities, to QoS customization [27], e.g. the opportunity for the customer to pick a certain level of guaranteed security in financial transactions, or resources customization [24], e.g. the opportunity for customers to directly choose which resources will be used in the process.

We focus on a management aspect that has not yet been extensively investigated by the literature on BPM and customization, i.e. monitoring. Generally, business entities need information about the activities taking place in the business landscape (at internal units, partner organizations, or third parties) so that they can react and/or adapt to them. Thus, business entities need mechanisms that ensure the collection of relevant data from the environment (referred to as *monitoring* [1]). In B2B, customers may use monitoring information, for instance, to synchronize their own internal processes or to assess the provider behavior, especially in dynamic relationships where customers and providers are often not likely to have conducted business together in the past. In this context, each customer is likely to have specific monitoring requirements in terms, for instance, of what information the provider should make available and how this has to be communicated, e.g. along which channel, with which frequency. Customization of monitoring requires focus on both control flow aspects, specifying the way to capture monitoring information and make it available, and resources, specifying what has to be monitored and at which stage of the process.

A successful mass-customizer should develop three capabilities [19]: (i) *solution space development*, i.e. understanding customer needs and options over which those are likely to diverge, (ii) *robust process design*, i.e. design an infrastructure to offer such options, making sure that customization does not hinder the company's ability to execute its processes in an efficient and effective way, and (iii) *choice navigation*, i.e. support customer selection of options minimizing the burden of choice.

Focusing on the provider designing the monitoring infrastructure, this paper aims at supporting the capabilities (i) and (ii). To support the provider in the identification of the *solution space*, in fact, we provide a set of patterns for the conceptual design of mass-customizable business process monitoring infrastructures. The patterns are positioned in a framework providing a multidimensional classification space. The multidimensional space is constructed by reasoning on the literature for software monitoring, Web service, and business process monitoring and by using existing, context-specific solutions as verification and illustration techniques. To support the *robust design* of customized monitoring infrastructures, we show how to combine patterns depending on specific contextual requirements using an example in online advertising.

The paper is organized as follows. In Section 2, we introduce the multidimensional space of our framework. In Section 3, we provide the patterns for the options defined within the multi-dimensional space. Section 4 discusses an example of the application of the framework, showing how it can be used to improve current practice in online advertising to achieve mass-customizable business process monitoring. In Section 5, we discuss relevant literature on business process customization and monitoring. The paper ends with our conclusions and plans for future work.

## 2 Monitoring Dimensions and Options

This paper considers monitoring of business processes supported by information systems. In Fig. 1(a), we sketch the general outline of a monitoring architecture. In the business process provider domain, a *monitoring infrastructure* (also called sensor or observer [2-3]) capturing relevant information is built around a *business process engine*. A business process engine is meant in a broad sense - it is a component producing process information (activity or process states, data values, etc.). Although the monitoring infrastructure can be built in (or even be an integral part of) the business process engine [2], conceptually, it is a separate entity [3]. Considering it as a separate entity promotes separation of concerns with respect to the underlying process engine, leading to a more focused and context-independent analysis of the monitoring infrastructure component and its possible customization. In the customer (also called controller [2-3]) domain, a *monitoring client* obtains the information captured by the monitoring infrastructure. It processes the information obtained and if desired exerts control over the business process engine.

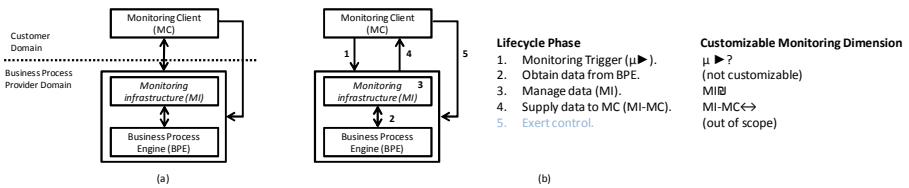
In many scenarios, the monitoring client can be an intermediary for the actual business entity that requires the monitoring information. Monitoring may take place within an organization (between independent business units) or in a cross-organizational setting. Furthermore, the client of the monitoring infrastructure may be an autonomous application or a human user. As these cases do not introduce any specifics in our work, in the sequel of the paper, we abstract from them and discuss the general scenario depicted in Fig.1(a).

The dimensions of our framework identify what parts of the scenario in Fig.1(a) can be customized by the monitoring client, i.e. they define the monitoring *solution space* for the customer of the business process.

The first two dimensions concern the context in which monitoring has to occur. In particular, we consider the *monitoring variable* and the *anchoring points*, defined as follows:

**Monitoring Variable (MV):** The MV specifies the object of monitoring, i.e. the process information that the *Monitoring Infrastructure* (MI) obtains from the *Business Process Engine* (BPE) and makes available to the *Monitoring Client* (MC). It has a domain, which specifies the (range of) values that it can assume, and a unit of measure, if needed. In the world of software programs monitoring, the monitoring variable is the target, for instance, of a watch for debugging. In business process monitoring, the monitoring variable may range from infrastructure-level data, such as timestamps of service calls [5], to application-level process data, such as the status of an activity or domain specific data produced by an activity [6-7]. Note that a value of the monitoring variable represents a single data element captured by MI during the execution of the business process, e.g. the timestamp of an order, the warehouse level at a specific point in time, the unique id of a user executing a specific activity. Captured values can be stored by MI during the execution of a business process and made available in batches to MC.

**Anchoring Point (AP).** The MI is enabled within a specific scope of the process to be monitored. Anchoring points specify the scope of the process within which the MI is enabled. The notion of anchoring point derives from the literature on software program monitoring, where running the monitoring program in the same memory space as the monitored program can be costly and, therefore, the monitoring program has to be enabled only when strictly necessary [8]. In a business setting, while in many cases we can make the hypothesis that monitoring is permanently enabled, defining anchoring points may be helpful when capturing and making available the values of MV to MC is very costly. Intrusive process monitoring [5] is an example of this scenario, since the execution of monitoring statements blocks and, therefore, delays, the execution of the process. In such a scenario, the MC may want to enable monitoring only when strictly necessary. In the remainder, we refer to AP-START and AP-END as the anchoring points enabling and disabling the Monitoring Infrastructure MI, respectively.



**Fig. 1.** (a) Monitoring Scenario; (b) Business Process Monitoring Lifecycle

Fig.1(b) refines the conceptual outline of a monitoring solution by showing the lifecycle of communications among the different elements of the architecture. We use the phases of such lifecycle to derive the next three dimensions of our framework. In particular, the first phase concerns commanding the acquisition of monitoring data. This can be done either by MC or by MI (Phase 1). For instance, MC may specify that MI has to acquire values of a specific MV periodically or may want to be allowed to command the acquisition pro-actively.

The second phase concerns MI obtaining the required data from BPE (Phase 2). Since this phase is internal to the business process provider domain and cannot be customized, we do not consider it in this paper. Then, MI may have to manage, e.g. store in batches, the monitoring data obtained by BPE (Phase 3) before supplying them to MC (Phase 4). Eventually, MC processes monitoring data and may decide to exert control on the monitored process executed by BPE (Phase 5). This last phase is out of scope in this paper.

Phases 1, 3, and 4 of the lifecycle in Fig. 1(b) are characterized by one monitoring dimension in our framework, since they involve aspects that are customizable by MC. Each dimension has a set of options. Options represent the solution space for the customization of process monitoring, that is, a customized monitoring infrastructure can be built once the customer has chosen one option for each possible monitoring dimension. In the remainder of this section we present the monitoring dimensions and their possible options.



**Monitoring Trigger ( $\mu \blacktriangleright$ ).** The monitoring trigger phase is characterized by one dimension that identifies the entity commanding the acquisition of values of MV ( $\mu \blacktriangleright ?$ ). The acquisition of a monitoring value can be triggered by MC or by MI. In the former case ( $\mu \blacktriangleright ? = MC\text{-}trg$ ), MC makes a request to the provider for commanding the acquisition of the value of MV. In the latter case ( $\mu \blacktriangleright ? = MI\text{-}trg$ ) MI acquire values of MV proactively. Note that in this case MC should be able to customize the way in which MI will acquire values, for instance periodically at a certain frequency, or when a threshold is exceeded, or on change. The third option ( $\mu \blacktriangleright ? = mix\text{-}trg$ ) fits cases in which acquisition is triggered by MI, e.g. periodically, but also the client MC wants to be allowed to request acquisition. Once acquisition is triggered, MI will obtain monitoring data from BPE. This phase of the monitoring lifecycle is internal to the business domain of the provider and, therefore, it cannot be customized by MC.

**Manage Data (MI).** For this lifecycle phase we define one monitoring dimension ( $MI \boxplus$ ), which captures MI's logic in managing the values obtained from BPE before supplying them to MC. New values obtained from BPE can rewrite old values captured for the same MV or the obtained values can be stored (persisted), e.g. to produce historical series of values of MV. When supplied to MC, values can be consumed, i.e. they will not be available in the future to MC, or they can just be read, remaining available also in the future. Thus, we identify four options for  $MI \boxplus$ , i.e. (i) *rewrite-consume*, (ii) *rewrite-read*, (iii) *persist-consume*, and (iv) *persist-read*.

**Supply Data to MC (MI-MC).** This phase is characterized by the dimension  $MI\text{-}MC \leftrightarrow$  referring to the direction of the interaction between MI and MC. For  $MI\text{-}MC \leftrightarrow$  we define the options *push* and *pull*. The option *push* models cases in which MI pushes monitoring values to MC, whereas *pull* models cases in which MI sends values of MV only after having received a request from MC. Generally, communication between MI and MC is a distributed systems communication issue and its customization may require the definition of additional dimensions, such as space and time decoupling option [9, 10]. However, since those aspects are not monitoring-specific, we do not further discuss them here.

### 3 Monitoring Patterns for Monitoring Dimensions

In this section, we use the monitoring dimensions and present the design of internal data and control flows of the Monitoring Infrastructure (MI). This is required for the provider to offer mass-customization monitoring capabilities to its customers. Following common principles in the design of distributed systems [3], we first define the interfaces required between MI and MC, and MI and BPE, respectively, to support the monitoring dimensions. Then, we propose a modeling pattern for each option characterizing the monitoring dimensions. Modeling patterns specify the data and control flow of MI's internal implementation of the interfaces. Eventually, Section 4 presents an example in the online advertising industry to combine the patterns in a monitoring infrastructure with certain desired monitoring properties chosen by customers.

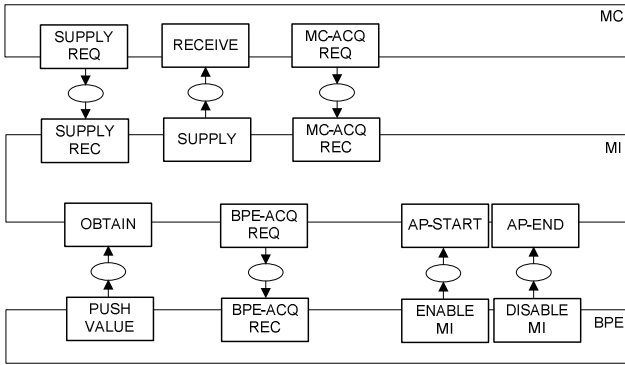


Fig. 2. Interfaces between MI, MC, and BPE

For business process modeling purposes, we use Colored Petri Nets (CPNs) [10-11]. CPNs have been chosen because they have a graphical representation, they have mature and freely available tool support, e.g. CPN Tools, and they have a precise semantic that can be translated to or directly implemented in other languages, e.g. Event-driven Process Chains [12] or YAWL [6], used by commercial and non-commercial workflow engines. Readers unfamiliar with CPNs’ notation are referred to [11]. We use two token colors in our patterns, i.e. UNIT and MV. UNIT represents the default color of black tokens, and it is used to model the control flow in our modeling patterns. Tokens of color MV represent a single value of the monitoring variable MV. The precise definition of the color MV, e.g. possible values and unit of measure, is part of the definition of the monitoring context and we do not further discuss it in this paper.

Fig. 2 shows the interfaces between the elements of the monitoring solution MI, MC, and BPE. Note that, although aspects related to the interaction between MC and BPE are internal to the provider domain, and therefore not subject to possible customization, considering the interfaces between MI and BPE allows separation of concerns between business process execution and monitoring and, therefore, for more robust patterns for customizable process monitoring infrastructures.

MI requires one interface to receive acquisition triggers from MC (MC-ACQ-REC) and supply data to MC (SUPPLY). Optionally, MI may require also an interface to receive supply requests from MC (SUPPLY REQ). This interface is necessary only when MC decides to pull monitoring data from MI. The interface between MI and BPE is constituted by a generic interface for receiving monitoring values (OBTAIN), by the two anchoring points, and by interface BPE-ACQ-REQ allowing MI to command acquisition of values of MV by BPE. Note that Fig. 2 does not show the color of tokens in places connecting interfaces. As shown later, the color of such places is determined by the modeling pattern realizing the MI’s internal implementation of the corresponding interface(s).

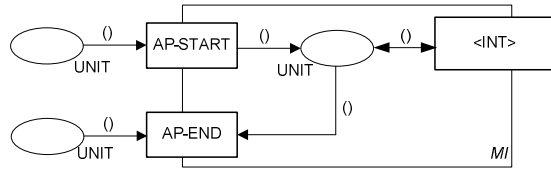


Fig. 3. Pattern for anchoring point implementation

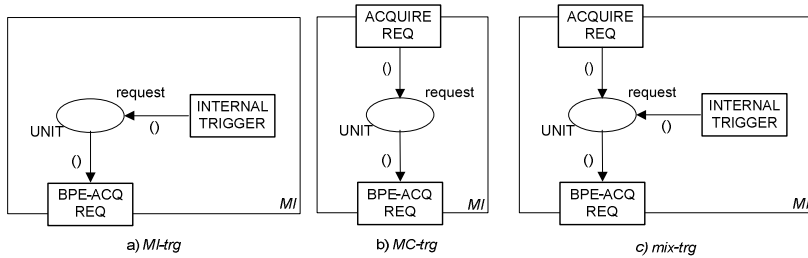


Fig. 4. Monitoring patterns for  $\mu \blacktriangleright ?$  dimension options

Concerning the context, MI is required to expose interfaces to define the anchoring points of monitoring to the business process executed by BPE. The tokens required to fire the transitions AP-START and AP-END are produced by BPE when the process execution reaches the point enabling and disabling the monitoring, respectively. Fig.3 shows the pattern for the implementation within MI of the anchoring point business logic. Specifically, the generic interface <INT>, which can be any of the ones defined in Fig. 2, becomes enabled after the firing of the AP-START transition and is no longer enabled after the AP-END transition has fired.

The remainder of this section presents the monitoring patterns for the remaining monitoring dimensions. Each pattern models one option of one monitoring dimension identified in Section 2.

**Monitoring Trigger ( $\mu \blacktriangleright ?$ ).** The patterns modeling the options of dimension  $\mu \blacktriangleright ?$  implement the ACQUIRE-REQ and BPE-ACQ-REQ interfaces of MI.

The patterns corresponding to the three values *MI-trg*, *MC-trg*, and *mix-trg* are shown in Fig.4. In Fig.4(a), the transition INTERNAL TRIGGER captures MI’s business logic to trigger acquisition, e.g. periodically or on change, which can be customized by MC. In Fig.4(b) the acquisition is commanded by MC, by firing the transition ACQUIRE REQ, whereas Fig.4(c) shows the mix case, i.e. MC or MI can both trigger an acquisition request.

**Supply data to MC (MI-MC).** The patterns for the *push* and *pull* options of the dimension MI-MC $\leftrightarrow$  are shown in Fig. 5(a) and Fig. 5(b), respectively. The *push* option requires MI to only expose the SUPPLY interface. The transition SUPPLY TRIGGER fires accordingly to the logic chosen by the customer to receive monitoring values. The customer, for instance, may require monitoring values periodically or only when values exceed a certain threshold. For the *pull* option, the SUPPLY interface can fire, i.e. supply a monitoring value, only after a request is received. Note that that this implemented by limiting the control exerted by the anchoring point to the SUPPLY-REQ interface.

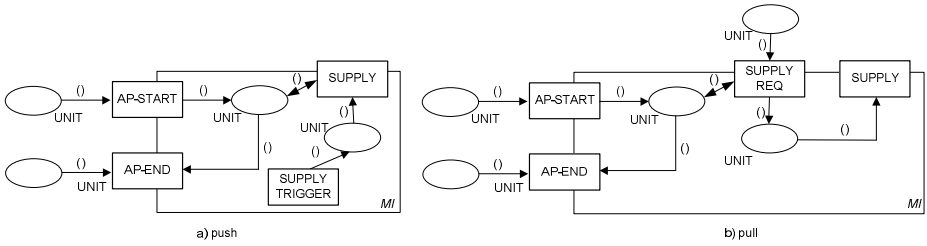


Fig. 5. Monitoring patterns for the MI-MC $\leftrightarrow$  dimension options

**Manage data (MD).** The patterns for the MI  $\rightleftharpoons$  dimension implement the connection between the interfaces SUPPLY and OBTAIN exposed by MI. Fig. 6 shows the patterns for the four options.

Note that the place *bpe* stores values of MV ready to be obtained by MI, whereas the place *mc* stores the value or set of values supplied to the monitoring client MC. Also, note that the color *L\_MV* represents a list of tokens of color MV.

In the *rewrite-* options, the new value of MV *v\_new* always replaces the old value *v\_old*. In the *persist-* options, the values of MV are stored by MI in a list *lv* (the list in this case represents a generic data structure); a new monitoring value *v\_new* is simply inserted into *lv*.

In the *-consume* options, the SUPPLY transition, when fired, replaces the current monitoring value stored by MI with the default, whereas in the *-read* options the monitoring value is put back in the place *mi\_db* after being read and can be read again in the future by MC. Note that the default monitoring value is the empty list [] for the *persist-* options and the token with value “dft” of color MV for the *rewrite-* options.

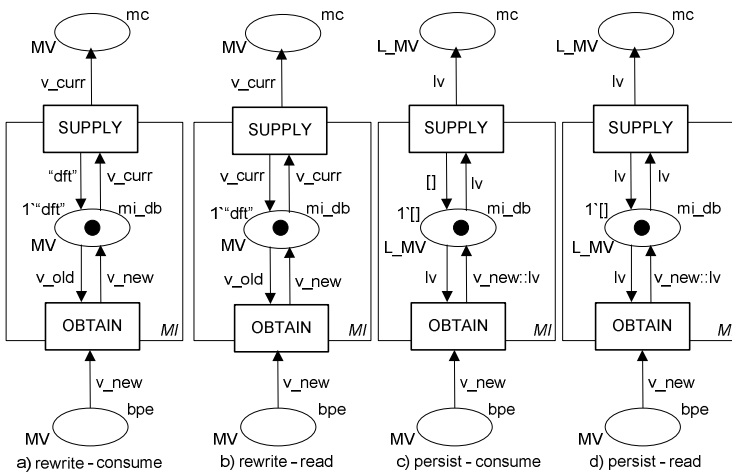
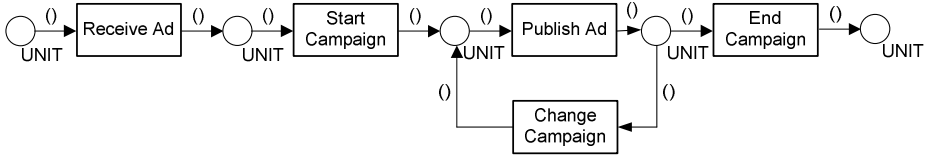


Fig. 6. Patterns for MI  $\rightleftharpoons$  dimension options

## 4 Combining Patterns to Design Monitoring Infrastructures

To demonstrate the value of the framework, we provide an example for its application in an on-line advertising scenario. The example shows how the framework can be applied for the design of a customizable monitoring solution and the advantages that it offers to the designers of the monitoring infrastructure at the provider side.



**Fig. 7.** Excerpt from the on-line advertising process

The advertising provider (e.g. a newspaper) offers advertising space to companies (customers). In a simplified scenario (see Fig. 7), the customer sends the advertisement to the provider and when the time agreed to start the campaign comes, the provider starts the advertising campaign by publishing the ad in its newspaper. In a basic scenario, the ad is shown each time a reader loads the newspaper page, while in a more complex case, the most relevant and highest priced ad is shown in a specific advertising spot (e.g. in Google AdWords). When the budget of the customer is depleted and/or the number of agreed appearances of the ad is reached, the campaign ends. The customer may ask to change its campaign if it observes that the ad is not reaching the target audience, the campaign has little impact, etc. This business case is described in greater detail in [28].

In a traditional advertising setting, the provider offers a fixed set of tools to all customers for monitoring their campaigns. For example, the customers can monitor the IP addresses of the readers seeing their ad, the number of shows of their ad, number of clicks on their ad, the spots where the ad has been published (banner, column, pop-up, in-text), etc. Typically, customers have to access their account to view this information. This monitoring advertising scheme does not address the individual preferences of the customers. Customers may be interested in receiving the monitoring information directly instead of having to query it from the provider; they may be interested in obtaining the information in an aggregated format at the end of the campaign or be constantly updated to be able to adapt their campaign; they may be interested, motivated by a cheaper price, in obtaining only some of the monitoring information instead of monitoring all possible variables, etc. Next, we describe how our framework can be applied to set up a mass-customizable monitoring infrastructure. For brevity, we focus on the construction of monitoring infrastructure for only one monitoring variable, i.e., the number of clicks on an ad, which is the most straightforward indicator of the ad success and profitability over time.

To apply the framework for the mass-customization of a MV, the provider has to consider the possible options from the monitoring dimensions presented in Section 2

to offer to its customers (see Table 1). The MV in this case is the current value of number of clicks on the ad of the customer. We hypothesize that the provider (newspaper), through an advertising engine, keeps track continuously of this value. The customer wants to monitor MV along the campaign. Therefore, the anchoring point enabling and disabling monitoring are the start and the end of the campaign, respectively. Note that, in principles, customers may choose a different anchoring point, for instance enabling monitoring only after the campaign has been changed a first time.

**Table 1.** Possible monitoring dimension values for the IP address MV

<i>Monitoring Variable</i>	Number of clicks on the customer’s ad (cumulative)
<i>Anchoring Point</i>	“Start Campaign” (enabling); “End Campaign” (disabling)
<i>Monitoring Trigger</i>	$\mu \blacktriangleright ?$ : <i>MC-trg</i> , <i>MI-trg</i>
<i>Manage Data</i>	$MI \sqcap$ : <i>persist-read</i> , <i>persist-consume</i>
<i>Supply data to MC</i>	$MI-MC \leftrightarrow$ : <i>push</i> , <i>pull</i>

A customer may like to be pushed monitoring information or to pull it (thus,  $MI-MC \leftrightarrow = push$  or  $MI-MC \leftrightarrow = pull$ ). The customer may prefer to monitor the MV only at a specific point in time that cannot be revealed (e.g., it is not known) to the provider (hence,  $\mu \blacktriangleright ? = MC-trg$ ) but may also like to delegate the acquisition of the monitoring values to the provider at a pre-agreed time, periodically or when the number of clicks exceeds a certain threshold (hence,  $\mu \blacktriangleright ? = MI-trg$ ). Typically, customers would prefer to have all the monitored values stored by the provider, so that these can be queried any time later on and used to analyze the number of clicks trend over time (hence,  $MI \sqcap = persist-read$ ). Of course, if storage space is crucial for the provider, it may offer some incentives (e.g., financial) to the customers to choose also  $MI \sqcap = persist-consume$ .

Thus, having the framework as a guiding tool in the set of possible monitoring styles, the provider has straightforwardly defined all possible monitoring styles for the number of clicks MV. Each customer interested in monitoring information on the cumulative number of clicks is presented with a set of possible options. Table 2 presents two possible sets of choices for customers A and B. Customer A delegates the acquisition of monitoring values to the provider at the beginning of its campaign,

**Table 2.** Selected monitoring dimension options by Customers A and B

	<i>Customer A</i>	<i>Customer B</i>
<i>Monitoring Trigger</i>	$\mu \blacktriangleright ?$ : <i>MI-trg</i>	$\mu \blacktriangleright ?$ : <i>MC-trg</i>
<i>Manage Data</i>	$MI \sqcap$ : <i>persist-read</i>	$MI \sqcap$ : <i>persist-consume</i>
<i>Supply Data to MC</i>	$MI-MC \leftrightarrow$ : <i>pull</i>	$MI-MC \leftrightarrow$ : <i>push</i>



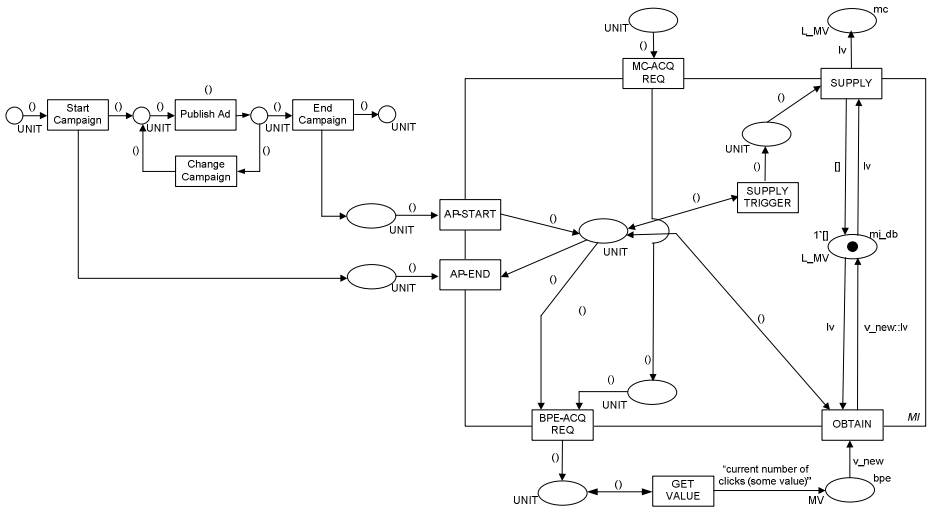


Fig. 9. Customized business process monitoring for Customer B

## 5 Related Work

Process customization is a paramount activity in the implementation of complex enterprise systems, such as ERP [25]. Traditional process customization in enterprise systems is a design time concern, which aims at designing standard processes across the implementing organization. While process standardization across the enterprise promotes uniformity and interoperability [25], it is also seen as a constraining institutional factor for large companies with diverse business units [26], limiting the flexibility of the company. In this paper, we take a much more dynamic perspective on process customization, allowing each single client to specify his or her own monitoring requirements at runtime and supporting the derivation of a monitoring infrastructure to support those.

As remarked in the Introduction, the literature on business process customization has not extensively considered monitoring as an aspect that can be customized. As an example, the work in [21] proposes an approach to add control flow options, such as adding, skipping, or redoing an activity, to an original process model, but with no explicit mention of monitoring options. In a different perspective, QoS-based Web service selection can be seen as a form of customization, since the building blocks of a business process are selected at run-time to satisfy the QoS requirements expressed by the user [27]. However, also in this case, we did not find approaches considering monitoring as a customizable aspect.

Business process customization is usually opposed to *configuration*, which aims at designing reference process models capturing the behavior of a set of process variants serving the same business goal [22-24]. We argue that the business process configuration approach does not suit mass-customization, since capturing all possible process variants in a single process model leads to very complex models that are not



easily understandable by the process customers. Note that a reference configurable process in our example should contain all possible combinations of monitoring options values for each possible combination of MV and AP in the original process. In a nutshell, configuration remains a design-time concern of process designers [23], whereas customization should be performed directly by the customer right before the process enactment. In our review of related work, we were not able to find approaches to process configuration explicitly capturing monitoring options.

To compile a list of possible behaviors of the monitoring infrastructure and its communication with the business process and the client application, we analyzed the literature on traditional software program monitoring, Web service-based monitoring and business process monitoring.

A survey on software programs and software requirements monitoring can be found in [8]. From this survey, we take the notion of monitoring points. Monitoring points define the anchors of the monitoring program to the monitored program. Similarly, in our model we define *anchor points* for monitoring options to the monitored process.

The survey in [8] is used by [5] to classify approaches to Web service-based process monitoring. In particular, [1] considers the modality to notify monitoring information as classification criterion. Monitoring information, usually captured by an instrumentation of the Web service container, can be either pushed to the monitor or pulled by it. Web service monitoring usually takes an event source-listener approach [9], where the instrumented Web service container is the source that pushes monitoring related event to the monitor (listener) [15]. When monitoring information is pushed, the work in [5] also considers the multiplicity and frequency with which monitoring information is made available to the monitor. Still in the context of Web service monitoring, the model in [16] considers the concept of monitoring socket, i.e. a generic component which is responsible for the generation of monitoring data, which can then be pushed to or pulled by the monitor.

## 6 Conclusions

In this paper, we present a conceptual framework for the modeling of mass-customized business process monitoring infrastructures. The framework identifies a set of orthogonal dimensions and options along which the customer monitoring requirements may vary. For each option, we provide patterns that model the process and data aspect of the monitoring infrastructure. The customized process monitoring infrastructure is then modeled through the combination of patterns from the dimensions. We illustrate the applicability of the framework and discuss its value using an online advertising example process.

The embedding of our framework in a wider context raises several issues that must be paid closer attention. Firstly, our framework is of descriptive nature and does not specify concrete steps and guidelines that need to be followed for its usage. A method that describes its application would improve its value as a design and analysis tool. Secondly, exercising control over a business process is the logical continuation of

monitoring activities. Thus, the value of the framework can be further extended by incorporating it in a generic framework for monitoring and control. This is the focus of our ongoing work [18].

Finally, we are looking at a possible implementation of our framework. Process configuration requires extensions of currently available workflow management systems to enable injecting behavior into existing process specifications. The efficient execution of customized processes may rely on the cloud computing paradigm, in which the resources required by each customer (or by a class of similar customers) for their customized monitoring can be bundled and provisioned as a service.

## References

1. zur Muehlen, M.: *Workflow-Based Process Controlling*. Springer, Berlin (2005)
2. Curtis, G., Cobham, D.: *Business Information Systems: Analysis, Design and Practice*, 6th edn. Financial Times/ Prentice Hall (2008)
3. Wieringa, R.: *Design Methods for Reactive Systems: Yourdon, Statemate, and the UML*. Morgan Kaufmann Publishers (2003)
4. Baresi, L., Guinea, S., Nano, O., Spanoudakis, G.: Comprehensive Monitoring of BPEL Processes. *IEEE Internet Computing*, 50–57 (2010)
5. ter Hofstede, A.M., van der Aalst, W.M.P., Adams, M., Russell, N.: *Modern Business Process Automation: YAWL and its Support Environment*. Springer (2010)
6. IBM Corporation: *WebSphere Business Monitor*, <http://www-01.ibm.com/software/integration/wbimonitor/>
7. Delgado, N., Gates, A.Q., Roach, S.: A Taxonomy and Catalog of Runtime Software-Fault Monitoring Tools. *IEEE Transactions on Software Engineering* 30(12), 859–872 (2004)
8. Eugster, P.T., Felber, P.A., Guerraoui, R., Kermarrec, A.-M.: The many faces of Publish/Subscribe. *ACM Computing Surveys* 35(2), 114–131 (2003)
9. Aldred, L., et al.: Dimensions of Coupling in Middleware, Concurrency, and Computation. *Practice and Experience* 21, 2233–2269 (2009)
10. Jensen, K., Kristensen, L.M.: *Coloured Petri Nets*. Springer (2009)
11. Scheer, A.W.: *ARIS Business Process Modeling*. Springer (2000)
12. Spanoudakis, G., Mahbub, K.: Non Intrusive Monitoring of Service Based Systems. *International Journal of Cooperative Information Systems* 15(3), 325–358 (2006)
13. Sadiq, W., Governatori, G., Namiri, K.: Modeling Control Objectives for Business Process Compliance. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) *BPM 2007*. LNCS, vol. 4714, pp. 149–164. Springer, Heidelberg (2007)
14. Angelov, S., Vonk, J., Grefen, P., Vidyasankar, K.: Enhancing Business Collaborations with Client-Oriented Process Control. *International Journal of Cooperative Information Systems* 20(1), 1–37 (2011)
15. Salvador, F., de Holan, P.M., Piller, F.: Cracking the Code of Mass Customization. *SLOAN Management Review* 50, 71–78 (2009)
16. Pine, J.B.: *Mass Customization - The New Frontier in Business Competition*. Harvard Business School Press, Cambridge (1993)
17. Hallerbach, A., Bauer, T., Reichert, M.: Capturing variability in business process models: the Provop approach. *J. Softw. Maint. Evol-R* 22, 519–546 (2010)

18. Lapouchnian, A., Yu, Y., Mylopoulos, J.: Requirements-Driven Design and Configuration Management of Business Processes. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 246–261. Springer, Heidelberg (2007)
19. Gottschalk, F., van der Aalst, W.M.P., Jansen-Vullers, M.H., La Rosa, M.: Configurable Workflow Models. *International Journal of Cooperative Information Systems* 17(2), 177–221 (2008)
20. La Rosa, M., Dumas, M., ter Hofstede, A.H.M., Mendling, J., Gottschalk, F.: Beyond Control-Flow: Extending Business Process Configuration to Roles and Objects. In: Li, Q., Spaccapietra, S., Yu, E., Olivé, A. (eds.) ER 2008. LNCS, vol. 5231, pp. 199–215. Springer, Heidelberg (2008)
21. Jacobs, F.R., Whybark, C.: *Why ERP?* Irwin/McGraw Hill, New York (2000)
22. Gattiker, T.F., Goodhue, D.L.: What Happens After ERP Implementation: Understanding the Impact of Interdependence and Differentiation on Plant-Level Outcomes. *MIS Quarterly* 29, 559–585 (2005)
23. Gmach, D., Krompass, S., Scholz, A., Wimmer, M., Kemper, A.: Adaptive Quality of Service Management for Enterprise Services. *ACM Trans. Web*, 2(1), Article 8 (2008)
24. Angelov, S., Grefen, P.: Supporting the Diversity of B2B e-Contracting Processes. *International Journal of Electronic Commerce* 12(4), 39–70 (2008)

# Predicting QoS in Scheduled Crowdsourcing

Roman Khazankin, Daniel Schall, and Schahram Dustdar

Distributed Systems Group, Vienna University of Technology,  
Argentinierstrasse 8/184-1, A-1040 Vienna, Austria

[lastname@infosys.tuwien.ac.at](mailto:lastname@infosys.tuwien.ac.at)

<http://www.infosys.tuwien.ac.at>

**Abstract.** Crowdsourcing has emerged as a new paradigm for outsourcing simple for humans yet hard to automate tasks to an undefined network of people. Crowdsourcing platforms like Amazon Mechanical Turk provide scalability and flexibility for customers that need to get manifold similar independent jobs done. However, such platforms do not provide certain guarantees for their services regarding the expected job quality and the time of processing, although such guarantees are advantageous from the perspective of Business Process Management. In this paper, we consider an alternative architecture of a crowdsourcing platform, where the workers are assigned to tasks by the platform according to their availability and skills. We propose the technique for estimating accomplishable guarantees and negotiating Service Level Agreements in such an environment.

**Keywords:** Crowdsourcing, Scheduling, QoS, SLA, Negotiation.

## 1 Introduction

Enterprises seek to automate all sorts of their front and back office operations to cut costs and eliminate human factors. However, humans inevitably remain as key elements of many business processes. It is not only creative, management, and communication activities that are hard to reproduce with technology. Other tasks that require basic human skills, such as image recognition or categorization, as well as specific skills, such as text translation or usability testing, are also difficult to fulfill with software only. Crowdsourcing remains a prominent paradigm for solving such tasks and providing human workforce on demand. Recent efforts demonstrate the successful adoption of crowdsourcing techniques at an ever-increasing rate, and the amounts of both platforms and workers in such platforms are expected to grow rapidly [5].

Crowdsourcing systems are diverse in their architecture, target problems, and user interaction styles [3]. We focus on platforms that deal with tasks consisting of manifold similar independent jobs provided by consumers, such as translation, reviewing, voting, tagging, content creation, image recognition, data quality improvement, and so on. Currently, such platforms have a market-like operation chain where the tasks received from customers are announced at the portal and the workers choose among the assorted mass of tasks those they like to process.

Examples of such systems include Amazon Mechanical Turk<sup>1</sup>(AMT), CrowdFlower<sup>2</sup> and CloudCrowd<sup>3</sup>.

Although the aforementioned systems are considered quite successful, we argue that purely market-like architecture lacks some features that could realize more potential of crowdsourcing platforms. As in a market-like system the job assignments are initiated by the workers themselves, it is hardly possible for the system to have an active influence upon assignments. As a result, the platform is unable to give any certain guarantees for the consumers, neither about the time of processing a task nor about the outcome quality they can expect<sup>9</sup>. From a Business Process Management (BPM) perspective, such guarantees can provide an additional value for crowdsourcing services. First, it can strengthen the certainty and predictability in process planning and design. Second, if such guarantees are given in form of Service Level Agreements (SLAs), it allows for crowdsourcing in QoS(Quality of Service)-sensitive business processes <sup>[1,2,6]</sup>.

In our previous work <sup>[12]</sup> we presented a crowdsourcing platform model where the workers are assigned to tasks *by the platform* according to current workers' availability, skills of workers, skill requirements provided by consumers, and service level agreements with consumers (described in Sec. <sup>2</sup>). We refer to this model as *scheduled crowdsourcing*. While providing the same flexibility, scheduled crowdsourcing comprises a number of advantages:

- **Quality.** Skills of the workers are manifold. The tasks submitted to the platform are also diverse in their requirements. One can assume that the more the worker is suitable for a task, the better the expected outcome quality is. We refer to this indicator as *suitability*. Hence, by considering the worker-task suitability, it is possible to improve the overall results by assigning tasks to most suitable workers.
- **Deadlines.** In market-like platforms task completion times span from several to thousands of hours <sup>8</sup>. As in scheduled crowdsourcing the assignments are controlled by the platform, tasks can be scheduled according to specified deadlines.
- **Predictions and SLAs.** Considering the short-term information about workers' availability on the one hand and tasks in progress on the other hand, the platform can predict the available workforce and, thus, estimate what can be offered or guaranteed to a consumer who wants to submit a particular task.

In this paper we focus on prediction and SLA negotiation in scheduled crowdsourcing. As mentioned above, SLAs provide an additional value for services. However, when an SLA is negotiated with a customer, the platform has to make sure that this SLA is feasible and will not endanger other agreements. Specifically, we address the following questions:

---

<sup>1</sup> <http://www.mturk.com/>

<sup>2</sup> <http://www.crowdfunder.com/>

<sup>3</sup> <http://www.cloudcrowd.com/>

- *How fast a task can be done?* If too many jobs are scheduled to the same period, there could be not enough available workers to withstand the workload, so some deadlines will be broken. Thus the platform should determine the *earliest deadline* which the customer could set up for his/her task such that the timely execution of other tasks is not endangered.
- *What is the expected quality of the result?* As mentioned above, different outcome quality can be expected from different workers. Thus, if a close deadline is set, then more workers must be involved, and, therefore, the average result quality could be lower than in the late deadline case, where the smaller amount of best workers would do all the jobs. Therefore, there is a trade-off between the task deadline and the resulting average quality of the task. Estimating and explicitly presenting such a trade-off to the consumer would clarify what s/he can expect when submitting a task. Moreover, it can serve as a basis for SLAs which, as mentioned before, are crucial in a service-oriented environment. To achieve this, the platform needs to predict *quality by deadline* efficiently for multiple deadlines.

The paper introduces a technique and a base algorithm for predictions in scheduled crowdsourcing environment. The general idea of the technique is to simulate the work of the platform using the prior experience data. The precision and performance of the approach are evaluated through experiments.

The paper is organized as follows: Section 2 describes the environment and the platform model. The approach and the algorithm are presented in Sec. 3 and evaluated in Sec. 4. Section 5 discusses the related work. In Sec. 6, we shed light on some disputable aspects of our model. Section 7 concludes the paper.

## 2 Scheduled Crowdsourcing

This section describes a platform model that supports scheduled crowdsourcing [12]. The platform receives tasks from consumers and distributes these tasks for execution to the crowd. A task comprises manifold similar independent jobs which can be assigned to workers. When a job is done, the result is returned to the consumer, which is invited to provide a quality feedback on this result (see Fig. 1).

Before a consumer submits a task, an SLA for this task is negotiated. The SLA includes temporal and quality requirements. Figure 2 depicts the process of negotiating the SLA. At first, the consumer provides the parameters and skill requirements of the task to the platform. The parameters include the time of expected submission, the amount of jobs, and the job duration. Secondly, the platform estimates possible options regarding the processing time and the average outcome quality considering the status of the crowd and other active tasks or scheduled tasks. After that, the consumer decides, which option is the most suitable, and, finally, the agreement is established. The platform takes this agreement into consideration when negotiating other agreements and scheduling the tasks. If the consumer doesn't have the actual task contents at the moment, but is certain to provide it in the near future and knows the parameters of the task, then the SLA can be negotiated in advance of the actual submission.

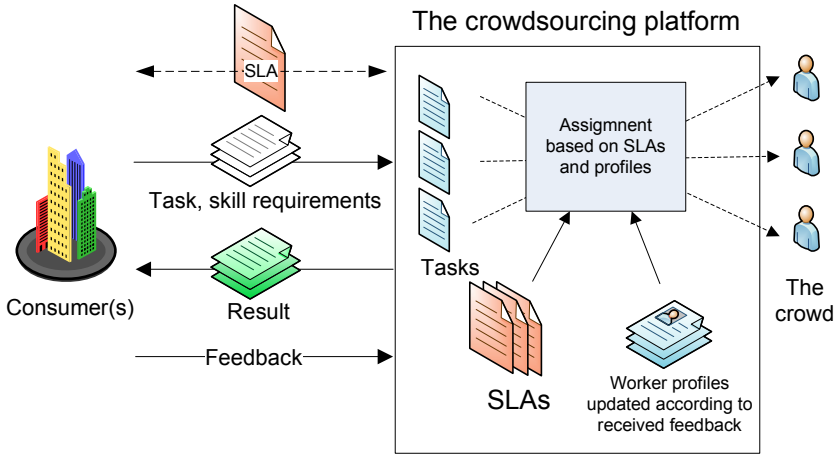


Fig. 1. Scheduled crowdsourcing

Active jobs are assigned to workers according to worker-task suitability, negotiated SLAs, and short-term availability information provided by workers. After all jobs of a task are done, the average outcome quality is calculated for the task. If it does not exceed the guaranteed quality, then the provider might incur penalties towards the consumer. If an assignment is refused by a worker despite his claim for availability, various penalty sanctions can be imposed to this worker.

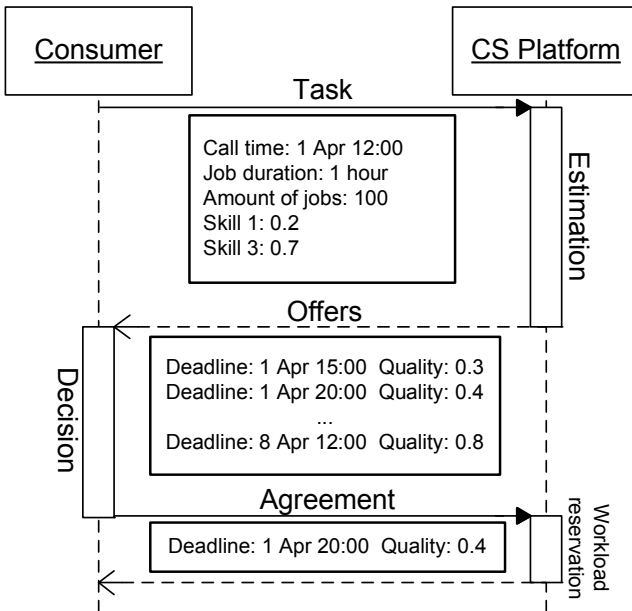


Fig. 2. SLA negotiation

The suitability is calculated as a match between required skills for the task and the skills of a worker. Skills of workers are maintained in their profiles. Initially, skill information is provided by the workers themselves. However, the profile of a worker can be modified by the platform if the expected quality (suitability) differs from the real quality that was reported by the consumer as a feedback. The maintenance of workers' skills is performed in the platform by analyzing the feedback and trying to keep the skills in the profile aligned with the real skills of the worker. A skill maintenance and update approach was considered in detail in [12], and therefore is out of this paper's focus. Moreover, the rules for calculating the suitability are independent of the scheduling and prediction logic. The suitability is represented by a single real value in  $[0, 1]$  (0 - not suitable at all, 1 - perfectly suitable) which summarizes the expectations regarding the quality of the result. This assertion completely decouples the technique, which is used to calculate the suitability, from scheduling and prediction algorithms. Therefore, the architecture is compatible with other skill and suitability assessment approaches, such as in [15] or [10].

### 3 Prediction and Estimation

In this section we propose a mechanism for earliest deadline estimation and average quality by deadline predictions in scheduled crowdsourcing. The general idea of our approach is to actually simulate the work of the platform using the statistical data about workers' behaviour and availability, considering earlier submitted or negotiated tasks. Then it is possible to predict which and how many workers might be available for the task which is being negotiated. As was shown in [12], the greedy assignment algorithm is generally good enough for scheduling in crowdsourcing environments due to a large number of available workers. Therefore, due to the linear complexity of the algorithm, the simulation can perform near real-time.

To achieve a verisimilar simulation, the simulated environment should mimic the real conditions. From the scheduling perspective, the following characteristics are important:

- **Worker's availability.** Although it is impossible to predict whether a particular worker is available at particular time, the approximate availability of each worker can be predicted from the history and the reported short-time availability. In our implementation, the availability of each worker is generated randomly for the simulated period based on his/her recent schedule.
- **Job duration accuracy.** The time that a worker needs to finish a job can differ from the specified job duration. The reasons can be an inaccurate estimation of the job duration from the consumer, the slow speed of the worker, or the difficulty of the particular job. We discuss this issue further in Sec. 6. The accuracy for already submitted tasks can be estimated based on prior assignments of these tasks. We estimate the overall job accuracy in our simulation.



- **Suitability.** As mentioned in Sec. 2, worker-task suitability is calculated by the platform and can be modified with time. If a task of a kind is submitted to the platform for the first time, the suitability can be calculated imprecisely. However, the following task submissions of this kind (with the same skill requirements) will use refined values. We discuss this issue further in Sec. 6. In any case, the simulation can only make use of the latest calculated suitability values and assume the quality of a job equal to the suitability of the performer and the corresponding task.

A deterministic time model is used in the simulator, so the time is discrete and is represented by sequential equally long time periods. A time period can represent, e.g., an hour. Quality is described by a real number in  $[0, 1]$  (0 - lowest quality, 1 - perfect quality).

The simulation period starts with the current state of the real platform. The size of the period can be either fixed (e.g., predictions for up to 10 days) or depend on the quality increment (e.g., if by prolonging the deadline by 1 day the expected quality is increased by less than 0.05, then stop the prediction). Durations of jobs are simulated according to the estimated accuracy.

At each step, the submitted (or pre-submitted) tasks are assigned to workers using the greedy scheduling algorithm (see Alg. 1), the objective of which is to maximize the overall quality, while fulfilling deadlines. The parameters of a task include the time of expected submission, amount of jobs, and job duration. The algorithm iterates over tasks in the order of times their SLAs were agreed. For each task, it tries to assign best suitable workers, so that each *task duration* an equal amount of assignments is performed. If the deadline is less than 2 *task durations* away, then all the jobs of the task are assigned. The assignment is interrupted if no more workers are available.

After conducting the assignment, the prediction algorithm is executed for the current step (see Alg. 2): workers, that are available and were not assigned by the scheduling algorithm, are examined as candidates for the negotiated task *nTask*.

---

#### Algorithm 1. Greedy scheduling algorithm.

---

**Require:** *currentTime* current time

**Require:** *tasks* active tasks

```

1: for task ∈ tasks in the order of ascending task.agreementTime do
2:   stepsToDeadline = (task.deadline - currentTime + 1) / task.duration - 1
3:   if stepsToDeadline > 0 then
4:     if (task.deadline - currentTime + 1) % task.duration > 0 then
5:       toTake = 0
6:     else
7:       toTake = Trunc(task.numberOfWorksToDo / stepsToDeadline)
8:     end if
9:   else
10:    toTake = task.numberOfWorksToDo
11:  end if
12:  while toTake > 0 AND some workers are still available do
13:    Assign a job of task to most suitable available worker
14:    toTake = toTake - 1
15:  end while
16: end for

```

---

This is performed for each job duration period of `nTask` using so-called array of average suitability of best workers (`avgSuit`). The  $k$ th element of this array represents an approximated suitability value of the  $k$ th most suitable worker for all prior simulation steps. This element contains the summed suitability and the amount of workers that were considered at this position, so the average value can be calculated at each step. The algorithm thus adds the *suitability* value and increments the *amount* for each element that corresponds to an unassigned worker (lines 3-10 of Alg. 2).

After that, if the total amount of available workers at previous steps exceeds the amount of jobs in `nTask` with certain excess, the prediction of quality is calculated for the current step. The algorithm assumes that best available workers would be evenly assigned for the task (true for the greedy scheduling algorithm). Using `avgSuit` array, it estimates the expected quality produces by most suitable available workers for all previous steps using `avgSuit` array (lines 11-21 of Alg. 2). It is assumed that a worker is late with the job with probability of 0.5 (this assumption holds if the job duration is set accurately). Eventually, the average quality which represents the prediction for *quality by deadline* for the current step, as if it was the deadline, is calculated. The *earliest deadline* is the step where it was first estimated that the number of available workers at previous steps exceeds the amount of jobs.

---

**Algorithm 2.** Prediction algorithm.
 

---

```

Require: time current time
Require: nTask negotiated task
Require: avgSuit the array of average suitability of best workers
Require: ttlAvWorkers total number of available workers
Require:  $\Delta$  excess ratio (0.8 used)
1: if (time - nTask.callTime) % nTask.jobDuration == 0
   and (time > nTask.callTime) then
2:   i = 0
3:   for worker  $\in$  workers in the order of descending suitability do
4:     if worker was not assigned and is available then
5:       avgSuit[i].suitability + = suitability of worker
6:       avgSuit[i].amount + +
7:       i + +
8:       ttlAvWorkers + +
9:     end if
10:  end for
11:  if ttlAvWorkers * 0.5 *  $\Delta$  > nTask.numberOfJobs then
12:    toTake = nTask.numberOfJobs
13:    i = 0
14:    q = 0
15:    while toTake > 0 do
16:      take = Max(1, floor(Min(avgSuit[i].amount * 0.5, toTake)))
17:      toTake - = take
18:      q + = avgSuit[i].suitability * take
19:      i + +
20:    end while
21:    return {time, q / nTask.numberOfJobs}
22:  end if
23: end if

```

---

As the scheduling algorithm prioritizes tasks by submit time, no “collisions” are expected: on the one hand, the prediction doesn’t take already reserved resources into account, on the other hand, if the task is submitted, the assessed resources will not be assumed available for the tasks submitted afterwards.

## 4 Experiments

To explore the potentiality of our approach, we implemented and tested the prediction mechanism in a simulated crowdsourcing environment. Simulation of such an environment is challenging due to the lack of comprehensive statistical data in this area. Although we don’t rely on real data in our simulations, we tried our best to prognosticate the meaningful simulation parameters based on our knowledge and experience.

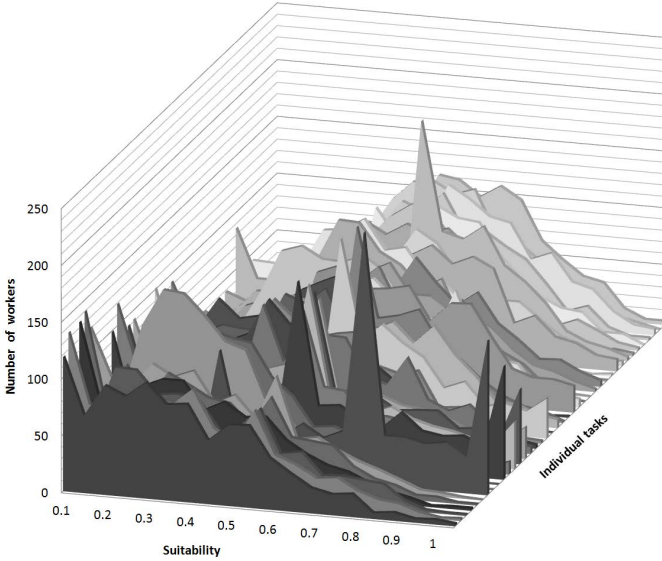
One could argue that using the simulation to predict the outcome of a simulated environment is meaningless. However, in our experiments, the prediction mechanism operates completely separately from the simulated environment. The parameters of the prediction’s simulator such as availability of workers and job duration accuracy, are estimated or generated based on the simulated environment’s prior activity only. Also, the duration of individual assignments, if those happen to take place in both simulators, would be different.

### 4.1 Setup

**Customers.** Tasks from customers are submitted randomly while ensuring the average crowd workload. At each simulation time period, if the *Task Limit* has not been reached yet, a new task is submitted to the system with *Task Concentration* probability. The job duration is calculated as  $Min(1 + abs(\phi/2 * \sigma), \sigma + 1)$ , where  $\phi$  is a normally distributed random value with mean 0 and standard deviation 1. The deadline is assigned randomly according to *Steps To Deadline* parameter. The number of jobs is calculated so that the crowd workload is near equally distributed among the tasks, and the average workload remains close to *Intended Schedule Density*. The parameters and their values are described in Table 1.

The experiments embrace diverse tasks: some can be successfully done by most of workers, some require a special set of skills, so there are only few very suitable workers, etc. Instead of explaining the generation procedure, the suitability data from experiments for some randomly selected tasks is depicted in Fig. 3.

**Crowd Workers.** The crowd size in experiments was 1000 workers (except for performance tests). This size is big enough to enclose the diversity of workers, but still allows for fast simulation. In our experiments we use a *Workers Unavailability* parameter which indicates the mean ratio of unavailable workers for each period of time (values used: 0.3 – 0.6, step 0.1). The busy periods are generated randomly, but have a continuous form which reproduces human behavior. The amount of time that takes a worker to finish the job is the *Job Duration* with injected variations. In our experiments we used values of 30-50%, which means that a job can be executed for 0.5 – 1.5 job durations.



**Fig. 3.** Suitability of the crowd for a random set of tasks

**Table 1.** Task generation parameters

Name	Description	Value(s)
<i>Tasks Limit</i>	The total number of submitted tasks	200
<i>Job Duration Sigma (<math>\sigma</math>)</i>	Describes the deviation and the maximum for job durations	20
<i>Steps To Deadline</i>	Average maximum number of jobs of a task that a single worker can finish until the deadline.	50
<i>Task Concentration</i>	The probability of new task submission for each time period.	0.35
<i>Intended Schedule Density</i>	Target assignment ratio for each time period.	0.4 - 0.7 (step 0.1)

Such circumstances as rejecting the assignment or failing to deliver are not explicitly simulated in our setting. However, inaccurate job durations and randomized worker availability partially cover these cases. For example, a situation where one worker spends too much time for a task and another worker is temporarily unavailable after that time is similar to a situation where one worker fails to deliver and another worker gets this job re-assigned.

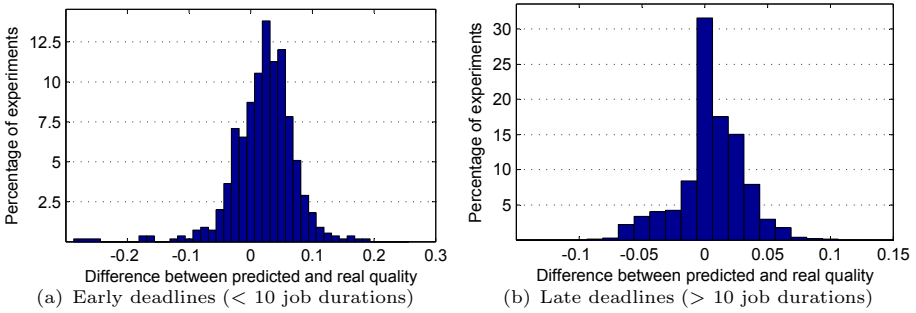
## 4.2 Experiment Types and Results

In the experiments we evaluated the prediction accuracy and the performance of the approach.

**Prediction Accuracy.** First, we made the predictions for 50 tasks in the middle of simulation for 500 time periods. It better reflects a real crowdsourcing environment, as there are both tasks being in progress and new tasks being

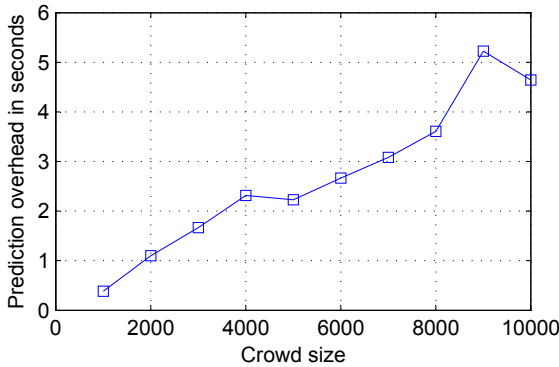
submitted. Then, we checked each prediction by varying the deadlines of corresponding tasks and running the simulation in the identical setting. The resulting accuracy is depicted in Fig. 4. From the total of 2041 experiment, in 98% the deviation was less than 0.1, in 85% of experiment - less than 0.05. The average deviation was approximately 0.025. Evidently, the prediction is less accurate for early deadlines, and more accurate for late dealines.

The results indicate that the algorithm can be successfully applied for negotiating the agreements. Moreover, the guaranteed values can be calculated depending on deadline remoteness. For example, the guarantee can be given as the predicted quality reduced by 0.2 in case of early deadlines, and reduced by 0.1 in case of late deadlines.



**Fig. 4.** Prediction accuracy. Histograms describe the amount of experiments (in percentage of the total number of experiments performed) that produced one or another accuracy. Subfigure (a) corresponds to experiments in which the deadline was set to be less than 10 job durations of the task, SLA of which is being negotiated; Subfigure (b) corresponds to experiments in which the deadline was set to be more than 10 job durations.

**Performance.** We ran the prediction in the same setting while varying the size of the crowd from 1000 to 10000. The prediction overhead is depicted in Fig. 5.



**Fig. 5.** Prediction performance

System parameters were Intel Core 2 Quad 2.40 GhZ with 6 GB of RAM (the algorithm is not parallelized so only one core was actually used).

The results show that the approach can be used in a near real-time setting. The overhead of several seconds would not play a huge role when negotiating the agreement and is therefore acceptable.

## 5 Related Work

Although the idea of QoS-enhanced crowdsourcing was discussed before [11], to the best of our knowledge, no work was devoted to deadline- and quality-centric predictions and guarantees in crowdsourcing.

In [14], semi-automatic assignment mechanism was proposed. This work assumes that SLAs are established with the workers, and Community Brokers are hold responsible for assignments in various crowd segments. However, no SLAs with crowdsourcing service consumers are considered.

Advanced market-based crowdsourcing platform which takes the suitability of workers to tasks into account was proposed in [15]. For a given task, it organizes an auction among only the most suitable workers to increase the overall quality and motivate workers to improve their skills. Again, this model doesn't provide predicting capabilities and doesn't support SLAs.

The considered scheduling problem is based on worker-task suitability, task deadlines, and workers availability with the objective of maximizing the job quality, and does not correspond to any of well-known scheduling problems [13]. The formulation can be boiled down to the problem of unrelated machines in parallel with deadlines, but the optimization objective is different from objectives related to processing time which are commonly studied in the domain.

Staff scheduling [4] designs workers' schedules which should fulfill certain requirements and cover the tasks that need to be done in a given planning horizon. Crowdsourcing's idea is the opposite: the workers define their schedule by themselves, and the platform assigns available workers to tasks in progress dynamically.

## 6 Discussion

In this section we discuss some disputable aspects of our approach.

The operation of the platform is influenced by several subjective characteristics provided by consumers, such as skill requirements, job duration, or feedback. However, it is of consumer's interest to specify them accurately. For example, if s/he underestimates job duration, then some deadlines might be missed, or the resulting quality can be lower than agreed. This situation can be spotted by the platform as the majority of workers would do the jobs longer than expected. Then, the input data from the consumer can be considered misleading, and the agreement can be denounced. If the consumer overestimates the job duration, then agreements are not endangered as most of workers would be faster than expected, however, this would produce underestimated predictions.

The same can be applied for other characteristics: if the platform spots that the majority of assignments do not correspond to expectations - then they were probably specified inaccurately. Moreover, guarantees are given according to the platform's sight, so the agreement can include the condition that the platform cannot be responsible for the outcome quality if the input data was inaccurate, and the consumer is responsible for the accuracy of his/her input. The consumer in turn can rely on the prior experience or submit some sample tasks to adjust these parameters.

Crowdsourcing presumes a substantial amount of registered workers, and the scheduled crowdsourcing puts even more restrictions on what the workers can do and when. The feasibility of real-world deployment of such a platform can thus be questioned. Some contrary arguments, however, are that about 20% of AMT workers consider AMT as their primary income source, and about 20% of AMT workers complete more than 200 jobs per week [7]. Considering that AMT claims that more than 500 000 workers are registered in the platform, one can conclude that there is a significant amount of people who are willing to perform jobs at the regular basis. Moreover, the payments for jobs in scheduled crowdsourcing can be bigger due to the added value of SLAs. Finally, the scheduling mechanism can be upgraded to take account of workers' preferences, while keeping the assignments compliant to the established agreements.

In this paper we don't discuss the cost of the work and payments in detail. Although it is an important factor, in our vision, it can be seamlessly integrated into the platform. One design solution could be that the workers specify the minimal cost for their work and the consumers would pay as much as they want as in a traditional crowdsourcing platform. Therefore, the more the customer is willing to pay, the more workers would be considered for assignment, and, as it is sensible to assume, more suitable workers could be found. However, such a design will not change the basics and the algorithms of our platform substantially. Thus, for the sake of simplicity, in this work we assumed that all the jobs cost correspondingly to their specified duration.

## 7 Conclusion

A technique and a base algorithm for predictions in scheduled crowdsourcing environment are proposed in the paper. The potentiality of the approach is evaluated thorough experiments which show that such an environment is predictive in spite of its inherent uncertainty. The proposed base algorithm can be considered rather precise (average quality deviation 0.025), and, therefore, can be applied for negotiating the agreements. The experiments show that the prediction accuracy depends on deadline remoteness, the guaranteed values therefore can be adjusted accordingly.

These results show that crowdsourcing platforms can be organized to provide quality guarantees for the consumers. They can strengthen the certainty and predictability in process planning and design, and enable Service Level Agreements with customers. From a Business Process Management perspective, such guarantees provide an additional value, thus promoting more advantageous crowdsourcing services.

In our future work, we will focus on pricing of services for scheduled crowdsourcing. Also, we foresee that a practically-applicable solution should be hybrid, i.e., support both market-like and scheduled approaches, so we plan to investigate the possible hybrid architectures and correspondent integration issues in this field.

**Acknowledgement.** This work was supported by the Vienna Science and Technology Fund (WWTF), project ICT08-032.

## References

1. Adams, C.: Managing crowdsourcing assignments. *Cutter IT Journal* 24(6), 6–11 (2011)
2. Adams, C., Ramos, I.: The past, present and future of social networking and outsourcing: impact on theory and practice. In: *UK Academy for Information Systems Conference Proceedings 2010: Information Systems: Past, Present and Looking to the Future* (2010)
3. Brabham, D.: Crowdsourcing as a model for problem solving. *Convergence: The International Journal of Research into New Media Technologies* 14(1), 75 (2008)
4. Caprara, A., Monaci, M., Toth, P.: Models and algorithms for a staff scheduling problem. *Math. Program.* 98(1-3), 445–476 (2003)
5. Doan, A., Ramakrishnan, R., Halevy, A.Y.: Crowdsourcing systems on the worldwide web. *Commun. ACM* 54, 86–96 (2011)
6. Frankova, G., Séguran, M., Gilcher, F., Trabelsi, S., Dörflinger, J., Aiello, M.: Deriving business processes with service level agreements from early requirements. *Journal of Systems and Software* 84(8), 1351–1363 (2011)
7. Ipeirotis, P.: Demographics of mechanical turk. New York University, Tech. Rep. (2010)
8. Ipeirotis, P.G.: Analyzing the Amazon Mechanical Turk Marketplace. *SSRN eLibrary* 17(2), 16–21 (2010)
9. Karnin, E.D., Walach, E., Drory, T.: Crowdsourcing in the Document Processing Practice. In: Daniel, F., Facca, F.M. (eds.) *ICWE 2010. LNCS, vol. 6385*, pp. 408–411. Springer, Heidelberg (2010)
10. Kern, R., Thies, H., Satzger, G.: Statistical Quality Control for Human-Based Electronic Services. In: Maglio, P.P., Weske, M., Yang, J., Fantinato, M. (eds.) *ICSOC 2010. LNCS, vol. 6470*, pp. 243–257. Springer, Heidelberg (2010)
11. Kern, R., Zirpins, C., Agarwal, S.: Managing Quality of Human-Based eServices. In: Feuerlicht, G., Lamersdorf, W. (eds.) *ICSOC 2008. LNCS, vol. 5472*, pp. 304–309. Springer, Heidelberg (2009)
12. Khazankin, R., Psailer, H., Schall, D., Dustdar, S.: QoS-Based Task Scheduling in Crowdsourcing Environments. In: Kappel, G., Maamar, Z., Motahari-Nezhad, H.R. (eds.) *ICSOC 2011. LNCS, vol. 7084*, pp. 297–311. Springer, Heidelberg (2011)
13. Pinedo, M.: *Scheduling: theory, algorithms, and systems*. Springer (2008)
14. Psailer, H., Skopik, F., Schall, D., Dustdar, S.: Resource and agreement management in dynamic crowdcomputing environments. In: *2011 15th IEEE International Enterprise Distributed Object Computing Conference (EDOC)*, pp. 193–202 (September 2011)
15. Satzger, B., Psailer, H., Schall, D., Dustdar, S.: Stimulating Skill Evolution in Market-Based Crowdsourcing. In: Rinderle-Ma, S., Toumani, F., Wolf, K. (eds.) *BPM 2011. LNCS, vol. 6896*, pp. 66–82. Springer, Heidelberg (2011)



# Towards Proactive Web Service Adaptation

Ahmed Moustafa and Minjie Zhang

School of Computer Science and Software Engineering,  
University of Wollongong, Gwnneville, NSW 2500, Australia  
{aase995,minjie}@uowmail.edu.au  
<http://www.uow.edu.au>

**Abstract.** With the rapid development of web service technology, next generations of web service applications need to be able to predict problems, such as potential degradation scenarios, future erroneous behaviors and deviations from expected behaviors, and move towards resolving those problems not just reactively, but even proactively, i.e., before the problems occur. Service oriented applications are thus driven by the requirements that bring the concepts of decentralization, dynamism, adaptation, and automation to an extreme. In this paper, an approach is proposed that depends mainly on the concept of proactive adaptation by the use of reinforcement learning to achieve an autonomous dynamic behavior of web service composition considering potential degradation and emergence in *QoS* values. Experimental results show the effectiveness of the proposed approach in dynamic web service composition environments.

**Keywords:** Web services, dynamic composition, proactive adaptation.

## 1 Introduction

Research in web service composition has drawn a great deal of attention in recent years. The service composition process can be distinguished as fulfilling user requirements using available web services repositories. Those requirements can be functional or non-functional. While the former is in relation to targeting the overall outcome concerning the the underway business process. The latter pertains to the quality of the composition as a whole in terms of availability, reliability, cost and response time.

Although many standards are being adopted in this domain, lack of QoS support is a long standing main challenge and an obvious drawback of current composition engines. Hence, efficient web services applications need to be managed not only from a functional perspective but also non-functional qualities should be incorporated.

The importance of service adaptation has generated a cross domain interest spanning a number of research fields. Web service adaptation is of adjusting web services to changing user requirements, QoS degradations and other prospective run time environment changes. Changes in run time environment can be in the form of QoS change, functional requirements change or environment change.

Any service composition approach should take into consideration the above forms of run time environment changes and work towards getting through those limitations. This paper mainly targets overcoming the consequent limitation of *QoS* change with an aim to provide an adaptive approach for service composition that can work in a proactive mode.

Initial attempts of *QoS* monitoring and adaptation for composite Web services focused on AI planing techniques, Genetic Algorithms and Formal Methods [12,1]. Though those methods still lack proactivity, they assume the corrective adaptation behavior to happen after a problem occurs which is not the case for many real world applications. For example, critical real time systems need to accommodate predictive proactive activity that is able to forecast the erroneous behavior and possible degradations in *QoS* before it takes place.

In this paper, we propose an approach to integrate both proactive and adaptive behavior into web service composition. The proposed approach has three contributions in terms of its abilities of prediction, incorporation and learning of web service composition attitude. Those abilities can be briefed as: the ability to predict possible downfall of the composite services quality during run time, the ability to incorporate single services and composite workflows dynamically into the running composition and the ability to learn *QoS* attributes of the undergoing services dynamically at run time. Having those qualities, the proposed approach outperforms other existing approaches of service composition and optimization that assume an abstract static composition environment which is not the case for real world applications.

The rest of this paper is organized as follows. Section 2 communicates a detailed methodology and an approach description. Experiments are conducted and results are assessed in Section 3. Section 4 gives a brief review of related work and discussions. Finally, conclusions are drawn in Section 5.

## 2 Methodology

In this section, a multi-phase approach is proposed to address the problem of web service adaptation in a composite dynamic environment. The main approach consists of three phases as follows:

### **Phase One: Modeling the Service Composition Process**

Having a user request, the approach matches the user functional requirements and generates an abstract logical workflow that is subsequently adopted by selecting the best available *QoS*.

### **Phase Two: Web Service Log Analysis**

Once deployed, the proposed approach dynamically monitors the composition via analyzing the historical data in the web service execution log.

### **Phase Three: Proactive Adaptation**

Recognizing the dynamic environment, the proposed approach adaptively learns the best execution policy and proactively takes corrective actions to keep the workflow running in a dynamic and smooth manner.

The three phases are introduced in detail in the following subsections, respectively.

## 2.1 Phase 1: Modeling the Service Composition Process

Markov Decision Process (MDP) is an AI method to model sequential decision processes and has been also used in different applications. In this phase, we employ MDP concept to model the process of service composition as it can enable to model service composition in dynamic environments as a series of steps showing stochastic nature [5,7]. We employed MDP concept in our approach to model the service composition process.

In general, a web service can be described in terms of service ID and quality of service  $QoS$ . A web service can be formally defined by Definition 1.

**Definition 1 (Web Service).** A *Web Service*  $WS$  is defined as a tuple  $WS = \langle ID, QoS \rangle$ , where  $ID$  is the identifier of the web service,  $QoS$  is the quality of the service represented by a  $n$ -tuple  $\langle Q_1; Q_2; \dots; Q_n \rangle$ , where each  $Q_i$  denotes a  $QoS$  attribute of  $WS$ .

A general Markov Decision Process (MDP) is defined as follows [5,7].

**Definition 2 (Markov Decision Process (MDP)).** A *MDP* is defined as a 4-tuple  $MDP = \langle S, A(\cdot), P, R \rangle$ , where

- $S$  is a finite set of states of the world.
- $A(s)$  is a finite set of actions depending on the current state  $s \in S$ .
- $P$  is a probability value, i.e., when an action  $a \in A$  is performed, the world makes a probabilistic transition from its current state  $s$  to a resulting state  $s'$  according to a probability distribution  $P(s' | s, a)$
- $R$  is a reward function. Similarly, when action  $a$  is performed the world makes its transition from  $s$  to  $s'$ , the composition receives a real-valued reward  $r$ , whose expected value is  $r = R(s' | s, a)$ .

A MDP involves multiple actions and paths to choose. By using this feature to model service compositions, it can enable us to integrate multiple alternative workflows and services into a single composition by simply replacing the actions in a MDP with single web services or composite workflows. Based on this consideration, we develop a MDP-based service composition model in this phase, which is formally defined by Definition 3.

**Definition 3 (A MDP-Based Web Service Composition (MDP-WSC)).** A *MDP-Based Web Service Composition* is defined as a 6-tuple,  $MDP\text{-}WSC = \langle S, s_0, S_r, A(\cdot), P, R \rangle$ , where

- $S$  is a finite set of states of the world;
- $s_0 \in S$  is the initial state and any execution of the the service composition usually starts from this state;
- $S_r \subset S$  is the set of terminal states. Upon arriving at one of those states, an execution of the service composition terminates.

- $A(s)$  is the set of web services that can be executed in state  $s \in S$ , a service  $WS$  belongs to  $A(s)$ , only if the precondition  $WS^P$  is satisfied by  $s$ .
- $P$  is the probability when a web service  $WS \in A(s)$  is invoked when the world makes a transition from its current state  $s$  to a resulting state  $s'$ , where the effect of  $WS$  is satisfied. For each  $s$ , the transition occurs with a probability  $P(s'|s, WS)$ .
- $R$  is a reward function when a web service  $WS \in A(s)$  is invoked, the environment makes a transition from  $s$  to  $s'$ , and the service consumer receives an immediate reward  $r$ , whose expected value is  $R(s'|s, WS)$ .

the above MDP-WSC model is expressive enough to describe the control flows of general service compositions. In particular, the proposed MDP-WSC model can express a super service composition which in turn is composed of multiple smaller compositions. This feature enables to incorporate single services as well as alternative composite workflows to the deployed service composition dynamically.

### 2.2 Phase 2: Web Service Log Analysis

In software applications, logging refers to the recording of activities [15,6]. Throughout the service composition course, logging can be conducted at three different levels, which are an operational level, an interaction level and a workflow level. In this research, logging at the workflow level is employed to record for the behavior of the whole workflow. Execution logs accounting for the transactions of web services that are part of an ongoing workflow are collected from various web servers. Those logs are then merged into one coherent file.

Table 1 shows a sample of log file data auditing for  $QoS$  values on a transaction by transaction basis.

A service  $WS$  is represented by the vector  $WS = \langle ID, QoS \rangle$  (refer to Definition 1), where  $ID$  is the identifier of the web service and  $QoS$  is an  $n$ -tuple  $\langle Q_1; Q_2; \dots; Q_n \rangle$ , which represents the case of multiple attributed  $QoS$ . For every transaction  $t \in WS$ , an aggregated value for  $QoS$  attributes is calculated as follows:

$$G(t) = \sum w \times \frac{Q_i^t - Q_i^{min}}{Q_i^{max} - Q_i^{min}} \tag{1}$$

where  $Q_i^t$  represents the observed value of the  $i$ th attribute of a transaction  $t$ .  $Q_i^{max}$  and  $Q_i^{min}$  represent the maximum and minimum values of  $Q_i$  for all

**Table 1.** Composition Transactions Log

Service	Transaction	Time	Availability	Reliability	$G(t)$
$WS_1$	$t_1$	10:40:39	0.92	0.4	0.73
$WS_2$	$t_1$	10:40:44	0.93	0.88	0.9
$WS_1$	$t_2$	10:40:50	0.94	0.77	0.83
$WS_2$	$t_2$	10:40:54	0.9	0.7	0.79
$WS_2$	$t_3$	10:41:01	0.8	0.92	0.83
$WS_2$	$t_4$	10:41:02	0.76	.95	0.85
$WS_1$	$t_3$	10:41:03	0.96	0.99	0.97
$WS_1$	$t_4$	10:41:09	0.99	0.45	0.74
$WS_2$	$t_5$	10:40:57	0.99	0.91	0.95

transactions recorded within the same service respectively.  $w$  is the weighting factor of  $Q_i$ .  $w$  is positive if users prefer  $Q_i$  to be high, e.g. availability and reliability. Respectively,  $w$  is negative if users prefer  $Q_i$  to be low, e.g. service fee and response time. Calculated  $G(t)$  values are then used as a basis for the rest of the approach.

### 2.3 Phase 3: Proactive Adaptation

In this phase, we use Q-learning [16] to address adaptation in web service environments. Q-learning is one of the methods in reinforcement learning. The reason for us to employ Q-learning in our adaptation phase is that (1) Q-learning is considered an effective strategy for learning optimal decisions during run time dynamic environments showing stochastic nature [11,10] and (2) Q-learning does not require pre-training process which fits the situations of service oriented domains.

By using Q-learning inside our composition engine, it enables learning the optimal policy adaptively. Considering the MDP-WSC model introduced previously, the task of the service composition engine becomes distinguishing the optimal workflow that offers the best cumulative reward.

**Definition 4 (Optimal Workflow).** Let  $W$  be the set of candidate workflows. Let  $R_i^w$  be the reward associated with each Web service invocation  $WS_i$  for some workflow  $w$ . Let  $N$  be a fixed maximum number of invocations in each of the candidate workflows. The workflow  $w^*$ , that results in the maximum expected reward is called an optimal workflow.

This reward value is calculated using Q-learning [11,10] as follows:

$$Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)] \quad (2)$$

where  $s$  represents the state space for all the possible workflows the composition engine can go through to achieve a user's request,  $a$  is the action vector representing the available web services,  $r$  is the reward given by selecting particular service,  $\alpha$  is a learning rate which controls convergence and  $\gamma$  is the discount factor that reflects the learning policy. At the right side of Equation 2,  $\alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$  represents the new immediate reward gained by selecting any service belongs to  $a$  at any state belongs to  $s$ ; whereas  $Q(s, a)$  is the previously observed reward.

The learning process is conducted recursively through a sequence of episodes, in each episode the proposed approach starts from the initial state and selects a sequence of actions by following a learning policy [3,4] till reaches the goal state.

By incrementally updating the accumulative reward values gained each episode, the proposed approach can mix execution and learning at the same time. As the reward values are subject to change in a dynamic environment settings, the proposed approach can adapt to new rewards accordingly.

This feature enables the proposed approach to promote real time adaptivity to dynamic service composition at run time.

Building on its decision policy [3,4], a trade off between exploitation of previously chosen workflows and exploration of newly available ones can be decided by the proposed approach.

**Definition 5 (Decision Policy).** A decision policy  $\delta_t$ , for time  $t$ , determines a probability distribution of actions over  $A(i)$ . It is the principle that guides taking action at any state. A policy is a sequence of decision rules. A policy is denoted by  $\pi$  and takes the form  $\pi = (\delta_1, \delta_2, \dots, \delta_t, \dots)$ . Symbol  $\pi$  is used for both infinite and finite horizon problems. To obtain a finite horizon policy from an infinite horizon policy, its operation must be restricted to a finite number of time units. A policy tells how to determine actions for any time unit of the process.

To incorporate service composition proactivity, Algorithm 1 is proposed to set the reward values  $r$  as follows: for each transaction  $t_j \in WS_i$ , the the difference between the aggregated  $QoS$  values  $G(t_j)$  (refer to Equation 1) and the mean of the aggregated  $QoS$  values for all transactions  $n$  belonging to the same service is computed.

$$d(j) = G(t_j) - \frac{\sum_{j=1}^n G(t_j)}{n} \quad (3)$$

Then, the sum of  $d(j)$  for the last  $n$  transactions, referred to as Quality of Expectation  $QoE$ , is updated correspondingly.

$$QoE = \sum_{j=1}^n d(j) \quad (4)$$

This parameter,  $QoE$  is used as an indicator for possible upward or downward trends in  $QoS$  values of service  $WS_i$ . Low values of  $QoE$  reflect the potential degradation of  $QoS$ , while high values of  $QoE$  accommodate for the prospective  $QoE$  growth. When  $QoE$  moves to a negative value, this means that the web service has experienced a series of  $QoS$  degradations and is vulnerable, while positive  $QoE$  values ensure the web service tendency to stability.

The last  $n$  transactions are used as a dynamic sliding window to ensure  $QoE$  values keep reflecting latest service activity.

To build proactivity into the proposed approach,  $QoE$  values are used as rewards,

$$r = QoE \quad (5)$$

This empowers the proposed approach to learn the workflow with the least possible potentiality of  $QoS$  degradation, by avoiding degrading workflows and enacting emerging alternatives.

Algorithm 1 presents the proactive adaptation process described above.

---

**Algorithm 1.** Proactive Adaptation Algorithm

---

```

for each  $ws_j$  do
  for  $t_i \in ws_j$  do
     $d(j) = G(t_j) - \frac{\sum_{j=1}^n G(t_j)}{n}$ ;
     $QoE = \sum_{j=1}^n d(j)$ ;
  end for
   $r = QoE$ ;
end for

```

---

### 3 Experiment and Analysis

#### 3.1 Experiment Set Up

Two experiments are conducted based on two different considerations. The first experiment explores the optimal learning policy that the proposed approach should follow, whereas the second experiment examines the proactive and adaptive performance of the proposed approach in a highly dynamic web service composition environment. For both experiments, a number of parameters are set up as follows. To simulate a fairly complex dynamic web service composition environment, the number of states  $s$ , denoting workflows, is set to 3000 and the number of actions  $a$ , denoting services, is set to 3 per state. To ensure the highest learning efficiency, the learning rate  $\alpha$  is set to 1 (refer to Equation 2). The  $QoS$  attributes  $Q_i$  that will be measured are availability and reliability (refer to Definition 1) and their relative weight  $w$  is set to 0.5 each (refer to Equation 1). This experiment is conducted on 3.33 GHz Intel core 2 Duo PC with 3 GB of RAM.

#### 3.2 Result Analysis

The results of the two experiments are demonstrated and analyzed in details in the following subsections.

**Experiment 1.** In this experiment, the optimal learning policy is addressed through the discount factor  $\gamma$  (refer to Equation 2). To enact an effective learning policy, a trade off between exploration and exploitation should be considered when setting the value of  $\gamma$ . Low values of  $\gamma$  drive that the proposed approach is greedy to exploit the already adopted workflows with immediate rewards, where high values make the proposed approach eager to explore other workflows possibly giving higher expected rewards in the long term. The proposed approach converges to a near optimal policy once it receives the same approximate value of accumulative rewards for a number of successive episodes. Those accumulated rewards are compared episode by episode and the difference is projected against a threshold. For both experiments, this threshold value is set to 0.0001, and the number of successive episodes is set to 1000.

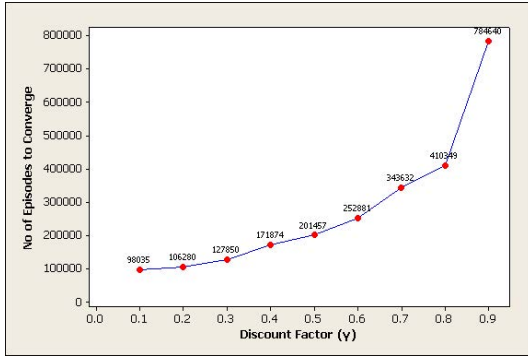


Fig. 1. Learning Policy

Table 2. Learning Policy

Discount Factor $\gamma$	No of Episodes to Converge
0.1	98035
0.2	106280
0.3	127850
0.4	171874
0.5	201457
0.6	252881
0.7	343632
0.8	410349
0.9	784640

Table 2 shows a scale of  $\gamma$  values from 0.1 to 0.9 which are implemented to the proposed approach and the number of episodes till convergence is figured, correspondingly.

Figure 1 depicts the relationship between this group of  $\gamma$  values ( $x$  axis) and the corresponding number of episodes needed to converge ( $y$  axis). The trend line shows an upward movement describing a linear relationship, with the number of episodes to converge surges for values of  $\gamma \geq 0.8$ . This means that more run time and cost incurred in considering longer term rewards. For the rest of the two experiments, the value of  $\gamma$  is chosen to be 0.8 as the authors believe that it makes a good trade off between the accumulated rewards gained in the long term and the execution time and cost.

**Experiment 2.** Experiment 2 tends to reveal how the proposed approach adaptively corresponds to the change in the composition environment. The dynamic change in the composition environment is expressed by the change in the QoS values. Those QoS values may go up or drop down for many reasons. Hence, affecting the reward values  $r$  given to the proposed approach (refer to Equations



3, 4, 5). The dynamic change in the composition environment is measured in this experiment by two factors. The first factor is the scale of change, that means how many services in the ongoing composition are exposed to QoS changes. The second factor is the frequency of this change, that means how frequently those services realize new QoS values. To recognize the former factor, the QoS values are periodically varied with 1%, 5% and 10% respectively, every 5000 episodes. Figure 2 depicts the outcome of this change in the composition environment on the performance of the proposed approach. In Figure 2,  $x$  axis represents the percentage of  $QoS$  change and  $y$  axis represents the convergence time. We can see from Table 3 that it takes 123 and 171 seconds respectively for the proposed approach to converge in a composition environment with 1% and 5% periodic scale of change. Compared to 120 seconds in a static composition environment with 0% scale of change. Those values also show the effectiveness of the proposed approach in learning the optimal policy proactively in a complex composition environment. This convergence time increases slightly to 304 seconds within 10% scale of change which is rational in such highly adaptive and a fairly complex composition environment.

To experiment the frequency of change, a 5% scale of change in  $QoS$  is selected and an experiment is conducted to explore the impact of the frequency of change on the time needed by the proposed approach to converge. The frequency of change is represented by the number of episodes the proposed approach executes before receiving new QoS values. Figure 3 depicts this impact with  $x$  axis represents the frequency of change and  $y$  axis represents the convergence time.

As in Table 4, the proposed approach spends 162 and 171 seconds respectively to converge under frequency of change as 5000 and 10000 episodes respectively. This small difference in time proves the effectiveness of the proposed approach though the frequency of change is doubled from 5000 to 10000 episodes. Finally, the proposed approach takes 230 seconds to converge under a frequency of change every 2500 episodes which is reasonable as in such highly dynamic composition environment.

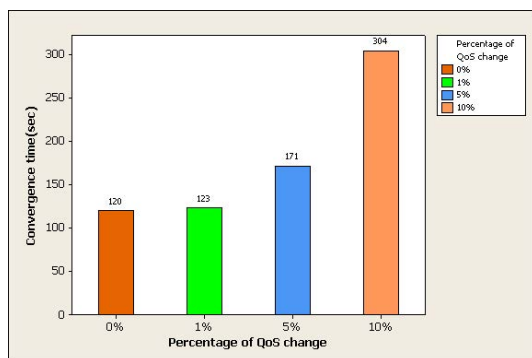
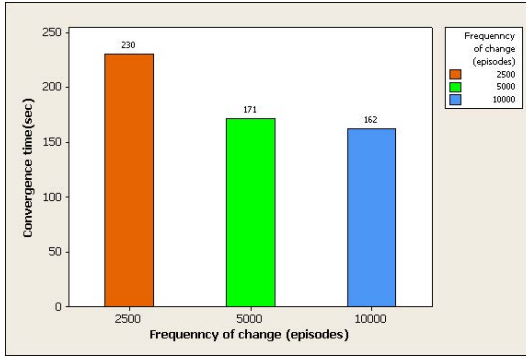


Fig. 2. Scale of Change

**Table 3.** Scale of Change

Percentage of <i>QoS</i> Change	Convergence Time(sec)
0%	120
1%	123
5%	171
10%	304

**Fig. 3.** Frequency of Change**Table 4.** Frequency of Change

Frequency of Change (episodes)	Convergence Time (sec)
2500	162
5000	171
10000	230

## 4 Related Work and Discussions

Various approaches have been proposed in literature to tackle the problem of dynamic composition of web services. In those approaches, two goals have been set for the sake of this purpose. As one point of view, a service composition should provide the functionalities required by service requesters. As another point of view, the quality of a service composition should be maximized. The previous approaches mostly address the two issues in two distinct stages.

In the first stage, a set of tools are used to create an abstract workflow of a service composition, which aims to satisfy users requirements. An example is the work of [9]. They proposed an approach to locate services conforming to independent global constraints. In this approach a greedy algorithm is used to identify conforming values and to locate composite services. Given user constraints, complying values are derived using domain rules. Also, those domain rules are used to combine services assigning those values to their restricted attributes.

It was observed that the requester and the provider both constrains play a role in composition plan generation involving sequential and parallel execution flows.

Artificial Intelligence (AI) planning methods have also been used for creating abstract workflows of service compositions. In such techniques, both the initial state and the goal state are specified in the requirement by web service requester. AI planning composition techniques employ various methods including: Declarative composition, Situation calculus, Rule based composition, Theorem proving and Graph plan based composition [14,18,12]. The search based composition methods are categorized as: forward search based composition and backward search based composition. In goal driven forward search based composition [12], the composition plan is generated in two phases. Firstly, selecting web services that match the requester's input values. Secondly, expanding the plan by matching other services inputs with the outputs of the services identified earlier. In backward search based methods, the composition plan is initialized by recognizing a suitable service that satisfies the requester's outputs. Generally, backward search is more efficient as it does not permit meaningless expansion to the composition plan. An apparent draw back of the previous approaches is that they do not capture all the features of the composition. For example, single web services cannot be directly integrated into the running composition dynamically. In contrast, our learning based approach is able to incorporate both single web services and composite workflows into the undergoing composition automatically at run time.

In the second stage, an optimal set of concrete services are selected to deploy the abstract workflow. This stage is also known as service optimization. UML based modeling techniques have been applied to Aspect Oriented Programming (AOP) technologies for service optimization. In [17], an approach is proposed to address optimization and dynamic adaptation problems. In [13], a decoupling approach based upon the separation of core service logics from context related functionalities was proposed. This separation is conducted via adopting a model driven approach based upon a modified version of the context UML meta model. At the modeling level, core service logics and context handling are treated as independent separate concerns; later AOP encapsulates context dependent behavior.

Few researchers have used the finite state machine (FSM) to model service behavior towards the automatic composition and optimization of web services. In [8], the authors worked on an augmentation based method to model services as FSMs with linear counters to integrate activity processing costs to the model. Automatic e-service optimization is proposed in [2] which presents a framework to describe the exported behavior of an e-service in terms of its possible executions trees. The framework is also specialized to consider exported behavior (i.e. the execution tree of the e-Service) that is represented by a finite state machine. However, as those approaches assume that information about *QoS* is known and up to date, they cannot deal with the real world cases where such information is unknown. A basic difference between our approach and the previous approaches is that our service composition approach does not depend on

an static workflow. In real service composition environments, the environment parameters, represented as *QoS* values, keep changing in a random and dynamic way. The *QoS* values of some services may drop down for many reasons, e.g., the service provider stops functioning. Also, The discovery of new web services with higher *QoS* values and getting them involved in the composition workflow to fulfill the requestor's complex requirements. As web services requests are inherently stochastic and web service execution environments are not deterministic, it becomes inappropriate to select and compose web services in a static way. Conversely, service composition in our approach is able to integrate multiple alternative workflows and services. Moreover, our approach does not need to know the non-functional properties of composed services a-priori. Instead, it applies reinforcement learning to derive the optimal solution by analyzing the results of execution. Therefore, our approach of service composition can be highly adaptive to a dynamic environment. Furthermore, by using *QoE* values as rewards for Q-learning, our approach is capable to adapt to dynamic changes in the composition environment proactively.

## 5 Conclusion

In this paper, the focus is to incorporate proactive and adaptive behaviors in web service composition under dynamic environments. The proposed approach uses reinforcement learning to proactively adapt to dynamic changes in complex web service composition environments. The sum of the differences of the aggregated *QoS* values is used as an accumulative reward value. The proposed approach is able to adapt proactively, not only reactively to *QoS* fluctuations considering both potential degradation and emergence of *QoS* values. The results of our experiments show the effectiveness of the proposed approach in highly complex and dynamic web service composition environments.

## References

1. Agarwal, V., Jalote, P.: From specification to adaptation: An integrated qos-driven approach for dynamic adaptation of web service compositions. In: Proc. IEEE Int Web Services (ICWS) Conf., pp. 275–282 (2010)
2. Berardi, D., Calvanese, D., De Giacomo, G., Lenzerini, M., Mecella, M.: Automatic Composition of *E*-services That Export Their Behavior. In: Orłowska, M.E., Weerawarana, S., Papazoglou, M.P., Yang, J. (eds.) ICSOC 2003. LNCS, vol. 2910, pp. 43–58. Springer, Heidelberg (2003)
3. Chen, G., Cao, W., Chen, X., Wu, M.: Multi-agent q-learning with joint state value approximation. In: Proc. 30th Chinese Control Conf (CCC), pp. 4878–4882 (2011)
4. Clausen, C., Wechsler, H.: Quad-q-learning 11(2), 279–294 (2000)
5. Doshi, P.: Dynamic workflow composition using markov decision processes. International Journal of Web Services Research 2, 1–17 (2005)
6. Gaaloul, K., Walid, Godart, C.: Log-based mining techniques applied to web service composition reengineering. Service Oriented Computing and Applications 2, 93–110 (2008), doi:10.1007/s11761-008-0023-6

7. Gao, A., Yang, D., Tang, S., Zhang, M.: Web Service Composition Using Markov Decision Processes. In: Fan, W., Wu, Z., Yang, J. (eds.) WAIM 2005. LNCS, vol. 3739, pp. 308–319. Springer, Heidelberg (2005)
8. Gerede, C.E., Ibarra, O.H., Ravikumar, B., Su, J.: Minimum-cost delegation in service composition. *Theor. Comput. Sci.* 409, 417–431 (2008)
9. Gooneratne, N., Tari, Z.: Matching independent global constraints for composite web services. In: *Proceeding of the 17th International Conference on World Wide Web, WWW 2008*, pp. 765–774. ACM, New York (2008)
10. Lee, J.W.: Stock price prediction using reinforcement learning. In: *Proceedings of IEEE International Symposium on Industrial Electronics, ISIE 2001*, vol. 1, pp. 690–695 (2001)
11. Li, H., Dagli, C.H., Enke, D.: Short-term stock market timing prediction under reinforcement learning schemes. In: *IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning, ADPRL 2007*, pp. 233–240 (April 2007)
12. Oh, S.-C., Lee, D., Kumara, S.R.T.: A comparative illustration of ai planning-based web services composition. *SIGecom Exch.* 5, 1–10 (2006)
13. Prezerakos, G.N., Tselikas, N.D., Cortese, G.: Model-driven composition of context-aware web services using contextuml and aspects. In: *IEEE International Conference on Web Services, ICWS 2007*, pp. 320–329 (July 2007)
14. Rao, J., Su, X.: A Survey of Automated Web Service Composition Methods. In: Cardoso, J., Sheth, A.P. (eds.) *SWSWPC 2004*. LNCS, vol. 3387, pp. 43–54. Springer, Heidelberg (2005)
15. Tang, R., Zou, Y.: An approach for mining web service composition patterns from execution logs. In: *Proc. IEEE Int Web Services (ICWS) Conf.*, pp. 678–679 (2010)
16. Watkins, C.: *Learning from Delayed Rewards*. PhD thesis, Cambridge University, England (1989)
17. Xu, Y., Tang, S., Xu, Y., Tang, Z.: Towards aspect oriented web service composition with uml. In: *6th IEEE/ACIS International Conference on Computer and Information Science, ICIS 2007*, pp. 279–284 (July 2007)
18. Zahoor, E., Perrin, O., Godart, C.: Rule-based semi automatic web services composition. In: *Proceedings of the 2009 Congress on Services - I*, pp. 805–812. IEEE Computer Society, Washington, DC (2009)

# Clouding Services for Linked Data Exploration

Silvana Castano, Alfio Ferrara, and Stefano Montanelli

Università degli Studi di Milano,  
DICO - Via Comelico, 39 - 20135 Milano  
{silvana.castano,alfio.ferrara,stefano.montanelli}@unimi.it

**Abstract.** Exploration of linked data aims at providing tools and techniques that enable to effectively explore a dataset through concepts, relationships, and properties by means of SPARQL endpoints and visual interfaces. In this paper, we present a set of clouding services for linked data exploration, to enable the end-user to personalize and focus her/his exploration by interactively configuring high-level conceptual structures called *inClouds*.

**Keywords:** Linked data, exploration, clouding services.

## 1 Introduction

The Linked Data paradigm promoted a new way of exposing, sharing, and connecting pieces of data, information, and knowledge on the Semantic Web, based on URIs (Universal Resource Identifier) and RDF (Resource Description Framework) [1]. However, due to the inherent flat organization of linked data repositories, the user interested in getting data about a certain search target has usually to face a multi-step and loosely-intuitive browsing activity to build a (more or less) comprehensive picture of the data of interest [5,7,13].

In this paper, we address the exploration of linked data, namely the activity aiming at providing tools and techniques that enable to effectively explore a dataset through concepts, relationships, and properties by means of SPARQL endpoints and visual interfaces. We define a set of clouding services to enable the end-user to personalize and focus her/his exploration by interactively configuring high-level conceptual structures called *inClouds*. *inClouds* and associated clouding services allow the exploration of underlying data through i) *concepts*, that are representative of sets of linked data built through similarity-based clustering and that can be browsed and dynamically reconfigured upon user request; ii) *proximity relations*, that express the existence of a similarity relationship between concepts and related clusters and that are used as navigation paths to browse and combine data pertaining to different but related concepts; iii) *ranking properties*, expressed in form of prominence values associated with concepts and proximity values associated with relations, that enable the user to intuitively focus on the most relevant concepts and their most significant relations to explore the underlying data.

After introducing the main features of *inClouds* and their life-cycle (Section 2), the paper focuses on describing the clouding services to support the initial bootstrapping and the subsequent interactive tailoring of *inClouds* for enabling more effective and focused exploration modalities (Sections 3, 4, and 5). Related work (Section 6) and concluding remarks (Section 7) finally close the paper.

## 2 Linked Data Clouds

To go beyond the flat organization of linked data repositories, in [4], we introduced the *inClouds* as a high-level, intuitive data structure capable of representing at a glance a (generally wide) collection of linked data. After briefly recalling the main features of *inClouds*, we then focus on the main contribution of the paper, namely on the definition of the *inCloud* life-cycle and associated clouding services.

### 2.1 The *inCloud* Structure

An *inCloud* originates from a set  $\mathcal{S}$  of linked data extracted from a repository  $\mathcal{R}$  (e.g., Freebase<sup>1</sup>, DBpedia<sup>2</sup>) through a combination of SPARQL queries starting from a *seed*  $s$ , that is an URI of  $\mathcal{R}$  chosen by the user as the “point of origin” for linked data extraction (see Section 2.2).

**Definition 1.** *inCloud*. Given a seed  $s$ , an *inCloud* is defined as a graph  $iC_s = (N, E)$ , where a node  $n_i \in N$  represents a *concept* with an associated *cluster* of similar linked data belonging to  $\mathcal{S}$  and an edge  $e(n_i, n_j) \in E$  represents a relation of *proximity* between  $n_i$  and  $n_j$ , denoting the fact that the two concepts and the respective clusters are somehow related.

In the *inCloud* graph, a *concept*  $n_i \in N$  is defined as  $n_i = (K_i, T_i, cl_i)$  where  $K_i$  is a set of keywords,  $T_i$  is a set of types, and  $cl_i$  is a cluster of linked data, respectively. The cluster  $cl_i$  contains a subset of the linked data in  $\mathcal{S}$  and it is built through aggregation of those linked data that are similar according to a considered matching function.  $K_i$  and  $T_i$  are defined over the linked data of  $cl_i$  by choosing the most frequently occurring terms and types in the specification of the linked data of  $cl_i$ , respectively. Each concept  $n_i \in N$  is characterized by a *prominence*  $p_i$ , which denotes the relative importance of  $n_i$  within the overall *inCloud*. The concept prominence is proportional to the number and the strength of proximity relations holding between  $n_i$  and the other concepts of the *inCloud*. The prominence affects the visual organization of the corresponding concept/cluster of the *inCloud*, in that the most prominent concepts are highlighted in foreground.

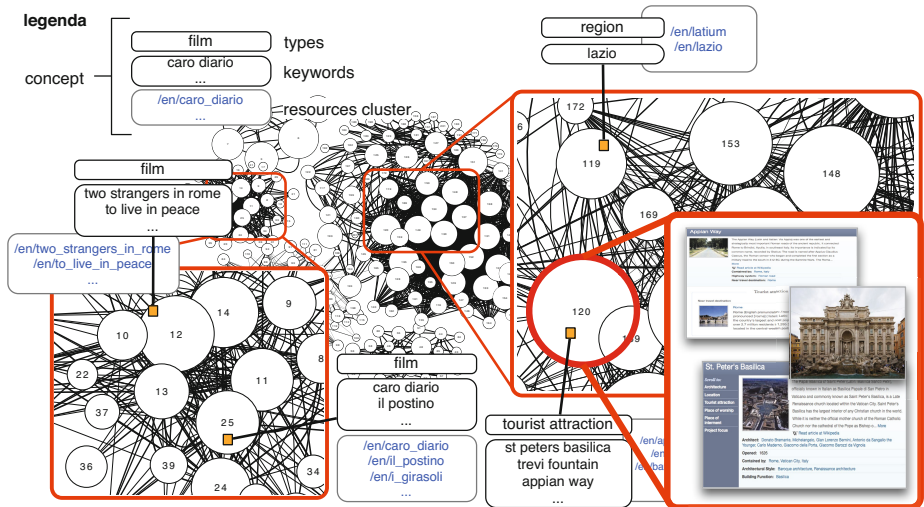
A *proximity relation*  $e(n_i, n_j)$  represents a similarity-based relationship between the concepts  $n_i$  and  $n_j$ . The nature of the proximity relation depends on the matching function employed for similarity evaluation. For instance,

<sup>1</sup> <http://www.freebase.com/>

<sup>2</sup> <http://dbpedia.org/>

when geo-spatial properties are considered to calculate linked data similarity, a proximity relation  $e(n_i, n_j)$  denotes that the concepts  $n_i$  and  $n_j$  contain a number of geographically close elements in their respective clusters  $cl_i$  and  $cl_j$ . A proximity relation  $e(n_i, n_j)$  is associated with a *degree of proximity*  $x_{ij}$  which denotes the strength of the relationship between the concepts  $n_i$  and  $n_j$ . The proximity degree  $x_{ij}$  depends on the number of elements that are similar (i.e., matching) across the clusters  $cl_i$  and  $cl_j$ , respectively. The higher the number of similar elements, the higher the proximity degree  $x_{ij}$ <sup>3</sup>. Proximity relations suggest possible exploration paths across the concepts of the *inCloud*, thus enabling a user to navigate from one concept to another following a similarity-based criterion.

*Example.* An example of *inCloud* extracted from the Freebase repository for the seed  $s = /en/italy$  is shown in Figure 1, providing concepts about Italy and some related countries in Europe (e.g., France, Germany). In the figure, we highlight some concepts clustering data about cities, regions, tourist attractions, and films. In Figure 1, keywords  $K_i$  and types  $T_i$  of a concepts  $n_i$  are represented as boxes, while a circle with an ID is used to represent the corresponding cluster  $cl_i$ . The circle size of a cluster  $cl_i$  can vary from one cluster to another, and it is proportional to the prominence  $p_i$  associated with the concept  $n_i$ . A proximity relation  $e(n_i, n_j)$  is represented as a solid line whose thickness is proportional to the degree of proximity  $x_{ij}$  holding between the concepts  $n_i$  and  $n_j$ .



**Fig. 1.** An example of *inCloud* extracted from the Freebase repository for the seed  $s = /en/italy$

<sup>3</sup> A proximity relation  $e(n_i, n_j)$  and its degree of proximity  $x_{ij}$  are defined between concepts  $n_i$  and  $n_j$  and are calculated over the matching elements of the corresponding clusters  $cl_i$  and  $cl_j$ .



For a concept  $n_i$ , a portion of the linked data resources contained in the cluster  $cl_i$  is also shown. For instance, the concept  $n_{120}$  of Figure 1 is described by the keyword set  $K_{120} = \{st\ peters\ basilica, trevi\ fountain, appian\ way\}$ , the the type set  $T_{120} = \{tourist\ attraction\}$ , and the cluster  $cl_{120}$ .

## 2.2 The inCloud Life-Cycle

An *inCloud* is characterized by a two-phase life-cycle based on *bootstrapping* and *tailoring* activities (see Figure 2).

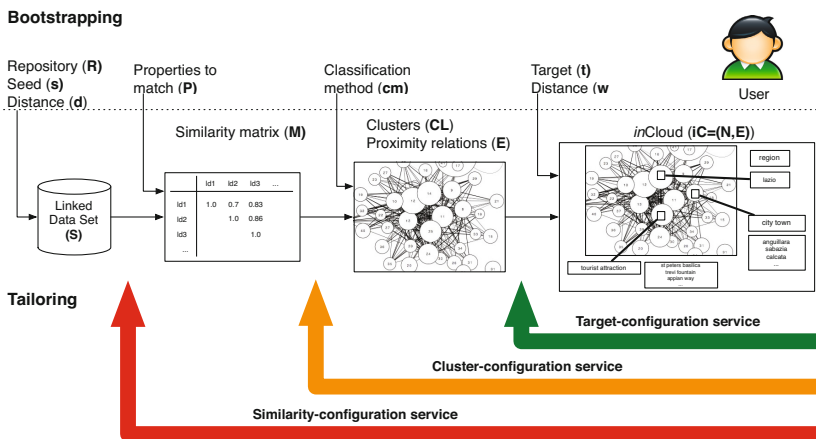


Fig. 2. The *inCloud* life-cycle

**Bootstrapping.** This phase has the goal to generate the initial *inCloud* about a given user interest. To this end, the user specifies the repository  $\mathcal{R}$  and the seed  $s$  to use for extracting the set of linked data  $\mathcal{S}$  that is the basis for the *inCloud* construction. The set  $\mathcal{S}$  is constituted by those linked data of  $\mathcal{R}$  that are pertinent to the seed  $s$ , namely those resources that are connected to  $s$  through a property path of length  $\leq d$ . The distance  $d$  is a parameter to set the extension at which linked data extraction has to be enforced and thus the size of the resulting *inCloud*<sup>4</sup>. The *inCloud* creation is articulated in three steps.

*Similarity Evaluation.* A *similarity matrix* ( $M$ ) is defined by calculating all the pairs of similar linked data resources in the set  $\mathcal{S}$ . An entry  $M[i, j] \in (0, 1]$  denotes the similarity value between the  $i$ -th and the  $j$ -th linked data resources in  $\mathcal{S}$ . Techniques for linked data matching based on property names and values are employed for calculation of the similarity values of  $M$ . By default, all the properties of the linked data resources in  $\mathcal{S}$  are considered for matching.

<sup>4</sup> More details about linked data extraction from a repository  $\mathcal{R}$  are provided in [4].

*Cluster Definition.* A *classification method* ( $cm$ ) is employed to generate a set of similarity-based clusters  $CL = \{cl_1, \dots, cl_k\}$  out of the matrix  $M$ . Different classification methods can be chosen by the user. In particular, we enforce classification methods that allow the insertion of a linked data resource in a single cluster only (i.e., *single-partitioning*), the default one, and those supporting the insertion in multiple clusters (i.e., *multiple-partitioning*).

*Concept Abstraction.* A set of concepts  $N$  and a set of proximity links  $E$  featuring the *inCloud* are finally generated out of similarity clusters. In particular, a concept  $n_i \in N$  is defined by extracting a set of keywords  $K_i$  and a set of types  $T_i$  from the linked data contained in a cluster  $cl_i \in CL$ . Prominence value  $p_i$  and degree of proximity  $x_{ij}$  are then calculated for each concept  $n_i$ . Based on the value  $x_{ij}$ , the proximity relation  $e(n_i, n_j) \in E$  is defined for each pair of concepts  $n_i$  and  $n_j$ .

**Tailoring.** This phase has the goal to enable the user to dynamically configure the *inCloud* for adaptation and re-organization according to her/his exploration preferences. The *similarity-configuration*, *cluster-configuration*, and *target-configuration* services are defined enforcing tailoring at three different levels of deepness.

*Target-Configuration Service.* This service enables the user to re-organize the *inCloud* structure according to a specific target query of interest. Target-configuration is a “lightweight” re-organization service of the *inCloud* that works only on prominence while keeping the similarity matrix  $M$  and related clusters  $CL$  unaltered.

*Cluster-Configuration Service.* This service enables the user to change the clustering method to produce new clusters/concepts providing a different view on the underlying linked data. Cluster-configuration is a “middleweight” re-organization service, in that it allows to produce a different aggregation of data while keeping the similarity matrix  $M$  unaltered.

*Similarity-Configuration Service.* This service enables the user to change the properties to consider for evaluating the similarity of linked data, and thus to set the “dimensions” of similarity to emphasize. Similarity-configuration is a “heavyweight” re-organization service that deeply changes the structure of the *inCloud* since it directly works on the set of linked data  $\mathcal{S}$  extracted from the repository  $\mathcal{R}$ .

### 3 Target-Configuration Service

The target-configuration service makes it possible for a user to restrict the exploration of the *inCloud* according to a target query, which expresses in form of keywords a specific theme/topic, or a specific entity described by the linked data which compose the *inCloud*, such as a real-world object/person, an event, a situation, or any similar subject. For example, in our *inCloud* example of Figure [11](#),

a user may be interested in focusing on a specific city, like Rome, or either on the notion of “city”, seen as the collection of all the cities described in the cloud. The target query is processed against the *inCloud* concepts, and a new *inCloud* is produced containing only the concepts matching the target, as shown in the example of Figure 3.

Given a target query  $t$  and an *inCloud*  $iC$ , the target-configuration service produces a new *inCloud*  $iC'$  which contains one or more cut-outs of  $iC$  composed by the subset of the concepts of  $iC$  that “have to do” with  $t$ . In particular, a cut-out contains the concepts  $\overline{C}$  of  $iC$  that directly match  $t$ , and can also contain the concepts of  $iC$  at a distance  $w$  from at least one concept in  $\overline{C}$ . Moreover, concepts in the new *inCloud* are characterized by a new value of prominence for  $iC'$ .

The core functions of target-configuration are the *filtering* function and the *prominence* function.

### 3.1 The Filtering Function

In order to satisfy an interest, a user formulates her target as a keyword-based query  $t$ , composed over a controlled vocabulary of terms extracted from the sets of keywords and types associated with the concepts of the *inCloud*. The filtering function takes the actual *inCloud*, the target query  $t$  and a distance parameter  $w \geq 0$  setting the extension of the filtering and produces the new *inCloud*  $iC'$ :

$$filtering(iC, t, w) \rightarrow iC'$$

The query  $t$  is processed using standard tokenization techniques for extracting constituent terms. Each constituent term  $\bar{t}$  is matched against sets  $K_i$  and  $T_i$  of each concept  $n_i \in N$ . Each  $n_i$  containing  $\bar{t}$  in  $K_i$  and/or  $T_i$  (looking at the type names) is selected and added to the set of filtered concepts  $N' \in iC'$ .  $N'$  is then expanded by adding, for each concept  $n_i \in N'$ , all the adjacent concepts  $n_j \in N$  for which  $e(n_i, n_j) \in E$ . This last step is iterated  $w$  times. According to this procedure, the resulting *inCloud*  $iC'$  is composed by all the concepts matching the target  $t$  and all the concepts at distance  $w$  in  $iC$  from them.

### 3.2 The Prominence Function

The prominence  $p_i$  of a concept  $n_i \in iC'$  is a measure of the relevance of  $n_i$  in  $iC'$  according to the number of proximity relations holding between  $n_i$  and the other concepts in  $N'$ , as follows:

$$prominence(iC') \rightarrow [0, 1]$$

Concept prominence is computed on the basis of the random walks procedure that has been proposed in [11]. This measure is calculated by counting how often a concept  $n_i$  is traversed by a random walk between two other concepts, using proximity relations between concepts as paths in the *inCloud*. In particular, the probability of using a given proximity relation in a random walk for moving from a concept  $n_i$  to another concept  $n_j$  is proportional to the proximity degree  $x_{ij}$  between  $n_i$  and  $n_j$ . We first label each concept with a same initial small value

of prominence and then we start walking in the *inCloud* moving from a concept to the subsequent one using their proximity relation as a path. Each time a concept is visited, we augment its prominence. When more than one concept is reachable from a starting concept, we randomly choose the proximity relation to follow. The probability of each proximity relation to be chosen as a step in the random walk is proportional to its proximity degree. For the sake of simplicity, we simply stop the procedure after a pre-defined number of times. According to this approach, the concepts with the highest levels of prominence at the end of the process are those which are more connected with others through proximity relation with high degrees of proximity.

### 3.3 Example

As an example of target configuration, we suppose that the user is interested in switching her view of the *inCloud* shown in Figure 1 on the target “rome”. This configuration produces two main cut-outs in the resulting *inCloud*, one containing geographical and touristic information (Figure 3 (a)) and the other one containing movies (Figure 3 (b)). In the figure, concepts directly matching the target (i.e.,  $w = 0$ ) are highlighted, and adjacent concepts (i.e.,  $w = 1$ ) are shown as gray circles. Concepts directly matching the target are related to the city of Rome and to movies located in Rome. Adjacent concepts contain information about resources that are indirectly referred to the city of Rome, such as for example Italian villages surrounding the capital or movies similar to the one located in Rome. The new prominence values for the resulting *inCloud* show that the five concepts directly matching the target are the most prominent, since the new cloud is built starting from them. However, looking at the

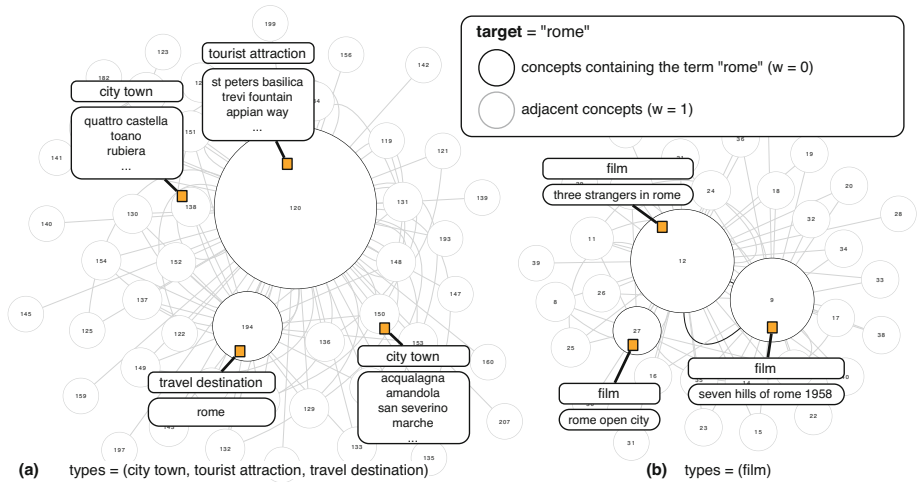


Fig. 3. An example of target-configuration of the *inCloud* of Figure 1 with target “rome”

cut-out of Figure 3 (a), it is interesting to note how the most prominent concept is the one containing touristic attractions, since it is the one containing resources associated not only with Rome but also with the other towns surrounding the capital.

### 4 Cluster-Configuration Service

The cluster-configuration service is conceived to enforce a dynamic re-organization of the *inCloud* through the use of a different classification method for linked data clustering. As an example, we consider the *inCloud* fragment of Figure 4(a) taken from Figure 1 where clusters are produced using a hierarchical agglomerative clustering. In this example, numerous and very focused clusters characterize the *inCloud*, thus enforcing a sort of “analytic” view of the underlying linked data. By invoking the cluster-configuration service, the user can switch from the *inCloud* of Figure 4(a) to the *inCloud* of Figure 4(b). The example of Figure 4(b) is built by employing clique percolation as classification method. We observe that the clusters of Figure 4(b) provide a sort of “thematic” view of the underlying linked data where a higher level of aggregation is enforced and a lower number of clusters is produced as a result.

The cluster-configuration service changes the cluster structure of the *inCloud* and it is conceived to switch from one classification method to another according to the kind of view that the user aims to enforce (i.e., analytic vs. thematic). The cluster-configuration service works on a similarity matrix  $M$  and it generates a new set of clusters  $CL'$  with new proximity relations and associated degrees of proximity.

The core functions of cluster-configuration are the *clustering* function and the *proximity* function.

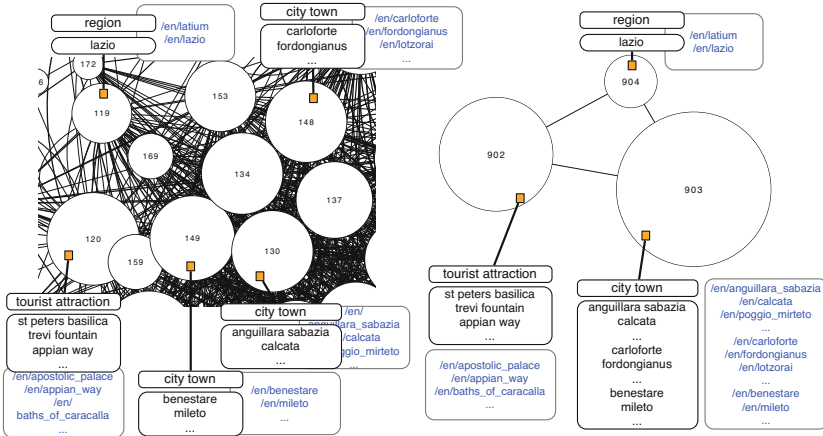


Fig. 4. An example of *inCloud* generated with a) hierarchical clustering and b) clique percolation

## 4.1 The Clustering Function

The clustering function produces a set of clusters  $CL'$  calculated by relying on a similarity matrix  $M$  through the use of the classification method  $cm$  as follows:

$$\text{clustering}(M, cm) \rightarrow CL'$$

In the literature, a number of clustering algorithms are available to be employed as classification method  $cm$ . In [10], the main existing clustering algorithms are surveyed according to the partitioning solution used for segmenting the dataset to cluster. In the following, we distinguish the possible classification methods in two different families, namely *single-partitioning* and *multiple-partitioning*.

*Single-Partitioning Classification Methods.* They are based on the idea that a linked data resource is placed in only one cluster and thus cluster overlapping is not possible. An example of single-partitioning classification method is agglomerative hierarchical clustering [3]. Hierarchical clustering works on the similarity matrix  $M$  through a series of successive merging operations of linked data resources into groups. The algorithm follows a bottom-up approach to compose a tree where each leaf corresponds to a linked data resource, while intermediate nodes represent virtual elements, (i.e., cluster “centroids”) that group the child nodes of the tree. Initially, a singleton cluster  $cl_i$  is created for each linked data resource stored in the matrix  $M$ . Then, in each successive iteration, the closest pair of clusters (i.e., the clusters with the highest similarity value in  $M$ ) are merged and inserted in the tree. The iteration terminates when no further merge operations are possible. By exploiting the cluster tree, clusters with a desired level of similarity can be selected by setting an appropriate similarity threshold  $th$ . In the example of Figure 4(a), hierarchical clustering with a similarity threshold  $th = 0.7$  has been adopted, generating a total number of 210 clusters. More technical details about similarity-based hierarchical clustering of linked data can be found in [3].

As a general remark, we note that single-partitioning methods are usually characterized by quadratic/cubic computational complexity and they tend to produce a sort of “analytic” view of the linked data in  $\mathcal{S}$  composed of small, homogeneous clusters. The readability of clusters is high due to the focused information therein contained. However, retrieving a resource within clusters becomes hard when some incorrect grouping operation has happened and a linked data resource can be misplaced into an inappropriate cluster.

*Multiple-Partitioning Classification Methods.* They allow the possible insertion of a linked data resource in more than one cluster. An example of multiple-partitioning classification method is the clique percolation method (CPM) [12]. The CPM is a clustering algorithm that works on a linked data graph  $\mathcal{S}^+$  as input for generating the set of clusters  $CL$  as output.  $\mathcal{S}^+$  corresponds to the RDF graph of the linked data in  $\mathcal{S}$  “augmented” with additional edges to directly connect the pairs of similar linked data resources that have an entry in the similarity matrix  $M$ . The CPM recognizes as a cluster a region of nodes in  $\mathcal{S}^+$  which are

more densely connected to each other than to the nodes outside the region. The CPM is based on the notion of *k-clique* which corresponds to a complete (fully-connected) sub-graph of *k* nodes within the graph  $\mathcal{S}^+$ . Two *k*-cliques are defined as *adjacent k-cliques* if they share *k*−1 nodes. The CPM determines clusters from *k*-cliques. In particular, a cluster, or more precisely, a *k-clique-cluster*, is defined as the union of all *k*-cliques that can be reached from each other through a series of adjacent *k*-cliques. In the example of Figure 4(b), the CPM has produced a total number of 26 clusters. More technical details about the CPM classification method can be found in [4].

As a general remark, we note that multiple-partitioning methods are usually characterized by exponential computational complexity and they tend to produce a sort of “thematic” view of the linked data in  $\mathcal{S}$  composed of large, varied clusters. Each cluster provides a “bird-eye” of the contained linked data and a certain resource can belong to different clusters due to the fact that a linked data can match with more than one linked data resource.

### 4.2 The Proximity Function

The proximity function produces a set of edges  $E'$  calculated by relying on a set of clusters  $CL'$  and a similarity matrix  $M$  as follows:

$$proximity(CL', M) \rightarrow E'$$

The proximity function exploits the clusters in  $CL'$  and it calculates the proximity degree  $x_{ij}$  for each pair of clusters  $cl_i, cl_j \in CL'$ . An edge is inserted in  $E'$  when a proximity degree  $x_{ij} > 0$  is found. The proximity degree  $x_{ij}$  measures the level of similarity between the contents of the clusters  $cl_i$  and  $cl_j$ . Thus,  $x_{ij}$  is proportional to the number of similar linked data resources between  $cl_i$  and  $cl_j$  according to the similarity matrix  $M$ . The proximity degree  $x_{ij}$  is calculated as follows:

$$x_{ij} = \frac{|s_{ij}|}{|cl_i|}$$

where  $s_{ij} = \{M[k, z] > 0 : k \in cl_i, z \in cl_j\}$  is the set of similar linked data between  $cl_i$  and  $cl_j$  and  $|cl_i|$  is the cardinality of the cluster  $cl_i$ , that is the number of linked data resources belonging to  $cl_i$ . The higher the number of similar linked data between  $cl_i$  and  $cl_j$ , the higher the corresponding proximity degree  $x_{ij}$ . We note that the proximity degree between two clusters  $cl_i$  and  $cl_j$  is asymmetric, denoting the fact that the cluster  $cl_i$  can be more specific than  $cl_j$  and somehow “contained” in it, or viceversa.

## 5 Similarity-Configuration Service

The similarity configuration service is invoked in order set the dimension of interest for similarity evaluation in terms of properties to be considered for matching

resources in  $\mathcal{S}$ . We call *dimension*  $\mathcal{D}$  a subset of the set of properties  $\mathcal{P}$  featuring linked data resources in  $\mathcal{S}$ . If, for example, a user is interested in the “geographical” dimension of data for similarity evaluation, she will choose only those properties representing geo-oriented attributes of resources, such as geo-location coordinates, regions, countries. The core functions of similarity configuration are the *dimensioning* function and the *similarity* function.

### 5.1 The Dimensioning Function

The dimensioning function defines the composition of the dimension  $\mathcal{D}$  by selecting the constituent properties among those in set  $\mathcal{P}$ , as follows:

$$\textit{dimensioning}(\mathcal{P}) \rightarrow \mathcal{D}$$

Dimensioning is performed by the end-user who interactively selects the properties of interest. The choice is assisted by giving the user the chance to select all the properties having one or more types of interest in  $\mathcal{S}$  as domain, in order to support a faster process of property selection. In the bootstrapping phase, a general dimension  $\mathcal{D} \equiv \mathcal{P}$  is applied to evaluate a comprehensive value of similarity. During the tailoring phase, dimensioning allows the modification of the similarity criterion by focusing on a specific subset of properties. Since the similarity matrix is calculated through matching techniques by taking into account only the properties in  $\mathcal{D}$ , it happens that the property choice affects the criterion used in order to consider two resources to be similar. For example, in case of geographical information, the notion of “similar” will be intended as “near”, since similar resources will be the ones having similar values for their geographic properties.

### 5.2 The Similarity Function

The similarity function returns a similarity matrix  $M'$  starting from the linked data resources in  $\mathcal{S}$  by applying the dimension  $\mathcal{D}$ , as follows:

$$\textit{similarity}(\mathcal{S}, \mathcal{D}) \rightarrow M'$$

$M'$  is calculated through linked data matching techniques working on the set of linked data properties specified in  $\mathcal{D}$ . In order to implement *similarity*, we rely on the matching library of HMatch 2.0 matching system, where a wide set of matching functions are available to accommodate different matching requirements and cases. In particular, given two resources  $r_i, r_j \in \mathcal{S}$ , their similarity coefficient  $M'[i, j]$  in  $M'$  is computed as follows:

$$M'[i, j] = \frac{\sum_{k=0}^{|\mathcal{D}|} \textit{match}(p_k(r_i), p_k(r_j))}{|\mathcal{D}|}$$

where  $p_k(r_i)$  denotes the value of the property  $p_k \in \mathcal{D}$  for the resource  $r_i$ . The function *match* returns 0 if one of the two resources has no values for the



property at hand; otherwise, *match* returns a similarity degree between the two property values in form of a coefficient in the range [0,1]. The similarity degree between property values is calculated by exploiting different matching functions depending on the type of the property values at hand. In particular, if we are matching dates or numbers, we rely on specific matching functions which measure the distance between the values at hand, in such a way that a higher similarity degree corresponds to a smaller distance between the dates/numbers that are matched. Instead, in case of textual values, conventional, state-of-the-art string matching functions like I-Sub, Q-Gram, Edit-Distance, and Jaro-Winkler are available in HMatch 2.0.

### 5.3 Example

As an example of how the choice of a dimension may change the resulting similarity matrix and, as a consequence, the resulting concepts/clusters in the corresponding *inCloud*, we take into account four resources, namely the URIs */en/italy*, */en/france*, */en/spain*, and */en/united\_kingdom* representing the countries Italy, France, Spain, and UK in Freebase, respectively. A portion of the properties and corresponding values of these resources are shown in Figure 5.

At the bootstrapping, the degree of similarity between the four countries is quite low (i.e., not exceeding 0.42), since countries are matched according to the general dimension based on all their properties. In fact, in spite of the presence of some properties with similar or equal values, there are many country-specific properties and values that are different from one country to the other, such as for example the historical events and the local products (in the example we reported the case of beers produced in each country). The picture changes if we configure more specific dimensions such as the geographical dimension  $\mathcal{D} = \{\text{area, containedby, time\_zones}\}$  and the political dimension  $\mathcal{D}' = \{\text{org\_founded, form\_of\_government, currency\_used, official\_language}\}$ . As we can see in Figure 6, the

property	<i>/en/italy</i>	<i>/en/france</i>	<i>/en/spain</i>	<i>/en/united_kingdom</i>
area →	301338.0	674843.0	504030.0	244820.0
containedby →	europe	europe	europe	europe
containedby →	southern_europe	western_europe	southern_europe	western_europe
time_zones →	CET	CET	CET	GMT
events →	social_war	july_revolution	first_carlist_war	battle_of_britain
events →	...	...	...	...
beers_from_here →	peroni	fischer_tradition	cerveza_reina	greens_discovery
beers_from_here →	...	...	...	...
form_of_government →	republic	republic	monarchy	monarchy
org_founded →	council_of_europe	council_of_europe	-	council_of_europe
currency_used →	euro	euro	euro	uk_pound
official_language →	italian_language	french_language	spanish_language	english_language
official_language →	-	-	catalan_language	-

**Fig. 5.** A portion of properties and corresponding values featuring the resources */en/italy*, */en/france*, */en/spain*, and */en/united\_kingdom* in Freebase

Geographical dimension				
	/en/italy	/en/france	/en/spain	/en/united_kingdom
/en/italy	1.0			
/en/france	0.81	1.0		
/en/spain	0.9	0.89	1.0	
/en/united_kingdom	0.74	0.67	0.66	1.0

Political dimension				
	/en/italy	/en/france	/en/spain	/en/united_kingdom
/en/italy	1.0			
/en/france	0.91	1.0		
/en/spain	0.62	0.58	1.0	
/en/united_kingdom	0.57	0.6	0.7	1.0

Fig. 6. Example of similarity matrices for different dimensions

similarity between countries of the example changes if we adopt the geographical or the political dimension, respectively. From a geographic point of view, the most similar countries are Italy, Spain and France, and thus we can define two clusters of countries containing Italy, Spain and France on one side, and United Kingdom on the other side. This result is mainly caused by the fact that the three countries have a similar dimension and adopt the same time zone. According to the political dimension, we note that Italy and France are the most similar countries and this would result in two different clusters: one containing Italy and France and the other one containing Spain and United Kingdom, respectively.

## 6 Related Work

Work more strictly related to our approach is focused on improving retrieval, search, and exploration of data belonging to the Linked Data cloud [28]. In this context, some tools are recently being appearing, like for example Parallax (<http://www.freebase.com/labs/parallax/>), gFacet (<http://www.visualdataweb.org/gfacet.php>), Sigma (<http://sig.ma/>), and Microsoft Pivot (<http://www.microsoft.com/silverlight/pivotviewer/>). Mainly, the idea of these tools is to work on the presentation aspects of the Web of Data and to provide functionalities for smart browsing in form of visual interfaces based on graphs, mashups, and histograms. In a similar direction, structured and collaborative search engines are being emerging as a promising solution for presenting query results in a sort of structured form with the aim at focusing on understanding the user information need. Examples in this field are Wolfram Alpha (<http://www.wolframalpha.com>) and YAGO2 (<http://www.mpi-inf.mpg.de/yago-naga/yago>).

Other work related to *inClouds* includes approaches for linked data organization and presentation. Examples of solutions in this respect are [69], where tools for exploration of DBpedia and Freebase are presented, not only via directed RDF links, but also via newly-discovered knowledge associations and visual navigation paths. Aggregation techniques are proposed to combine related topics in unified nodes, providing also a textual description of each node. In other

approaches, like Marbles (<http://www5.wiwiss.fu-berlin.de/marbles>) and LESS (<http://less.aksw.org>), information about resources of interest is presented exploiting HTML and RSS and by using different colors to distinguish sources. In comparison with recent approaches to graph summarization [14], we stress that *inClouds* do not aim at providing efficient graph compression techniques for visualization of a potentially large dataset. Instead, the ultimate goal of *inClouds* is similarity-based aggregating of linked data resources for exploration purposes. Summarization and compression are interesting side effects of our clustering methods that could be integrated in our clouding services for improving visualization in case of large linked data sets to consider.

*Main Contribution of the Proposed Approach.* The main contribution of our clouding services for linked data exploration regards the definition of techniques to move from the flat and static organization of linked data to a high-level and dynamic thematic view. This new organization of data makes it possible the definition of powerful services for linked data exploration as well as new paradigms for linked data presentation. We introduce an intuitive visualization of linked data in terms of concepts, which synthesize the contents of thematic clusters. Moreover, we define a dynamic environment where *inClouds* are not seen as a static data structures, but where they can be dynamically (re-)configured during their life-cycle by the user choosing different techniques for matching, clustering and filtering of contents.

## 7 Concluding Remarks

In the paper, we defined clouding services for smart and effective exploration of linked data through interactive configuration of *inClouds*. Besides aggregation-based visualization of linked data, clouding services aim at providing the essential functionalities for allowing the end-user to dynamically change the view over a certain linked data set of interest according to her/his personal preferences. In this sense, target-, cluster-, and similarity-configuration services are to be seen as a sort of *dashboard*, enabling the user to take control of *inClouds* and thus of the underlying data through powerful and intuitive re-organization/manipulation tools. Ongoing work is devoted to the development of the proposed services in our prototype for *inCloud* construction and management (<http://islab.dico.unimi.it/inCloud/>). Some experimental results have already been collected by focusing on evaluation of *inClouds* with respect to matching/clustering accuracy and user-perceived quality of data cloud organization [3]. In particular, we presented an experiment run with a group of 18 students of the Databases course of the Master Degree in Computer Science held at the University of Milan. The students had a similar background on Linked Data and Semantic Web, mainly based on some classes delivered on these topics in the course. Students were required to work on three test cases corresponding to different kinds of initial seeds of interest and *inClouds* involving different datasources, including Freebase and DBpedia. In particular, we asked each student to compare *inClouds* with respect

to conventional web tools for accessing the linked data contents, such as the web interfaces of Freebase and Wikipedia. The main goal of the experiment was to collect a feedback concerning the effectiveness and advantages of our approach for linked data exploration. The answers were positive. For about the 75% of the users, *inClouds* provide relevant and sufficient information about the data of interest and the perceived quality of the thematic organization is generally good. Moreover, the majority of the involved students reported that *inClouds* provide an advantage in terms of effectiveness and usability with respect to conventional web tools for linked data exploration. A more specific evaluation of dimension-based configuration is being conducted using a method analogous to the one used for the *inCloud* quality evaluation. However, the *inCloud* quality evaluation results are promising also for the usefulness of dimension-based matching and re-configuration of *inClouds*. In fact, from the experiments we got the suggestions of providing mechanisms to interactively focus the *inClouds* organization according to different user needs (i.e., matching dimensions).

## References

1. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data - The Story So Far. *Int. Journal on Semantic Web and Information Systems* 5(3) (2009)
2. Bozzon, A., Brambilla, M., Valle, E.D., Pasini, C.: A conceptual framework for linked data exploration. In: *Proc. of the 1st ICWE Int. Workshop on Search, Exploration and Navigation of Web Data Sources (ExploreWeb 2011)*, Paphos, Cyprus (2011)
3. Castano, S., Ferrara, A., Montanelli, S.: Structured Data Clouding across Multiple Webs. *Information Systems* (2011) (to appear)
4. Castano, S., Ferrara, A., Montanelli, S.: Thematic Exploration of Linked Data. In: *Proc. of the 1st VLDB Int. Workshop on Searching and Integrating New Web Data Sources (VLDS 2011)*, Seattle, USA (2011)
5. Davies, S., Hatfield, J., Donaher, C., Zeitz, J.: User Interface Design Considerations for Linked Data Authoring Environments. In: *Proc. of the WWW Int. Workshop on Linked Data on the Web (LDOW 2010)*, Raleigh, NC, USA (2010)
6. Hirsch, C., et al.: Interactive Visualization Tools for Exploring the Semantic Graph of Large Knowledge Spaces. In: *Proc. of the IUI Int. Workshop on Visual Interfaces to the Social and the Semantic Web*, Sanibel Island, USA (2009)
7. Hogan, A., Harth, A., Passant, A., Decker, S., Polleres, A.: Weaving the Pedantic Web. In: *Proc. of the WWW Int. Workshop on Linked Data on the Web (LDOW 2010)*, Raleigh, NC, USA (2010)
8. Marchionini, G.: Exploratory Search: from Finding to Understanding. *Communications of the ACM* 49(4) (2006)
9. Mirizzi, R., Ragone, A., Di Noia, T., Di Sciascio, E.: Semantic Wonder Cloud: Exploratory Search in DBpedia. In: *Proc. of the ICWE 2nd Int. Workshop on Semantic Web Information Management (SWIM 2010)*, Vienna, Austria (2010)
10. Müller, E., Günemann, S., Färber, I., Seidl, T.: Discovering Multiple Clustering Solutions: Grouping Objects in Different Views of the Data. In: *Proc. of the 10th IEEE Int. Conference on Data Mining (ICDM 2010)*, Sydney, Australia (2010)
11. Newman, M.J.: A Measure of Betweenness Centrality based on Random Walks. *Social Networks* 27(1) (2005)

12. Palla, G., Derényi, I., Farkas, I., Vicsek, T.: Uncovering the Overlapping Community Structure of Complex Networks in Nature and Society. *Nature* 435 (2005)
13. Petrelli, D., Mazumdar, S., Dadzie, A.-S., Ciravegna, F.: Multi Visualization and Dynamic Query for Effective Exploration of Semantic Data. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) *ISWC 2009*. LNCS, vol. 5823, pp. 505–520. Springer, Heidelberg (2009)
14. Tian, Y., Hankins, R., Patel, J.: Efficient Aggregation for Graph Summarization. In: *Proc. of the 2008 ACM SIGMOD Int. Conference on Management of Data, Vancouver, Canada* (2008)

# Business Intelligence Modeling in Action: A Hospital Case Study

Daniele Barone<sup>1</sup>, Thodoros Topaloglou<sup>2</sup>, and John Mylopoulos<sup>1</sup>

<sup>1</sup> Computer Science Department, University of Toronto, Canada  
{barone, jm}@cs.toronto.edu

<sup>2</sup> Rouge Valley Health System, Toronto, Canada  
ttopaloglou@rougevalley.ca

**Abstract.** Business Intelligence (BI) projects are long and painful endeavors that employ a variety of design methodologies, inspired mostly by software engineering and project management lifecycle models. In recent BI research, new design methodologies are emerging founded on conceptual business models that capture business objectives, strategies, and more. Their claim is that they facilitate the description of the problem-at-hand, its analysis towards a solution, and the implementation of that solution. The key question explored in this work is: *Are such models actually useful to BI design practitioners?* To answer this question, we conducted an in situ empirical evaluation based on an on-going BI project for a Toronto hospital. The lessons learned from the study include: *confirmation* that the BI implementation is well-supported by models founded on business concepts; *evidence* that these models enhance communication within the project team and business stakeholders; and, *evidence* that there is a need for business modeling to capture BI requirements and, from those, derive and implement BI designs.

**Keywords:** Business Modeling, Balanced Scorecards, Performance Management, Key Performance Indicators, Business Intelligence.

## 1 Introduction

Organizations that choose to adopt Business Intelligence (BI) solutions face major risks and frequent failures [1]. Prominent among these risks is the conceptual impedance mismatch between business users and data sources. Business users think in terms of goals, action plans, initiatives, processes, risks, and metrics. Data sources contain records representing entities that might be relevant or not to the problem that business users want to address. Successful linking of the two worlds is not an easy task. Existing BI platforms are not designed for business users, and inevitably BI projects end up being IT projects that follow traditional software lifecycle models, and require professionals with the right mix of technological and business skills. Consequently, there is a pervasive risk that the end result of a BI development project may not deliver full value to business stakeholders.

We have been developing a strategic business modeling language, named BIM (“Business Intelligence Model”), intended to support the description of organizations

at a business level. In BIM, an organization is modeled in terms of its strategic *objectives* and tactical *plans* [11], along with the *situations* that can affect them. BIM also supports the notions of *indicator* to measure performance [9], and business *process* to represent “means” for the achievement of objectives, or operationalization of plans. The ultimate goal of BIM is to support creation of conceptual strategic models of an organization.

Our hypothesis is that the conceptualization of BI problems and solutions in terms of business concepts rather than technical ones, reduces the BI implementation complexity and risks mentioned above. The use of such concepts strengthens communication among BI solution designers and end users during the project planning and business requirement phases.

To test this hypothesis, we have conducted an in situ case study in a healthcare organization during BI implementation. The case study involved researchers working side-by-side with a BI development team. The goal of the research team was to shadow the implementation effort, and generate models that capture the requirements and design choices of the implementation. The study objective has been to investigate whether or not these models are actually useful in the definition of BI requirements and the design of BI solutions.

The main questions we have attempted to answer in this case study are: (i) *What is the value of BIM in a BI implementation project?* (ii) *Are the primitive concepts of BIM adequate for supporting business modeling needs?* (iii) *Who are the users of BIM?* (iv) *Can BIM be used in conjunction with state-of-practice methodologies for developing BI solutions?* (v) *How does BIM facilitate the transition from business models to database schemas, queries, and data?*

The case study offers evidence that, indeed, BIM is founded on a useful set of primitive concepts for describing BI problems, and supporting the design of BI solutions. In addition, the case study showed us the way in defining a BI-specific *requirements analysis* phase, and in defining a *hybrid approach* for designing data marts that fits one of the *de facto standard* methodological frameworks for data warehousing and BI solutions.

The rest of the paper is structured as follows. Section 2 offers an overview of the project the case study is based on, and describes the business problem at hand. Section 3 amalgamates BIM concepts within a state-of-practice methodology for designing data warehousing and BI solutions, and proposes a hybrid approach for the design of a data mart. Section 4 describes our experiences in business requirements gathering, offering evidence for the usefulness of BIM concepts. Finally, Sections 5 and 6 present, respectively, lessons learned and conclusions.

## 2 Case Study

### 2.1 Background

This case study was conducted in partnership with the *Rouge Valley Health System* (RVHS), a large community hospital with two campuses in eastern greater Toronto. Three years ago, RVHS adopted *Lean* [2] as an enterprise-wide management and

quality improvement philosophy. It also established a robust performance management framework, and a corporate balanced scorecard.

Consistent with the Lean philosophy, on the technology side, last year RVHS launched two transformative IT initiatives to: (a) Create a competency center in business process management (BPM project); and, (b) Develop an enterprise BI system. The vision for the first was to enable managers and staff to document and monitor hospital processes in a visual fashion, and to create a repository with process knowledge. With processes providing the context, and the strategic plan and scorecard providing the direction, the vision of the BI system was to provide the data and the analytic tools to measure operational performance and monitor the execution of the hospital's strategic plan.

The motivation for the BI system was to use the wealth of data that the organization produces in order to uncover facts related to the quality and efficiency of its processes, the utilization of its resources, and the outcomes of its operating. The goal of the BI system is to harness this information in order to generate insights into operating performance with the goal of improving efficiencies, and the quality of patient care. Specifically, the BI initiative set out to: (a) provide an integrated repository of data from disparate sources organized to meet the information needs of business and clinical users; and, (b) offer an intuitive data access interface that enables business users to access information on their own.

The promised value of the BI system was to enable managers and staff to make facts-based decisions on their own by monitoring specific performance indicators, tracking service quality and outcomes, detecting problems, and assessing cost performance of their areas.

The end product of the BI initiative was envisioned to be a series of linked and cascading scorecards, each one containing a collection of metrics (a.k.a. indicators) measuring the performance of the internal processes. This case study is limited to one, but critical, hospital process that manages patient flow, with a focus on the Emergency Department.

## **2.2 Business Problem: Emergency Department Patient Flow**

In almost every hospital in Canada, patients go through the same steps as part of their emergency department (ED) visit. Patients arrive to the ED, and have an initial assessment by a triage nurse. During triage, the acuity of the patient condition is assessed, and a *Canadian Triage Acuity Score* (CTAS) is assigned in a scale of 1 to 5 — 1 being the most critical. The next step is the *Physician Initial Assessment* (PIA). The time to PIA can vary widely depending on CTAS, patient volume, and staff resources. A typical range is 2-4 hours. A decision to admit or discharge to home is the next step in the process. For patients who are admitted, there is often a significant delay waiting for an inpatient bed. This creates a (complex) flow problem that depends on bed capacity and turnover. When patients finish their acute care treatment but require other services, such as rehabilitation or long-term care services (that are in short supply), they end up occupying an acute service bed longer than anticipated, thereby creating a backlog in the ED.



In Canadian healthcare, this process is known to have bottlenecks leading to long wait times that impair access to care. Health authorities attempted to tackle wait-times by introducing standard indicators measuring hospital performance. At RVHS access-to-care is a strategic object which is measured by several indicators, one of which is: “Length of stay for 9 out of 10 non-admitted patients with less critical conditions (CTAS 4-5) not to exceed 3.8 hours”. This indicator is cascaded down to the ED and other departmental scorecards, and linked to a number of processes and improvement initiatives that are themselves measured by other metrics. One of the motivations of using BIM is to be able to model and document strategic objectives, operating processes, associated metrics; and the relationships between all of the above.

A specific requirement of the BI system is to support a set of metrics known as the *Daily Access Reporting Tool (DART)* indicators [13]. DART indicators are mandated by the Ontario Ministry of Health and Long-Term Care, for all hospitals participating to the Emergency Department Process Improvement Program [10], as a way to monitor improvements and reduce ED waiting times. The DART indicators are calculated based on data elements specific to the patient flow process covering both the ED visit, and the admission and discharge processes at the inpatient acute care units. The process in Figure 1 illustrates the ED process, and some of the relevant data elements.

In summary, the BI system and corresponding data warehouse must support metrics designed to manage access to care performance, which include the mandated DART indicators, as well as metrics that RVHS staff uses for real time process control. The next step is to represent these requirements by means of a conceptual model that will drive the design of a dimensional data warehouse model.

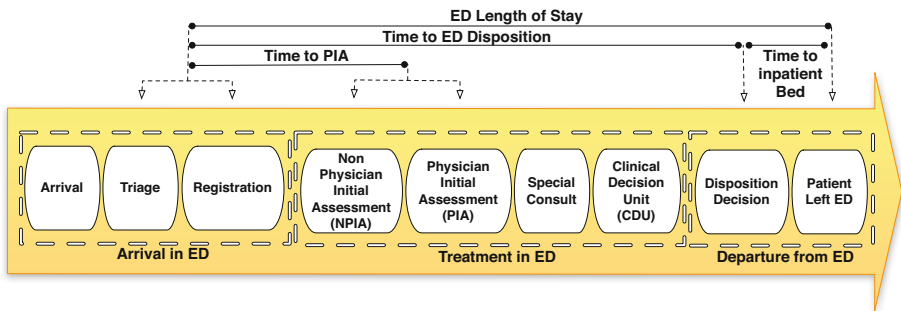


Fig. 1. The ED process with (some) performance metrics

### 3 From Business Requirements to Data Warehouse Design

At the core of a BI solution, there are one or more dimensional models (data marts) to support business level queries, may these be indicators in a report, or dashboards responding to business requirements. Although BI implementations are often considered business projects, in reality, they are mostly system implementations where the business side is mainly engaged at the business requirements definition

stage. From there and after, they are IT projects following a traditional system development lifecycle model. The Kimball Group (<http://www.kimballgroup.com/>) has proposed one of the most widely used lifecycle models for data warehouse design. Kimball's framework reinforces three fundamental concepts [3]: (a) A focus on adding business value across the enterprise; (b) Structuring data in a dimensional model; and, (c) Iterative solution development. While we are in agreement with these concepts, we see a weak point in the informal representation of business requirements, typically captured in text, that renders them difficult to analyze in the data design phase. With such a framework, the risk of communication break-ups between business and IT is high.

The vision of BIM is, like UML in software, to provide a variety of model types that are accessible to business users, and can document business requirements in a graphical and yet formal way. Business requirements now become managed metadata artifacts that act as a blueprint for the implementation and can be updated, versioned, and queried.

The focus of the case study has been to apply BIM modeling, and generate models that link business user needs with the data sources. In other words, our objective was to define the data requirements for the BI dimensional model. With this aim, we follow a seven-phase process described in [1], and briefly summarized in Table 1.

**Table 1.** The seven phases for the design of a data mart

Phase	Input	Output	Role Involved
Analysis and reconciliation of data sources	Operational source schemata	Reconciled schema	Designer, data processing center staff
Requirements analysis	Strategic goals	Requirement specifications, preliminary workload	Designer, end users
Conceptual design	Reconciled schema, requirement specification	Fact schemata	Designer, end users
Workload refinement, validation of conceptual schemata	Fact schemata, preliminary workload	Workload, data volume, validated fact schemata	Designer, end users
Logical design	Fact schemata, target logical model, workload	Logical data mart schema	Designer
Data-staging design	Source schemata, reconciled schema, logical data mart schema	ETL procedures	Designer, databases administrators
Physical design	Logical data mart schema, target DBMS, workload	Physical data mart schema	Designer

The seven phases can be combined and executed in different order, leading to different approaches [4] for data warehouse designing. According to [1], these approaches can be classified as: (a) *data-driven* (or *supply-driven*); and, (b) *requirements-driven* (or *demand-driven*). In the former, the data warehouse structure is derived from an analysis of the sources (supply) in light of user requirements. In the latter, the user requirements (demand) are turned into a data warehouse conceptual design, leaving unanswered the problem of mapping source data to the “desired” warehouse at a latter stage of the process (see *Data-staging design* phase in Table 1).

The case study BI implementation followed a hybrid approach where requirements and data source analysis were conducted concurrently. In this scenario, BIM models

proved to be an efficient and effective vehicle for hybrid design. Moreover, BIM offered strong support for formally capturing requirements during the business requirements analysis phase.

Figure 2 shows our hybrid approach, and how the seven phases are interconnected to each other through inputs and outputs.

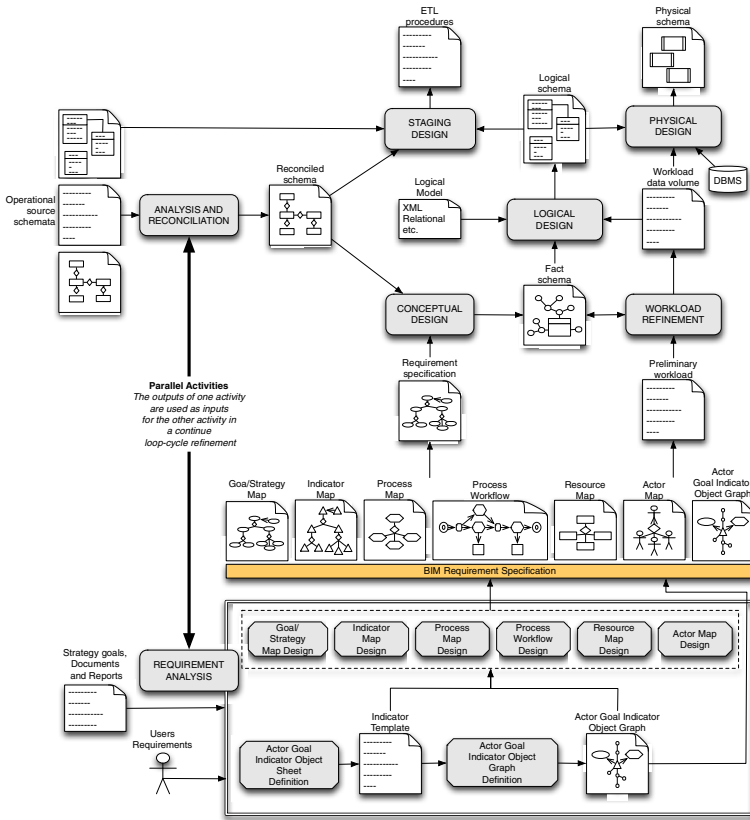


Fig. 2. The hybrid approach for the design of a data mart

A key result of our work was the integration of BIM concepts with the business requirement analysis phase. The output of this phase is a series of “maps” (models), each one representing a set of requirements covering a particular aspect of the business. This is analogous to UML’s diagrams, but adapted for BI implementations. The BIM meta-model supports interrelationships and cross-references between different maps, thereby facilitating interoperability among different business views. Section 4 will show examples.

Among the various types of maps, the *Indicator Map* is a sufficient requirement specification model for the definition of the data mart structure. However, in the case study we produced all the different requirement specification models described in Figure 2, for the following reasons: (a) Provide a comprehensive set of maps to aid

the communication of business users, subject matter experts, and database designers; (b) create reusable maps that can benefit other ongoing IT initiatives at RVHS, such as the BPM project; and, (c) test the full power of BIM.

Requirements models expressed in BIM support variability-in-design. This means that they can be adapted to the goals and needs of the business analyst / designer, allowing her to choose which and how many of the sub-phases to perform during the Requirement Analysis phase. In the following section, we provide a brief overview of the sub-phases, and the outputs we obtained for each.

## 4 BI Data Requirements for ED Activity Reporting

The business problem, as defined in Section 2.2, is to facilitate improvements in access-to-care, by reporting daily a list of metrics known as the DART report. As the specification of the DART report is given (mandated), we focus on the design of the corresponding ED data mart. We foresee scenarios where the BI report requirements are not available, or new reports and set of performance measures need to be created. Such scenarios are handled by variations of the BIM business user requirement analysis approach (see [5] for a comprehensive survey).

In the following sections, we trace the steps of the phases of BIM modeling in Figure 2 by using DART as our working example.

### 4.1 Sub-phase: Actor Goal Indicator Object Sheet Definition

The main input for this sub-phase was the DART report. Each one of the 37 DART indicators for ED and Inpatient care units represents a (business) requirement that the BI system-to-be must support. For each indicator/requirement, we documented the indicator's meta-data in the *Actor Goal Indicator Object (AGIO<sup>1</sup>) Sheet*, briefly summarized in Table 2.

We have found the AGIO sheet to be an effective instrument for focusing attention on different informational perspectives related to an indicator. AGIO Sheets were compiled through interviews [6] with domain experts. Each interview, corresponding to one sheet, had an average duration of 40 minutes, and consisted of a combination of open-ended, closed, and evidential questions [7] to the domain expert. For the 37 DART indicators, this time summed up to a six days period for requirements gathering: the period includes also the time spent for the design of the maps extrapolated from the AGIO sheets, and described in the following sections.

Table 3 shows a (partial) example of an AGIO sheet for the indicator "Percentage of ED Left Without Being Seen (LWBS) patients" (this sheet is also used in Figure 3). The right-hand column offers a sample of the questions posed to the domain experts.

---

<sup>1</sup> "Agio" is an Italian word that means "ease", intended to suggest an absence of difficulty or effort for a domain expert to communicate business requirements in a unique view.

**Table 2.** Summary of the AGIO Sheet's content

Perspectives	Information collected	Overview
<b>General Description</b>	<i>ID, Name, Description, Type, Scorecards.</i>	General information about the indicator. Notice the "Type" attribute, which defines if an indicator is <i>positive, negative, or bidirectional</i> [9].
<b>Organization Context</b>	<i>Goal, Goal responsible, Measured object.</i>	Core information to motivate an indicator, and to understand what it is measuring.
<b>Measurement</b>	<i>IT responsible, Metric, Numerator and Denominator, Sources for Numerator and Denominator, Dependencies, Frequency.</i>	Detailed information about the measurement that must be performed.
<b>Data mart and Navigability</b>	<i>Data mart name, Dimensions, Levels.</i>	Information needed to build the fact table, and to define user preferences in analyzing the events of the fact table.
<b>Performance Parameters</b>	<i>Criteria for Restriction and Aggregation, Targets, Positive and negative thresholds, Parameters motivation and interpretation.</i>	Detailed information for evaluating measurements with respect to the user performance requirements; and <i>restriction and aggregation</i> criteria [1] for grouping data for the performance parameters.
<b>Data Source Details</b>	<i>Database, Table, Field, Query, Data collection process.</i>	Information about the data sources needed to feed the fact table.
<b>Access Control</b>	<i>Confidential level</i>	Information about credential levels needed to access the indicator values.
<b>User Visualization Requirements</b>	<i>Graph Type, User, Criteria for Restriction and Aggregation</i>	User preferences in the visualization of indicator information.
<b>Information and Data Quality</b>	<i>Issues</i>	Quality issues in the data used to calculate the indicator. They give a general idea on the confidence users should have when making decisions based on the indicator.

**Table 3.** Partial AGIO sheet for "Percentage of ED LWBS patients"

General Description	Question
<i>ID</i> 7	<i>Not applicable</i>
<i>Name</i> Percentage of ED LWBS patients	What is the indicator name?
<i>Description</i> The indicator calculates the percentage of ED patients that leave the ED without being seen by the doctor.	Can you provide a high level description of the indicator?
<i>Type</i> Negative	Do you want to reduce, increase, or balance the value of the indicator?
<i>Scorecards</i> DART	Does the indicator belong to some report?
Organizational Context	Question
<i>Goal</i> Reduce the percentage of LWBS patients	Which is that goal you try to achieve monitored by the indicator?
<i>Goal Responsible</i> ED Manager	Who is the responsible of such a goal?
<i>Measured Object</i> Physician Initial Assessment	Which is the operative process or resource for which the indicator is measuring the performance of?

**4.2 Sub-phase: Actor Goal Indicator Object Graph Definition**

The objective of this sub-phase is to extrapolate from each AGIO sheet a corresponding *AGIO graph*. An AGIO graph has one objective, to provide a compact view of an indicator (a.k.a., "Story" of an indicator) by answering questions such as: *Why an indicator is necessary? What is measuring? Who is interested in its performance? Which are the perspectives that can be used to investigate its values?* AGIO graphs constitute the main input for those sub-phases that produce different BIM maps (see Figure 2).

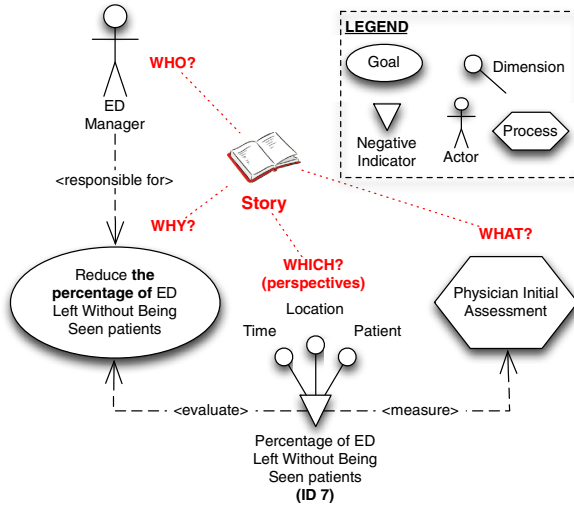


Fig. 3. The AGIO graph for the indicator “Percentage of ED LWBS patients

Figure 3 shows an example of an AGIO graph for the indicator described in Table 3. The graph provides a pictorial representation of the AGIO sheet’s requirements. Notice the orientation of the triangle symbol, with the tip towards the bottom, for representing a *negative* indicator<sup>2</sup>, i.e., an indicator whose value needs to be reduced. This is consistent with the description of the associated goal “Reduce the percentage of ED LWBS patients” defined by using the Goal-Question-Metric [8] approach, where a goal description follows the pattern *purpose, issue, object* (process or resource), and *viewpoint*. For example, for the above goal, we have that: *purpose* = reduce, *issue* = the percentage of ED LWBS, *object* = patients, *viewpoint*<sup>3</sup> = ED Manager.

### 4.3 Sub-phases: Goal/Strategy and Indicator Maps Design

Figure 4 shows a combined view of the goal/strategy map and indicator map, representing part of the strategy undertaken to improve the high level goal “Improve access to care for patients” (the topmost goal in the figure). This combination is very useful to understand how the collected measures are aligned with RVHS strategy; and how, in turn, the measures are connected to the operational layer (see Workflow Map in Section 4.5).

<sup>2</sup> For a *positive* indicator, the tip of triangle points towards the top; while, for *bidirectional* indicators, we have a side-by-side positive and negative indicators symbol. See Figure 4 for some examples, and refer to [9] for more details on these three types of indicators.

<sup>3</sup> In the AGIO graph, the viewpoint is external, and is represented by the actor responsible for the goal associated with the indicator.

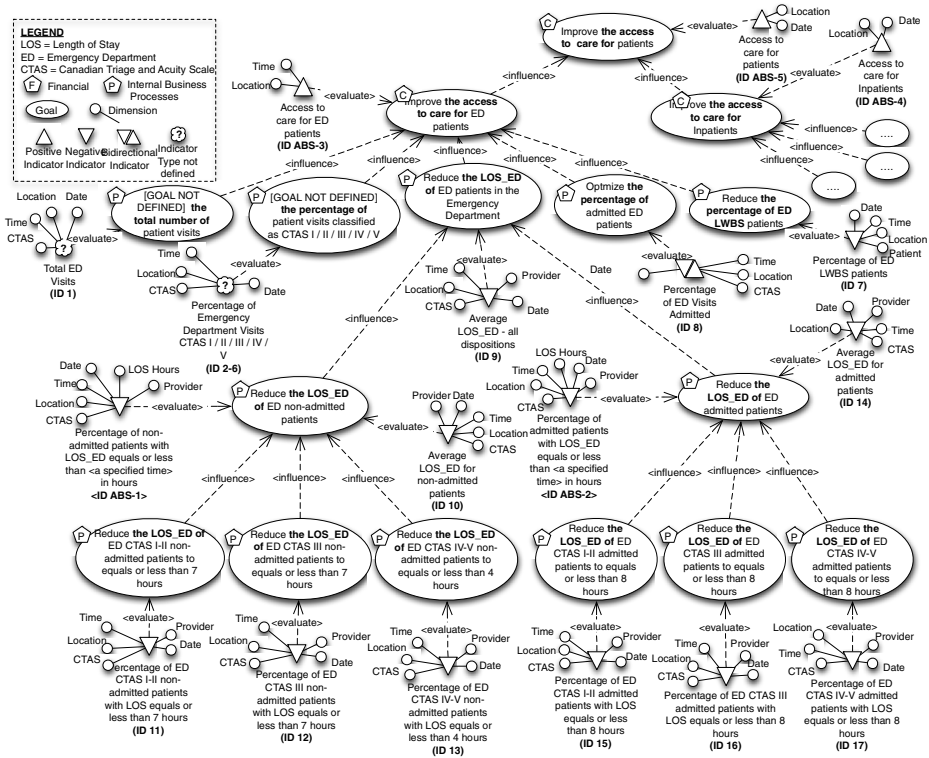


Fig. 4. Combined view of Goal/Strategy and Indicator Maps

Notice also, how the strategy map is aligned with the four perspectives of Kaplan and Norton [9].

To build a goal/strategy map, the designer starts from the set of goals collected in AGIO graphs; then, by interviewing domain experts, s/he defines influence relationships among goals, together with those high level goals not described in the AGIO graphs, e.g., the goal “Improve the access to care for patients”. In general, for the later designing task, the designer can count on external documents describing the strategy of an organization, and on interviews with the domain experts.

#### 4.4 Sub-phases: Process, Resource, and Actor Maps Design

The outputs of these sub-phases are maps that describe objects (*processes, resources, and actors*), and sub-objects involved in the RVHS operational layer. Due to space limitations, Figure 5 shows only an example of an actor map.

A careful reader may argue that common models for the representation of an organization structure, e.g., hierarchy charts, can be used in place of our Actor map. We stress the fact that BIM is a (underlying) metamodel that integrates concepts and relationships from different domains (business, requirements and software engineering, data warehouse, etc.) for designing Business Intelligence Solutions.

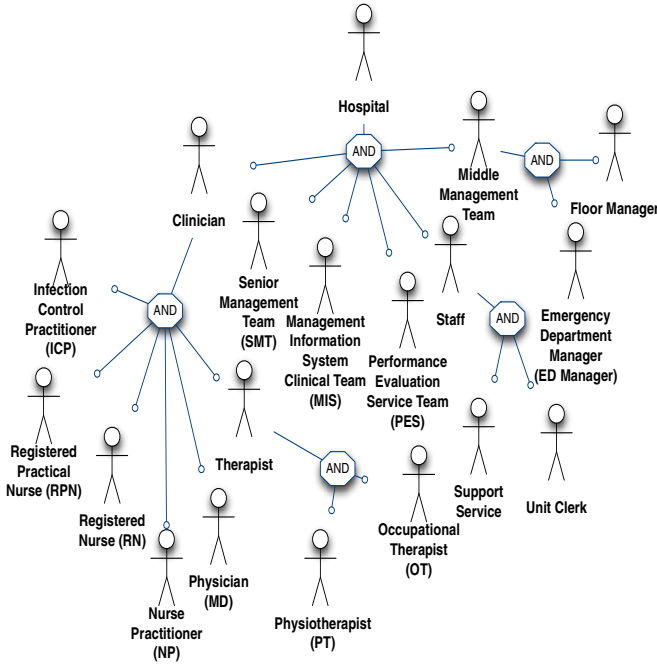


Fig. 5. RVHS’s Actor map, a fragment

For actors, BIM has to offer more than the simple view illustrated in Figure 5. As described in [9,11], different relationships can be instantiated from an actor to other concepts in the (meta)model. For example, an actor can “perform” a process, “desire” a goal, or be “responsible for” an indicator. These relationships cannot be handled by a hierarchy chart model, but can be easily added to an Actor map by relying on the BIM metamodel. Further versions of a visual editor prototype described in [9] will have collapse/explode and filters features to customize and combine BIM maps with respect to customer needs.

#### 4.5 Sub-phase: Process Workflow Design

The output of this sub-phase is an abstract representation of relevant process workflows in an organization. With this aim, the designer can start from the set of (sub-)processes defined in the AGIO graph, representing a footprint of the complete operative processes used to achieve the strategy and operational goals of an organization. The designer can then produce a complete design of the process workflows, supported by resource and process maps, as well as further interviews with domain experts. Figure 6 shows an example for the RVHS’s ED patient workflow.

Notice how indicators can be easily attached to sub-processes or tasks, thanks to the connection made in the AGIO graphs. The sub-model of BIM we used for the design of workflows attempts to be as simple as possible to be easily understood by the domain experts.





During conceptual design, a fact schema is constructed as follows<sup>4</sup>:

- 1) Extrapolate all indicators defined in the AGIO graphs, and assigning them to the *set of measures*;
- 2) Extrapolate all *dimensions* and *levels* attached to the above indicators, and assigning them to the *set of (conformed) dimensions*.
- 3) Insert in the fact schema all measures (from the sets above) that: (a) share the same conformed dimensions; (b) has the same granularity; and, (c) concern the same fact.

It must be noted that more than one fact schema may result from the previous steps. Also that information from the Workload Refinement phase (see Figure 2) may be taken into account.

A properly designed fact schema is fundamental for feeding reports and dashboards with the proper data. Figure 7.b shows a prototype of a dashboard using the fact schema.

## 5 Lessons Learned and Reflections

The case study experience suggests that the BIM approach can have a positive impact in the design and development of BI solutions. BIM addresses the impedance mismatch between business users and IT. BIM models function both as communication tool, and design blueprint. Below, we present the lessons learned from the two different perspectives of the project team: the research team and the RVHS project team.

We validated that the primitive concepts provided by BIM were sufficient to support the business requirements elicitation phase for the data mart modeling (see Figure 2).

In the course of the case study, we had the opportunity to test the application of different BIM features, and receive feedback from domain experts. This led to some refinements of BIM.

When BIM was first introduced, it was not clear “who” would be the actual users of such a model: managers? domain experts? business analyst/designer? The case study gave us some revealing answers. BIM was proven useful to designers during the phase business requirements elicitation, and helped domain experts to understand whether or not what they had in mind (in terms of requirements) was properly translated into the “to-be” BI system.

An important outcome of the case study is the definition of the hybrid data mart design approach using BIM. When we started the project, we did not have a preconceived design methodology that would utilize BIM in developing a BI solution. We decided to look at existing methodologies in BI, and extend them with BIM in order to take advantage of its novel features. In this way, the use of BIM is done within a familiar BI development framework, and we can therefore assess the

---

<sup>4</sup> [1] defines terms such as *fact*, *measure*, *granularity*, *conformed dimension*, and *level*.

improvement in communication between business users and IT. At the same time, we were able to make a first step towards the definition of a new way of doing BI based on the next generation of BI tools like BIM. The development methodology, introduced in Section 3, indicates that BIM is capable to communicate business requirements, as well as assist in the data mart design.

The feedback received by the domain experts in the project was generally positive, suggesting that BIM concepts are contributing in the definition of BI requirements. The domain experts felt comfortable with the notation, and the produced models. In many occasions, it was noted that the graphical models produced by BIM were concise representations of the domain and related business requirements; and they were found helpful in communicating requirements in a complete and accurate way to an external BI vendor. The project team appreciated the creation of the graphical BIM maps which served as a *roadmap* throughout the project. The maps supported step-wise refinement, and guided the development of the RVHS BI solution.

From the RVHS perspective, BIM added value in the BI project in the following ways. First, the AGIO sheets captured essential metadata in simple but complete form, which helped decision support and IT analysts to develop a common language and understanding of the domain. The AGIO sheets were extremely helpful in communicating indicator requirements to an external implementation partner. Furthermore, the AGIO maps helped to connect roles to goals to measures to processes in a simple and unambiguous way that further surpasses the traditional textual descriptions.

As the organization already had a performance management culture and corporate scorecard, the combined goals and indicators map nicely aligned with the scorecard. In fact, the general perception was that the goals and indicators map provided a two dimensional view of the scorecard design, with clear depiction of the 'influence' relationships.

Process maps described the RVHS operations layer accurately. RVHS had already adopted a business process modeling methodology to support continuous process improvement. Similarly to BIM, the goal has been to tag processes with indicators, and design BI dashboards to monitor process performance. The expectations of using BIM in this project were: (a) to produce and maintain metadata of the BI requirements and design that different stakeholders will use with ease; (b) reduce project complexity, and accelerate BI project timelines. The first expectation was fully met by producing a complete and accurate meta-data repository, easily consultable by the stakeholders. For the second, the feeling is that the project timeline was positively impacted, although it is difficult to separate the various factors affecting timelines. An area where timelines were positively impacted was the scorecard production where the combination of AGIO graphs, RVHS scorecard, and Performance Management Framework (PMF)<sup>5</sup> streamlined and accelerated the development process, and shortened the time.

---

<sup>5</sup> PMF is a commercial product by Information Builders [12] that RVHS licensed to produce scorecards.

Overall, the case study confirmed that BIM drastically reduced the risks discussed in [1]. Properly define requirements eliminated the risks related to data and design. Organizational risks were also lower, since end users felt more involved in the development process, and part of the final outcome: they are now able to take advantage of the results achieved. Indirectly, also the risks related to project management and risks related to technology may have been influenced (although we did not experience such risks at RVHS).

The formalized knowledge derived from the requirements analysis, such as strategy process, indicator maps etc., is useful for supporting the RVHS performance management needs.

## 6 Conclusions

We presented a case study of developing BI solutions through BIM concepts. Our modeling and methodological framework has proven adequate for the task. On the way, we extended widely practiced BI solution development techniques by enriching them with BIM concepts.

The case study suggests that BIM concepts enhance communication and collaboration between designers and domain experts; they also reduce common risk factors that a BI solution may face during its lifecycle. It also gave us inputs on simplifying BIM and fitting it to a methodology.

As for directions for further research, the case study revealed an opportunity to evolve BIM towards a model for performance management.

**Acknowledgments.** We would like to thank our colleagues, the members of the RVHS BI project team, Deepak Sharma, Andrea Gates, Thomas Thuraiiratnam, Ken Kuganesan, Yen Luong, and Patricia Moreno; and, our collaborators, Ganesh Iyer, and Steve Huang.

## References

1. Golfarelli, M., Rizzi, S.: *Data Warehouse Design: Modern Principles and Methodologies*. McGraw-Hill (2009)
2. Pettersen, J.: Defining lean production: some conceptual and practical issues. *The TQM Journal* 21(2), 127–142 (2009)
3. Kimball group: Design Tip #115 Kimball Lifecycle in a Nutshell (2009), <http://www.kimballgroup.com>
4. Winter, R., Strauch, B.: A method for demand-driven information requirements analysis in data warehousing projects. In: *Proceedings Hawaii International Conference on System Sciences*, pp. 1359–1365 (2003)
5. Golfarelli, M.: From user requirements to conceptual design in data warehouse design: A survey. In: Bellatreche, L. (ed.) *Data Warehousing Design and Advanced Engineering Applications: Methods for Complex Construction*. IGI Global (2009)
6. Kimball, R., Reeves, L., Ross, M., Thornthwaite, W.: *The Data Warehouse Lifecycle Toolkit*. John Wiley & Sons (1998)

7. Kendall, K., Kendall, J.: System analysis and Design. Prentice-Hall International (2002)
8. Basili, V.R., Caldiera, G., Rombach, D.H.: The Goal Question Metric. In: Encyclopedia of Software Engineering. Wiley (1994)
9. Barone, D., Jiang, L., Amyot, D., Mylopoulos, J.: Reasoning with Key Performance Indicators. In: Johannesson, P., Krogstie, J., Opdahl, A.L. (eds.) PoEM 2011. LNBIP, vol. 92, pp. 82–96. Springer, Heidelberg (2011)
10. Emergency Department Process Improvement Program (2011), <http://www.patientflowtoolkit.ca/>
11. Jiang, L., Barone, D., Amyot, D., Mylopoulos, J.: Strategic Models for Business Intelligence. In: Jeusfeld, M., Delcambre, L., Ling, T.-W. (eds.) ER 2011. LNCS, vol. 6998, pp. 429–439. Springer, Heidelberg (2011)
12. Butler Group: WebFOCUS Performance Management Framework v5 (2009)
13. Ontario Ministry of Health and Long-Term Care. Daily Access Reporting Tool (2011), <http://www.health.gov.on.ca/>

# RadioMarché: Distributed Voice- and Web-Interfaced Market Information Systems under Rural Conditions

Victor de Boer<sup>1</sup>, Pieter De Leenheer<sup>1</sup>, Anna Bon<sup>2</sup>, Nana Baah Gyan<sup>1</sup>,  
Chris van Aart<sup>1</sup>, Christophe Guéret<sup>1</sup>, Wendelien Tuyp<sup>2</sup>, Stephane Boyera<sup>3</sup>,  
Mary Allen<sup>4</sup>, and Hans Akkermans<sup>1</sup>

<sup>1</sup> Department of Computer Science, VU University, Amsterdam, The Netherlands  
{v.de.boer, pieter.de.leenheer, n.b.gyan,  
c.j.van.aart, c.d.m.gueret}@vu.nl, hans.akkermans@akmc.nl

<sup>2</sup> Centre for International Cooperation, VU University, Amsterdam, The Netherlands  
{a.bon, w.tuijp}@vu.nl

<sup>3</sup> World Wide Web Foundation  
stephane@boyera.net

<sup>4</sup> Sahel Eco, ACI 200 Rue 402, 03 BP 259, Bamako, Mali  
mary.sahelco@afribonemali.net

**Abstract.** Despite its tremendous success, the World Wide Web is still inaccessible to 4.5 billion people - mainly in developing countries - who lack a proper internet infrastructure, a reliable power supply, and often the ability to read and write. Hence, alternative or complementary technologies are needed to make the Web accessible to all, given the limiting conditions. These technologies must serve a large audience, who then may start contributing to the Web by creating content and services. In this paper we propose RadioMarché, a voice- and web-based market information system aimed at stimulating agricultural trade in Sahel countries. To overcome interfacing and infrastructural issues, RadioMarché has a mobile-voice interface and is easy to deploy. Furthermore, we will show how data from regionally distributed instances of RadioMarché, can be aggregated and exposed using Linked Data approaches, so that new opportunities for product and service innovation in agriculture and other domains can be unleashed.

**Keywords:** market information system, voice-based interfaces, linked data approaches, service innovation, generativity.

## 1 Introduction

The World Wide Web connects millions of people and organizations, empowering them to socialize, express opinion, and co-create at a scale and speed never seen before. It was not a carefully top-down planning, but a set of elementary internet technologies designed for de-centralized use that allowed for a Web with such a dramatic level of complexity and scale to emerge in less than two decades. Examples of such technologies are W3C-recommended open standards such as HTTP or HTML. By carefully excluding features that are not universally useful these technologies became easily adopted

on a massive scale and gave the Web a *generative* character, that is, the capacity to produce unanticipated change through unfiltered contributions from a broad and varied audience [1].

An upcoming trend is to publish structured data from different sources such as governments<sup>1</sup> and organizations<sup>2</sup>. More specifically, we follow the Linked Data guidelines and provide HTTP URIs for the resources (persons, places, products etc.) and describe the relations between them using the W3C-recommended open standard Resource Description Framework (RDF) [2]. In RDF, information is represented using Subject-Predicate-Object *triples*. The data is stored in a RDF databases, also known as a *triple store*. The Web of Data emerging from interlinked triple stores is an extension of the Web: it serves the data using Linked Data approaches so that machines can process them, rather than merely publishing them for human consumption [3]. By treating data as an asset, by sharing and trading it, an *open innovation* platform for all kinds of services will flourish, linking and augmenting data across domains [4].

Despite its success so far, the Web implicitly assumes a wide availability of high-bandwidth Internet infrastructure and reliable power supply. Interfacing the Web requires Personal Computers and various skills of which the most pertinent are reading and writing abilities. According to the Web Foundation<sup>3</sup>, there is an estimated 4.5 billion people, mostly living in developing countries, that cannot benefit from the Web for one or more of these reasons. This limits the Web's generative character per se. For our case study in Mali, only 1.8% of the population has Internet access<sup>4</sup>, only 10% has access to the electricity network<sup>5</sup>, and only 26.2% is literate<sup>6</sup>.

For a truly worldwide diffusion of innovations brought forward by the Web, we must devise new types of technologies immune to these infrastructural and interface problems. Hence, complementary or even alternative technologies to the ones we know are needed. Moreover, to guarantee these technologies will be applied and content will be contributed on a large scale, we have to identify value propositions that are interesting enough for a wider audience.

The proposition we consider in this paper is targeted at reducing poverty and hunger in Sub-Saharan Africa through better agricultural and rural development. According to the International Food Policy Institute, small subsistence farmers account for more than 90% of Africa's agricultural production and are usually at the very bottom of the pyramid [5]. In Africa, agriculture is the primary source of livelihood for about 65% of the population, it represents 40% of Africa's GDP and 60% of Africa's total export. Farmers who can count on different sources of income are less vulnerable in periods of

<sup>1</sup> e.g., <http://data.gov> and <http://data.gov.uk>

<sup>2</sup> such as public transport schemas, scientific results, etc.

<sup>3</sup> <http://www.webfoundation.org>

<sup>4</sup> source: <http://www.internetworldstats.com/> Internet World Statistics, Mini-watts Marketing Group.

<sup>5</sup> source:

<http://www.developingrenewables.org/energyrecipes/reports/genericData/Africa/061129%20RECIPES%20country%20info%20Mali.pdf>

<sup>6</sup> source <http://www.indexmundi.com/facts/indicators/SE.ADT.LITR.ZS> Index Mundi 2011.

drought. Trading is the best way to increase their income; to this end, better communication and access to customers and market information are key challenges. Our focus now lies on non-timber forest products (NTFPs) because they have a very long tradition and their production involves leadership by men as well as women.

According to the United Nations Food and Agriculture Organisation, *market information systems* (MISs) play an important role in rural agricultural supply chains and are the key to lower food cost and to raising producer and trader incomes [6]. MISs are information systems that gather, analyze and publish information about prices and other augmented information relevant to stakeholders involved in handling agricultural products and services. Indeed, farmers have to know the trends in demand to adapt production, find out where to find customers, and be able to determine a reasonable price by comparing with prices from other markets. Hence, there is an urgent need for effective and fair marketing delivered by transparent information [7]. Moreover, costs related to logistics are usually ignored. However, farmers at remote locations have to focus on products that can weigh up for such high prices implied by production as well as transportation costs. Opportunities for innovation through new cultivation techniques, new types of seeds, or by-products remain under-exploited due to a lack of market information needed to deal with the higher production costs.

In Africa, mobile telephony has become the primary mode of telecommunication [8]. In 2006, an estimated 45 percent of Sub-Saharan African villages were covered by a mobile signal [9]. And in 2009, Africa showed the fastest rate of subscriber growth, introducing 96 million new mobile subscribers in a period of only twelve months [10]. The widespread availability of mobile phones and increasing level of coverage creates great opportunities for new services.

RadioMarché is being developed within the context of the VOICES (VOICE-based Community cEntric mobile Services) project [1].

The contributions of this paper are :

- The introduction of RadioMarché (RM), a MIS concept adapted for rural conditions in the African Sahel.

Regarding the above mentioned challenges, RM is not dependent on Internet infrastructure, and has voice-based and sms-based interfaces. By exploiting the upward trend in (first-generation) mobile phone usage and the traditionally central role of radio in these areas, we believe in the generativity; hence a wide adoption of the RM concept in many regions of the Sahel.

- The proposition of a Linked Data model to address data integration issues across different regions.

On a large scale, we deal with the issue of aggregation and management of distributed market data by adopting Linked Data approaches. We show how our design choices offers opportunities to link aggregated market information to datasets from other domains. The resulting “Web of Data” provides an open innovation platform to develop services with augmented reasoning capabilities for *e.g.* NGOs, governments, policy makers, traders and scientists.

- A report on a first deployment of RadioMarché conducted in Mali, along with the explanation of the Living Lab approach applied to drive this activity.

---

<sup>7</sup> <http://mvoices.eu>



The structure of this paper is as follows. In Section 2 we discuss related work. In Section 3 we describe RadioMarché and discuss the instantiation of one single RadioMarché instance in the Tominian region in Mali. Next, we describe a scenario where there are multiple RM instances in different regions. In Section 4 we discuss how we apply the Living Lab methodology for a non-disruptive deployment of RM under rural conditions, and report on the current status of the implementation and the technical setup. Finally, in Section 5 we outline the next steps including validation and the economical sustainability assessment of RM.

## 2 Related Work

Related work on voice technologies started in the 1930s in research on speech recognition. The first commercial deployments of voice-based services took place in the early 1970s. Major achievements on language recognition took place in the 1980s and 1990s, but this was mainly focused on English. While Text-To-Speech and Speech Recognition are key in voice application development, the creation of the VoiceXML standard by the W3C Voice Browser group, in 1999, further facilitated the development of voice applications [11].

Agarwal et al. from IBM Research India, developed a system to enable authorship of voice content for 2G phone in a Web space, they named the WWTW (World-Wide Telecom Web). A dedicated voice browser is hosted by the telecom operator and communicates on behalf of the end-user, and holds all user data about link history and e.g. user preferences. The system is not connected to the World Wide Web and does not allow indexing e.g. by third party search engines. The whole system creates a closed web space, within the phone network. Linking from one voice site to the other is done through a protocol HSTP, created by IBM. Especially the lack of open search possibility constrains its growth [12].

Several automated market information systems have been developed and built to support farmers and agricultural trade in developing countries. One of the well-known market information systems is ESOKO [13], an online market system, developed and built in Ghana. ESOKO enables sellers and buyers to exchange market information. The system is web-based and allows entry of market information and offers from farmers through SMS text messages. The ESOKO system is not an open source platform; the software is proprietary; licenses can be purchased. The existing platform that serves the market in Ghana is accessible for paying subscribers, mainly wholesale buyers. In contrast to RM, ESOKO does not target the poorest group of subsistence farmers in Africa, that account for 90% of Africa's agricultural production. The costs for subscriptions are relatively high and the system is not well-adapted for illiterate people. The scope of ESOKO is larger than the small-scale regional trading of small amounts of produce on local markets that RM targets.

Google started a project in Uganda in 2009, partnering with MTN and Grameen Foundation to develop mobile applications that serve the needs of poor and other vulnerable individuals and communities, most of whom have limited access to information and communications technology [14]. This system is based on SMS but does not allow voice access.

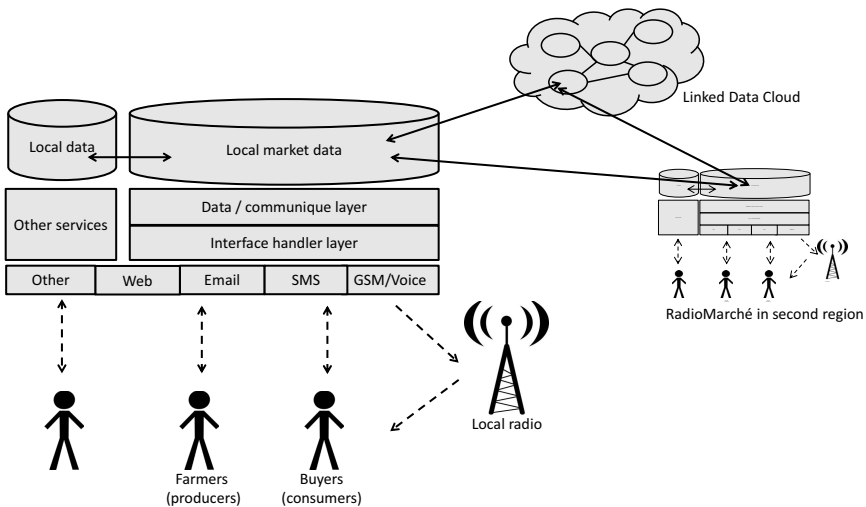
A related project on Linked Data for developing countries is described by Guéret et al. [15]. The SemanticXO is a system that connects rugged, low-power, low-cost robust small laptops (a.k.a. the XO promoted by the One Laptop Per Child organisation) for empowerment of poor communities in developing countries. This Semantic XO is based on the same Linked Data approaches as the data aggregation between instances of RM we will describe in this paper. Both systems are Open Source and use the Web to publish previously unpublished data.

### 3 Conceptual Design

In this section, we describe the overall design of the RadioMarché system. We first describe a single instance of the market information system for one single region, and discuss the opportunity for other services to reuse the market information within the region. Next, in Section 3.3 we extend the setting from one to multiple instances of RM across different regions and describe the distributed market data aggregation using Linked Data approaches. Finally, we describe how this aggregated market data can be linked with external data sets from other domains, leading to an open innovation platform for unanticipated services that consume this linked data. Figure 1 shows an overview of the system architecture.

#### 3.1 Regional Instance of RadioMarché

**Data.** A local instance of RadioMarché has one data store with rudimentary market information such as product offerings (including product type, quality, quantity,



**Fig. 1.** Conceptual design of the RadioMarché system. The system provides alternative interfaces based on voice or SMS via phone or radio, enabling a wider audience to consume and contribute content. The data design is optimized for (i) effective aggregation with other RM instances and data sources from other domains in the Cloud; and (ii) reuse by other services.

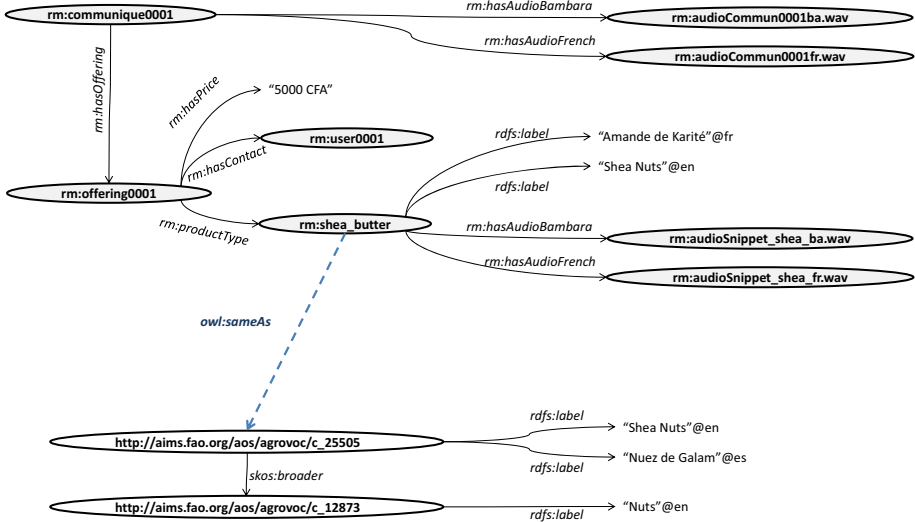
location and logistical issues) and contact details from sellers and buyers. To maximize the reusability across different domains and regions and allow for automatic machine processing, we adopt Linked Data standards to represent the data. Linked Data approaches provide a particularly light-weight way to share, re-use and integrate various data sets using Web standards such as URIs and RDF [16]. It does not require the definition of a specific database schema for a dataset [17]. Our implementation methodology assumes that we start from a legacy system. Although the specifics of the locally produced data will differ from region to region, Linked Data provides us with a standard way of integrating the parts of the data that different regions have in common. Also, because we do not impose a single overarching schema on the data, data reuse for new services is easier, both within a region and across regions.

An additional advantage is that Linked Data is well-suited to deal with multiple languages as its core concepts are resources rather than textual terms. A single resource, identified by a URI (ie. [http://example.org/shea\\_nuts](http://example.org/shea_nuts)) can have multiple labels (eg. “Shea Nuts”@en and “Amande de Karité”@fr). Other than textual labels, for our voice-services we add audio to the resources with language-specific voice snippets, also identified through URIs. Figure 2 shows an example of how a very small part of the data would be represented using RDF.

**Application Layers.** The raw Linked Data is handled and aggregated into communiqués by the data/communiqué layer which interfaces with the RDF triple store using standard Linked-Data querying and data-posting APIs (for example using the SPARQL query language). This is where the market information is aggregated and it is decided what information is accessible to which user. In the interface handler layer, this information is represented in multiple views. It is here that the audio versions of communiqués are constructed. The interface handler layer is also used to process user input such as that of the NGO agent entering new market data through a web form or local producers doing the same through voice menus.

**Interfaces.** The interface layer is the technical layer consisting of the actual interfaces channels: each with its own limitations to user interface design. The RadioMarché design foresees multiple interfaces for producing and consuming market information.

1. The voice-based interface allows non-intrusive market information access for all users having a first-generation mobile phone. It allows farmers to navigate a voice-based menu and enter product offerings using a call-in service at a local telephone number. The voice service is available in the local languages relevant to the specific region. For the voice-based interface, we adopt the industry standard VoiceXML. Since we cannot assume that text-to-speech (TTS) libraries are available for the local languages, we currently use prerecorded phrases in the local languages for the voice menus.
2. The SMS-based interface provide for literate users a more effective way of adding and consuming market data.
3. Through the traditional Web channels or via e-mail, users can get weekly digests of the latest offerings or add their own using a predetermined and machine-readable mail format. Standard Web access naturally allows for users to access market data using web browsers.



**Fig. 2.** Example of a snippet of RDF market information data in the RadioMarché triple store. Resources have URIs and are represented using ellipses, typed relations between resources and to literal labels are represented using arrows. “rm” is a shorthand for the namespace <http://radiomarche.com/>. The figure shows how multilingual audio resources are related to the communiqués. These are built up from audio snippets related to the content of the offerings making up the communiqué. The bottom part of the figure shows how the local market data can be linked to the Linked Data version of the Agrovoc thesaurus, opening up the possibility for mutual re-use and reasoning. Note that only a small part of the market data relating to offerings and product types are shown.

By offering multiple interfaces to RadioMarché, the system is open to contributions from a wider audience of users with less capabilities, both in terms of hardware as well as literacy; hence extending its generativity. The multi-interface approach also ensures that when local development causes new hardware and connectivity to become available to the users, they can access the same system in these new ways.

**Radio.** Although users can directly interface with RadioMarché using any of the interfaces described above, local radio stations provide an interface to potential market information consumers such as buyers. Every week, the market information is sent to local radio stations, that broadcast them to their listeners. Community radio is an important communication channel in rural agricultural areas with a recognized potential for change and development [18]. By integrating community radios explicitly in our system design, we aim to expand the range of potential buyers to users that have no access to mobile phones or web. The radio stations themselves access the system using each of the communication channels. Some radio stations have computer hardware and connections that allow them to receive the market data via the web. Radio stations that lack this infrastructure can use the voice channel, where they call in to the system and play the market information in audio form live on the radio.

These layers make up the market information system service on top of the data store. This system provides the ability to contribute market information by relevant actors, for example by producers wishing to add their offering to the system. The service also disperses the relevant market information to potential buyers.

### 3.2 Information Re-use Within a Region

Our data design allows to integrate the market information with other types of information thereby increasing its value and potential for reuse by other services. One additional service that is planned as a second case study in the VOICES project is a meeting scheduling system. Such a system would provide local NGOs with a more effective way to transfer agricultural knowledge about non-timber forest products to their farmer community. The services developed in this case study provide voice access to personal and scheduling information. By integrating this information with the market information from RadioMarché, personal profiles can be enriched with information about the type of products that specific farmers have been producing within a given period. Here a new scheduling and notification service can re-use the market information within a region.

A second use case that is currently under development is a voice-based journalism platform, which allows both professional and citizen journalists to send voice-recorded news items to local community radios. The target region for this use case consists of agricultural communities and there is a large possibility for re-use of both technical infrastructure as well as data.

To do this, the re-usable resources (e.g. person data, geographical or product information) in the market information data are linked to the relevant resources in the target data set using Linked Data standard relations.

### 3.3 Information Aggregation across Regions

Consider the setting where there are multiple RM instances running across different regions. This brings the opportunity to aggregate very large volumes of market information and link it with other data services, increasing its value for potential buyers but also for other stakeholders.

An example is the following scenario. In the Tominian region, farmers and buyers use local RM to express their offers and demand for shea butter. The RM's historical data learns us that there is an average supply-demand ratio of 5/1. The same is done in the more urbanized region of Bamako, where the RM instance informs us there is a ratio of 1/5. Given the low demand for shea butter in Tominian, it makes no sense to spend radio time to communicate the offer. However, having a global picture of the ratios across regions, the oversupply in Tominian could be offered in Bamako where there is no production of shea butter at all. This augmented capability allows farmers to think more commercially; hence finding new markets to increase their income on excess production.

The role of RM as a concept has now been implemented specifically for market information for NTFPs. The distributed and aggregated information services enables producers and consumers of NTFPs across regions to connect more efficiently. It also

enables producers to adapt their prices comparing with price information from other markets available in the MIS. The concept however is generic enough to be applied for harvesting data on supply-demand ratios for other (innovative) products such as new types of seeds or processing tools; and even services including trainings in advanced cultivation techniques and transport. E.g., the fact that the farmer learnt from RM that he can sell his excess shea butter production on far-away markets in Bamako, creates opportunities for optimizing transportation services.

Ultimately, by reproducing the simple RM concept on a large scale, a web of high volumes of aggregated data about supply-and-demand ratios for many different types of services or products contributed by different regions may emerge. This allows us to model and analyze the actual *value networks* in the Sahel using our *e<sup>3</sup>-value* [\[8\]](#) ontologies [\[19\]](#). From this semantically enriched knowledge we can apply our *e<sup>3</sup>service* technology to automatically discover desirable or undesirable patterns in exchange of tangibles as well as intangibles, and build customer and product catalogues accordingly [\[20\]](#). E.g., when the value networks for two different products are isomorphic and they can be produced by one source farmer (or cooperative); one may decide to publish innovative offerings combining these products for better prices at higher volumes. For our recent survey on service network approaches, see [\[21\]](#).

At the same time, local and national governments as well as NGOs can exploit the aggregated market information for analytic purposes, monitoring the trade in NTFPs within and across regions. By linking the market information to existing agricultural vocabularies such as FAO's Agrovoc thesaurus [\[9\]](#), the CAB Thesaurus [\[10\]](#), or the USDA's National Agricultural Library NAL [\[11\]](#), the aggregated market data can be used for specific analyses for government or NGO purposes.

In the example shown in Figure [2](#) the RadioMarché resource representing Shea Nuts is linked to the same concept in the Agrovoc thesaurus. Through the Agrovoc hierarchy, a reasoner can now infer that the offering in the example concerns a type of nut.

## 4 Deploying RadioMarché in the Tominian Area, Mali

We are currently in the process of implementing a specific instance of the described system in the Tominian Area in Mali, Africa. This use case was identified within the VOICES project.

### 4.1 Living Labs-Based Approach

Our methodology for developing, testing and deploying the RadioMarché system is based on the Living Labs principles. Living Labs (LL) are experimentation and validation environments of ICT-based innovation activities. They are characterised by the early involvement of user communities, by openness in establishing a close cooperation between developers, users and other stakeholders, and by the creation of rapid learning

<sup>8</sup> <http://www.e3value.com>

<sup>9</sup> <http://aims.fao.org/website/AGROVOC-Thesaurus/sub>

<sup>10</sup> <http://www.cabi.org/cabthesaurus/>

<sup>11</sup> <http://agclass.nal.usda.gov/>

cycles accelerating the innovation process. As highlighted by [22], LL are a good match for deploying information and communication technologies in rural areas.

As part of our LL-like approach, we employ a strategy of first explicitly analysing the current situation and identifying use cases. We selected a paper-based system as the initial system. In the next step parts of the system are augmented with ICT, which is then analysed with respect to the effectiveness and local acceptance. The results of this first cycle determine the content of the next development-test cycle. By employing this iterative methodology rather than deploying a single end-application at once, we aim to promote local ownership of the system through early involvement and ensure that we understand and are able to meet all local requirements. A recent global research report by UNICEF on mobiles for development (M4D) [23], state that among the reasons why many M4D projects fail is the lack of local content. By starting with an existing system, our development strategy presents a way of surmounting this challenge.

### 4.2 Local Situation

The RadioMarché system in the Tominian area starts from an already running “legacy” MIS that was set up by our project partner Sahel Eco<sup>12</sup> in 2010. Sahel Eco is an NGO dedicated to promoting sustainable use of forest resources and develop small businesses based on NTFPs. The main product focus of the MIS is on shea nuts, shea butter, honey, wild fruits and nuts. The MIS is currently used to distribute up-to-date market information via community radio in the area.

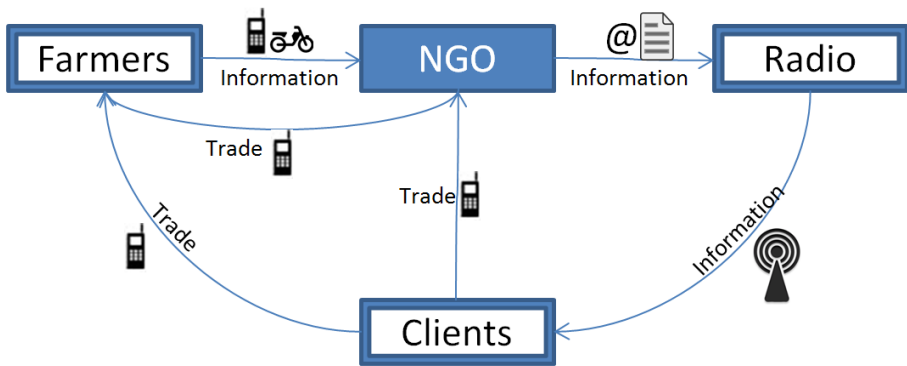


Fig. 3. Model of the current value network for the MIS in Tominian, Mali

In the current situation (shown in Figure 3), a Sahel Eco staff member receives offerings from local farmer’s representatives in the form of an SMS text message, containing info about offer, quantity, quality, price, name of the seller, village, phone number, etc. The SMS info is entered manually into the system. Every week, a “communiqué” is drafted by the staff member and from a cyber cafe sent to three radio stations (ORTM Ségou, Koutiala, ORTM Mopti). Only ORTM Ségou is connected to the internet, the

<sup>12</sup> <http://www.sahleeco.net>

## INFORMATION SUR LES PRODUIT FORESTIER NON LIGNEUX DU CERCLE DE TOMINIAN

Zone de production (commune)	Villages	Nom du produit	Unité de mesure	quantité disponible	qualité du produit	prix au kg en F CFA	contacts
Mafouné	Souté	amande de karité	kg	1800	amande ébouillantés	200	Mandiakuy Philippe TEL: 75 [REDACTED]
	Bokuy-Mankoina	miel	Litre	72	miel non brûlé	000	Zakari DIARRA TEL: 78 [REDACTED]
	Bokuy-Mankoina	Beurre de karité	kg	60	beurre issu des amandes ébouillanté	000	Zakari DIARRA TEL: 7 [REDACTED]
KOULA	Tiéblénikuy	Beurre de karité	kg	165	beurre issu des amandes ébouillanté	200	Gérard TRAORE TEL: 77 [REDACTED]

NB : Pour plus d'information contactez Monsieur Amadou TANGARA SAHEL ECO TOMINIAN TEL. 79 [REDACTED] ou le point focal de la radio que vous écoutez

Fig. 4. Example of a communiqué. Phone numbers are blurred for privacy reasons.

other two radio stations receive their message by going to a nearby cyber cafe and printing out the email attachment. A fourth radio station, Radio Moutian in Tominian, has no internet access whatsoever. The staff member worker prints out a hard copy of the information which is physically brought to that radio station. Figure 4 shows a recent example of such a communiqué.

The radio stations each have an employee that reads the communiqués live on the radio multiple times per week. The radio stations are paid a fee for the broadcast. The potential buyers listen to the community radio and contact the sellers to buy.

### 4.3 Current Status

In the current situation, we augmented the current MIS in a number of ways. First, we designed and deployed a web form, which allows registered users to add and edit market information to a database. Currently, this is used only by the Sahel Eco staff member. The system stores all communiqués allowing for aggregating and analysing historical market information.

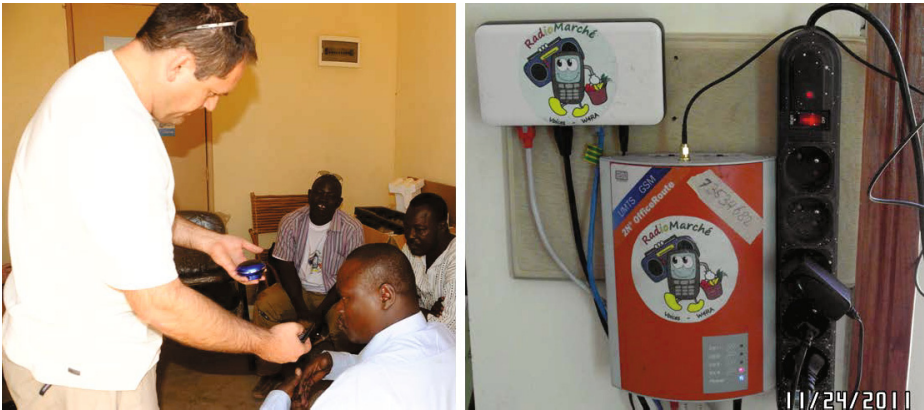
The Sahel Eco staff member can generate a new communiqué from the current market information. At that moment, the communiqué is available for the radio stations that have web connection in text form. Alternatively, this text version can be sent via email or printed on paper. At the same time, an audio version of the communiqué is generated from pre-recorded voice fragments. These fragments have been obtained during local recording sessions. During separate sessions the quality of the automatically generated communiqués was evaluated by local radio producers. Figure 5 shows such a session.

Currently, this audio communiqué can only be produced in the Malian dialect of French, but we are developing these services for local languages such as Bambara<sup>13</sup> and Boma<sup>14</sup>. The audio communiqué is created using local Malian voices. Two of the

<sup>13</sup> [http://en.wikipedia.org/wiki/Bambara\\_language](http://en.wikipedia.org/wiki/Bambara_language)

<sup>14</sup> [http://www.ethnologue.com/show\\_language.asp?code=bmg](http://www.ethnologue.com/show_language.asp?code=bmg)





**Fig. 5.** Implementation of the RadioMarché system in Tominian, Mali. On the left, an audio recording and evaluation session is shown. The right part of the image shows part of the hardware setup, including the OfficeRoute GSM gateway.

local radio stations journalists' voices have been recorded and are used for the audio communiqués. By using local voices, we intend to increase recognisability and trust.

The audio communiqué is also accessible to the radio stations from the web as well as through a voice channel. As soon as a new communiqué is generated, the radio station employees can call a local telephone number. After identifying the radio-station through a voice-menu, the latest audio communiqué specifically created for that radio station is played. The audio can be played directly on the radio, or recorded using local equipment for later broadcasting.

#### 4.4 Technical Setup

The above system was realized using two separate technical implementations. In one version, we use cloud-based services to host the web form and database. The local telephone company provides the system with voice-based access by linking a number of local telephone numbers to this system. This is done using France Telecom Orange Emerginov platform<sup>[15]</sup>.

The second version of the system is entirely local. This version has the web form and database running on a dedicated laptop. Radio stations that have internet connection can access this network directly via the web. The voice channel is provided by a voice browser (currently using the prophecy VXML browser by Voxeo<sup>[16]</sup>) and a GSM gateway (2N OfficeRoute) device that allows phone calls to be handled by the RadioMarché system on the laptop. The OfficeRoute is connected to the laptop. Figure 5 shows the GSM gateway as well as the ethernet switch used to connect it to a local laptop as it is currently installed on location.

The local version has the advantage that the system can be updated and is accessible through the voice channel even in the absence of an internet connection. The fact that the

<sup>15</sup> <http://www.emerginov.org>

<sup>16</sup> <http://www.voxeo.com>

system is completely localized might also improve local ownership and makes the set-up less dependent on telecom partners. The cloud-based version on the other hand has the advantage that it comes with extensive support, robustness and scalability. The two versions of the system are currently both being tested in the field. Moreover, one version can act as a backup to the other in a redundancy-based setting, increasing robustness of the system.

## 5 Discussion and Next Steps

In the previous section, we described the first steps of implementing the conceptual design of Section 3. More specifically, we have implemented the web-based interface that the NGO agent can use to enter market information into the system and the email- and voice-based interface channels for radio stations to receive the new communiqués. The current system, shown in Figure 3 is therefore augmented in a limited number of places. Over the next period, we will be evaluating this setup. Evaluation will focus on:

**usability:** Is the system usable for the proposed users? More specifically, are the web forms adequate and convenient. Although initial evaluation on the sound quality of the GSM voice interface have been very positive, longer term usage will be evaluated over the next period of time. When more information becomes accessible to the users, human-computer interaction issues such as voice-menu design will become more prevalent.

**robustness:** The current system, with its redundant setup, is designed for robustness. This will be tested over the next period, particularly with respect to local conditions particularly with respect to local conditions such as unreliable Internet connections and power-outages. At the same time, technical maintenance of the system should be feasible by local operators.

**efficiency gain:** Do the new interface channels, the web access and the digitized database actually improve the efficiency of the market information system? Are more people reached, are more products traded and is this done with less resources?

The next development cycle will include the implementation of the Linked Data layer, which will be populated with historical communiqué data (based on archived communiqués such as the one displayed in Figure 4) as well as the current market information data, as entered in our system via the new interfaces. This data will be represented using RDF and linked to a number of data sources, specifically geographical thesauri such as GeoNames<sup>17</sup> and the agricultural thesauri noted in Section 3.3. In this way the local data, created in rural development areas can become part of the growing Linked Data cloud<sup>18</sup>. At the same time, we will be developing the meeting scheduler use case that was described in Section 3.1, linking all common classes and instances, including places, people and products.

The current system is only equipped to produce audio communiqués in French. An important step is to record the required audio fragments in other, regional languages

<sup>17</sup> <http://www.geonames.org>

<sup>18</sup> <http://linkeddata.org/>

such as Bambara and Bomu as well, and to adapt the audio communiqué construction methods to be able to deal with these languages. At the same time we will open the system to non-radio users, allowing arbitrary potential buyers or sellers to receive the latest market communiqués.

Another dimension that will be further investigated is the economic sustainability of the proposed distributed market information system. This includes assessing local situations, stakeholder analysis and cost models for developing rural regions such as the Tominian area. RadioMarché is designed to be a low-cost, easy-maintenance system. Early results of the market analysis suggest that it can be economically sustainable even with a limited number of users. Moreover, the system can be easily replicated across regions and application domains and therefore is designed to scale up well.

Part of the Living Labs development methodology is that we starting off with an initial prototype, which will be used to get the local community involved in co-creating next development steps and new services. The developed software will be published as an open source toolkit, including the local language resources and voice interfaces. Local entrepreneurs will receive training to maintain existing services as well as develop new applications using this design. Other than the meeting scheduling and citizen journalism services described in Section 3.3, we will investigate developing voice-based services regarding social networks, yellow pages, medical services, weather services etc. Through this effort we are developing building blocks for a Web of Data accessible through voice for users in developing countries.

**Acknowledgements.** This research is partly funded by the European Union through the 7th Framework Programme (FP7) under grant agreement Num. 269954.

## References

1. Zittrain, J.: The Future of the Internet and How to Stop it. Yale University Press (2008)
2. Berners-Lee, T.: Linked data - design issues (2006), <http://www.w3.org/DesignIssues/LinkedData.html>
3. Fischetti, M.: The web turns 20: Linked data gives people power. Scientific American Magazine (2010)
4. Chesbrough, H.: Open Services Innovation. Wiley (2011)
5. International Food Policy Research Institute, I.: Ending hunger in africa, prospects for the small farmer (2004), <http://www.ifpri.org/sites/default/files/pubs/pubs/ib/ib16.pdf>
6. FAO: community-based tree and forest product enterprises: market analysis and development manual (2011), <http://www.fao.org/docrep/014/i2394e/i2394e00.pdf>
7. FAO: Financing agriculture and rural development in africa: Issues, constraints and perspectives. In: Twenty-third Regional Conference for Africa, Johannesburg, South Africa, March 1-5 (2004)
8. UNCTD: Science and technology for development: the new paradigm of ict. Information Economy Report 2007-2008, United Nations Conference on Trade and Development (2007)
9. ITU: Telecommunications/ict markets and trends in africa. International Telecommunications Union report (2007)

10. Economist, T.: Mobile marvels. Special report on telecoms in emerging markets. *The Economist* (September 2009)
11. W3C: Voice extensible markup language (voicexml) version 2.0. W3C Recommendation, March 16 (2004), <http://www.w3.org/TR/voicexml20/>
12. Agarwal, S.K., Jain, A., Kumar, A., Rajput, N.: The world wide telecom web browser. In: *Proceedings of the First ACM Symposium on Computing for Development, ACM DEV 2010*, pp. 4:1–4:9. ACM, New York (2010)
13. ESOKO: Esoko (2011), <http://www.esoko.com/>
14. AppLab, G.: Google sms to serve needs of poor in uganda (2009), <http://blog.google.org/2009/06/google-sms-to-serve-needs-of-poor-in.html>
15. Guéret, C., Schlobach, S.: SemanticXO: Connecting the XO with the World's Largest Information Network. In: Yonazi, J.J., Sedoyeka, E., Ariwa, E., El-Qawasmeh, E. (eds.) *ICeND 2011*. CCIS, vol. 171, pp. 261–275. Springer, Heidelberg (2011)
16. Bizer, C., Heath, T., Berners-Lee, T.: Linked data - the story so far. *International Journal on Semantic Web and Information Systems* 5, 1–22 (2009)
17. Domingue, J., Pedrinaci, C., Maleshkova, M., Norton, B., Krummenacher, R.: Fostering a relationship between linked data and the internet of services. In: Domingue, J., Galis, A., Gavras, A., Zahariadis, T., Lambert, D., Cleary, F., Daras, P., Krco, S., Müller, H., Li, M.S., Schaffers, H., Lotz, V., Alvarez, F., Stiller, B., Karnouskos, S., Avessta, S., Nilsson, M. (eds.) *The Future Internet. LNCS*, vol. 6656, pp. 351–364. Springer, Heidelberg (2011)
18. Fraser, C., Restrepo-Estrada, S.: Community radio for change and development. *Development* 45, 69–73(5) (2002)
19. Akkermans, H., Baida, Z., Gordijn, J., Peiia, N., Altuna, A., Laresgoiti, I.: Value webs: using ontologies to bundle real-world services. *IEEE Intelligent Systems* 19(4), 57–66 (2004)
20. Razo-Zapata, I., Gordijn, J., De Leenheer, P., Akkermans, H.: Dynamic service bundling: a value-oriented framework. In: *Proc. of IEEE Conference on Commerce and Enterprise Computing (CEC)*, Luxembourg. IEEE Press (2011)
21. Razo-Zapata, I., De Leenheer, P., Gordijn, J., Akkermans, H.: Service network approaches. In: Barros, A., Oberle, D. (eds.) *Handbook of Service Description: USDL and its Methods*, pp. 45–74. Springer, New York (2011)
22. Guzman, J.G., Schaffers, H., Bilicki, V., Merz, C., Valenzuela, M.: Living labs fostering open innovation and rural development: Methodology and results. In: *Proceedings of the Conference on International Conference on Concurrent Enterprising, ICE* (2008)
23. Boakye, K., Scott, N., Smyth, C.: Mobiles for development. UNICEF Technical Report p65-66 (2010)

# Publication of Geodetic Documentation Center Resources on Internet

Marcin Luckner<sup>1</sup> and Waldemar Izdebski<sup>2</sup>

<sup>1</sup> Warsaw University of Technology, Faculty of Mathematics and Information Science,  
pl. Politechniki 1, 00-661 Warsaw, Poland  
mluckner@mini.pw.edu.pl

<http://www.mini.pw.edu.pl/~lucknerm/en/>

<sup>2</sup> Warsaw University of Technology, Faculty of Geodesy and Cartography,  
pl. Politechniki 1, 00-661 Warsaw, Poland  
waldemar@izdebski.edu.pl

<http://www.izdebski.edu.pl/>

**Abstract.** Geodetic Documentation Centers collect geodetic and cartographic resources. The resources include spatial data and their metadata. European Union INSPIRE directive imposes an obligation on GDC to publish selected data in the Internet. In this paper, an adequate form of publication is discussed on the base of iGeoMap application.

The Internet application iGeoMap merges data from various resources. Depending on the data to present, different types of resources are used. The application can publish spatial data from files (text or binary), a database specialized in spatial data service (PostgreSQL, ORACLE), or web services (Web Map Service, Web Feature Service). Utilization of various data sources by the application is presented in this paper.

As a part of the subject, searches of the most popular data (parcels, address points, and control points) are discussed. Various data sources and searching mechanisms involved by the searches in iGeoMap are presented in use cases.

**Keywords:** E-Government, Spatial data, Web Mapping, Web Map Service, Web Feature Service.

## 1 Introduction

Geodetic Documentation Centers collect geodetic and cartographic resources. The resources include spatial data and their description. European Union INSPIRE [1] directive imposes an obligation on the centers to publish selected data on the Internet.

The centers manage data in various forms. Usually, spatial data is collected as files in vector or raster format. Sometimes databases with spatial data support, such as Oracle or Postgres, are used. Publication on the Internet is usually performed using a dedicated application compatible with internal data format. However, in recent years web services have been used more widely for such

purposes. Such form of publication is more user-friendly since it allows the data to be referenced in personal resources.

In this paper, an Internet application iGeoMap is presented as a tool for spatial data publication. The application supports several types of data sources. Spatial data can be acquired from files, databases, and web services.

In subsequent sections there are presented sources of spatial data utilized in Geodetic Documentation Centers (Section 2) and iGeoMap application for a data publication in the Internet. For this application several typical use cases, encompassing various data types, are discussed (Section 3). Finally, the conclusions are presented (Section 4).

## 2 Sources of Data

Geodetic Documentation Centers utilize various types of resources. The centers collect spatial data mainly as files in a closed, proprietary data format enforced by the data management software in use.

For example, the centers, which use iGeoMap, utilize a mapfile format. Usually final users do not have adequate software to read such files and a conversion to the most popular and open data formats such as GPX and KML is necessary.

In worse scenario, the centers only have graphical data available in the form of scanned map sheets. However, such files can be also published if additional georeference information is given. In the worst case, local centers do not have any data in digital form, but this situation will not be discussed here.

Several centers, mostly in big agglomerations, collect data in databases. This situation is typical when more than one application have to access the same data. For spatial data dedicated systems should be used (Oracle, Postgres). For security reasons, database engines cannot be made directly accessible to end users and the data must be accessed via an authorized application or a web service instead.

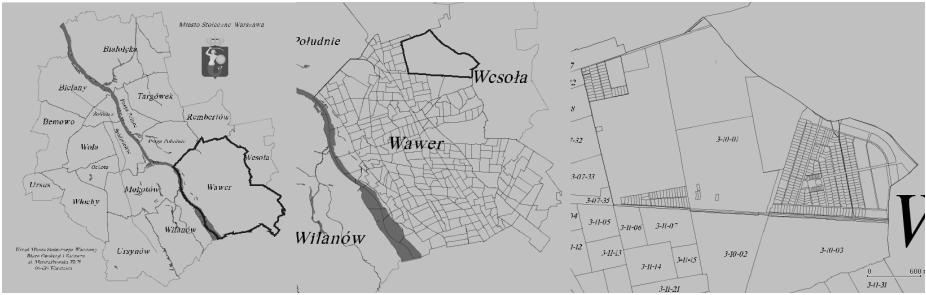
Among web services there exist some dedicated to spatial data publication. The most important is the WebMap Service, which creates an image of the requested area. This service has a limited ability to generate spatial objects descriptions.

When detailed geospatial data is necessary, the Web Feature Service should be used instead. Finally, overloaded WMS can be replaced with the Tile Map Service, which delivers pre-generated tiles.

A detailed description of the above-mentioned data sources together with information on how iGeoMap application supports them are given in the following sections.

### 2.1 Files

The iGeoMap application is able to read files in mapfile format. This format is used by Geodetic Documentation Centers on daily basis. The format is dedicated to presentation of spatial data collected by a local government administration.



**Fig. 1.** A hierarchical administrative structure mapped into a mapfiles structure (from the left: a district, a region, parcels). Links are presented as a bold boundary.

The application allows its user to export selected data in common file formats. The KML format is used for a presentation of 3D objects. The GPX format is used for import data into GPS devices.

These files store formatted text. However, spatial data can be also collected in binary sources. The first example are shapefiles as a commonly used and supported source of spatial data. The second, image files, are very often the only digital source of some data in centers and can be published if georeferences for images are given.

**Mapfile.** The mapfile format is a text format describing spatial data. A single file contains a list of objects. Each object is described by the object geometry and a set of descriptive features. An example is given below.

*Example of an area object from mapfile*

```
*5213 0 19 0
:A2[146516_8]
:A3[Wilanow]
:TX[@Obreby/Obreby_Wilanow.MAP]
P 1 -9701.560 3538.640 ,20121
P 1 -9676.950 3520.410 ,20120
P 1 -9646.360 3511.880 ,20118
P 1 -9642.500 3482.740 ,20117
P 0 -9701.560 3538.640 ,20121
```

In this example, the Wilanow district is described. The object geometry is a list of points. Each point is given by a labeled pair of coordinates in a local metric system. Different centers use different systems. It impedes the exchange of data between centers.

Descriptive features are stored in attributes. The district is characterized by a unique number called teryt (stored in the attribute A2) and holds links to objects from lower administrative level (in the attribute TX). These two attributes allow administrative structure to be mapped into file structure.



**Fig. 2.** Information about buildings exported into a KML file

The number called *teryt* holds information about position of the object in a local government administrative structure. The first two digits determine the first level voivodeship. The next two digits correspond to the administrative level of Geodetic Documentation Centers. The remaining digits define the district and its type (agglomeration or country).

Districts group regions denoted with numbers, which are unique only in the context of the *teryt*. In the example presented above the file that represents an administrative level of Geodetic Documentation Centers contains an object that is described by the *teryt* as a district. This object has a link to the next file. This successor includes a set of regions. The following links bring user to a demanded level of the local administration, as it is presented in Fig. 1.

The mapfile format is used in all Geodetic Documentation Centers, which publish spatial data in Internet using iGeoMap. Each day the data from the internal geospatial system are exported as mapfiles. These files contain information about parcels, buildings, and others.

Text files are not the best solution for publication of large data sets on the Internet because of transmission bandwidth limits. For this reason, the data is published as gzip files. For the final user, which uses Java applet this compression is imperceptible, because Java supports compressed streams.

**XML Based Files.** Formats, which are based on the syntax and file format of XML, such as Keyhole Markup Language (KML) and The GPS Exchange Format (GPX) are used to export selected data from the application.

Keyhole Markup Language is a descriptive language developed by Google Company. KML complements Geography Markup Language (GML) (standard defined by Open Geospatial Consortium) as a format for describing and storing geographical information including threedimensional objects.



The format was developed for Google Earth. In Google Earth, user–data is presented against a background of satellite photos, a three–dimensional Earths model, and third-party data. Google Earth brings user–friendly interface, which enables presentation of data as a layered structure. A user can modify various aspects of data presentation including zoom, position of the camera, and a visibility of layers. The format can be successfully applied in various GIS tasks [18,9].

In the centers, the KML format is used for the presentation of selected resources as a three–dimensional model. A visualization of buildings is given in Fig. 2. The model can be generated only by officials and cannot be generated via Internet.

The GPS Exchange Format [16] is a lightweight XML data format for the interchange of GPS data (waypoints, routes, and tracks) between applications and Web services on the Internet.

In the centers, the GPX format is used to export specific data. The result files can be loaded into a GPS device. Such approach makes localization of characteristic point in the terrain easy. A control point representation in the GPX format is given below. The file can be generated on demand by registered users.

*Example of a control point in a GPX file*

```
<gpx xmlns="http://www.topografix.com/GPX/1/1" creator="iGeoMap">
  <wpt lat="52.35376381765543" lon="20.873053706771273">
    <time>2011-02-28T09:54:17Z</time>
    <name>423.1288</name>
    <desc>Control point</desc>
    <sym>default</sym>
  </wpt>
</gpx>
```

Both types of files (KML and GPX) are based on the syntax and file format of XML. For that reason, the size of file is larger than in case of the mapfile. This problem can be reduced using compression.

Coordinates of points in exported formats are given in latitude/longitude convention. The coordinates are placed on an ellipsoid. A conversion, which bases on a complex transformation from a plain metric system is necessary.

**Shapefiles.** A shapefile [4] stores geometry and attribute information for the spatial features in the data set. The feature geometry is stored as a shape comprising a set of vector coordinates. Shapefiles are very popular in spatial data presentation but have several limitations [17].

Shapefiles have a relatively fast drawing speed and editing capabilities. Shapefiles can handle single features that overlap or are non–contiguous. Typically they also require less disk space to store. On the other hand, a single shapefile can store only one type of geometry at the same time.

Shapefiles support point, line, and area objects. Attributes are stored in a dBASE format file. This format is somewhat outdated, but can be sufficient in

typical applications. Each attribute record has a one-to-one relationship with the associated shape record. The length of attribute names is limited. Only Integer, Real, String and Date field types are supported. Various list and binary field types cannot be created.

A representative shapefile consists of three files: a main file SHP, an index SHX and a database table DBF. This division makes distribution of files on the Internet difficult.

The Shapefile format explicitly uses 32-bit addressing and thus cannot go over 8GB. In addition, there is a 2 GB size limit for any Shapefile component file, which translates to a maximum of roughly 70 million point features.

Because of its popularity, shapfiles are used to transfer data between various spatial applications in Geodetic Documentation Centers.

**Image Files.** Images can be published as spatial data when georeference files are attached. The georeference file has the same name as the image file and an extension constructed as the first and the last letter of an image extension with an additional *w* letter. For example, the name of a georeference file for a file *image.jpg* is *image.jgw*.

The georeference file gives information about pixel size in map units (usually in meters) in both directions. Coordinates of the first pixel are also given. The last georeference element is the rotation angle about two axes.

Image files are mostly used to present archival data, which are not longer modified.

## 2.2 Databases

Databases are usually better sources of data than files. However, for spatial data a specialized database has to be used. PostgreSQL with Postgis extension allows to store spatial data. The same is with ORACLE. iGeoMap can work with both databases directly or via PHP interface.

ORACLE database may be too expensive for local centers and the free PostgreSQL database is preferred.

**PostgreSQL and PostGIS.** PostgreSQL is an object-relational database management system. This open source system can successfully compete against other commercial databases. PostgreSQL with PostGIS extension is dedicated for spatial data managing [19].

Unlike dBASE, PostgreSQL supports not only basic types of data but also, with PostGIS extension, spatial data types, such as point, line, polygon, and so.

PostgreSQL can continue to maintain good performance, even when the database size reaches 100GB. A communication with the database can be performed on the Internet through script languages, like PHP.

PostGIS is an extension to PostgreSQL dedicated to spatial data. PostGIS supports storing and using spatial data, as well as improves capabilities of managing spatial data. Spatial data and attribute data are stored in the database

by PostGIS. With specialized functions, relationships between entities of spatial data can be easily analyzed.

The SQL language can be used to directly query both spatial data and attribute data, so it is very easy to provide a variety of database services or web services. In case of data stored in files, when a user wants to look through data from a Geodetic Documentation Center, he has to download either all the data of selected type (for shapefiles) or all the data from a district (for mapfiles). When data are stored in a database a query may be limited to single objects. Moreover, PostGis extension can limit data only to a current view.

A downside of the approach is that data are downloaded once again when the view is revisited. To avoid this, the data may be cached in local memory and a database manager has to synchronize data from a database with local data.

### 2.3 Web Services

Web services are dedicated for data publication in Internet. Spatial data can be exposed as a Web Map Service. This service returns spatial data as an image. When vector data is required Web Feature Service has to be used instead. Both services are supported by iGeoMap as well as Tile Map Service, which delivers data divided into tiles.

**Web Map Service.** A Web Map Service [13] produces a visual representation of spatial data, which is not the data itself. A result map is an image in one of popular formats, such as PNG, GIF, or JPEG.

The specification of the service defines three operations: *GetCapabilities*, *GetMap* and *GetFeatureInfo*. All of them are encoded as a URL that is invoked on the WMS using HTTP GET and POST operations.

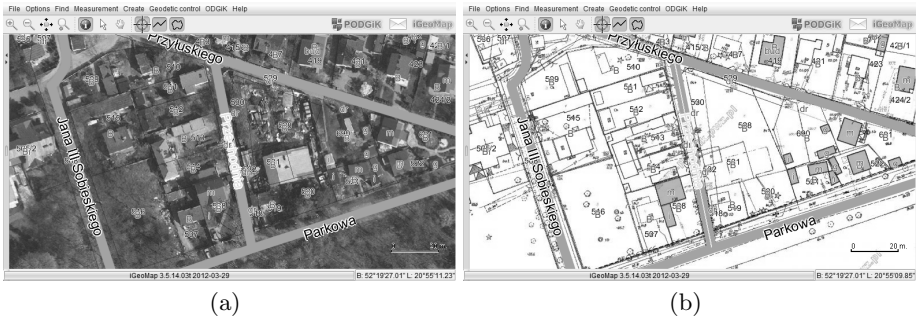
Every WMS server supports *GetCapabilities* operation. *GetCapabilities* operation returns general information about the service itself and specific information about the available maps.

*GetMap* operation produces a pictorial map image whose geospatial and dimensional parameters are defined by request parameters. This map is defined to be either a pictorial image or a set of graphical elements.

Unlike *GetCapabilities* and *GetMap*, *GetFeatureInfo* is an optional operation. It provides information about features in the maps that were returned by previous *GetMap* requests. This operation provides only information about an object defined by coordinates in the image. WMS service does not have advanced search engine itself [2].

In Geodetic Documentation Centers, WMS services are usually used to generate background for vector data. For example, airplane and satellite photos are presented as JPEGs generated by WMS server. Very often, connected WMS servers generate images from thirdparty resources.

Not all map resources have been converted into the vector format yet. For this reason, the resources are presented using a hybrid technique. Existing vector data are presented as objects from mapfiles. The rest of data is presented as an image



**Fig. 3.** Various utilizations of WMS. An opaque airplane photo as a background for vector data [3\(a\)](#) and a non-opaque PNG image as an extension of vector data [3\(b\)](#).

generated by a WMS service. In this case, a non-opaque PNG image is generated to merge both sources. Utilization of WMS is presented in Fig. [3](#).

Images are generated for a current view only. As in the case of PostGIS this approach limits the amount of downloaded data. Once again the problem of revisited area arises. In this case however, because of memory limitation of Java Virtual Machine, downloaded images cannot be stored and managed on a client-side. When the view is changing, the image memory has to be deallocated and a new request is generated even if the view is revisited.

**Web Feature Service.** Web Feature Service [12](#) enables the client to retrieve geospatial data through HTTP protocol [3](#). It allows clients and servers to share data without having to convert data between proprietary formats [15](#). WFS server provide an interface (defined in XML) for online data access. A request sent to a WFS server is a query for features. The result is encoded in Geography Markup Language [11](#).

In Geodetic Documentation Centers, WFS services are used as an interface for the data. The same data are a source for WMS service. Data delivered as an image can be managed by the client application. Examples are presented in the following sections.

**Tile Map Service.** Tile Map Service [10](#) works as WMS. However, data is presented as tiles determined by the grid from TMS specification. Each tile has the same size. A request that miss tiles defined generates an error. Tiles are defined on several levels. This allows generation of images on different levels of details [20](#).

The main problem with TMS is that the construction of adequate image from tiles has to be done by the client application. For fixed tiles with limited number of scales, it is not a trivial task. The main advantage of TMS is lower generation time than for WMS technique [14](#).

In Geodetic Documentation Centers TMS is used instead of WMS for data, which are frequently requested from the server.

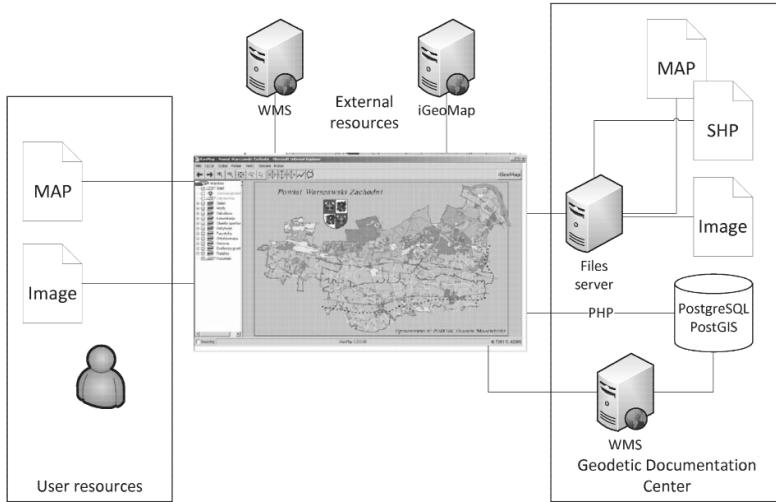


Fig. 4. Various sources for the iGeoMap application

### 3 iGeoMap

An Internet application iGeoMap [5] for presentation of Geodetic Documentation Center resources is developed in Java as an applet. The application groups data from various types of files (mapfiles, shapefiles and XML based files), databases (PostgreSQL and ORACLE), and web services (Web Map Service, Web Feature Service, and Tile Map Service). Data collected from all connected sources are presented in a tree structure as layers.

Apart from Geodetic Documentation Center resources the application may presents external data or user data. As an external source WMS servers may be used, for example, a national geoportal. Other centers that use iGeoMap can make their own data structure available. Users may add own mapfiles or images with georeference. An appropriate diagram is presented in Fig. 4.

Theoretically, the implementation in Java produces a multilanguage and multiplatform solution, which runs in most Internet browsers and operating systems. It is not true in practice. The interface is in fact multilanguage, but presented data from centers are described only in Polish. Moreover, data export based on IO operations is limited to Microsoft Windows systems

The application presents spatial data to two groups of users. Common citizens may get unrestricted information about parcels, buildings, and address points. Registered geodesists can also access restricted information such as control points description, geodetic maps, and registered geodetic works.

The following sections present typical use cases for both groups. The presented cases use data from various types of resources. The resource types were selected to optimize performance.

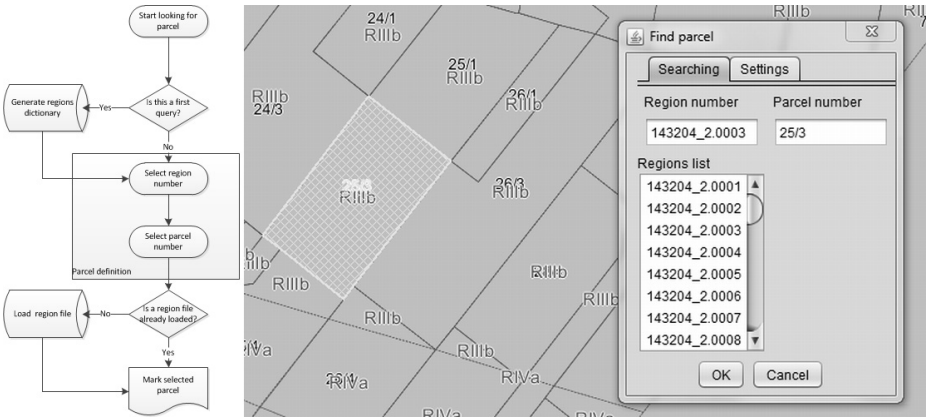


Fig. 5. A search of a parcel on the base of a region number and a parcel number

### 3.1 Use Cases

The most popular data are parcels, address points, and control points. A typical user looks mostly for these data. What is important for each of these user groups a different data source and searching mechanism is used.

**Parcels.** A parcel can be found when region and parcel numbers are known. The region number should be prefixed with the whole teryt number. This is not a problem for a geodesist. Common users can use usual names attached to regions.

Initially, parcels are not downloaded. The application can be started on two different administrative levels. In the first case, preloaded file contains districts.

In the second case, a frame of regions is created. In both cases, the preloaded file is a mapfile. The structure of a mapfile enables identification of districts or regions on the base of the teryt number. An identified object has information about linked files.

If the application starts on the district level, a dictionary of available regions can be created. Otherwise, the user should know the complete teryt number. In both cases, the region object holds information about linked parcels file.

This file is downloaded and the requested parcel is identified among downloaded objects. The procedure described is presented in Fig. 5.

The downloaded file stays in the memory and future queries concerning parcels from the region will be accomplished locally. Downloaded files maintain hierarchy so the queries among local data will be limited to the given region.

In this approach, downloaded data is limited only to essential information. In most cases, users are focused on the selected region or nearest neighborhood of the parcel. For that reason, detailed data from different administrative levels are not necessary. On the other hand, once downloaded, data may be freely managed in the client application. Such approach eliminates the repetitive downloading problem.



**Fig. 6.** A search of an address point on the base of a street name and a number

**Address Points.**

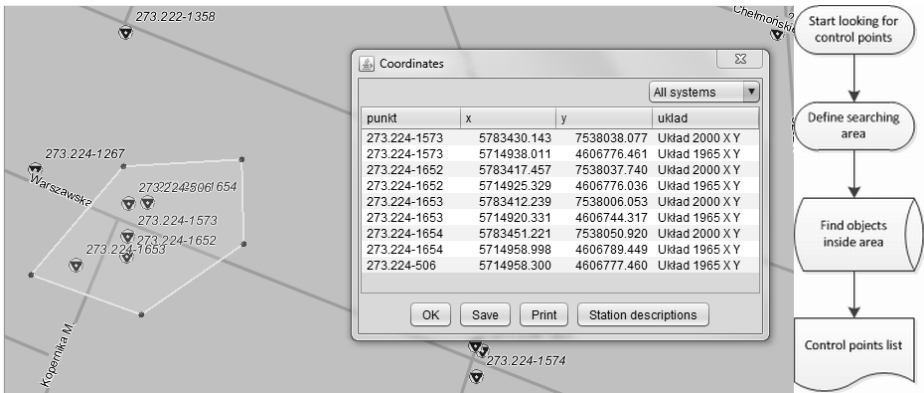
An address point defines the spatial localization of an address. In country districts, it is given by a city name, a street name, and a building number. In agglomerations, the city name is ignored. In both cases, all parameters should be known by the user, but the system enables also localization with partial data only. This possibility may be useful for ambiguous values such as *15a* or *15A* in the building number.

The main problem is the number of address points. In big agglomerations, the whole set of the address points cannot be displayed at once. To solve this problem two approaches are proposed. In the first one, the detailed information about address points are distributed among files. When a user starts a search for an address point a dictionary of streets is created on the base of a special file. Each position in the dictionary is mapped to a file. Such file contains a subset of address points limited to a few streets and if the street name is selected an appropriate file is downloaded. The main disadvantage of this approach is the limitation of the points presented only to downloaded streets. It is not a problem when the user wants to find a specific address, but a whole view of the addresses in the defined neighborhood cannot be presented.

In the second approach, WMS is used. The address points are drawn as an image. All address points in the current view are presented. In this case obtaining detailed data about presented points is more complicated.

The detailed information about a single address point can be obtained using *GetFeatureInfo* operation. For the selected address point visible in the image from WMS, the information about mapped object are presented.

The search procedure is more complex. When a user asks system addresses for the first time in a session, a street dictionary is generated from the database. It is not necessary in the following questions. The same dictionary can be used again. For the selected street and a building number, an SQL query is generated. The query can be sent to the database via PHP or as a WFS request. As a result, a localization of the address point is generated. The application focuses



**Fig. 7.** A search of detailed information about control points on the base of the localization

the view on the localization and an appropriate WMS image is generated. The image presents all address points in the view, but selected ones are marked. The described procedure with search parameters and the result is presented in Fig. 6.

**Control Points.** A control point presents information about coordinates of the point. The coordinates are given in several systems. For this reason, this information cannot be presented directly on a map.

Mostly geodesists are interested in control points, since information about coordinates is necessary in surveys. The control points have unique numbers and the description of a single point can be obtained basing on this identifier. However, the geodesist mostly needs information about all points in the surveyor’s works area.

In the application, the user has to mark an area of interest. If the control points were downloaded as a mapfile and appropriate objects are inside the marked area, unique points numbers would be extracted from the objects. The numbers are primary keys in the database, so detailed information can be obtained on their base.

In this approach, only base data (localization and a number) are downloaded as a file. Detailed information is generated only on special request. The described procedure Fig. 7.

In case of control points, the same problem as with address points occurs. In large areas, the number of points is significant. In such case, the control points may be represented by a WMS image. A marker is created and described as a PostGiS geometry. Using PostGiS functions, points inside this geometry are localized in the PostgreSQL database. The result brings detailed information about points.



### 3.2 Comparison with Other Approaches

Despite the fact that iGeoMap is a good solution to publish spatial data on the Internet a different approach to the problem can be chosen. Two of them, based on Google Earth and PHP scripts will be briefly presented.

**Google Earth.** One of the best applications for spatial data presentation is Google Earth. The application presents fast loading satellite and aerial images. It has also layers that contain geodetic data discussed in this paper such as streets and administrative objects.

Geodetic Documentation Centers can publish its data in Google Earth as KML files or WMS services. However, this form of publication can be unsuitable for restricted information such as detail information about control points. End users can also add local KML files to create a visualization of their spatial data against the background of the Centers data. The visualization may include 3D objects and time-based data [9]. Such effects cannot be achieved in iGeoMap.

The problem with Google Earth as a form of data publication lies in its search engine. The search is limited to the internal application layers. User cannot search added data.

The only exception are the information from Geodetic Documentation Centers published as a part of Google Earth layers, for example address points. However, such data are not transferred directly from the centers and can be outdated.

In sum, Google Earth should be preferred when a visualization is the main aim of the data presentation. However, iGeoMap is a better solution when a validity of data is crucial.

**PHP Scripts.** iGeoMap is a Java applet. This can be a problem for some groups of users. There are systems and mobile devices that do not support Java. In institutions with significant number of computers but without IT support it can be a problem to install Java Virtual Machine.

In such cases, a better solution is a publication based on PHP scripts. This form was selected in several projects including WebEWID [8], eMapa [6], and Geoportal 2 [7]. All above-mentioned projects use a WMS to produce a visual representation of spatial data. For this reason, navigation over the map results in a significant lag. Moreover, search engines are very limited. WebEWID and eMapa have search engines to look for parcels and addresses, but do not have tools for marking groups of objects similar to the mechanism described in the section 3.1. All the deficiencies pointed out are consequences of WMS limitations.

Search engines in the project discussed can be improved if conclusions from the presented search cases are drawn. There should not be any technical problems in application of solutions presented in the section 3.1 to PHP projects.

On the other hand, a serious technical problems occur when a thin html client is used instead of a Java applet. Unlike PHP, iGeoMap supports an import of multi-file structures such as SHP sources and a streams compression.



Fig. 8. Geodetic Documentation Centers, which use iGeoMap system

### 3.3 Reception

The iGeoMap application was introduced in several Geodetic Documentation Centers. Implementation locations are presented in Fig. 8. Among these centers are both ones from country districts as well as from major agglomeration such as Warsaw and Poznan.

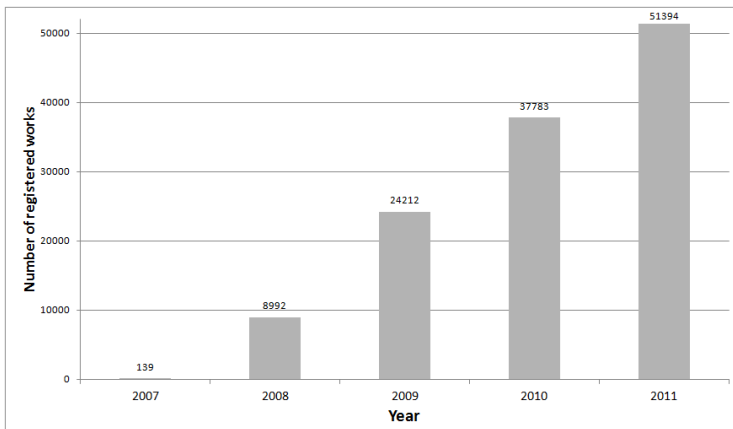


Fig. 9. Registered geodetic works illustrating a development of the application

A part of the published data is not available to common users. Geodetic Centers are obliged by the law to limit access to detailed information about geodetic data. For this reason, a description of control points can be exported only by a geodesist who has registered work in the area. Under this obligation geodesists have to register works in a system. The number of registered works illustrates the development of the system. Collected data, presented in Fig. 9, shows popularity of this solution among specialists.

The solution based on iGeoMap introduced in The Center of Geodetic and Cartographical Documentation in Warszawski Zachodni District was rewarded with The Best SDI Practice Award 2009 during eSDI-NET+ domestic competition.

## 4 Conclusions

In this paper, the problem of Internet publication of resources managed in Geodetic Documentation Centers was discussed. The main difficulty is a variety of spatial data sources. The centers manage files and databases, which can be made available as web services. All sources have their advantages and disadvantages, some of them were described in the paper, but the solution that publish resources on the Internet should work with all of them.

The proposed solution is iGeoMap. This application, developed as a Java applet, manages the resources mentioned. The application utilizes a mapfile format, which can be used to map levels of a local government administration into a files structure. The reception of the system shows that the solution has been accepted among specialists.

Different types of data sources are managed in an effective way. In the typical use cases, the selection of data source that limits number of downloaded data without limitation of functionality was presented.

The presented solutions can be use to develop applications based on PHP scripts. Such applications are popular because of portability and small technical requirements. However, search engines used in the existing projects are very limited. A search for spatial data in the projects can be improved when schemas presented in this paper and created on the base of our experiences with iGeoMap will be adopted.

## References

1. Directive 2007/2/ec of the european parliament and of the council of 14 march 2007 establishing an infrastructure for spatial information in the european community (inspire). Official Journal of the European Union 50 (April 2007)
2. Bai, Y., Yang, C., Guo, L., Cheng, Q.: Opengis wms-based prototype system of spatial information search engine. In: Proceedings of IEEE International Geoscience and Remote Sensing Symposium, IGARSS 2003., vol. 6, pp. 3558–3560 (July 2003)
3. Chunithipaisan, S., Supavetch, S.: The development of web processing service using the power of spatial database. In: 2009 2nd International Conference on Emerging Trends in Engineering and Technology (ICETET), pp. 832–837 (December 2009)

4. Environmental Systems Research Institute, Inc.: ESRI Shapefile TechnicalDescription (1998)
5. Geo-System (2011), <http://www.igeomap.pl>
6. Geo-System (2011), <http://e-mapa.net/>
7. GeoBid (2011), <http://www.geoportal2.pl/>
8. Geomatyka (2011), <http://geomatyka-krakow.pl/>
9. Grzenda, M., Kaczmarek, K., Kobos, M., Luckner, M.: Geospatial presentation of purchase transactions data. In: Ganzha, M., Maciaszek, L.A., Paprzycki, M. (eds.) FedCSIS, pp. 291–296 (2011), <http://dblp.uni-trier.de/db/conf/fedcsis/fedcsis2011.html#GrzendaKKL11>
10. Liu, Z., Pierce, M., Fox, G., Devadasan, N.: Implementing a caching and tiling map server: a web 2.0 case study. In: International Symposium on Collaborative Technologies and Systems, CTS 2007, pp. 247–256 (May 2007)
11. Open Geospatial Consortium: OpenGIS Geography Markup Language (GML) Encoding Standard (Version 3.2.1) [EB/OL] (2007)
12. Open Geospatial Consortium: OpenGIS Web Feature Service (WFS) Implementation Specification Version 1.1.0 (2009)
13. Open Geospatial Consortium: OpenGIS Web Map Service (WMS) Implementation Specification Version 1.3.0 (2009)
14. Qun, L.: Key technologies on rapid web publication of spatial data. In: 2010 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT), vol. 7, pp. 282–285 (July 2010)
15. Ribeiro, J., de Farias, O., Roque, L.: A syntactic and lexicon analyzer for the geography markup language (gml). In: Proceedings of IEEE International Geoscience and Remote Sensing Symposium, IGARSS 2004, vol. 5, pp. 2896–2899 (September 2004)
16. TopoGrafix: GPX: the GPS Exchange Format (2007), <http://www.topografix.com/gpx.asp>
17. Ye, L., Mao, J.: Spatial analysis on sql geography and geometry data. In: 2010 International Conference on Audio Language and Image Processing (ICALIP), pp. 1749–1752 (November 2010)
18. Ying-jun, D., Chong-chong, Y., Jie, L.: A study of gis development based on kml and google earth. In: INC, IMS and IDC, 2009, NCM 2009, pp. 1581–1585 (2009)
19. Zhang, L., Yi, J.: Management methods of spatial data based on postgis. In: 2010 Second Pacific-Asia Conference on Circuits, Communications and System (PACCS), vol. 1, pp. 410–413 (August 2010)
20. Zhang, Y., Li, D., Zhu, Z.: A server side caching system for efficient web map services. In: International Conference on Embedded Software and Systems Symposia, ICESSE Symposia 2008, pp. 32–37 (July 2008)

# Business Process Design from Virtual Organization Intentional Models

Luz María Priego-Roche<sup>1</sup>, Lucinéia Heloisa Thom<sup>1,2,\*</sup>, Agnès Front<sup>1</sup>,  
Dominique Rieu<sup>1</sup>, and Jan Mendling<sup>3</sup>

<sup>1</sup> Laboratoire LIG, Equipe SIGMA, BP 72, 38402 Saint Martin d'Hères, France  
{Luz-Maria.Priego-Roche, Dominique.Rieu, Agnes.Front}@imag.fr

<sup>2</sup> Universidade Federal do Rio Grande do Sul, Instituto de Informática,  
Av. Bento Gonçalves, 91501-970 Porto Alegre, Brasil  
lucineia@inf.ufrgs.br

<sup>3</sup> Wirtschaftsuniversität Wien, Augasse 2-6, 1090 Vienna, Austria  
jan.mendling@wu.ac.at

**Abstract.** Virtual Organizations (VO) have emerged as a new type of inter-organizational relationship for dealing with emerging challenges. The information system support for a VO poses new challenges to system design. Recent works define three levels of abstraction, namely an intentional level, an organizational level, and an operational level. For such a staged design, it is fundamental that artifacts defined on the different layers are consistent with each other. We address this problem based on a transformation approach. We illustrate the transformation from the intentional level towards the organizational level based on the 360° VisiOn and the BPMN process modeling language. The approach has been implemented in a prototype and validated using a case study from a regional stockbreeder union in Mexico.

**Keywords:** Virtual organizations, intentional modeling, organizational modeling, workflow activity patterns.

## 1 Introduction

Virtual Organizations (VOs) have emerged as a new type of inter-organizational relationship to deal with emerging challenges such as new competitors, new markets, or new customer needs. A VO is an alliance for integrating competencies and resources from independent companies, potentially geographically dispersed [13]. This integration has to be facilitated by appropriate information system support. A new challenge in this context is the need for methods for streamlining discussions among participant organizations, examining stakeholders requirements and analyzing the environment [2].

Different approaches of requirement elicitation can be used for modeling traditional organizations including i\* [23], KAOS [20], Maps [15], and scenarios [9]. The e<sup>3</sup>value approach [5] has been designed for service-oriented collaboration. In our work, the 360° VisiOn approach has been addressed to VOs. This approach defines three levels

---

\* Special thanks to the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior from the Brazilian government. In particular to the PNPd Program.

of abstraction for building the information system of a VO. First, the intentional level describes the “problem space” by identifying the objectives and collaboration. Second, the organizational level specifies the “solution space” by establishing the business process. Third, the operational level defines the “implementation space” by developing the information system. A central challenge is to establish a comprehensive specification which is consistent between the intra-organizational, the inter-organizational, and the extra-organizational perspective.

In this paper, we address the problem of defining business processes on the organizational level which are consistent with models on the intentional level. Our approach builds on a pattern-based mapping between the constructs of the first and the second level. The transformation requirements are aligned with the concept of workflow activity patterns (WAPs) in order to achieve semantically rich process models. The approach is based on a 360° VisiOn modeling tool as a transformation to BPMN. We use the case of a regional stockbreeder union in Mexico to demonstrate the applicability of our approach.

The paper is structured accordingly. Section 2 gives an introduction to the 360° VisiOn framework, Section 3 describes the intentional level of the framework and its underlying metamodel. Section 4 gives an overview of BPMN and defines the transformation rules from 360° VisiOn to BPMN. Section 5 introduces the WAPs for the enrichment process at the organization level. Section 6 shows the application of the approach for the case of the stockbreeder union in Mexico. Section 7 discusses the results in the light of related work. And finally, Section 8 summarizes our proposals and prospects for future study.

## 2 Overview of the 360° VisiOn

The 360° VisiOn proposes a framework for eliciting VO’s requirements from a vertical and a horizontal approach [13]. The levels of analysis are (see Figure 1.a):

- Vertical: Intentional (emphasizing the alliance, collaboration and common objectives), Organizational (formalizing the business processes) and Operational (executing the BP).

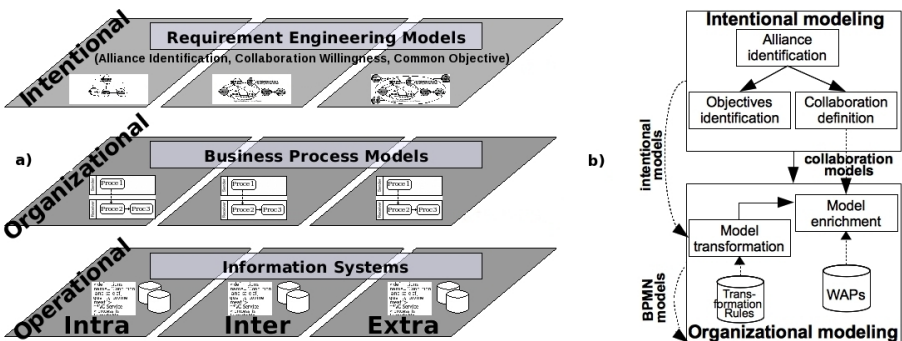


Fig. 1. The 360° VisiOn and building the VO’s business processes

- Horizontal: Intra-organizational (focusing on the relationships inside each organization forming the VO), Inter-organizational (focusing on the relationships among the organizations forming the VO) and Extra-organizational (focusing on the organizations of the external environment).

Figure 1b shows a simplified view of the intentional modeling based on the 360° VisiOn approach. These intentional models are the input for building the BPMN diagrams. We use the latter as a target model notation at the organizational level, since it is a standard notation being adopted by the business community as well as by many BPM tools. We carry out a transformation process based on transformation rules. As seen in Figure 1b, in this paper the collaboration diagrams are improved with an enrichment process based on WAPs.

### 3 The Intentional Level

The characterization of the 360° VisiOn is composed of the following aspects that guide the intentional models development and are shown in Figure 1:

*Alliance Identification* of the agreement, actors and services offered. The alliance establishes the relationship preserving organization's independence for continuing their own projects. Actors (e.g. stakeholders, users, organizations) identification is a recommended way for starting eliciting IS requirements [6]. Services identification defines the general output expected from these actors and their role in the service generation.

*Collaboration Willingness* characterizes the compromises each Member Organization plans to give to the VO in terms of its availability to the new relationship, the investments willing to make, the elements to be coordinated and the regulation of the expected behavior that would assure VO members good performance.

*Common Objective* characterizes the shared goal and the directions to be followed for achieving it. The latter could answer customer's needs (integral services), satisfy companies' objectives (to share costs, to create more effective processes), make new business (markets, products or services) or confront difficulties (absence of knowledge).

The simplified UML<sup>1</sup> meta-model of the three aspects composing the 360° VisiOn is shown in Figure 2. To illustrate them, two aspects are instantiated with graphical and textual models, the implementation was done with a software tool prototype [13] shown in Figures 3 and 4.

The **Alliance Identification** states a set of *organizations* forming a VO by subscribing an *alliance* to *offer* a service. To explore this aspect, different driving forces are considered: at the Inter level, the end consumer as the reason of the organization network existence, at the Intra level, each organization's functional actor that takes part of the service delivery and at the Extra level, the environmental organizations that influence the service.

Inspired by [11], Figure 3 presents the Inter and Extra views. *Internal Organizations* participate in the service as *Member Organization* that agrees to join the alliance and accordingly to acquire compromises and rewards beyond the service offered (e.g. Stockbreeders, Stockbreeders Associations, Freight & Trucking, Slaughter House and Meat

---

<sup>1</sup> Unified Modeling Language.

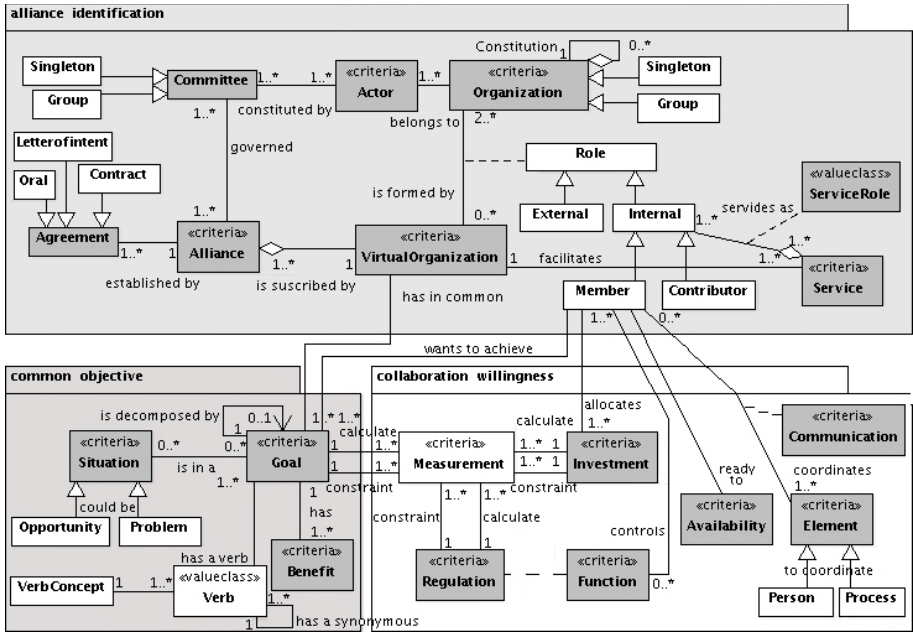


Fig. 2. The 360° VisiOn meta-model

Marketing), or as *Contributor Organization* that does not agree to join the alliance, however interacts directly with the VO either soliciting or rendering services. A *Contributor Organization* may become a *Member Organization* that may extend the VO boundaries (e. g. Supermarket Chains and Meat Consumers). The *Member Organizations* form the VO *Regional Stockbreeders Union of Tabasco (UGRT)* are delimited by a dotted oval. *External Organizations* influence the service like Government Regulators, Meat Importers and Leather Article Producers which have a respective Regulatory, Concurrently, and Complementary role in the Service. Members and Contributors provide one or more services, e.g. *Conditioning and Sale of Quality Bovine Meat*, detailed in the text model of the VO. These organizations play a *service role* (as service, direct, indirect, and auxiliary provider, or service consumer), the Service text model shows the Stockbreeder as an Indirect Provider that *Provides grass fed cattle livestock*.

A VO is subscribed by an *alliance* based on an *agreement* which is established orally or written with a letter of intent or a contract. This alliance is governed by a *committee* which is constituted by *actors* that belong to an organization. Either organizations and committees are formed by only one (*singleton*) or more than one (*group*) of them. Among the information agreed in the alliance is the facade to be expose between the VO and its clients or users: screen (only the VO is viewed), semi-screen (the VO and all or some organizations are viewed) and mirror (only the organizations are clearly viewed). In the Alliance Identification text model, a semi-screen facade was agreed allowing clients to view the UGRT and some of the member organizations.

The **Collaboration Willingness** arises Member Organizations responsibilities and advantages to the VO. The alliance formed among the Member Organizations allows to



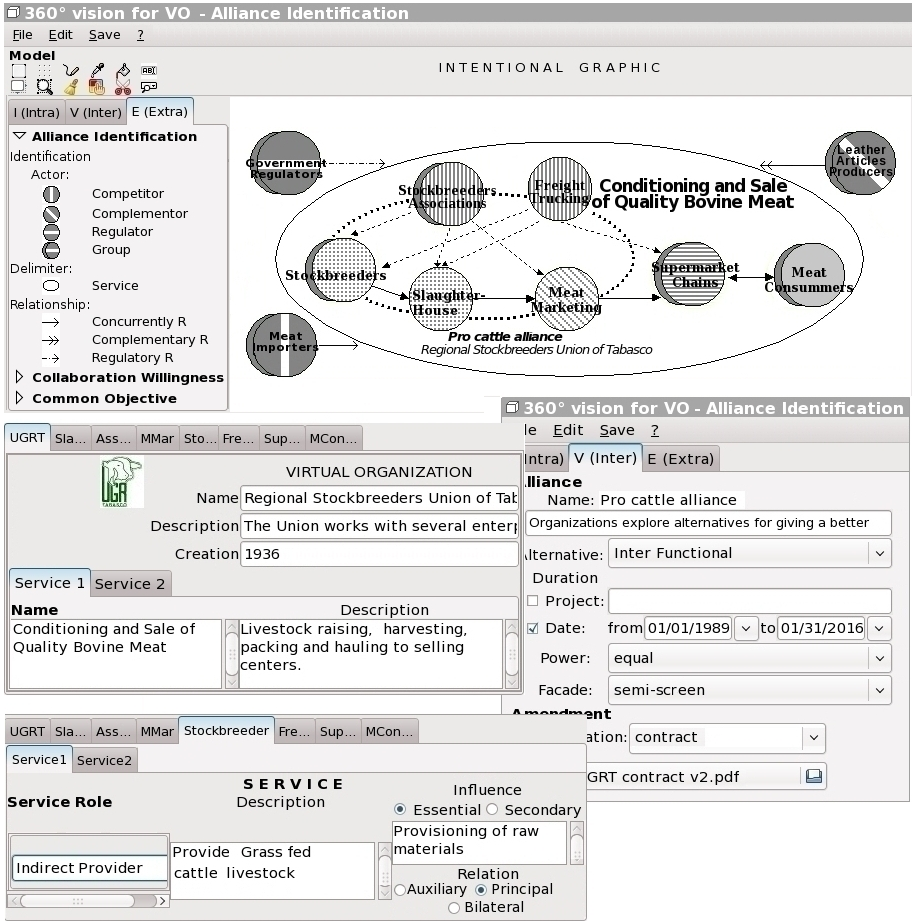


Fig. 3. The Alliance Identification models for the UGRT

explore collaboration not only for service provisioning but to find other means that benefit Member Organizations (e.g. better logistic coordination, resources optimization).

Firstly, from the meta-model each Member Organization is ready to give an *availability* to the VO in terms of time, priority to the relationship and adaptability to changes. It also allocates one or more *investments* to the VO. These investments can have one or many *measurement* to be calculated or to constraint them. A partial view of the investments graphic representation is given in Figure 4 showing a chart where the arcs bind the Stockbreeder Member Organization with the VO. The three investments are represented with dotted squares. The icon inside the square illustrates its type (Assets: financial or material, or Capital: human, relational or organizational [12]). The text model shows a financial asset investment (“Contribution”) described as “Give a contribution for each slaughtered cattle” which is fixed by an “Average Cost”. This contribution is event-triggered (when the cattle is slaughtered) and has a direct impact on the VO.

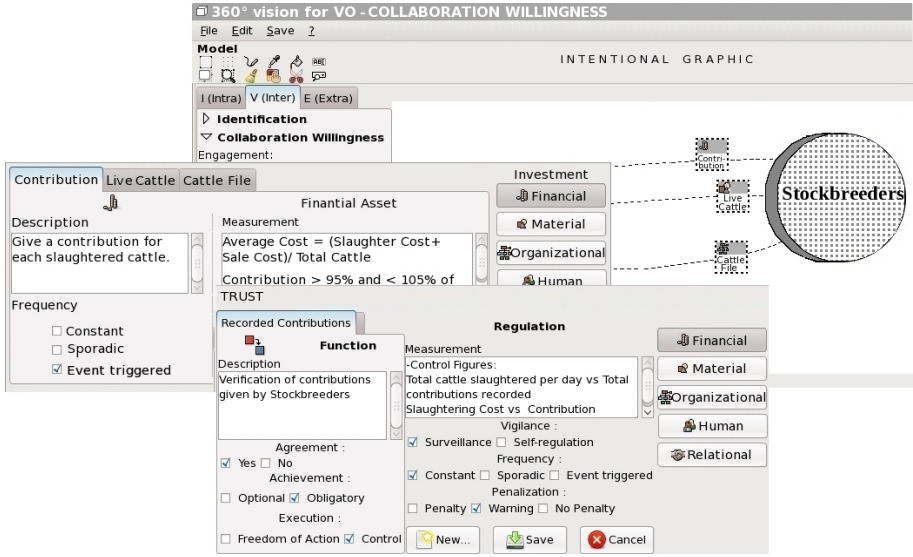


Fig. 4. The Investment models for the UGRТ

Secondly, each Member Organization has one or more *elements* to coordinate. An element can be a *person* or a *process*. The elements’ *communication* is described in terms of space (where), time (when) and movement (how) [17].

Finally, Trust fosters an effective collaboration. Each Member Organization may or may not have a *function* to establish alertness for assuring the compromises well execution. The *regulation* of these functions can be controlled or not by other Member Organization in a constant, sporadic or event-triggered frequency and in case of violation to these controls a penalization (penalty, no penalty, or warning) can be established. These regulations are quantified through *measurement*, a set of calculations and constraints. The text model linked to the “Contribution” Investment named “Regulation”, consist of a “Verification of contributions given by Stockbreeders”. Contributions are constantly surveyed by other organizations and there is a “Warning” in case of miss achievement.

Finally, the **Common Objective** characterizes the goals all Member Organizations have in common to form a VO (Inter level) and those each Member Organization wants to achieve to justify the alliance (Intra level). A goal is described by a verb, a targeted object and a complement. Goals have one or more expected *benefits* representing the foreseen goal yield and can be *measured* by constraints or calculations. Usually, objectives can emerge from a *situation* that could be classified in *opportunity* (circumstances favoring the alliance) or *problem* (difficulties that justify exploring the alliance).

## 4 Obtaining Basic Business Processes Models

The 360° VisiOn models provide a clearer understanding of a VO from an intentional perspective. Business process modeling allows organizations to understand and

communicate their internal business processes in a formal or semi-formal graphical notation from an organizational perspective. It also allows organizations to analyze, model, execute, and monitor their business processes. UML, EPC (Event Process Change), and BPMN are alternatives for representing business processes. The transformation from the intentional to the organizational models is a complex task that involves establishing a formal model to review and verify the transformation and selecting the correct equivalent representation between the two different models in order to assure accurate and coherent BPM.

#### 4.1 Transformation Process: From Intentional to Organizational Models

Once the alliance is identified (Figure 3), each Member Organization determines the resources willing to give and expecting to receive (Figure 4) as well as the common objective that unifies them for working together. When the intentional level modeling is considered complete, it is required to define the Member Organizations business processes needed by the VO for satisfying the service and contributing to the consented objectives. As stated before, one user defines the intentional level, then s/he gives it on to another user to complete the models at the organizational level.

A VO can facilitate many services, where each service is represented with one BPMN diagram. Based on [22] we state a high level business process as the description of a set of one or more linked procedures representing the know how of each organization which collectively satisfy the defined service. The complete BPMN notation from [11] is the basis of the translation from the 360° VisiOn to BPMN. The translation process can be seen as a partial function  $f : \mathbb{C} \rightarrow \mathcal{P}(B)$  mapping from 360° VisiOn Criteria domain  $\mathbb{C}$  to the BPMN Construct  $\mathbb{B}$  domain. The first step of the strategy followed consists in analyzing each criteria of the intentional aspects and mapping it to the more suitable BPMN constructs.

The first step is based on the alliance identification intentional models. Below, we define the rules for assigning these Criteria of the 360° VisiOn from the simplified UML meta-model given in Figure 2 to the BPMN constructors.

#### 4.2 From the Alliance Identification to BPMN Collaboration Diagram

The virtual organization is denoted by  $VO$  and it facilitates one or more services. Member organizations, contributor organizations and external organizations are denoted by the sets  $O_m$ ,  $O_c$  and  $O_e$  respectively. A virtual organization service  $VOS$  will be denoted as a tuple  $(MO_s, CO_s, EO_s, SR_s)$  where

- $MO_s$  is the part of the virtual organization related with members that facilitate the service,
- $CO_s$  is the part of the virtual organization related with contributors that facilitate the service and
- $EO_s$  is the part of the virtual organization related with externals that influence the service.
- $SR_s$  is the collection of Service Roles for the service.

$MO_s$ ,  $CO_s$ ,  $EO_s$  and  $SR_s$  do not have direct representations in the UML meta-models of Figure 2; nevertheless for formalization purposes, they will be considered as  $\mathbb{C}$  constructs that could be built from the UML meta-models.  $MO_s$  is represented by a tuple  $(O_{ms}, SR_{ms})$  where

- $O_{ms} = \{o_{ms1}, \dots, o_{msi}\} \subseteq O_m$ , is the set of member organizations that participate in the service, and
- $SR_{ms} = \{sr_{ms1}, \dots, sr_{msi}\} \subseteq SR_s$  is the set of roles of member organizations in the service and  $sr_{msa}$  corresponds to the service role served by  $o_{msa}$ .

$CO_s$  is represented by a tuple  $(O_{cs}, SR_{cs})$  where

- $O_{cs} = \{o_{cs1}, \dots, o_{csj}\} \subseteq O_c$  is the set of contributor organizations that participate in the service, and
- $SR_{cs} = \{sr_{cs1}, \dots, sr_{csj}\} \subseteq SR_s$  is the set of roles of the contributor organizations in the service and  $sr_{csa}$  corresponds to the service role served by  $o_{csa}$ .

$EO_s$  is a set of external organizations  $\{o_{es1}, \dots, o_{esk}\}$ .  
 $SR_s$  is a set of service roles  $\{sr_1, \dots, sr_n\}$ .

Therefore, the rules for assigning the Criteria of the Alliance Identification are:

**Rule 1:** Each Service facilitated by the VOS is represented by one BPMN Collaboration Diagram which contents are determined by the application of all the following rules, from Rule 2 to Rule 11.

**Rule 2:** Each  $MO_s$  is represented by one BPMN Pool contained in a BPMN Process.

**Rule 3:** Each  $CO_s$  is represented by one BPMN Pool contained in a BPMN Process.

**Rule 4:** Each  $EO_s$  is represented by one BPMN Pool contained in a BPMN Process.

**Rule 5:** The BPMN Pools obtained by the application of Rules 2 to 4 are included in the representation of the BPMN Collaboration construct.

**Rule 6:** Each Member Organization is represented by one BPMN Lane which includes a reference to its service role description.

**Rule 7:** The VO facade is represented by one BPMN Lane inside the  $MO_s$  BPMN Pool if and only if facade is screen or semi-screen.

**Rule 8:** Each Contributor Organization is represented by one BPMN Lane which includes a reference to its service role description.

**Rule 9:** Each External Organization is represented by one Lane which includes a reference to its service role description.

**Rule 10:** For each ServiceRole there is one Activity Object stereotyped with its Role-Name. This Activity Object contains three BPMN constructors: one sub-Process, several sequenceFlows (between organizations within the same pool) and messageFlows (between organizations in different pools).

**Rule 11:** For each ServiceRole there is a collection of messageFlows, one for each collaboration between service roles associated to organizations in other pools.

## 5 Organizational Level Enrichment

Based on transformation rules we have obtained the basic BPMN diagram that includes pool, lines, sub-processes and flows constructs needed to represent the service from the Alliance Identification models. To complement the business process development we use recurrent business functions called WAPs and illustrate the enrichment process using one of the collaboration willingness models. We extract the sequence of BPMN Construct from the  $\mathbb{B}$  domain that match with the WAPs domain  $\mathbb{W}$ , where  $\mathbb{W} \subseteq \mathbb{B}$ .

### 5.1 An Overview on Workflow Activity Patterns

A WAP refers to the description of a recurrent business function (e.g. task execution request, approval) as it can be frequently found in business processes. In this paper we use a subset of the seven activity patterns defined in [19] to enrich the transformations from the intentional to the organizational level of a VO and obtain process models fragments executed by the organizations. The activity patterns comprise the following behavior:

*Approval (WAP1):* An object (e.g. a document) has to be approved by one or more roles. The evaluation can be executed only once (single approval) or multiple times. In the latter, it can be done in sequence (iterative approval) or in parallel (concurrent approval).

*Question-Answer (WAP2):* An actor might have a question before or during an activity in the process. The pattern formulates such question, identifies a role who is able to answer it, sends the question, and waits for response (single question-answer). The question can be sent to multiple roles or actors resulting in multiple answers (multi-question-answer).

*Unidirectional Performative (WAP3):* A sender requests the execution of an activity from a receiver (e.g. a human) involved in the process. The sender continues execution of his part of the process immediately after having sent the request. A requestor sends an activity execution request to one receiver (Single-Request) or to many (Multi-Request).

*Bi-directional Performative (WAP4):* A sender requests the execution of a particular activity from another role involved in the process. The sender waits until the receiver notifies him that the requested activity has been performed.

*Notification (WAP5):* The status or result of an activity execution is communicated to one or more process participants.

*Information Request (WAP6):* An actor requests certain information from a process participant. S/he continues process execution after having received the desired information.

*Decision (WAP7):* During process enactment, the execution of one or multiple activities is requested. Depending on the executions results the process continues through one or several branches. The pattern allows to include a decision activity with connectors to different subsequent execution branches (each of them is associated with a specific transition condition if evaluated to true, its branches are selected).

Figure 5 provides a summary of the WAPs in the BPMN notation. The Approval Pattern (WAP1), for example, starts with a send activity indicating an approval request. A receiver performs the approval and the result of the approval is then sent to the requester. Generally, multiple activity patterns can be composed in a process model using workflow patterns like Sequence, AND-Split, AND-Join or XOR-Split. An empirical study, in which more than 200 real-world process models were analyzed, confirmed the existence of the seven activity patterns [18,19]. The study showed that the analyzed process models can be designed based on the investigated WAPs; i.e., the set of identified activity patterns is necessary as well as sufficient to design the 200 process models, at least at a certain level of granularity.

In [18] the frequency of co-occurring activity patterns in real world process models was also analyzed. In addition a process model mining tool was developed to be used for identifying the activity patterns co-occurrences. The miner allows analyzing process models instead of event logs as proposed in literature. This can be considered as a very important functionality to automatically identify activity patterns co-occurrences (e.g., the pattern pair APPROVAL  $\rightarrow$  NOTIFICATION) in real-world process models. To achieve a precise semantics the patterns had been formalized using Pi-calculus [19]. A process model specified in Pi-calculus can express the dynamic behavior of the process, thus making it possible to verify formal properties of the model like soundness (e.g., absence of deadlocks and livelocks) and model equivalence [8].

## 5.2 From the Collaboration Willingness to the BPMN Diagram

Complementing the BPMN diagram with the Collaboration Willingness models.

Based on the Collaboration Willingness, we illustrate part of the Engagement sub-spec that refines the *MO*'s sub-processes, to do so, we use transformation rules and the WAPs. The Investments each Member Organization makes to the VO is denoted as follow:

- $I_s$  is the collection of investments made to the service.
- $I_{ms} = \{i_{ms1}, \dots, i_{msp}\} \subseteq I_s$  is the set of investments of the member organizations in the service and  $i_{msa}$  correspond to the investment made by  $omsa$ .

**Rule 12:** One *Investment* given by a *MO* is represented by a BPD Collaboration construct which content is determined by the application of all the following rules, from Rule 13 to Rule 20.

**Rule 13:** The *Frequency* is represented by the BPMN Start Event to indicate the investment process start. If the *Constant* or *Sporadic* option is selected, a Timer Start Event is chosen to indicate the specific time-date or cycle. If the *Event triggered* option is selected, the BPMN Conditional Start Event is chosen to indicate that a condition is true.

**Rule 14:** The *Vigilance* from the Regulation sub-aspect expresses if the tasks performed by an organization have to be notified to other organizations. *Vigilance* is represented by a Bi-Directional Performative (WAP4) if the *Surveillance* option is selected with the variants: requested by one (Single-Request) or many (Multi-Request) organizations. If the *Self-Regulation* option is selected, the WAP3 Unidirectional Performative is used.

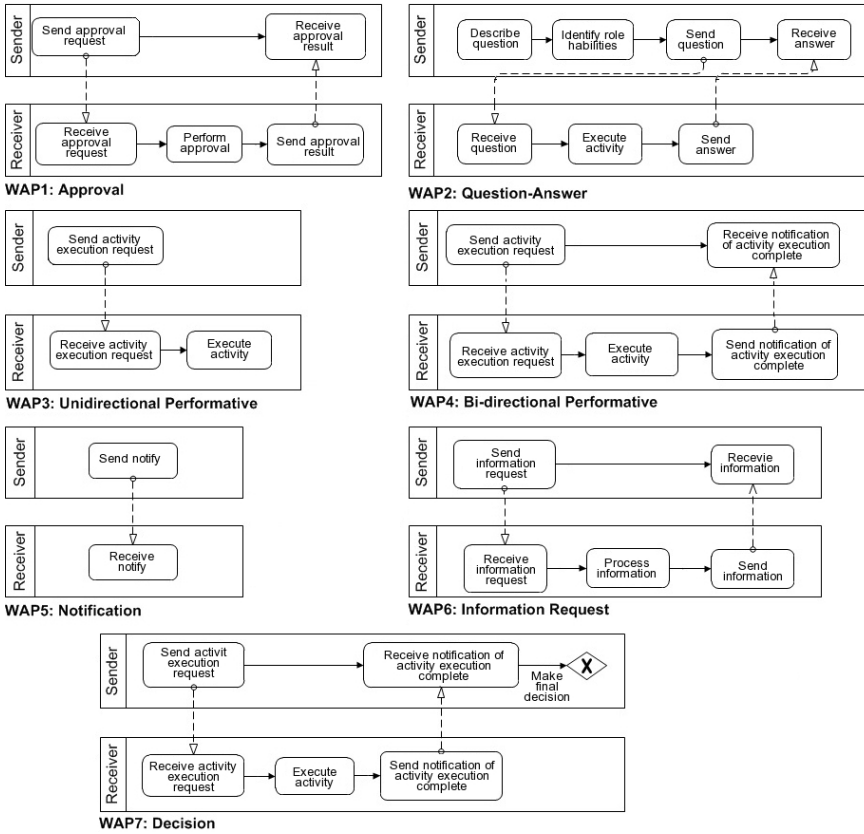


Fig. 5. Summary of Workflow Activity Patterns

- Rule 15:** The *Investment* given by the Member Organization is represented with the Bi-Directional Performative (WAP4) if the *Surveillance* option is selected, with either two variants: requested by one (Single-Request) or many (Multi-Request) organizations. The Unidirectional Performative (WAP3) is represented if the *Self-Regulation* option is selected.
- Rule 16:** The *Frequency* from the Regulation sub-aspect specifies the *Vigilance* for the BPD Collaboration when the *Surveillance* option is selected. *Constant* meaning that *Surveillance* (notification) is needed in all steps of the process, *Sporadic* when needed in some steps and *Event triggered* only when a condition is true.
- Rule 17:** The *Measurements* defined in the *Investment* are used to describe the Business Rules of the BPMN Business Rule Task in the WAP of Rule 14.
- Rule 18:** The *Measurements* defined in the *Regulation* is complemented with the BPMN Business Rule Task in the assigned WAP of Rule 17.
- Rule 19:** The Data Object of Rule 15 represents the *Investment* (Assets or Capital) to be given from one organization to another including a descriptor reference.

**Rule 20:** The *Penalization* from the Regulation sub-aspect is represented by one BPMN Exclusive Gateway to fork to the Activity Object stereotyped with the *Penalty* or *Warning* option selected.

Having the BPMN diagram, more semantic can be given to them by:

- Specifying the information of the BPMN Conditional Start Event or the Timer Start Event defined in Rule 13: *when cattle slaughtered* for the former; or *every two weeks, every month, at the end of the year* for the latter.
- Defining the Single or Multi Request variants of Rule 14 and 15 (e.g. to which organizations the execution notification has to be sent).
- Establishing the BPMN constructs for controlling the *Sporadic* or *Event triggered* Regulation. This is not explicit in Rules 14 and 15 since the WAP4 is used in the collaboration process to represent the “Constant Surveillance”.
- Representing Investments with Data Objects, to exchange elements like money, materials which are not available in BPMN.
- Having a multi instance participant for the lane construct (for representing the group of Stockbreeders for example which is not available in BPMN).

## 6 Prototypical Implementation for the Stockbreeder Union

Figure 6 partially shows the alliance identification mapping between the 360° VisiOn and the BPMN diagrams. Based on the models of Figure 3 it represents the service “Conditioning and Sale of Quality Bovine Meat” of the UGRT case study. On the left hand, the corresponding *f* definition that creates the collaboration diagram representing the service facilitated by the VO is shown, as well as other constructs. We use the following operators to write these functions: **if** <condition> **then** <value> for expressing conditionals, and **+** for string concatenation. Different font styles are used to distinguish elements in BPMN and XML constructs: **bold** for operators, *typewriter* for variables, *italic* for functions, and **\*\* FOR COMMENTS** to clarify the construct (for more detail see [14]).

The BPMN diagram corresponding to the mapping of the UGRT service is generated by *function* (a). Member Organizations (Stockbreeders Associations, Stockbreeders, Slaughterhouse, Freight Trucking and Meat Marketing, Rule 6) and the “semi-screen Facade” Rule 7 are lanes in the pool “UGRT Members” (Rule 2). They are generated by *functions* (b) and (c). Each Member has a service offered and represented as a sub-Process inter-connected with sequenceFlows (Rules 10 and 11). These sub-Processes are named as the service role, e.g. “Provide grass fed Cattle Live-stock” for the Stockbreeder and generated by *functions* (d) and (e).

Figure 7 shows the BPD diagram corresponding to the mapping of the Stockbreeder Investments shown in Figure 4 according to rules 12 to 20. The Member Organizations involved in this process are the Slaughterhouse, the Stockbreeders, and Meat Marketing. The “Contribution” investment is represented by this collaboration diagram (Rule 12) which starts with a Conditional Start Event named *cattle is slaughtered* as described in Rule 13. The Surveillance is described with the WAP4 (Rule 14) indicating that a notification of the activities execution is needed. This WAP4 is refined with the Measurements of Rule 17 in the form of Business Rules. These Business Rules define how



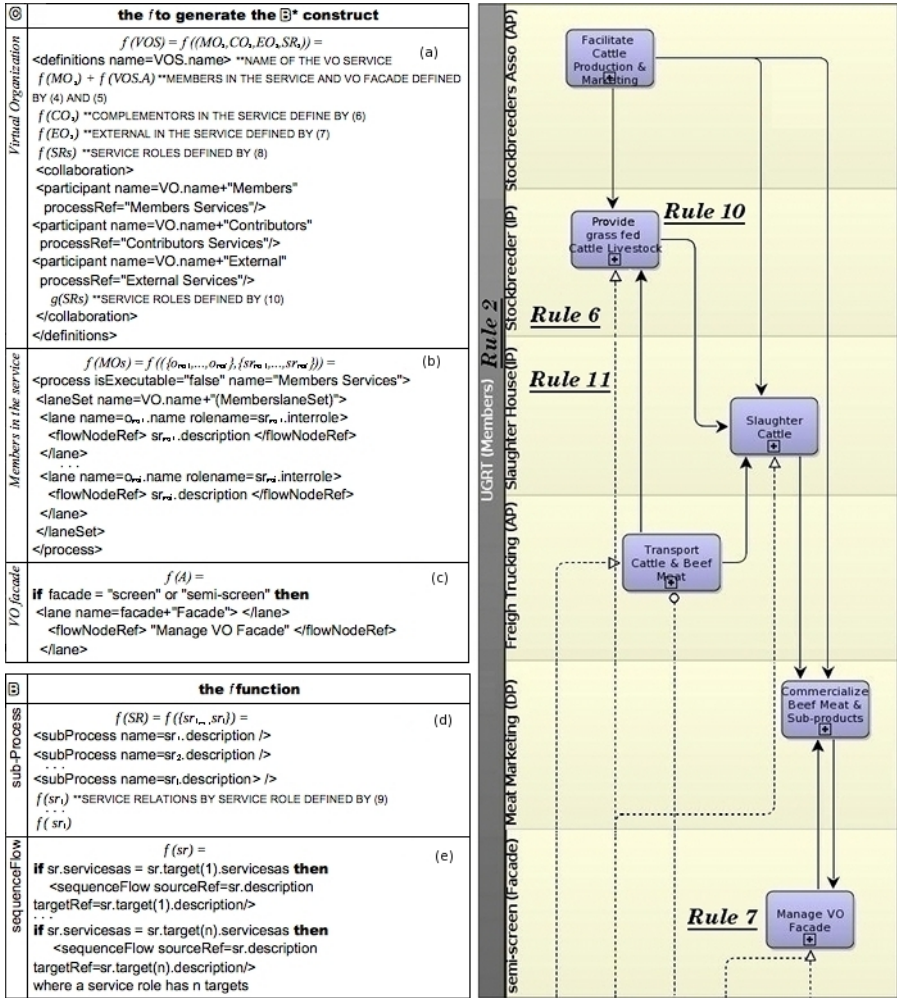


Fig. 6. Some of the Alliance Identification transformation rules

the Contribution is calculated e.g. “between 95% and 105% of the Average Cost”. A second activity pattern (WAP4), is used to give the Contribution (Rule 15), respecting that *Constant Surveillance* is required. The rules that complement this WAP are from the Regulation *Measurements* of Rule 18 (e.g. “Control Figures”). An activity task is used to express the *Penalization* chosen in case of miss achievement, in this example is a “Warning” (Rule 20).

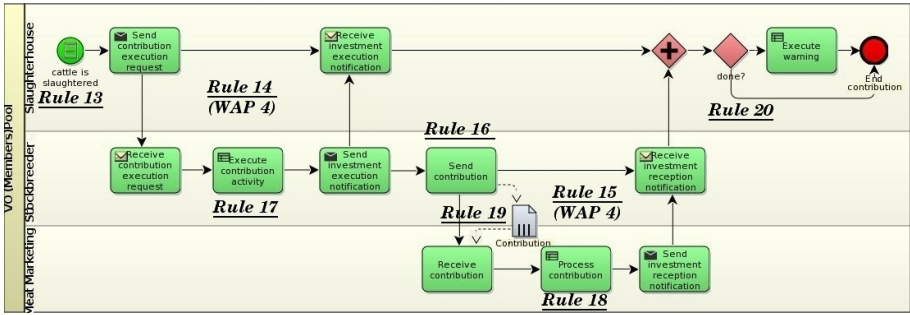


Fig. 7. The Investments of the Collaboration Willingness transformation rules

## 7 Related Work

i\*, KAOS, Map, scenarios and e<sup>3</sup>value are well known RE modelling approaches that have evolved and matured, they are a source of many publications and offer tools as compiled in [10,214]. Business processes are usually developed with no reference to intentional models. Nevertheless, several works argue the need of a combination of intentional models and BPMN. [7] applies i\* to express changes during the business process life cycle and [3] proposes reducing the gap between BPM and agent software paradigm. [4] determines goal satisfaction from KAOS model to business process, [16] shows how the Map formalism can capture goal achievement variability in business process models, and [21] identifies from e<sup>3</sup>value, the value objects concerned in the business process design. Firstly, instead of adapting the existing RE modeling approaches to VOs, in this paper we rely on the 360° VisiOn that as far as we know, is a novel and specifically conceived approach to guide and express requirements for VOs. Secondly, although these proposals define transformation rules, they lack of formalization. The work presented in this article offers a formalization to facilitate the automatization of the transformation process. In contrast to a manual approach and linked to the number of model elements, building an appropriate tool support saves time, avoids errors and provides standard quality BPM.

## 8 Conclusions

In this paper we reported on an approach which allows to design basic business process diagrams from information obtained at the intentional level of VOs. In particular we provided partial answers for the causals stated in the introduction:

The intentional level provides information about the problem space and the organizational level provides information about the possible solutions. We have formalized and developed a set of transformations rules principled on BPMN to assign the adequate BPMN items representations. WAPs showed to be very effective for representing single/multi participants either requesting the execution of activities or being notified about executed activities. The WAPs have also help to add more semantics and details

for the activities description. We use the XML semantic namespace defined by [11] to generate the BPMN-XML code to increase interoperability.

At the intentional level, a prototype tool has been developed for automatically generate the graphic models from the captured instances. A new functionality will be added to automate the model transformation based on the proposed method. The intentional level includes rich information that can be used in process models (e.g. object exchange, collaboration control) that can reduce process complexity, design time at the operational level. Nevertheless, assuring model traceability remains a difficult challenge not considered in this work.

Finally, we have tested our proposal in the agroindustry and health care domains. Future work will be done to validate it in other business domains and different types of VOs with and without this transformation approach. Moreover, we are going to test how the WAPs can be adapted and/or which are the new variants to be used for representing process models executed by different types of VO and the goals defined at the intentional level. We are considering to validate the completeness and correction of the obtained BPMN diagrams by comparing process design with and without using our approach, and by verifying the obtained BPMN diagrams with companies' end users.

## References

1. Basole, R.C., Rouse, W.B.: Complexity of service value networks: Conceptualisation and empirical investigation. *IBM Systems Journal* 47(1), 53–70 (2008)
2. Cheng, B.H.C., Atlee, J.M.: Research directions in requirements engineering. In: *FOSE 2007: 2007 Future of Software Engineering*, pp. 285–303. IEEE Computer Society, Washington, DC (2007)
3. Cysneiros, L.M., Yu, E.: Addressing agent autonomy in business process management-with case studies on the patient discharge process. In: *Proc. 2004 IRMA Conference* (2004)
4. Ghose, A.K., Koliadis, G.: Relating business process models to goal-oriented requirements models in kaos. *Faculty of Informatics-Papers*, p. 573 (2007)
5. Gordijn, J.: *Value-based Requirements Engineering: Exploring Innovative e-Commerce Ideas*. PhD thesis, Vrije Universiteit Amsterdam (2002)
6. Hickey, A.M., Davis, A.M.: Elicitation technique selection: How do experts do it. In: *Proceedings of the 11th IEEE International Conference on Requirements Engineering*, p. 169. IEEE Computer Society (2003)
7. Koliadis, G., Vranesevic, A., Bhuiyan, M.A., Krishna, A., Ghose, A.K.: Combining *i\** and BPMN for Business Process Model Lifecycle Management. In: Eder, J., Dustdar, S. (eds.) *BPM Workshops 2006*. LNCS, vol. 4103, pp. 416–427. Springer, Heidelberg (2006)
8. Li, C., Reichert, M., Wombacher, A.: On Measuring Process Model Similarity Based on High-Level Change Operations. In: Li, Q., Spaccapietra, S., Yu, E., Olivé, A. (eds.) *ER 2008*. LNCS, vol. 5231, pp. 248–264. Springer, Heidelberg (2008)
9. Maiden, N.A.M.: Crews-savre: Scenarios for acquiring and validating requirements. *Journal for Automate Software Engineering* 5(4), 419–446 (1998)
10. Nuseibeh, B., Easterbrook, S.: Requirements engineering: a roadmap. In: *ICSE 2000: Proceedings of the Conference on The Future of Software Engineering*, pp. 35–46. ACM, New York (2000)
11. OMG. Business process model and notation (BPMN). Technical Report 2, Object Management Group (June 2010)

12. Parung, J., Bititci, U.S.: A conceptual metric for managing collaborative networks. *Journal of Modelling in Management* 1(2), 116–136 (2006)
13. Priego-Roche, L.-M., Rieu, D., Front, A.: A 360° Vision for Virtual Organizations Characterization and Modelling: Two Intentional Level Aspects. In: Godart, C., Gronau, N., Sharma, S., Canals, G. (eds.) *I3E 2009. IFIP AICT*, vol. 305, pp. 427–442. Springer, Heidelberg (2009)
14. Priego-Roche, L.M.: *Intentional and Organizational Information Systems modelling for Virtual Organizations*. PhD thesis, University of Grenoble (2011)
15. Rolland, C., Prakash, N.: Bridging the gap between organisational needs and ERP functionality. *Requirements Engineering Journal* 5, 180–193 (2000)
16. Rolland, C., Prakash, N.: On the adequate modeling of business process families. In: *8th Workshop on Business Process Modeling, Development, and Support (BPMDS 2007)*, Electronic Ressource (2007)
17. Salvador, T., Scholtz, J., Larson, J.: The denver model for groupware design 5Yeeeeee Haaaaaa? In: *Conferencing Applications. CSCW 1992, CHI 1995, Denver Colorado*, vol. 28(1), pp. 52–58 (1996)
18. Thom, L.H., Reichert, M., Chiao, C.M., Iochpe, C., Hess, G.N.: Inventing Less, Reusing More, and Adding Intelligence to Business Process Modeling. In: Bhowmick, S.S., Küng, J., Wagner, R. (eds.) *DEXA 2008. LNCS*, vol. 5181, pp. 837–850. Springer, Heidelberg (2008)
19. Thom, L.H., Reichert, M., Iochpe, C.: Activity patterns in process-aware information systems: Basic concepts and empirical evidence. *International Journal of Business Process Integration and Management* 4(2), 93–110 (2009)
20. van Lamsweerde, A., Darimont, R., Letier, E.: Managing conflicts in goal-driven requirements engineering. *IEEE Transactions on Software Engineering* 24(11), 908–926 (1998)
21. Weigand, H., Johannesson, P., Andersson, B., Bergholtz, M., Edirisuriya, A., Ilayperuma, T.: Value object analysis and the transformation from value model to process model. In: *Enterprise Interoperability*, pp. 55–65 (2007)
22. WFMC. *Terminology & glossary. Technical Report 3, Workflow Management Coalition, UK* (February 1999)
23. Yu, E.S.K.: Towards modeling and reasoning support from early-phase requirements engineering. In: *IEEE 3th International Symposium on Requirements Engineering, Annapolis MD, Janvier 5-8*, pp. 226–235 (1997)

# A Novel Approach to Modeling Context-Aware and Social Collaboration Processes

Vitaliy Liptchinsky, Roman Khazankin,  
Hong-Linh Truong, and Schahram Dustdar

Distributed Systems Group, Vienna University of Technology,  
Argentinierstrasse 8/184-1, A-1040, Vienna, Austria

[lastname@infosys.tuwien.ac.at](mailto:lastname@infosys.tuwien.ac.at)

<http://www.infosys.tuwien.ac.at>

**Abstract.** Companies strive to retain the knowledge about their business processes by modeling them. However, non-routine people-intensive processes, such as distributed collaboration, are hard to model due to their unpredictable nature. Often such processes involve advanced activities, such as discovery of socially coherent teams or unbiased experts, or complex coordination towards reaching a consensus. Modeling such activities requires an expressive formal representation of process context, i.e. related actors and artifacts. Existing modeling approaches do not provide the necessary level of expressiveness to capture it. We therefore propose a novel modeling approach and a graphical notation, demonstrate their applicability and expressivity via several use cases, and discuss their strengths and weaknesses.

**Keywords:** Process Modeling, Social Context, Collaboration, Visual Language.

## 1 Introduction

Companies strive to retain the knowledge about their business processes by modeling them. If captured accurately, such knowledge allows us to analyze, improve, and execute those processes with higher efficiency. Although a variety of techniques and tools have been introduced for Business Process Modeling (BPM), nevertheless, modeling of highly dynamic non-routine processes such as human collaboration is still a subject for discussion in research [16].

While collaboration in general means working together to achieve a goal [8], the more narrow notion of creative human collaboration implies working together to design or improve some artifact (a piece of software, a wiki page, a product design, an article of law, a research paper, etc.). With proliferation of collaboration software, such as groupware or wikis, the manner of such collaboration has taken the form of incremental contributions to a network of shared documents. Relations between documents, actors, and other artifacts may influence the collaboration process. For example, some tasks should be done by actors chosen based on social relations, actions on some documents should not be performed

before related documents reach certain condition, or a change in a related document might force to re-do an activity. Although artifact-based process models have already been researched [19,24], existing modeling approaches do not emphasize the relations between artifacts as process “driving force”, and, therefore, either do not provide the needed expressivity to capture this logic, or are inefficient because of the excessive complexity. We thus propose a novel modeling approach and a graphical notation for collaboration processes, the key principle of which is to treat each document’s evolution as an individual process which is explicitly influenced by the states of related documents and patterns in surrounding social network. We propose to formalize the relations in line with the data from collaboration software, e.g., two developers can be considered related if they committed code to the same project folder in a source code repository. The amount of such data will grow with social computing pervading the enterprise IT<sup>1</sup>, thus allowing process modelers to create richer models of people-intensive processes.

The rest of this paper is organized as follows: Section 2 describes motivation behind the modeling approach and presents a motivating example. In Section 3 we show the lack of expressivity in existing modeling approaches with regards to the motivating example. Section 4 describes the proposed modeling paradigm and the corresponding graphical notation. Section 5 demonstrates the usability of the approach through sensible use-cases. Disadvantages of the modeling approach are discussed in Section 6. The paper is concluded in Section 7.

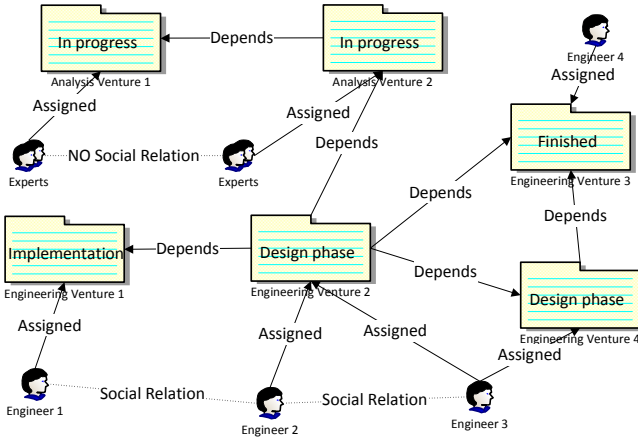
## 2 Motivation

Collaboration is a recursive [13] process comprised of human interactions towards realization of shared goals. Groupware and social software foster collaboration of individuals who work across time, space, cultural and organizational boundaries, i.e., virtual teams [17]. Using this type of software, people interact through conversations (e.g., e-mails and instant messages) and transactions (e.g., create/modify/assign/restructure a document) in order to augment a common deliverable, e.g., documentation of an idea, a technical specification, a source code file, or a wiki page. Typically, such interactions are chaotic, non-routine, and are hard to predict and model. However, as side-effects they produce semantical and social relations between actors and artifacts. Furthermore, artifacts usually are semantically connected into hierarchical or network structures, i.e., references in wiki pages, or dependencies between software components.

As a motivating example, let us consider in-house software engineering in a dot-com company. Projects, or ventures, in such company can be classified as engineering ventures (development of new functionality), or analysis ventures (incident investigation, proof-of-concepts). Both types of ventures produce deliverables, such as source code or technical documentation. Figure 1 demonstrates a snapshot of a collaboration process as a directed graph of venture deliverables

<sup>1</sup> <http://www.gartner.com/it/page.jsp?id=1470115>

and collaborating actors. Edges connecting Ventures represent functional dependencies (i.e., venture depends on either an investigation report or a software component produced by other ventures). Edges connecting Actors depict social relations, i.e., there is a regular communication over instant messaging channels between them, or they contribute to the same venture. Analysis ventures, representing rather creative and non-routine work, can reside only in two possible phases, namely **In Progress** and **Finished**, while engineering ventures, representing more structured and long-running work, can reside in more phases, such as **Design**, **Implementation**, **Testing**, and **Finished**.



**Fig. 1.** Software engineering collaboration process snapshot

Now, let us consider a process modeler that possesses knowledge of working environment, culture, and the scale of the company, and aims at modeling the following rules:

1. *A venture project team should be notified of any changes in the technical documentation of other ventures it depends on. However, if two functionally interdependent ventures share any team members, then enforced communication is not required.* This rule ensures proper knowledge sharing between functionally interdependent ventures while avoiding overcommunication. For example, any new technical reports of **Analysis Venture 2** should be communicated to the project team of **Engineering Venture 2**. However, the same synchronization between **Engineering Venture 2** and **Engineering Venture 4** is not critical, because **Engineer 3** is anyway aware of any such changes.
2. *Venture technical documentation (i.e., design, or a report) should be reviewed by an expert from a functionally dependent venture. Moreover, it is preferable to assign an expert socially unrelated to the venture team members.* This rule tries to avoid biased reviews by finding a socially unrelated experts. For example, it is more preferable to assign **Engineer 4**, than **Engineer 1**, as a

reviewer of **Engineering Venture 2**, as **Engineer 4** does not have strong social relations with the **Engineering Venture 2** team.

3. *An engineering venture can be started if at least one venture, it depends on, has passed **Design phase**.* This rule defines a balance between total serialization of dependent ventures **Design phases**, which results in a longer time-to-market, and total parallelization of **Design phases**, which results in more iterations. For example, **Engineering Venture 2** was started upon completion of **Design phase** of either **Engineering Venture 3** or **Engineering Venture 1**.
4. ***Design phase** of a venture cannot be finished if all ventures, it depends on, have not passed **Design phase**.* This rule minimizes chances of potential rework and wasted efforts. For example, **Design phase** of **Engineering Venture 2** can be finished only after **Engineering Venture 4** switches to **Implementation phase**.

We refer to such rules as context dependency rules (CDRs). As it can be seen from the examples above, they allow to capture the knowledge about the impact of social and structural relations on collaboration processes. Formal specification can help visualize and improve CDRs, thus reflecting management experience in enterprise.

### 3 Related Work

In this section we discuss the related works and show their shortcomings with regard to their ability to model context dependency rules (CDRs), examples of which are outlined in the previous section. To the best of our knowledge, no framework is capable of capturing CDRs in a formal and visual manner.

Information-centric modeling approaches, such as Case Handling [2] and Artifact-centric workflows [4], can capture the evolvement of collaboration entities into formal models, and capture the relations on a conceptual level using composite cases and 'is-a' relationships between Roles in Case Handling, and Entity-Relationship models in Artifact-centric workflows. However, *condition* elements in these approaches do not allow to specify CDRs. Conditions in case-handling are defined as sets of bindings where a binding is a set of values for specific data objects. Therefore, it is not possible to define a condition which examines all the objects in a specific relation to the object at hand (CDR example 3), or to specify that *all* the related objects must reside in a specific state (CDR example 4). Conditions in Artifact-centric workflows are specified in formulas written in first-order logic. However, the specification is restricted and does not allow to use quantifiers, which is crucial for expressing CDRs (e.g., CDR examples 3 or 4).

Traditional activity-oriented business process modeling approaches like BPMN [2] allow to model dependencies between processes via messages or events. Asynchronous messaging can be used to partially resemble CDRs, e.g., by sending notifications to related processes. However, it would not provide enough flexibility to capture such rules. Using external events is another way to model

<sup>2</sup> <http://www.bpmn.org/>



such logic, but, it would require the specification of events in natural language. Moreover, activity-oriented approaches are difficult to apply for collaboration processes, because it is hard to predefine exact steps to follow [16]. In addition, explicit communication and coordination entities (i.e., events, message channels), intended for publishing information, do not convey any functional load and, therefore, complicate and encumber process models. Agent-based or agent-inspired approaches for coordination of business processes as [111] also utilize explicit information publishing entities, thus sharing the same disadvantages.

Context-aware workflows [20] are a generic approach that advocates the augmentation of workflow technology with information about the physical world. It is an execution framework, and proposes to use an XML-based language to express context dependencies. Theoretically, CDRs could be implemented using this framework, however, it would be hardly intuitive to comprehend.

In [3] so-called *batch-tasks* were proposed to allow for a task that is executed for multiple workflow instances at the same time. Other similar approaches can be found in [5]. Partially, CDRs can be covered by batch-tasks, e.g., CDR example 4. For more complex rules, however, this approach is not flexible enough.

Team Automata [9,10] use communication via shared action spaces. Transitions, which include the same external action, are fired simultaneously in these Automata. Alike to batch-tasks, it doesn't provide the needed flexibility.

COREPRO modeling framework [14] proposes to model the dependencies between states of related processes via so-called external state transitions. Again, it provides limited expressivity for describing the dependencies, as it allows to specify only exact external state transitions.

Futhermore, neither of approaches discussed above focuses on functional or social relations between actors and artifacts and therefore does not provide corresponding modeling elements. This makes it difficult for a modeler to specify such relations and their impact in a natural way.

## 4 Modeling Paradigm

In this section we present the modeling approach for collaboration processes, which allows to express context dependency rules (CDRs, see Sec. 2). We explain the design features, outline the modeling paradigm, and present modeling elements and graphical notation.

The key modeling principles of our modeling paradigm are:

- *Information-centric.* As described in Sec. 2, collaboration can be seen as a network of evolving artifacts. In addition, activity-oriented approaches are difficult to apply to collaboration processes, because it is hard to predefine exact steps to follow [16]. For instance, people interactions, such as conversations and transactions, in a collaboration process are rather chaotic and unpredictable, therefore, it is easier to capture collaboration artifacts and corresponding social and semantic relations as side effects of interactions. Therefore, the information-centric modeling paradigm is chosen as a basis for the modeling approach.

- *Bottom-up and neighborhood-aware.* Modeling an evolution of a network of artifacts and people in a holistic view can be a daunting task. Contrarily, neglecting relations completely and modeling the progress of artifacts in isolation leads to context tunneling, and, therefore ineffective models. We thus propose to use a bottom-up hybrid approach, which models evolution of each artifact as an individual process explicitly influenced by its neighborhood. This approach allows to describe behavior at macro level (network of artifacts) by means of modeling behaviors at micro level (evolution of a single artifact). Additionally, it allows to provide simple processes coordination and secure encapsulation: a process can modify only its own state, it cannot impact related processes explicitly. This approach was inspired by a computational model of Cellular Automata(CA) [15].
- *Social.* Collaboration processes often involve non-routine activities, such as discovery of socially coherent teams, or complex decision-making by exploiting social hubs and unbiased experts. Therefore, the paradigm promotes modeling not only a network of evolving artifacts, but also an evolving network of people.

In the following subsection we introduce the modeling framework, which incorporates the key principles discussed above.

#### 4.1 Modeling Framework

Our modeling framework is defined as a set of basic modeling elements that a business process modeler can operate with in order to reflect context dependency rules (CDRs) within business process models:

1. *Collaboration artifacts* and their *states*. Artifacts should represent various aspects and deliverables of collaboration process (e.g., a software component, or a technical design). The states should represent the possible phases of collaboration. *Artifacts* and their *states* may be modeled using existing information-centric approaches, such as Artifact-centric workflows [4], making thus our approach rather complementary, than stand-alone.
2. *Relations*. Relations can be pre-defined (e.g., functional or structural dependency) or dynamic (e.g., temporal or social relations), i.e., produced as side effects of interactions and transactions. Proliferation of groupware and social software boosts the quantity and quality of dynamic relations data, thus empowering process modelers.
3. *Context-aware state transitions*. Context-aware state transitions define what *Relations* and *Artifacts* are relevant for a business process at various steps of its execution.

In order to better demonstrate how the framework basic modeling elements can be put together to model a business process, we present a graphical notation for the modeling framework. The notation is an extension of the conventional statecharts visual formalism [7]. The choice of statecharts is justified by their information-centric nature and widespread adoption as part of Unified Modeling

Language (UML)<sup>3</sup>. Being a natural visual representation of state machine mathematical model, statecharts include the following basic elements: (i) Clustered and refined *states*; (ii) *State transitions* comprised of *events* (external happenings such as user input or timeout), *conditions* (boolean expressions over events and state) and *actions* (e.g., sending an e-mail, or assigning a person to a task).

Our graphical notation, dealing with explicit modeling of relations, extends conventional statecharts with a new element **Context**, graphically depicted as a hexagon. **Context** element, being inseparable to **State** element, defines relations and artifacts, relevant to a particular state. Each **Context** element contains a query against the neighborhood of the artifact (i.e., related artifacts and people) asking for the presence of a specific pattern. Each **Context** can have several **Transition** elements attached: if the context query finds the corresponding pattern, then all transitions attached to this **Context** element are enabled, otherwise disabled. Similarly to **State** elements in statecharts, **Context** elements can be clustered using logical AND/OR/XOR operations.

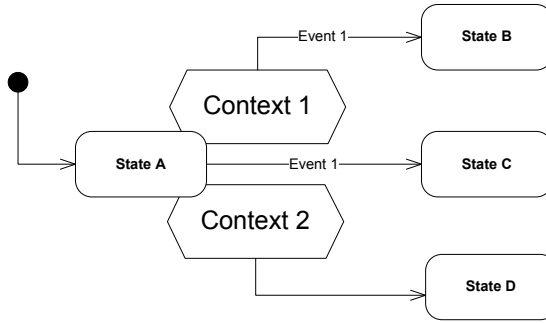
Figure 2 demonstrates the overall integration of **Context** element into statecharts (the context queries are omitted in this figure for the sake of simplicity). Two of three transitions in the figure are enabled by **Context** elements. By default, we assume that transitions attached to **Context** elements have a higher priority over other transitions, but generally it is up to a modeler to define the priorities. Below are enlisted possible transitions in the default prioritization order:

1. If a pattern described in **Context 2** is found, then the state machine switches to state D. Here we can see that an *event* element is optional, and if absent, then the *transition* is activated at once.
2. If **Event 1** is fired and a pattern described in **Context 1** is found, then the state machine switches to state B.
3. If **Event 1** is fired and a pattern described in **Context 1** is not found, then the state machine switches to state C.

When modeling the behavior of multiple interdependent concurrent process instances, a modeler should assume that state transitions are synchronized, i.e., *every* Context element is evaluated before activation of *any* state transition in any process. Thus, if some process switches to state A and then instantly to some other state, the fact that it has been in state A will be considered.

We believe that graphs *a priori* are rather a natural visual medium for describing artifact networks and relations. Therefore, we define a visual graph query language, which is used to specify queries in **Context** elements. Queries expressed in the language can easily be mapped to a *First-Order Logic* expressions, but vice versa does not hold. A query in the visual language is a directed connected multigraph with labeled edges and nodes. Labels can either denote atomic relations/states/types, or expressions over atomic entities based on propositional calculus expressions. Additionally, labels may be absent in general, denoting a placeholder (e.g., any relation/state/type). An edge direction in a graph is used to depict a non-commutative relation.

<sup>3</sup> <http://www.omg.org/spec/UML/>



**Fig. 2.** Integration of **Context** elements into statecharts

Interpretation of a graph query naturally corresponds to the way we read *First-Order Logic* expressions. Query graphs always have one initialized primary element, therefore, graph queries should be interpreted outwards: starting from the central primary element towards most distant nodes. For example, graph queries depicted in Fig. 3 can be interpreted as follows:

- **Context 1**: if the primary document is in state **A**, and there are no documents, related by content or author to the primary one, residing either in state **A** or state **B**, then the attached transition is enabled.
- **Context 2**: if the primary document is in state **A**, and every single document, related by content to the primary one, must reside in state **B** and have two socially unrelated Authors that contributed to it, one of which is **Active**, then the attached transition is enabled.

As depicted in Fig. 3, single line edges correspond to existence quantifiers, while double line and crossed dashed edges correspond to universal quantifiers. Nodes in query graphs may be labeled with variables, that can later be reused in Conditions and Activities of corresponding Transitions. Since multiple occurrences of a context pattern may be found in the neighborhood, Activities/Conditions may be also extended with quantifiers, i.e., send e-mail to any/every related contributor.

The success of a modeling approach depends, to a great extent, on the level of simplicity offered. Therefore, we favor simplicity over completeness and impose following constraints on the queries expressed in the visual language:

- Only basic operators from proposition calculus are allowed as literal expressions attached to edges and nodes: conjunction, disjunction and negation. Even though, conditional and biconditional operators may be expressed via the former ones, more complex operators may decrease understanding and make reasoning about the model more difficult.
- Under Open World Assumption [18] negation may introduce ambiguity, therefore only negation as a failure is allowed, i.e., negation on an edge can be used only if nodes connected by the edge are transitively connected to the central node with non-negative edges.

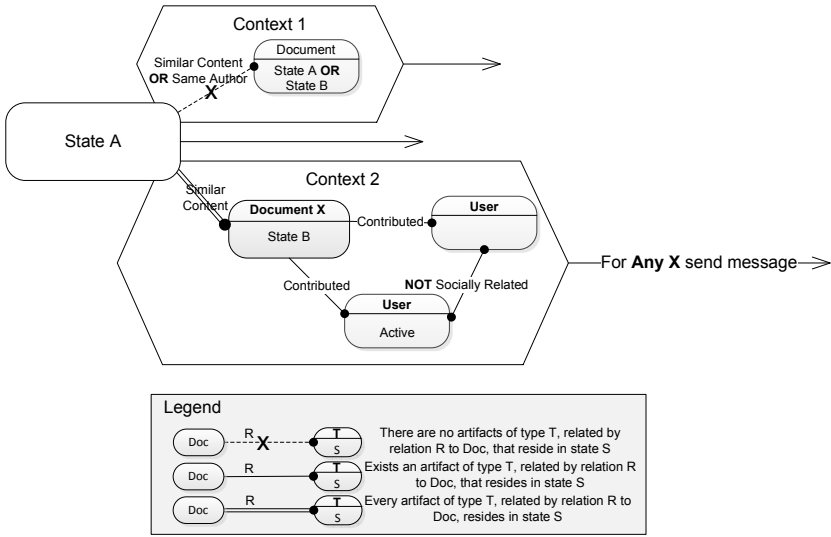


Fig. 3. Example of context queries in Context elements

- Edges with universal quantification can be adjacent only to the primary node. During our experiments with the modeling notation we observed that universal quantification can introduce ambiguity and would require assigning priorities to edges thus unnecessary complicating the modeling process. Nevertheless, implicit prioritization is still necessary: edges with universal quantification should have implicitly higher priority, than edges with existential quantification, in order to avoid ambiguities in case of cyclic query graphs.

A formal definition of our modeling notation is given below. In order to keep the definition succinct, we omit a formal definition of Statecharts, as it is available elsewhere, e.g., in [12].

**Definition 1.** Labels  $L$  in a query graph representing relations  $R$ , types  $T$  and states  $S$  of artifacts are defined as:

$$Label\ L \stackrel{def}{=} Atomic\ Condition \mid Placeholder \mid L \wedge L \mid L \vee L \mid \neg L, \quad (1)$$

*Placeholder denotes any value (no condition)*

**Definition 2.** Edges in a query graph, along with adjacent vertices, are interpreted in First-Order Logic as follows:

$$(A) \xrightarrow{\bullet R} (T, S) \stackrel{def}{=} \exists X : R(A, X) \wedge T(X) \wedge S(X) \quad (2)$$

$$(A) \xrightarrow{R} \bullet (T, S) \stackrel{def}{=} \forall X : R(A, X) \wedge T(X) \rightarrow S(X) \quad (3)$$

$$(A) \xrightarrow{-\times R} \bullet (T, S) \stackrel{def}{=} \nexists X : R(A, X) \wedge T(X) \wedge S(X) \quad (4)$$

Where, given that graph queries are interpreted outwards from the central primary element (vertex),  $A$  denotes an already interpreted vertex. Predicates  $T$  and  $S$  describe type and state of suitable artifacts respectively. Result of query graph interpretation is a logical conjunction of the First-Order Logic formulas corresponding to graph edges. Higher priority of edges with universal quantification ensure that the corresponding to these edges formulas always appear at the beginning of the resulting logical conjunction.

**Definition 3.** Query graph  $Q$  is a quadruple defined as follows:

$$\begin{aligned}
 Q &\stackrel{\text{def}}{=} (a, CE, V, EV), \text{ graph } Q \text{ is connected,} \\
 &a \text{ is the predefined central primary vertex (artifact),} \\
 &V \text{ is a set of vertices } (T, S), a \notin V, \\
 &CE \text{ is a set of edges } CE \subseteq \{a\} \times \{-\bullet, \Rightarrow, -\times\bullet\} \times V, \\
 &EV \text{ is a set of edges } EV \subseteq V \times \{-\bullet\} \times V
 \end{aligned} \tag{5}$$

**Definition 4.** Context element  $CTX$  in the modeling notation is a composition of query graphs CQ:

$$\begin{aligned}
 CQ &\stackrel{\text{def}}{=} Q \mid CQ' \text{ AND } CQ'' \mid CQ' \text{ OR } CQ'' \mid CQ' \text{ XOR } CQ'', \\
 CQ' &= (a', CE', V', EV'), CQ'' = (a'', CE'', V'', EV''), \\
 a' &= a'', CE' \cap CE'' = \emptyset, V' \cap V'' = \emptyset, EV' \cap EV'' = \emptyset
 \end{aligned} \tag{6}$$

**Definition 5.** Transition element  $CT$ , attached to Context element  $CTX$ , can be defined as:

$$\begin{aligned}
 CT &\stackrel{\text{def}}{=} (CTX, E, C, AC), \\
 E &\text{ is an external event,} \\
 C &\text{ is a condition, } C : QU \times ID \rightarrow \{\mathbf{true}, \mathbf{false}\}, \\
 AC &\text{ is an activity, } AC : QU \times ID \rightarrow \emptyset, \\
 ID &\text{ is a set of identifiers attached to vertices in } CTX \text{ graph,} \\
 QU &\text{ is a set of quantifiers, } QU = \{\mathbf{Any}, \mathbf{Every}, \mathbf{All}\}
 \end{aligned} \tag{7}$$

Our visual graph querying language was inspired by Graphlog language [6]. Graphlog is more complex, because it was designed as an execution language, as opposed to our language which aims rather at modeling. Our language assumes that central artifact is always present and exploits that to simplify universal quantification notation with special types of edges, while in Graphlog universal quantification is represented by a conjunction of existential quantification and negation.

## 5 Use Cases

This section describes three collaboration process use-cases which demonstrate the application of our modeling approach to various collaboration issues. As it

can be witnessed, the approach allows to easily express the dependency of a process on complex relations in its environment, and to compactly capture the dynamic co-influence between instances of the same process in one model. For clarity, in the use cases we attach to each **Context** element a free text description of its query.

### 5.1 Use Case - Design Game

**Goal.** The goal in this use case is to coordinate a design of a complex system consisting of interrelated projects. A set of expert virtual teams thus collaborate to reach a consensus. Assignment relation between teams and projects is one-to-one, but teams can share members. As some projects are dependent, it can happen that changes in the design of one project can be the reason for changes in the design of others. Finally, all project designs should be consistent with dependent ones.

**Model.** Each project of this system is regarded as a separate process (See Figure 4). In the beginning, it is in **In Progress** state which means the team is currently working on its design. When the team makes some changes to the design and commits it, the process goes into **Updated** state. If no changes to the design were made, i.e., the existing version was examined and considered valid, then the process switches to **Finalized** state. These two states represent superstate **Wait Input** which means that the project design is currently awaiting for some external actions. If the team suddenly decides to update the design (e.g., a better idea emerged), the process goes back into **In Progress** state.

Now, if the process is in **Wait Input** state, and if all the related projects are also in **Wait input** state and at least one is **Updated**, then the team should check the design of their project against inconsistencies with updated projects. Thus, the updated documents are sent to the team and the state is switched to **In progress**. An exception is the case when the project team shares a common expert with the team of an updated project(*relation Socially related*), who is expected to foresee any inconsistencies beforehand. Waiting the related projects

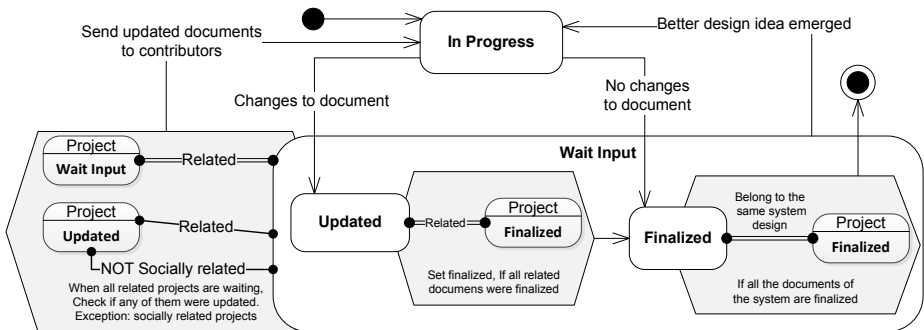


Fig. 4. Use case - design game

to be in “Wait input” ensures that all the updates of related documents will be taken into account.

When in **Updated** state, and if all the related projects are finalized, the process goes into finalized state, which ensures that if a document spawned no updates among related documents, it will not stay in **Updated** state.

The system may be considered in the final state when all the projects are in **Finalized** state.

**Advantages.** This use case demonstrates the modeling of collaboration as ordered iterative communication of project teams towards reaching a consensus. It shows that our modeling approach, as opposed to existing modeling approaches (See Sec. 3), is capable of expressing universal and existential quantification.

### 5.2 Use Case - Social Selection

**Goal.** The goal of this use case is to support a software development process with the selection of appropriate actors (e.g., developer, adviser, reviewer) based on relations with the other tasks and among the actors. Tasks are related if they belong to the same project, employees are related if they collaborated before.

**Model.** Figure 5 depicts the software development process. At first, the task is **Ready for implementation** state and is waiting for an appropriate developer to be assigned. Any available developer from a related task is assigned for this role, as he/she expected to be more productive because of being familiar with some related concepts. Alternatively, a manual assignment is performed. In either case, the process goes to **Implementation in progress** state. An impediment can occur during the implementation (**Impediment pending** state), in which case an

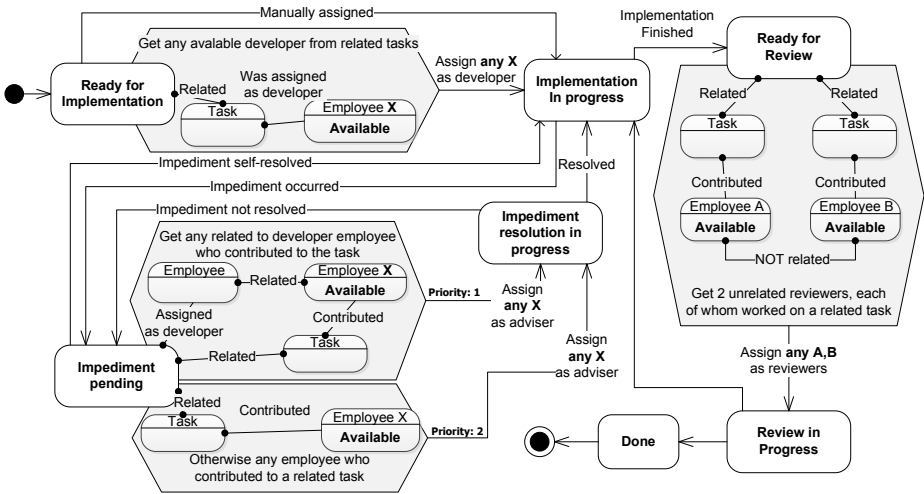


Fig. 5. Use case - social selection



adviser is needed for assistance. An adviser is preferably selected as a related to the developer employee who contributed to a related task, because of joint work experience. Otherwise, any related task contributor is chosen. If the adviser is found, the process goes into **Resolution in Progress** state, from where it can either go either back to **Implementation in Progress** or **Impediment pending** states, depending on whether the impediment has been resolved. Also, the developer can resolve the impediment by him/herself if no adviser was found. After the implementation is finished, the reviewers are selected (**Ready For Review** state): they are desired to have experience with related tasks but be unrelated to each other, which assures unbiased reviews. After the review process (**Review In Progress** state), either the implementation needs to be revised, or the task is considered finished.

**Advantages.** This use case demonstrates expressiveness of the modeling approach when visualizing social network environment, allowing thus to model processes that require discovery (e.g., compose a socially coherent team), unbiasedness (e.g., involve independent people), and negotiation (e.g. by exploiting of social hubs). It shows expressiveness of the graphical notation with regards to modeling patterns in a surrounding social network. Contrarily, existing modeling approaches do not model social relations between actors, therefore, are not capable of capturing such patterns (See Sec. 3).

### 5.3 Use Case - Dependent Components

**Goal.** The goal is to coordinate the development and testing of a software product, which consists of manifold components, some of which depend on others (we assume no cyclic dependencies). The development a component should proceed only when the components it depends on have reached certain progress.

**Model.** Figure 6 depicts the process which corresponds to a single component. It starts in **Open** state and switches over to **Implementation Phase** in either of two cases: it does not depend on any components, or at least one component which it depends on is in **Testing Phase**. This ensures some minimal basis for the development. After **Implementation phase**, the component is ready to switch over to **Testing Phase**, but, first, it should wait for all the components it depends on to be implemented, so the testing covers the combined functionality. The testing phase can reveal some flaws so the component will return into **Implementation Phase** for fixing those. If, while the component is in **Testing Phase**, any of the components it depends on suddenly goes into **Implementation Phase**, then the testing should be stopped in order not to waste the testing effort on outdated components. Lastly, if the component is in **Ready to Finalize** state, and all the components it depends on are **Finalized**, then the component can be finalized.

**Advantages.** This use case demonstrates the suitability of the modeling approach for expressing of coordination of project teams towards ensuring consistency and correctness of a complex product. It shows expressiveness of our modeling notation comparing to existing modeling approaches that would capture process coordination either in a text form or via events (See Sec. 3).

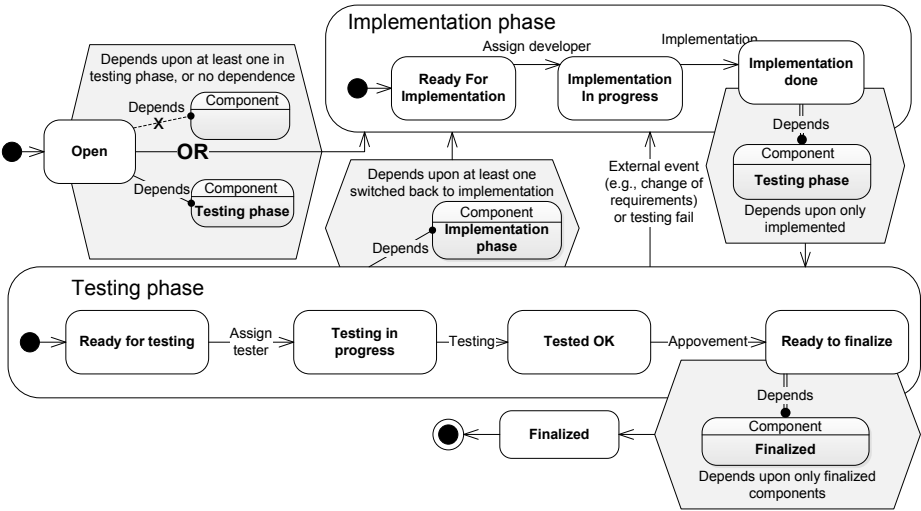


Fig. 6. Use case - dependent components

## 6 Discussion

Advantages of the modeling approach were demonstrated in the previous section. However, we perceive that our model also has particular disadvantages. Absence of explicit communication entities (events or messages) in the modeling approach is a strength, but also a weakness. A modeler cannot immediately see what parts of a business process (states) other processes rely upon. Given that definitions of events and messages represent a process interface, a modeler will not be able to remove or change process states without a risk of affecting other models. However, this problem can be remedied with State clustering available in statecharts.

We envision, that the discussed visual query language might need additional elements for greater expresiveness. For instance, aggregation elements to express query of exact number of neighbors residing in particular state, or aggregated state of all neighbors. However, in this paper we focused on fundamental concepts, thus trying to keep the appropriate level of detail.

## 7 Conclusion

This paper proposes a modeling approach and a corresponding graphical notation for creative human collaboration processes. The applicability of the approach was demonstrated through several use-cases, and its strengths and weaknesses were discussed.

Comparing to existing approaches, our contribution has two main distinguishable features: it is capable of capturing specific conditions in form of patterns in

related artifacts of the process, and it advocates a communication model where a process can modify only its own state and cannot explicitly impact the related processes. We have shown that these features are naturally suitable for modeling of collaboration processes. Although our approach was designed with this focus, we do not exclude its applicability in other areas.

Our future work includes the development of an associated execution framework and the integration with existing business process technologies and collaborative software.

## References

1. van der Aalst, W.M., Barthelmeß, P., Ellis, C., Wainer, J.: Workflow modeling using proclerts. In: Scheuermann, P., Etzion, O. (eds.) *CoopIS 2000*. LNCS, vol. 1901, pp. 198–209. Springer, Heidelberg (2000)
2. van der Aalst, W.M., Weske, M., Grnbauer, D.: Case handling: a new paradigm for business process support. *Data & Knowledge Engineering* 53(2), 129–162 (2005)
3. Barthelmeß, P., Wainer, J.: Workflow systems: a few definitions and a few suggestions. In: *Proceedings of Conference on Organizational Computing Systems, COCS 1995*, pp. 138–147. ACM, New York (1995)
4. Bhattacharya, K., Hull, R., Su, J.: A data-centric design methodology for business processes. In: *Handbook of Research on Business Process Modeling*, ch. 23, pp. 503–531 (2009)
5. Casati, F., Ceri, S., Pernici, B., Pozzi, G.: Conceptual Modeling of Workflows. In: Papazoglou, M.P. (ed.) *ER 1995 and ODER 1995*. LNCS, vol. 1021, pp. 341–354. Springer, Heidelberg (1995)
6. Consens, M.P., Mendelzon, A.O.: Graphlog: a visual formalism for real life recursion. In: *Proceedings of the Ninth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, PODS 1990*, pp. 404–416. ACM, New York (1990)
7. David Harel: Statecharts: a visual formalism for complex systems. *Science of Computer Programming* 8(3), 231–274 (1987)
8. Dickson, G.W., DeSanctis, G.: *Information Technology and the Future Enterprise: New Models for Managers*. Prentice Hall PTR, Upper Saddle River, NJ, USA (2000)
9. Ellis, C.: Team automata for groupware systems. In: *Proceedings of the International ACM SIGGROUP Conference on Supporting Group Work: the Integration Challenge, GROUP 1997*, pp. 415–424. ACM, New York (1997)
10. Engels, G., Groenewegen, L.: Towards Team-Automata-Driven Object-Oriented Collaborative Work. In: Brauer, W., Ehrig, H., Karhumäki, J., Salomaa, A. (eds.) *Formal and Natural Computing*. LNCS, vol. 2300, pp. 247–255. Springer, Heidelberg (2002)
11. Hagen, C., Alonso, G.: Beyond the black box: event-based inter-process communication in process support systems. In: *Proceedings of 19th IEEE International Conference on Distributed Computing Systems*, pp. 450–457 (1999)
12. Latella, D., Majzik, I., Massink, M.: Towards a formal operational semantics of uml statechart diagrams. In: *Proceedings of the IFIP TC6/WG6.1 Third International Conference on Formal Methods for Open Object-Based Distributed Systems (FMOODS)*, p. 465. Kluwer, B.V. (1999)

13. Martinez-Moyano, I.: Exploring the dynamics of collaboration in interorganizational settings. In: *Creating a Culture of Collaboration: The International Association of Facilitators Handbook 4*, p. 69 (2006)
14. Müller, D., Reichert, M., Herbst, J.: Data-Driven Modeling and Coordination of Large Process Structures. In: Meersman, R., Tari, Z. (eds.) *OTM 2007, Part I*. LNCS, vol. 4803, pp. 131–149. Springer, Heidelberg (2007)
15. Neumann, J.V.: *Theory of Self-Reproducing Automata*. University of Illinois Press, Champaign (1966)
16. Nurcan, S.: A survey on the flexibility requirements related to business processes and modeling artifacts. In: *Proceedings of the Proceedings of the 41st Annual Hawaii International Conference on System Sciences, HICSS 2008*, pp. 378–388. IEEE Computer Society, Washington, DC (2008)
17. Powell, A., Piccoli, G., Ives, B.: Virtual teams: a review of current literature and directions for future research. *SIGMIS Database* 35, 6–36 (2004)
18. Reiter, R.: *On closed world data bases*, pp. 300–310. Morgan Kaufmann Publishers Inc., San Francisco (1987)
19. Sanz, J.: Entity-centric operations modeling for business process management - a multidisciplinary review of the state-of-the-art. In: *2011 IEEE 6th International Symposium on Service Oriented System Engineering (SOSE)*, pp. 152–163 (December 2011)
20. Wieland, M., Kopp, O., Nicklas, D., Leymann, F.: Towards context-aware workflows. In: *CAISE 2007 Proceedings of the Workshops and Doctoral Consortium*. Citeseer (2007)

# Process Redesign for Liquidity Planning in Practice: An Empirical Assessment

Jochen Martin, Tobias Conte, and Athanasios Mazarakis

FZI Research Center for Information Technology,  
Haid-und-Neu-Str. 10-14,  
76131 Karlsruhe, Germany  
{martin, conte, mazarakis}@fzi.de

**Abstract.** The financial crisis has kept the world busy since 2007. The resulting difficulties in accessing liquidity and low interest rates on deposits strengthened the importance of proper liquidity planning. These challenges are even greater for globally spread enterprises in which currency-specific liquidity planning implies decentralized processes. These have to be coordinated within the local partitions such that proper and consistent overall financial planning is eventually ensured. Although extensive research has been conducted in the field of process redesign, most models lack applicability, either because of strict process restrictions or because they are too complex and, hence, hard to realize and communicate. To close this gap and to demonstrate the potential of business process redesign in practice, we (i) analyze the requirements of the financial planning domain to identify an appropriate redesign framework, and (ii) evaluate the impact of an industrially implemented process redesign with respect to process runtime and quality.

**Keywords:** business process redesign, financial planning, process quality, semi-structured processes.

## 1 Introduction

The financial crisis has kept the world busy from 2007 to the present day. As a result, the confidence in the creditworthiness of most enterprises, banks, as well as industrial corporations, suffered, and the risk premium for liquidity procurement rose. The fact that even large enterprises were hit by the crisis resulted in an increased awareness for the striking importance of a high credit rating to be able to access the capital market at reasonable costs. On the other hand, the governmental rescue measures induced cheap liquidity which accounted for low interest incomes and, hence, increased the cost of carry for companies that produce a liquidity surplus. The high risk premium along with the low interest on deposits strengthened the importance of proper liquidity planning, independent of a company's structure and size.

Certainly, it has been recognized long before the crisis that a precise forecast of business figures like sales, production and investments is essential to accomplish a correct liquidity and exposure planning and, thus, enables companies to cope with

uncertainties as exemplified above [1-3]. However, the financial crisis made it painfully obvious how difficult it is, even for large and apparently established companies with solid business models and secure sales, to assure constant liquidity. These challenges are even greater for globally spread companies, since the relevant data for the central currency-specific liquidity plan is distributed amongst the local partitions. Each of these local departments is confronted with the complex generation of planning data driven by the multitude of distributed data sources like sales development, operative goals, or macroeconomic indicators, altogether summing up to an extensive process [4-5]. Hence, an efficient and effective global risk management requires the definition of company-wide standards with respect to the structure of the planning data. To eventually ensure the compliance with these standard and to guarantee a proper and consistent overall financial planning, the decentralized planning processes have to be coordinated within the local partitions and internal transactions between them have to be monitored by a central entity.

For this reason, the process of data transmission and validation accompanied by an intensive communication between local and global management is of utmost importance. Regardless of all challenges arising within complex company structures, the requirements placed on sensitive processes like financial management continuously increase [6]. Simultaneously, due to the constant pressure to reduce costs, the number of employees remains constant or reduces.

In order to tackle this challenge, multinational companies oftentimes opt for a corporate financial portal, including IT support for the process of centralized currency-differentiated liquidity risk-management. Such a portal can, for example, provide services like report upload and validation as well as other corporate services [7-8], e.g. monitoring, or communication services. Implementing such an information system, major challenges arise from (i) the high heterogeneity of applications and business processes within multinational enterprises due to historically grown structures as well as mergers and acquisitions and (ii) a potentially low willingness to change and to abandon known structures and workflows of employees [9].

Literature has brought about several approaches to redesign and improve complex processes, however most articles include either strong process restrictions or complex redesign procedures which are hardly applicable in practice. The approach evaluated in this work is the flexible *objective-based process redesign model* [10], which we were able to realize in a renowned, globally operating large company acting in the chemical and pharmaceutical sector. Driven by the fact that process redesign literature is mostly of theoretical nature, this work addresses the following two research questions:

**RQ 1:** Does the objective-based process redesign reduce process runtime in practice?

**RQ 2:** Does the objective-based process redesign increase data quality in practice?

Since June 2010, the redesigned processes are a part of our industry partner's daily routines which provides us with empirical evidence regarding RQ1 and RQ2.

The remainder of this paper is structured as follows: In the next chapter, we introduce the related work with respect to (semi-structured) process redesign and the measurement of routines and structures in business processes along with a broader

motivation and introduction of the applied redesign model. Chapter 3 includes, besides the KPIs to be evaluated, the initial and final process structure as implemented at our industry partner along with a brief description of the technical implementation. In Chapter 4, we present the detailed evaluation of the realized redesign based upon real world data from our industrial partner. The evaluation includes a description of the sample data produced in two years of productive use, the applied methodology, and finally the results and their interpretation. Chapter 5 interprets and generalizes the results and experiences of the work at hand.

## 2 Related Work and Choice of Redesign Framework

Generally, business processes can be distinguished by their level of structure. In this vein, Deiters [11] distinguishes between structured, semi-structured, and completely unstructured business process as follows: structured processes are applied in standardized scenarios and, therein, the sequence of tasks and business rules is predetermined and prescribed. In semi-structured processes, some tasks are not ordered at all and some of the rules may be modified or added latter “on the fly”. Hence, only parts of the sequence of tasks and the business rules are structured. Finally, unstructured processes do not have any repeatable patterns at all, are executed spontaneously, and are difficult to automate.

To provide further insights into the characteristics of processes, lots of work has been performed on the analyses of processes’ structuredness. Pentland [12] introduces routines in business processes as a metric for the degree of standardization. Thereby, the identification of routine patterns allows for the definition of a lexicon per process and, hence, the comparison of different process parts. Furthermore, changes in the sequential execution of the identified process patterns reflect either development or variety and, thus, a lower level of structuredness in the process [13]. Rosenkranz et al. [14] try to derive a unique pattern base to compare different workflows and to detect joint aspects as a foundation for a process standardization approach.

However, their experiences in multiple case studies reveal complex challenges in redesigning business processes. In addition to these complex metrics for measurement and treatment of variety, recent academic work is strongly focused on either structured or unstructured processes. Van der Aalst [15], for example, introduced a framework to verify workflows which, however, only runs in a standardized scenario. Moreover, van der Aalst et al. [16] presented process support strategies for unstructured processes in which the unstructured parts of the process are handled as individual cases.

Academic literature also offers multiple criteria for an efficient process that can be applied as redesign goals: Reijers and Mansar [17] try to get rid of *(i)* unnecessary tasks, *(ii)* reduce contact and *(iii)* reduce waiting times. Moreover, Redman [18], presents solutions focused on *(iv)* task automation. In addition, the *(v)* focus on data quality is explicitly included [18]. Davenport et al. [19] enrich this data perspective by the need for *(vi)* data completeness. Data quality and completeness often depend on the process integration level. Therefore, van der Aalst and Weske [20] as well as Davenport et al. [19] claim the need for an increase of the level of integration. Finally,

Balasubramanian and Gupta [21] present a structural metric for business processes containing most of the above-mentioned objectives.

The process of financial data integration focused on in this paper is a semi-structured process as financial planning data transmission and interaction processes are usually driven by historically grown organizational characteristics in multinational enterprises. Due to the variety of process characteristics, the process redesign and optimization in processes with numerous workflow patterns is a highly complex process [22]. To date, literature has put forth a large body of models and frameworks that tackle the measurement and classification of routines in processes and, hence, enable the classification of semi-structured processes and process redesign in general. Yet, they are all of theoretical nature and hardly provide hands-on advices for flexible redesign necessary in practical applications. To address this research gap and to equip practitioners with a flexible redesign model, Martin et al. [10] propose the *objective-based process redesign model*. It is an approach for semi-structured, non-standardized processes, which inherits both aspects from workflow management systems (WFMS) and case handling and integrates the idea of sequential patterns. The resulting Redesign Model combines the above-detected general objectives (i) - (vi) with shortcomings and constraints dependent on the specific application domain. Based upon the detected process pattern and the predefined objectives, Martin et al. [10] identify process shortcomings and define serviceable process units taking the constraints into account. Along with the proposed stepwise implementation of these services, their approach represents an agile redesign model which allows for a practical handling of the process to be redesigned.

### 3 Use Case

After having made a decision for the appropriate redesign methodology (cp. Section 2), this chapter introduces the use case that underlies this paper. Section 3.1 briefly describes the initial situation we came across at our industrial partner, followed by an introduction of the key performance indicators (KPIs) to be evaluated (cp. Section 3.2). This chapter closes with a short description of the implemented redesign realization and its effects in our partner's processes.

#### 3.1 Financial Planning Process Prior to Redesign

As above-mentioned, financial planning processes in multinational companies are typically semi-structured. Fig. 1 shows the historically grown planning data delivery and validation process that is part of the actual planning process at our industry partner's site. It reflects the situation after introducing a financial portal as an interface between subsidiaries and holding, yet prior to any process redesign.

The process may be separated into three main layers, which represent the three roles involved in the planning data delivery. The first layer is the holding layer that is responsible for data integration. The holding exercises the central liquidity management function. The (enterprise) portal layer represents the gateway at which data is both delivered (by the subsidiaries, representing the decentralized data generation) and received (by the holding).



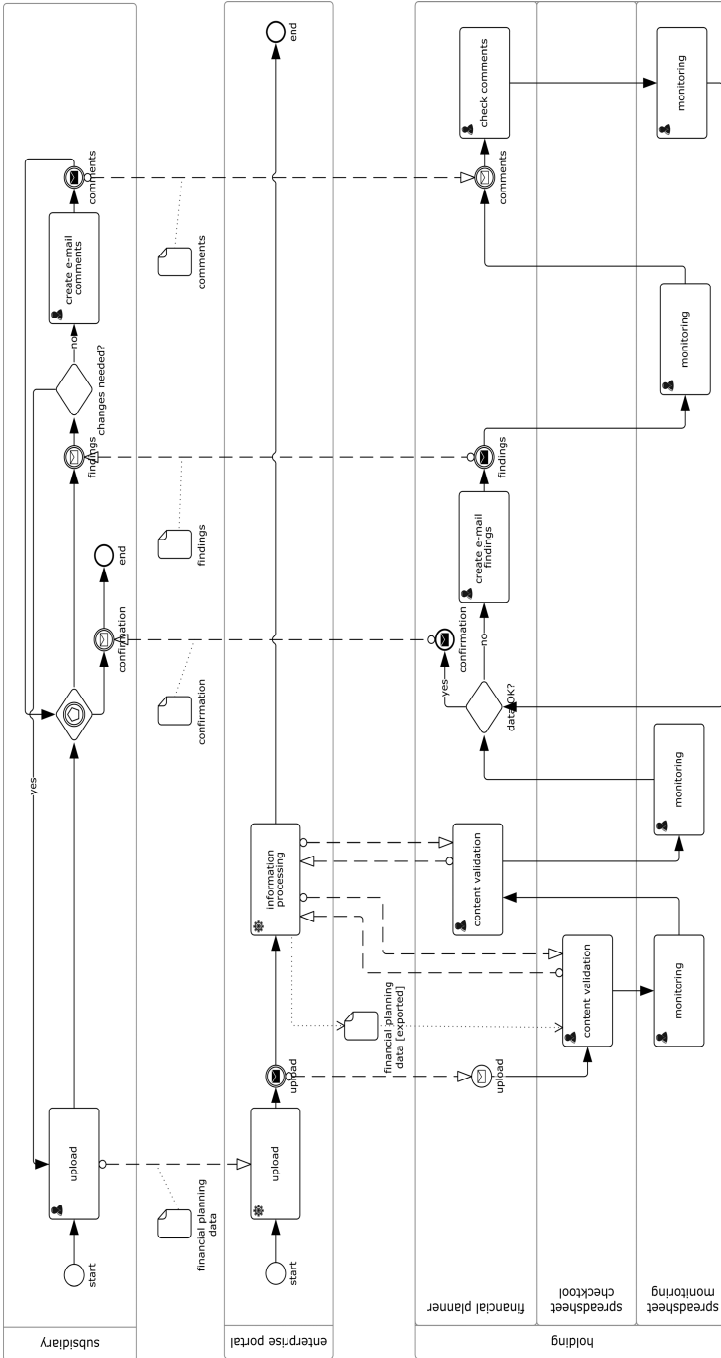


Fig. 1. Financial planning process prior to redesign. The Business Process Modeling Notation as the de facto standard in academic and practice communities for business process modeling is used for illustration [23-24].

The process as shown in Fig. 1 is initiated by the subsidiary which uploads a set of financial planning data. If the basic data structure is compliant with all business rules, an upload notification via email, addressing both subsidiary and holding, is sent. Receiving this email, a (human) financial planner performs a detailed, spreadsheet-based validation that goes beyond the structure of the data. If the financial planner does not identify any issues in the data, the subsidiary is informed by the holding and the process ceases. If issues are identified, they are communicated to the subsidiary via email. In this case, the subsidiary can either correct the reported issues or submit comments via email that justify the delivered data as-is. Data corrections trigger a new upload and validation process. If the data is still erroneous or the comment is declined, another feedback loop is initiated. Otherwise, the process is completed. During the complete process, the current status of the entities including information about validation results is documented in a monitoring spreadsheet.

### 3.2 Key Performance Indicators

As reflected in this work’s research questions (cp. Section 1), the evaluation presented is focused on the *time* necessary to perform all process tasks and the *quality* of the process, i.e. the data output. As we do not aim at cost reduction but rather at decreased runtime and increased quality, costs are required to remain constant. Hence, they are considered indirectly through the KPIs related to runtime. For instance, waiting time reflects waiting costs.

Buchsein and Machmeier [25] and Kuetz [26] describe a set of KPIs for a best practice process. In order to validate the KPIs proposed in literature, we performed semi-structured expert interviews among knowledge workers at our industrial partner’s site to ensure their practical relevance. The results are presented in Table 1 that is sub-divided into literature-based and expert interview-based KPIs.

**Table 1.** KPIs derived from literature and expert interviews

	Buchsein and Machmeier [25], Kuetz [26]	Expert interviews
Time	Processing Time (PrT), Waiting Time (WaT), Planning Time (PlT)	80% Resolution Time (ReT)
Quality	Number of Cycles (NoC) denoting increased communication	

The process time is expressed through the indicators *Processing Time*, *Waiting Time* and *Planning Time*. Waiting Time and Processing Time are considered from the holding perspective: Processing Time is defined as the time the holding is active. Such an activity is, for instance, the validation of the subsidiary’s planning data. Waiting Time is defined as the time the holding is passive, that is, waits for a subsidiary’s response. Finally, the Planning Time is defined as the sum of Processing and Waiting Time. In addition to these straightforward KPIs extracted from literature, the expert interviews suggested an indicator that documents the workload development of the knowledge workers both within the subsidiaries and the holding. The resulting *80% Resolution Time* represents the time interval between delivery

deadline and the completion of 80% of the considered subsidiary's plan. Thereby, the 80% benchmark is a value derived within the expert interviews.

Furthermore, Kuetz [26] proposes the *Number of Cycles* to reflect data quality. The expert interviews revealed that data quality is likely to increase with the number of validations. Since each validation causes communication activities, an increased Number of Cycles is likely to highlight quality improvements. Based thereupon, we defined the Number of Cycles as an indicator to be increased or at least to be kept on a constant level.

### 3.3 Redesign Realization

In order to realize the redesigned process and to enable a generalizable implementation, we have created several modular, de-coupled service units based upon the service oriented architecture (SOA) paradigm. The components that facilitate the redesigned process are: (1) an upload and validation service, (2) a comment management service, (3) a Rule Engine and database, (4) a management service for risk and operations, (5) monitoring services, and (6) an underpinning technical infrastructure (for the detailed implementation cp. Martin et al. [27]).

To assure the agility of the redesign realization, Martin et al. [10] propose an iterative implementation of the above-mentioned services. Based upon that, Chapter 4 presents the evaluation of the conceptualization and implementation of the first service. Therein, data validation and the communication of the results are automated which results in a process as depicted in Fig. 2.

The automated data validations include both checks of *intra-subsidiary planning* and *inter-subsidiary planning* that were conducted manually prior to the redesign. The former denotes validations based on the planning data within a single subsidiary and related to only one single planning period. For instance, comparison of invoices and payments for a specific time horizon. The latter defines validations that take up data delivered by at least two subsidiaries, yet still relates to one single period. That is, for example, the consolidation of invoices issued and received between two entities. Along with the automated checks, the respective notifications of the validations have also been automated. Additionally, new and more sophisticated verifications of data deliveries over several planning periods (*inter-subsidiary/inter-period planning*) to further increase data quality, have been added.

## 4 Evaluation

In this section, the research questions

**RQ 1:** Does the objectives-based process redesign reduce process runtime in practice?

**RQ 2:** Does the objectives-based process redesign increase data quality in practice?

are evaluated via empirical data accumulated during the real-world application of the redesigned financial planning process at our industrial partner's site. Section 4.1 presents the underlying sample data, followed by a brief overview of the applied

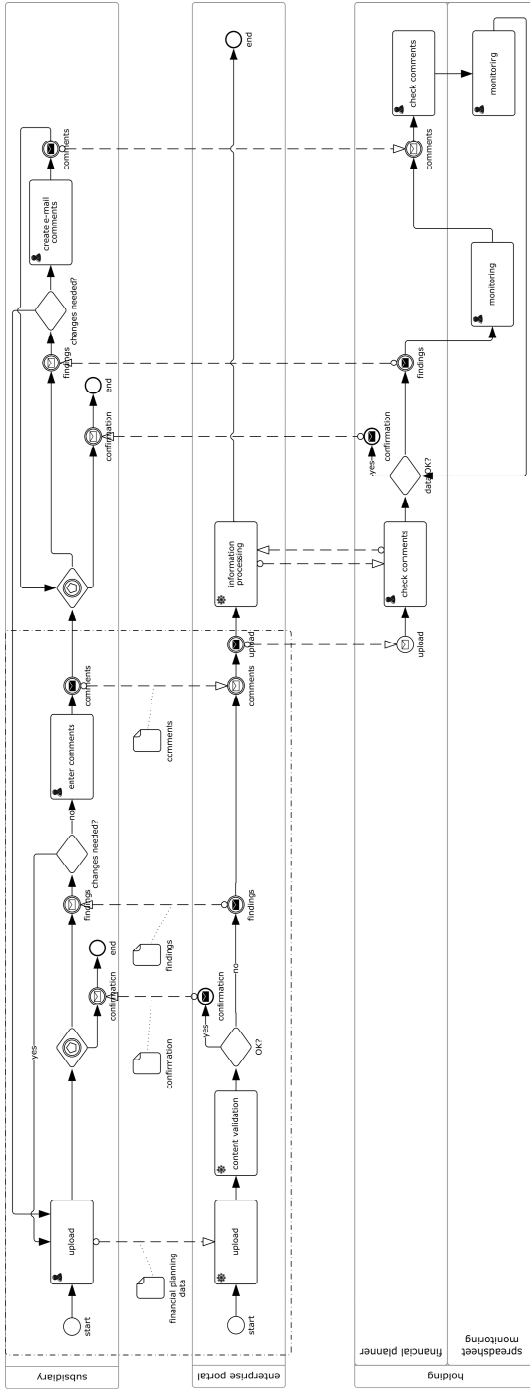


Fig. 2. Redesign Status – process after the redesign (automation of the data validation and the communication of the results)

methodology in Section 4.2. Section 4.3, as the major contribution of this paper, presents the statistical analysis of the empirical data. Chapter 4 concludes with an interpretation of the evaluation results and the implications to be drawn.

#### 4.1 Data Sample and Preparation

The data that underlies our evaluation includes seven data deliveries, starting in June 2009. The data has been delivered in regularly recurring time periods as shown in Tab. 2. Therein, the number of delivering subsidiaries ranges between 99 and 113, owed to mergers and acquisitions that took place during the evaluated time periods. 89 subsidiaries constantly delivered data in all seven periods, which serves as the data basis for the following evaluation. Furthermore, the data sample comprises two data delivery eras: the *pre-redesign phase*, which includes four deliveries from June 2009 (06/2009) to November 2009, and the *post-redesign phase*, including three deliveries from June 2010 to November 2010. The data delivered in March 2010 was distorted due to redesign implementation activities that took place during this delivery period. Therefore, this data set was excluded from the evaluation. It is important to note that our industrial partner has not only incorporated the redesign approach for the conceptual reorganization of the planning process, but has also integrated the services into its daily business immediately after their implementation (three pre-redesign deliveries are included in the evaluation). The data set per period includes the entire email communication between the holding and all subsidiaries that provides a detailed documentation of the validation and communication process. The email communication contains both manually and automatically generated messages. Automated notifications include information about the upload status and the validation results, while manual messages query, for instance, further planning data explanations.

**Table 2.** Number of subsidiaries per data delivery during the evaluation. There is an overlap of 89 subsidiaries between the deliveries, i.e. they constantly delivered data from 06/2009 to 11/2010.

Delivery	06/2009	09/2009	11/2009	03/2010	06/2010	09/2010	11/2010	Overlap
# Subsidiaries	99	100	106	104	113	113	113	89

The emails are classified into *Email Sender*, *Email Receiver*, *Email Subject*, and *Email Date*. That way, one can distinguish whether the email was sent automatically or not, whether it was sent by the holding or the subsidiary, and when it was sent. In this vein, an email sent by the holding to a subsidiary marks the switch from Processing Time to Waiting Time. Automated notifications are treated as generated on holding side, therefore, they also belong to this category. An email sent by a subsidiary marks the opposite switch, accordingly.

Since the structure of the data sample is essential for the correct choice of statistical analyses, a careful examination of the data is required. According to the

Shapiro-Wilk test for normality in SPSS 19, the distribution of the Planning Time is significantly non-normal (pre-redesign phase,  $W(267) = 0.72$ ,  $p < .001$ ; post-redesign phase,  $W(267) = 0.68$ ,  $p < .001$ ) and hence non-parametric inferential analyses have to be conducted (cp. next section).

## 4.2 Methodology

**Clustering:** In the context of a multinational enterprise, our expert interviews have revealed that a time reduction of one or two working days has only a small business impact. To increase practical relevance, we changed the data structure from working day intervals to *working weeks*. In more detail, the original data (measured in working days) has been transformed into the corresponding *number of working weeks after deadline* with a cap at the end of the fifth week to exclude extreme outliers. The transformation is defined by the following function:

$$t : R_+ \rightarrow \{1, 2, 3, 4, 5\}, t(x) = \begin{cases} \left\lfloor \frac{x}{5} \right\rfloor + 1 & \text{if } 0 \leq x < 20 \\ 5 & \text{else} \end{cases},$$

where  $\lfloor \cdot \rfloor$  denotes the floor function. Based upon the resulting values  $v \in [1, 5]$  we can calculate the average number of working weeks (compared to working days). The clustering is, however, not applicable to the Number of Cycles, since NoC is measured in a different unit, and the 80%-Resolution Rate, which is a strongly aggregated value per definition. As we focus on working weeks where possible, we perform an inferential analysis on the clustered values only.

**Comparability and Seasonal Regression:** Planning data generation depends on multiple inputs, some of which include seasonal effect. At our industrial partner, for instance, new controlling numbers for the following year are available in November which are included into the forecast generation. Therefore, the information available for planning is not constant for all periods which requires a differentiation of the planning data by its delivery period. To avoid any mistakes caused by these seasonal effects, we proceed twofold: (i) we compare values of the same month (in 2009 and 2010), and (ii) we increase the comparability within one year through a seasonal regression. It is based on the additive component model as it can be found, in standard literature. In this manner, we are able to compare values of the same year and between the years.

**Non-parametric Analyses:** For the inferential analysis we conduct the non-parametric Wilcoxon signed-rank test as the compared two data samples always include the same subsidiaries and the data sample distribution for all phases are non-normal (cp. Section 4.1). Based upon this inferential analysis we examined the deviations between the pre-redesign values in 2009 and the post-redesign values in 2010. According to Field [28], we always add the 1-tailed level of significance  $p$  and the test statistic  $T$  (denoting the smaller value of the two rank sums) to the reported absolute value.

### 4.3 Results

In this section we present the detailed evaluation results of the productive use of the “Upload and Validation Service” at our industrial partner using the sample data and methodology described in Sections 4.2 and 4.3.

Tab. 3 shows the results for the average Number of Cycles (NoC) and the average Processing Time (PrT). The latter is listed on a working day basis and on a working week basis (clustered). For each delivery period, Tab. 3 shows the KPI’s values of 2009 and 2010 along with the relative deviation  $\Delta(09,10)$ . In addition, the overall line prints out the aggregated results over all planning periods per year, representing the average over 267 subsidiaries.

**Table 3.** Evaluation results: absolute KPI values (Average Number of Cycles, Average Processing Time unclustered and clustered) for 2009 and 2010 along with the relative deviation and the 1-tailed significance level (\* $p < .05$ ; \*\* $p < .01$ ; and \*\*\* $p < .001$ ).

KPI / Period	Average Number of Cycles			Average Processing Time			Average Processing Time Clustered		
	Value 2009	Value 2010	$\Delta(09,10)$	Value 2009	Value 2010	$\Delta(09,10)$	Value 2009	Value 2010	$\Delta(09,10)$
Jun	1.90	1.89	-1%	3.68	1.91	-48%	1.51	1.21	-19%**
Sep	1.48	1.72	+16%	2.82	1.51	-47%	1.40	1.16	-18%*
Nov	1.66	1.91	+15%*	3.03	2.01	-34%	1.42	1.24	-13%*
Overall	1.68	1.84	+9%*	3.18	1.81	-43%	1.44	1.20	-17%***

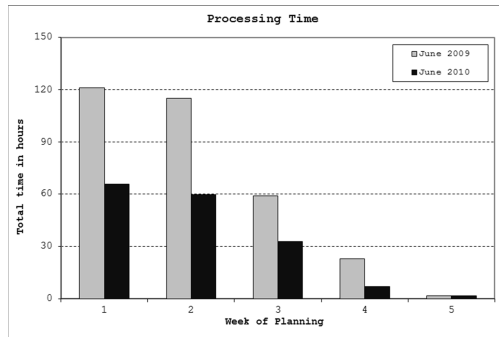
**Table 4.** Evaluation results: absolute KPI values (Average Waiting Time, Average Planning Time, both clustered, and 80%-Resolution Time) for 2009 and 2010 along with the relative deviation and the 1-tailed significance level (\* $p < .05$ ; \*\* $p < .01$ ; and \*\*\* $p < .001$ ).

KPI / Period	Average Waiting Time Clustered			Average Planning Time Clustered			80% Resolution Time		
	Value 2009	Value 2010	$\Delta(09,10)$	Value 2009	Value 2010	$\Delta(09,10)$	Value 2009	Value 2010	$\Delta(09,10)$
Jun	1.73	1.56	-10%	2.38	1.92	-19%**	18	13	-28%
Sep	1.44	1.38	-4%	1.91	1.67	-12%	15	11	-27%
Nov	1.56	1.39	-11%	2.00	1.72	-14%*	14	9	-36%
Overall	1.58	1.45	-8%*	2.10	1.77	-16%***	16	11	-30%

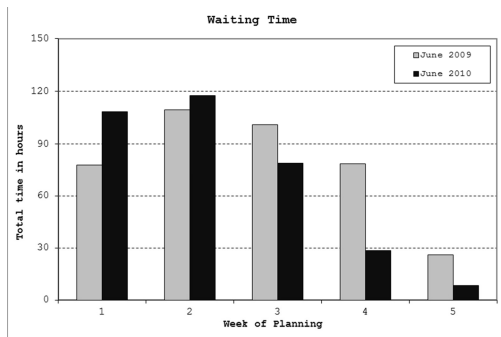
Starting with the average NoC, Tab. 3 indicates no deviation in June, however, clear increases of NoC in September (16%), November (15%) and in the overall numbers (9%). NoC in November (Overall) is significantly higher in 2010 than in 2009, with  $T = 497.50, p < .05$  ( $T = 3678.50, p < .05$ ). For the average PrT (working days), the improvement through redesign varies from 34% in November to 48% in June. For the clustered average PrT, the improvement ranges between 13% and 19%. The PrT is significantly lower in 2010 than in 2009 for all four observations,

$T = 73.50$  and  $T = 192$ ,  $p < .05$  in September and November,  $T = 119$ ,  $p < .01$  in June and even  $T = 1079.50$ ,  $p < .001$  in Overall.

Tab. 4 is structured analogously to Tab. 3. The Average Waiting Time (WaT), the average Planning Time (PIT) and the 80% Resolution Time (ReT) clearly decreases for all periods in 2010 compared to 2009. For WaT, the reduction ranges between 4% in September and 11% in November. The relatively small reduction in September is likely to be caused by the small absolute value (1.44 and 1.38 working weeks). Moreover, the Overall WaT 2010 is significantly lower than in 2009,  $T = 1723$ ,  $p < .05$ . The average PIT improves from 12% to 19% after the redesign. Again, Overall PIT 2010 is significantly lower than Overall PIT 2009,  $T = 365.50$ ,  $p < .05$  in November,  $T = 317$ ,  $p < .01$  in June and even  $T = 2905$ ,  $p < .001$  in Overall. Finally, ReT indicates a strong workload reduction. In 2009, ReT varied from 14 to 18 working days. In 2010, the highest value was observed in June with 13 working days. With respect to the Overall ReT decreasing from 16 to 11 working days, we can observe a reduction of one entire working week (which equals 30%).



**Fig. 3.** Processing Time in hours per week after delivery deadline. Comparison between June 2009 and June 2010.

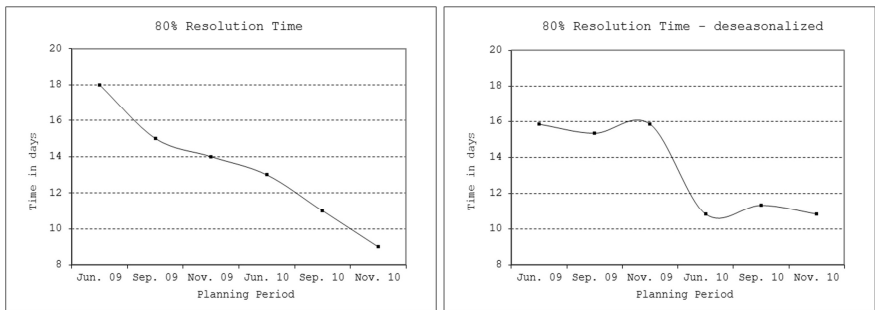


**Fig. 4.** Waiting Time in hours per week after delivery deadline. Comparison between June 2009 and June 2010.



We now turn our discussion to the results in Fig. 3 and Fig. 4 that illustrate the changes in PrT and WaT for the first delivery after the “Upload and Validation Service” was entirely rolled out (June 2010). They depict the working hours per week for the first five weeks after the delivery deadline. According to the original values printed out in Tab. 3, the PrT in June 2010 decreased by 48% compared to the values measured in June 2009. This development is clearly reflected in Fig. 3. In addition, Fig. 4 illustrates the reduced WaT in June 2010 which decreased by 10% compared to June 2009. Fig. 4 shows a second development which is certainly worth discussing: the WaT workload’s balance point shifts to the left, from approximately 2.7 in 2009 to approximately 2.1 in 2010. This shift towards the delivery deadline is a direct consequence of the reduced Processing Time. The subsidiaries receive the results of the holding validations earlier than before and, hence, can start to work on their response earlier, too. The Planning Time as the sum of PrT and WaT is consequently affected by both above-described effects. Since the shift of WaT towards the delivery deadline suggests that the validation process itself relocates, the ReT as the KPI to express a workload reduction or increase, is considered. Fig. 5 depicts the development of ReT for the six deliveries from June 2009 to November 2010 for (i) original values and (ii) for deseasonalized values (cp. Section 4.2 for more details on the seasonal regression applied).

The original values show a nearly linear decrease of ReT, from 18 working days in June 2009 to 9 working days in November 2010. If we are able to show that seasonal effects are present within the different delivery periods, a deseasonalized view of the data is appropriate. This is the case for the underlying data: ReT is calculated based upon PIT (cp. Section 3.2) and the aggregated 2009 and 2010 values for September significantly differ from the aggregated values for June ( $T = 1084,50, p < .01$ ). The same holds for the aggregated November values, which also significantly differ from the June values ( $T = 1544, p < .01$ ). Hence, it is legitimate to perform a seasonal regression on the ReT values. The result is depicted on the right side of Fig. 5 – the level of ReT before redesign (2009) is an entire working week higher than after redesign (2010), decreasing from an average of approximately 16 working days to an average of approximately 11 working days.



**Fig. 5.** Time necessary to finish 80% of the subsidiary plannings in working days. The original values are shown on the left side, the deseasonalized values are shown on the right side.

At large, all time-related KPIs (WaT, PrT, PIT) are significantly lower in 2010 (overall) than in 2009 ( $T = 1723$ ,  $p < .05$  for WaT,  $T = 1079.50$ ,  $p < 0.001$  for PrT,  $T = 2905$ ,  $p < .001$  for PIT). This reduction comes along with a one working week decrease in ReT. This improvement of all time-related KPIs is even more striking when we look at NoC at the same time. NoC, and hence the communication activity between subsidiaries and holding, increased significantly in 2010 (overall) compared to 2009 (overall),  $T = 267$ ,  $p < .05$ . In the following section, we discuss these findings and their implications, thereby also taking the industry perspective of our partner.

#### 4.4 Interpretation and Implications

The evaluation presented in this paper provides strong and significant improvements caused by process redesign in the financial planning domain. We are aware of potential biases in the data which may, for instance, be caused by organizational changes. Nevertheless, including the significantly increased NoC in November and additional expert interviews that were conducted after the study, we can definitely demonstrate the increased communication activity in September 2010 and November 2010. This activity is caused by the realization of intra- and inter-subsiary as well as inter-period planning validations. Such an extension of quality assurance measures would not have been possible without reducing the process runtime. It provides knowledge workers (who are the holding's financial planners) with additional capacity to perform valuable other tasks such as more sophisticated validations. These have, in turn, a huge potential to further improve liquidity management. Remarkably, the improvement of all time-related KPIs is striking even though the above-mentioned, new and probably time-consuming extended validations came into effect in March 2010. The reduction in process runtime unleashes valuable capacity and, hence, generates measurable business value.

Undoubtedly, the most important contribution of the evaluation performed in this work is its ability to actually quantify the benefit of a theoretical redesign model proposed in academia by implementing it in a *real-world setting of substantial business impact*. To date, such a redesign model has neither been implemented and integrated in business-relevant processes in a large company that acts worldwide nor been run over a significant time, producing a large set of real performance data.

## 5 Conclusion and Outlook

This article identified the requirements of redesigning semi-structured processes. Based upon that, the *objectives-based process redesign* [10] was chosen as the appropriate approach to put redesign into practice. We have implemented the proposed measures at our industry partner, a globally acting, large enterprise in the chemical and pharmaceutical sector, including a short note on its technical implementation "as-a-service". In this use case, we identified evidence to quantify the positive effects of the objectives-based redesign model. That way, we were able to show that theoretical redesign models cannot only be successfully realized in highly

relevant domains and enterprises, but also have the potential to significantly improve process time and data quality.

We scrutinized several key performance indicators of the financial planning process prior to and after its redesign to show and substantiate these findings. The core results of the evaluation demonstrated a reduction in Processing Time of up to 48% in working days and 19% in working weeks. Interferential analysis showed significant results for all deliveries in overall data. Moreover we showed an improvement of the overall Planning Time of up to 19% which is again significant for three of the four regarded data samples. These changes in Planning Time are strengthened by the KPI 80% resolution time. Here, we observed a reduction of up to one entire working week. These results clearly and favorably answer RQ1 (Does the objectives-based process redesign reduce process runtime in practice?)

Certainly, several business-related benefits for industry partner come along with the evaluation results (for detailed explanation cp. Section 4.4). Most importantly, the reduced Processing Time has unleashed resources that allowed for additional validations to increase financial planning accuracy and quality. The quality improvements yielded by the additional validations are strongly indicated by the significant NoC increase of 15% in November 2010 compared to November 2009. These findings provide a very strong indication to also positively answer RQ2 (Does the objective-based process redesign increase data quality in practice?). However, in order to be fully able to confirm RQ2, we will evaluate the results of future redesign iteration and service roll outs (cp. Section 3.3).

Critically assessing our approach, we were only able to implement iterations of the redesign model within one single enterprise so far. Nevertheless, our industry partner can be rated as archetypal for multinational enterprises and the challenges that arise from this kind of company structure. To enrich the practical experiences with the objective-based process redesign model, future work contains, besides the complete roll out of the implemented services, further evaluation cycles in intervals of one year to document the long-term effects of the redesign.

## References

1. de Kluyver, C.A., McNally, G.M.: Corporate Planning Using Simulation. *Interfaces* 10(3), 1–8 (1980)
2. Graham, J., Harvey, C.: The theory and practice of corporate finance: Evidence from the field. *Journal of Financial Economics* 60(2-3), 187–243 (2001)
3. Kim, C., Mauer, D., Sherman, A.: The determinants of corporate liquidity: Theory and evidence. *The Journal of Financial and Quantitative Analysis* 33(3), 335–359 (1998)
4. Lane, P., Milesi-Ferretti, G.: International financial integration. *IMF Staff Papers* 50, 82–113 (2003)
5. Pan, A., Vina, A.: An alternative architecture for financial data integration. *Communications of the ACM* 47(5), 37–40 (2004)
6. Lu, R., Sadiq, S.K., Governatori, G.: Compliance Aware Business Process Design. In: ter Hofstede, A.H.M., Benatallah, B., Paik, H.-Y. (eds.) *BPM Workshops 2007*. LNCS, vol. 4928, pp. 120–131. Springer, Heidelberg (2008)

7. Elsner, H., Vo, H.T.: Management of Portal Evolution - Introducing Evolution Management for the Corporate Financial Portal. In: *Wirtschaftsinformatik Proceedings*, vol. 2, pp. 337–352. Universitätsverlag Karlsruhe (2007)
8. Vo, H.T., Glaum, M., Wojciechowski, R.: Streamlining Foreign Exchange Management Operations with Corporate Financial Portals. In: Seese, D., Weinhardt, C., Schlottmann, F. (eds.) *Handbook on Information Technology in Finance*, pp. 123–139. Springer, Berlin (2008)
9. Riege, A.: Three-dozen knowledge-sharing barriers managers must consider. *Journal of Knowledge Management* 9(3), 18–35 (2005)
10. Martin, J., Conte, T., Knapper, R.: Towards Objectives-Based Process Redesign. In: 17th Americas' Conference on Information Systems, Paper 150, Detroit (2011)
11. Deiters, W.: Information gathering and process modeling in a petri net based approach. In: *Business Process Management*, pp. 291–314 (2000)
12. Pentland, B.T., Rueter, H.H.: Organizational routines as grammars of action. *Administrative Science Quarterly* 39, 484–510 (1994)
13. Pentland, B.T.: Sequential variety in work processes. *Organization Science* 14(5), 528–540 (2003)
14. Rosenkranz, C., Seidel, S., Mendling, J., Schaefermeyer, M., Recker, J.: Towards a Framework for Business Process Standardization. In: Rinderle-Ma, S., Sadiq, S., Leymann, F. (eds.) *BPM 2009. LNBP*, vol. 43, pp. 53–63. Springer, Heidelberg (2010)
15. Van der Aalst, W.M.: Workflow Verification: Finding Control-Flow Errors Using Petri-Net-Based Techniques. In: van der Aalst, W.M.P., Desel, J., Oberweis, A. (eds.) *Business Process Management. LNCS*, vol. 1806, pp. 161–183. Springer, Heidelberg (2000)
16. Van der Aalst, W.M., Weske, M., Grünbauer, D.: Case handling: a new paradigm for business process support. *Data and Knowledge Engineering* 53, 129–162 (2005)
17. Reijers, H.A., Mansar, S.L.: Best practices in business process redesign: an overview and qualitative evaluation of successful redesign heuristics. *Omega* 33(4), 283–306 (2005)
18. Redman, T.C.: Improve Data Quality for Competitive Advantage. *Sloan Management* 36(2), 99–107 (1995)
19. Davenport, T.H.: *Process innovation: reengineering work through information technology*. Harvard Business School Press, Boston (1993)
20. Van der Aalst, W.M., Weske, M.: The P2P Approach to Interorganizational Workflows. In: Dittrich, K.R., Geppert, A., Norrie, M. (eds.) *CAiSE 2001. LNCS*, vol. 2068, pp. 140–156. Springer, Heidelberg (2001)
21. Balasubramanian, S., Gupta, M.: Structural metrics for goal based business process design and evaluation. *Business Process Management Journal* 11(6), 680–694 (2005)
22. Seidel, S.: Toward a theory of managing creativity-intensive processes: a creative industries study. *Information Systems and e-Business Management* 9(4), 407–446 (2011)
23. Recker, J.: Opportunities and constraints: the current struggle with BPMN. *Business Process Management Journal* 16(1), 181–201 (2010)
24. Wohed, P., van der Aalst, W.M.P., Dumas, M., ter Hofstede, A.H.M., Russell, N.: On the Suitability of BPMN for Business Process Modelling. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) *BPM 2006. LNCS*, vol. 4102, pp. 161–176. Springer, Heidelberg (2006)
25. Buchsein, R., Machmeier, V.: *IT-Management mit ITILR V3: Strategien, Kennzahlen, Umsetzung*. Springer, Berlin (2008)
26. Kuetz, M.: *Kennzahlen in der IT: Werkzeuge für Controlling und Management*. Dpunkt, Heidelberg (2010)
27. Martin, J., Caton, S., Conte, T., Weinhardt, C.: Towards Financial Planning as a Service. In: 8th IEEE International Conference on Service Computing, pp. 767–768. IEEE Computer Society, Washington, D.C (2011)
28. Field, A.: *Discovering Statistics Using SPSS*. SAGE, London (2009)

# Building Information System Variants with Tailored Database Schemas Using Features

Martin Schäler<sup>1</sup>, Thomas Leich<sup>2</sup>, Marko Rosenmüller<sup>1</sup>, and Gunter Saake<sup>1</sup>

<sup>1</sup> School of Computer Science, University of Magdeburg, Germany  
{schaeler, rosenmueller, saake}@iti.cs.uni-magdeburg.de

<sup>2</sup> METOP Research Institute, Magdeburg, Germany  
thomas.leich@metop.de

**Abstract.** Database schemas are an integral part of many information systems (IS). New software-engineering methods, such as *software product lines*, allow engineers to create a high number of different programs tailored to the customer needs from a common code base. Unfortunately, these engineering methods usually do not take the database schema into account. Particularly, a tailored client program requires a tailored database schema as well to form a consistent IS. In this paper, we show the challenges of tailoring relational database schemas in software product lines. Furthermore, we present an approach to treat the client and database part of an IS in the same way using a variable database schema. Additionally, we show the benefits and discuss disadvantages of the approach during the evolution of an industrial case study, covering a time span of more than a year.

**Keywords:** Tailoring DB schemas, feasibility study, software product lines.

## 1 Introduction

Databases (DB) are an integral part of many information systems (IS). In *software product lines* (SPL), software engineers aim at creating tailor-made software systems with the help of reusable artifacts [47]. An SPL forms a group of similar software systems sharing a set of identical and different functions. The reuse of high-quality artifacts in SPLs reduces the effort for maintenance and further development [6]. Although the use of SPLs for producing executable program code has been researched quite intensively, the impacts on data management and especially DB schemas are still fragmentary [10,16,21,23]. To generate a certain program (*variant*) of an SPL, a customer selects the *features* that he wants to include into his variant. A feature is a characteristic of a program that is relevant for some stakeholder [6]. Imagine a simple workflow management system that has an optional feature: *logging*. This feature is not part of every variant of the system, but can be chosen by a customer. Depending on the implementation of the *logging* feature, the DB schema needs additional relations containing the log data whenever a customer chooses this feature. Therefore, we need a possibility to *tailor the DB schema* according to the application's features when creating a new IS. Note, that simply designing a distinct schema for every potential variant manually is practically impossible, because the number of potential variants increases exponentially

with the number of (optional) features. In addition, we argue that traditional engineering methods, such as using a global schema for all variants (see Section 3.2) raise the following challenges:

- Increased effort for maintenance and SPL evolution (e.g., integrate new features),
- Design limitations (e.g., no support for alternative features and table partitioning),
- Complex and thus hard to understand DB schema,
- Data integrity problems, because of missing integrity constraints in the DB schema,
- Solutions that do not scale, because of large number of variants.

In contrast to traditional engineering approaches, we aim at *generating a tailored* DB schema for every information-system variant of an SPL just as we tailor source code. To this end, we analyze whether the mechanisms used in SPL client programs, can be applied to the database schema to face the already mentioned problems. We extend previous work [21] suggesting to apply SPL techniques to ER modeling. This idea is adopted and extended to describe a holistic approach to model and generate an IS including client program and DB schema. Furthermore, we analyze the benefits and drawbacks of this approach. To the best of our knowledge, we provide the first empirical study that shows whether the generation of DB schema variants is possible for a specific data model and for an existing IS. Particularly, we investigate the following points:

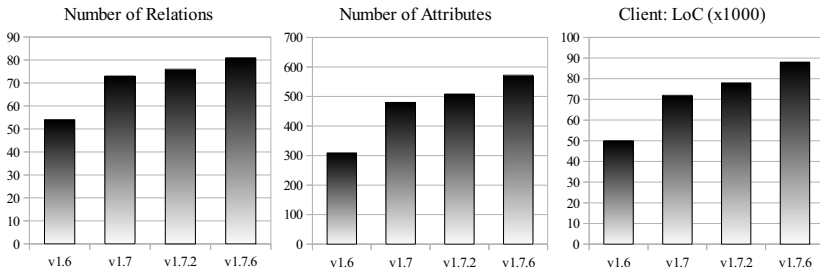
1. Analyze why currently used approaches are not sufficient to face the previously mentioned issues,
2. Suggest an approach to tailor a DB schema to the client-program requirements and thus a new holistic approach for IS engineering,
3. Determine the *feasibility* of our approach for real-world applications and show *advantages* and *disadvantages* compared to existing strategies,
4. Investigate how the benefits and drawbacks *change over time*, when integrating new features and extending existing ones within the IS.

## 2 Software Product Lines Case Study

In the following, we introduce an SPL case study that we use to show the feasibility of our approach as well as to evaluate its benefits and drawbacks.

### 2.1 The ViT-Manager

The ViT<sup>®</sup>-Manager is a family of industrial controlling tools for continuous improvement processes of the METOP Institute. A continuous improvement process defines a structured approach with regularly (continuous) meetings to identify and solve problems within a company, which increases productivity. Because the ViT-Manager is used commercially in several companies with different size, internal structure, and portfolio, the customers have a set of equal and differing requirements w.r.t. to one variant of the application. To face these challenges, the ViT Development Team refactored the application using a plug-in infrastructure to form an SPL. In the ViT Framework, a customer chooses additional features according to his needs, when generating a new variant of the



**Fig. 1.** Size of DB schema and client implementation for different versions of the ViT-Manager

application. While the client program was decomposed into *features* during the refactoring, the DB schema was not. Instead, a global DB schema for all variants was applied. We discuss disadvantages of this alternative solution in Section 3.2.

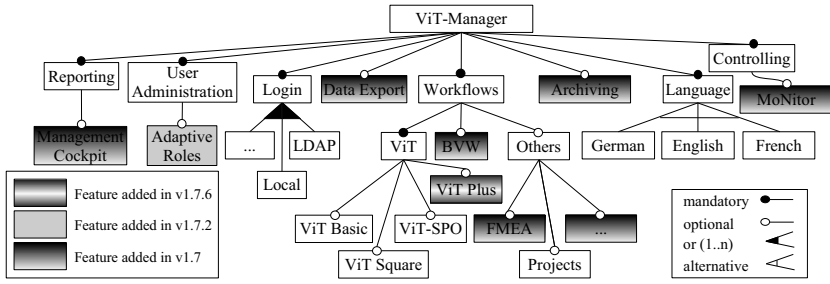
As we depict in Figure 1, the DB schema contained 53 relations with 309 attributes in version 1.6 (first version for which we implemented a variable schema, Feb. 2010). It increased to 81 relations including 572 attributes in version 1.7.6 (current version, Mar. 2011). Similarly, the size of the implementation on client side grew from about 50 thousand lines of code (KLoC) in version 1.6 to 88 KLoC in version 1.7.6. We consider this case study as appropriate to demonstrate our approach, because it is large enough to represent real-world challenges, while the evaluation complexity is manageable. Additional benefits are:

**Productive use.** The case study is used commercially. Hence, the danger of choosing a case study that does not include relevant problems is minimized. This improves the generalized results concluded from analyzing the case study.

**Changes over time.** Considering different versions of the case study (covering more than a year) allows us to extract results and conclusions about positive or negative effects of our approach during the evolution of the SPL case study. This includes adding new features to the SPL or extending and changing existing ones.

## 2.2 Feature Model of the Case Study

A *feature model* (FM) describes features and their relationships in a particular SPL [6]. Particularly, it defines, which feature combinations form a valid variant. Thus, an FM represents the variability of the SPL containing all valid feature combinations. In Figure 2 we depict the FM of the case study. There are four different types of feature relationships: mandatory, optional, alternative, and or-relationships. For instance, all variants of the case study have to contain a *Reporting* feature, because it is mandatory. In contrast, optional features do not have to be included. If a customer wishes to have a more detailed *Reporting* functionality, he chooses the optional *Management Cockpit*, which is then included into the generated program variant. Furthermore, one or more features connected via an *or*-relationship can be part of a particular variant. For instance, the *Login* of the case study can be performed via *LDAP*, *Local Authentication*, or both features may be included. Finally, the alternative-relationship defines that exactly one of the supported



**Fig. 2.** Excerpt of the feature model of the case study

features can be included into one variant. For instance, to generate a variant of the case study a customer selects one of the supported language features (e.g., French). In Figure 2 we visualize the version history as well. Features such as *Reporting* and *Login* have been part of v1.6, whereas, the major release v1.7 contains several new *optional* features such as *Archiving* and *BVW*. In contrast, the minor releases v1.7.2 and v1.7.6 contain just a small number of new features and improvements of existing ones.

### 3 Problem Statement and State of the Art

Different users of an application have different requirements to its functionality. In client programs, these differences can be faced by using SPL techniques, such as components. While these techniques have been intensively studied for variability of executable program code, the variability at the DB schema level is still fragmentary [10,16,20,23]. In the following, we analyze different currently used approaches to model DB schemas in SPLs. Therefore, we define requirements that a tailored DB schema must fulfill. Note, that we can create thousands of valid variants of the case study. Thus, it is not possible to create every single tailored DB schema variant manually.

#### 3.1 Requirements for Tailored DB Schemas in SPLs

To analyze different approaches, we consider several attributes of the process to create the single schema variants and of the resulting schema variants itself. We use two different criteria to describe the *modeling process* of a certain approach:

- Modeling complexity.** This criterion points out the complexity of creating the model of the (variable) DB Schema, which contains the basis for the DB schema variants.
- Expressiveness of the model.** In traditional modeling methods, the expressiveness of the model is limited. For instance, in a global schema, an engineer cannot integrate conflicting schema elements introduced by alternative features.

*Evaluation of Schema Variants.* The requirements for the single schema variants are:

- Completeness.** All DB schema elements (*relations and attributes*) necessary to perform the read and write operation in the single features included in this variant must be present in the variant's DB schema.



**Complexity of schema variants.** The size of the DB schema (number of *relations and attributes*) shall be reduced to a minimum. Particularly, there shall be no unused schema elements. We argue, that this improves the understandability of the schema variants, which is important for maintenance and further development.

**Data integrity.** All integrity constraints have to be included for the schema elements in a specific schema variant, to guarantee consistent data in the IS. This includes *primary and foreign keys, attribute value domains, and not null* as well as *check constraints*.

### 3.2 Limitations of Currently Used Approaches

Subsequently, we discuss three traditionally used approaches and describe which problems arise when applying them for SPL development. This discussion motivates the necessity for a new approach as well as it serves as basis when analyzing benefits and drawbacks of our new approach in Section 5.

**Global DB Schema.** Often, for every variant of an SPL the same global schema is applied [21]. This schema contains every schema element that is used at least in one variant. Thus, the schema contains every schema element that is necessary (*completeness*). On the other hand, major parts of the global schema can be unused in this particular variant. Consequently, the highly complex schema is unnecessarily hard to understand, which complicates maintenance and evolution of the SPL. Additionally, unused parts can impose integrity problems. Imagine an attribute used only in one variant. When it is used it shall be not null, or some check constraint is defined on it. Because it is part of *every* variant, you cannot define *integrity* constraints at the schema level, but on client-program side, which can lead to inconsistent data (e.g., by programming errors). Furthermore, the global schema does not exist in every case. As a result, conflicting schema elements introduced from alternative features cannot be defined in a global DB schema (expressiveness). To circumvent this problem, YE et al. discuss that overloading schema elements (i.e. using the same schema element for different purposes instead of renaming it) is possible, but causes highly confusing DB schemas [23]. Using a global schema can be seen as the standard approach, hence the modeling complexity is used as reference for all other approaches [21].

**View-Based Approaches.** View-based approaches [5] generate views on top of the global schema that emulate a schema variant for the client, which may be seen as an annotative approach [13]. Thus, the global schema is still part of every DB schema variant, the approach inherits the problems of the global schema. Unfortunately, the variant's schema complexity does even increase, because the additional schema elements for the view, emulating the variant, have to be included as well. Furthermore, there is additional effort to generate views when modeling the DB schema. This approach has benefits in data integrity, because the views emulating the schema variants can contain additional integrity constraints, which cannot be included into the global schema. Thus, the expressiveness of the model is also better than in the global schema approach.

**Framework Solutions.** In a framework approach, the plugins implement the features of the SPL and therefore this approach is a form of physical decomposition [13]. A plugin contains additional program code and an own DB schema. The DB schema variant is build from the single plugins that add the additionally required schema elements [21]. Thus, it fulfills the *completeness* requirement. Consequently, it also contains only schema elements that are needed in this variant. Unfortunately, using frameworks could lead to table partitioning, when two or more plugins use the same real world entity. Hence, this has negative impact on the complexity of the schema variants and limits the modeling expressiveness. Furthermore, consistency checking is implemented on client side, because there are no intra-plugin integrity constraints, which can lead to data integrity problems. The effort to model the DB schema is higher than modeling a global schema, because you have to take care of additional challenges such as naming conflicts, which are usually solved by naming conventions.

### 4 Solution - Holistic Approach for Client and DB Schema

This section contains the basic overall idea of our approach and explains how to obtain the variable DB schema. Therefore, we describe the relationship between features and DB-schema elements. Furthermore, we present a semi-automatic approach to relate schema elements to features.

**Basic Idea.** To create a tailored variant of the IS variant, a customer first selects the desired features (see Figure 3(a)). Every feature contains one model for its functions in the client program and another one for the DB schema (Figure 3(b)). Furthermore, each of the two models of a particular feature points to different implementation artifacts (e.g., source code files). Finally, a well defined composition strategy (see Section 4.3) creates the variant including the tailored client program and DB schema.

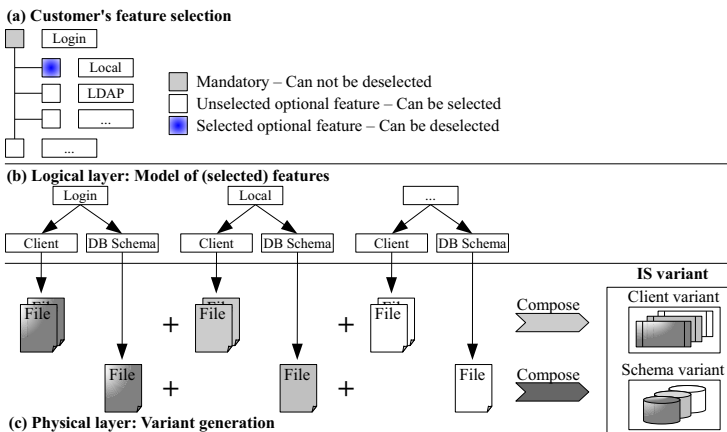


Fig. 3. Basic idea to create a particular variant

## 4.1 Relationship between Features and DB Schema Elements

As mentioned in Section 2, a feature is some characteristic that is important for some stakeholder. Furthermore, the features are the units a customer selects to create a tailored variant, which includes a DB schema. Therefore, we have to define the relationship between features and DB schema elements to model the *variable* DB schema. Moreover, we need to discuss the interaction of features on client and data-base-schema level. A feature at DB schema level contains all schema elements (relations, attributes, and integrity constraints) that this feature on client side needs to perform the read and write operations within its specific source code on client side. Thus, we tailor the DB schema w.r.t. the requirements of the feature on client side. As a result, there is one feature model for the whole IS allowing us to easily generate the single variants of the IS (see Section 4.3).

Note that the relationship between a feature on client and DB schema level ensures *completeness* and minimizes the *complexity* of the DB schema variant (see 3.1). Every schema element that a selected feature needs in a variant, is contained in the feature and therefore added to the schema variant during composition. Furthermore, the schema variant contains no unused schema elements for the same reason. Alternatively, the feature models could contain only the additionally required schema elements. This approach has the benefit that schema elements cannot be mapped to multiple features. This can lead to conflicting definitions of a schema element in *one* variant during the evolution of the SPL. The drawback of this alternative is, that we have to decide to what feature a schema element belongs, especially when they are only required by optional features. However, a high number of redundancy suggests a refactoring of the client programs source code.

**Mapping Features to DB Schema Elements.** Previously, we defined that a feature in the variable DB schema is a mapping of schema elements to this feature. Furthermore, we show how to create this mapping of schema elements to features representing the variable schema. To create the model of a variable DB schema there are two different alternatives:

1. The single features can be extracted from a previously used global DB schema.
2. Especially when creating the SPL from scratch, features on DB schema level are modeled in features instead of views, which have to be composed to form a schema variant.

Extracting the features from a global schema has the advantages that there is already a consistent DB schema. The main task in this case is mapping schema elements to features in a reasonable way. In contrast, modeling the features without a consistent schema, a development team faces additional challenges (e.g., naming conflicts) widely known from view integration [3].

Currently, SPL techniques are often used when traditionally methods are reaching their limitations [6]. That means that previously monolithic software is refactored into an SPL such as the case study. Thus, in these cases there also is a global schema and therefore we concentrate on this alternative.

## 4.2 From a Global DB Schema to a Variable Schema

Subsequently, we explain a semi automatic schema decomposition to map schema elements to features based on the client implementation. Therefore, we assume that a global previously used DB schema exists, because this is the standard case as we discussed in Section 3.2. The important point is that the mapping fulfills the *completeness* requirement recently defined.

**Preconditions.** The schema decomposition needs two preconditions: (1) Previously used global DB schema and (2) Client implementation that is separated into features. First, the decomposition needs a previously used global schema, because we need the schema elements, which includes element names, value domains (attributes), and integrity constraints defined for each particular element. The second precondition is that the client implementation is separated into features, which is true in SPLs. Consequently, the decomposition can identify all necessary schema elements (including integrity constraints) for a particular feature as postulated in the relationship of a feature on client and DB schema level.

**Semi Automatic DB Schema Decomposition.** The decomposition procedure determines the mapping of schema elements to every feature in the FM. First, for every feature in the FM the decomposition identifies the database operations (e.g., select and insert queries) in the source code. Second, for all of these operations it identifies the required schema elements and adds them to the feature's DB schema model. For instance in *Select* queries, the decomposition adds attributes used in the *select* and *where* part as well as the relations named in the *from* part to the model of the particular feature. Moreover, when adding a schema element to the feature, additional information extracted from the global schema, are added to the model, such as an attribute value domain and integrity constraints. In Figure 4 we show the result of the schema decomposition for two features of the case study. Each feature contains one relation (usually a feature contains several relations) and a specific number of attributes in a syntax form that is closely related to the SQL-Standard. Note, that integrity constraints, such as primary keys (PK) and not null constraints (NN) can be added to an attribute.

**Problem Referential Integrity.** As we presume that there is a previously used (consistent) global schema, there are no problems with integrity constraints (e.g., primary keys). By contrast, foreign keys are problematic if they reference from or to a schema element not existing in a particular variant. To avoid this problem, we propose rules that ensure that the referenced schema elements are present in the variant. The rules representing valid introductions of foreign key within a feature are: (1) The schema elements are located within the feature itself, (2) the referenced element is part of a parent feature, or (3) the referenced element is part of a feature the current feature has a *requires* relationship to. As a result, it is generally not possible to define foreign keys between optional features. For instance, in Figure 2 it is not possible to define a foreign key between the features *ViT Basic* and *ViT Square*. Whereas, it is possible to define foreign key in feature *Projects* to schema elements of feature *Others*, because *Others* is a parent feature (rule 2). If one really wants to have foreign keys between optional



superimposing DB schemas, the input consists of several DB schemas (model of features at DB side) and the result is a DB schema containing all schema elements of the input schemas, without any duplicate relations or attributes. To create a schema variant (V), the DB schemas of all selected features ( $f_i$ ) are superimposed.

**Structure of Features at Implementation Level.** The  $\bullet$ -operator works on *feature structure trees* (FST), which represent the internal structure of the implementation of a feature. They can be seen as simplification of an abstract syntax tree [2]. In Figure 5, we show the FST of the already known feature *Login* and *Local Authentication* of the case study on DB schema level. The FST can be generated quite intuitively from the model as shown in Figure 4. The result of the composition contains all attributes of the input features with the desired integrity constraints. FST nodes of two features that share the same path from the root node (e.g., attribute *login\_name*) are merged and thus show only once in the result as intended.

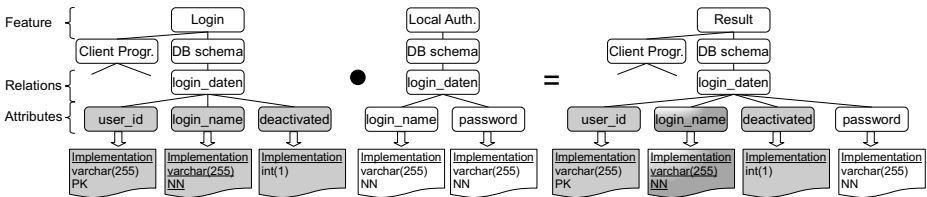


Fig. 5. Composition of feature structure trees (FST)

## 5 Evaluation

In the following, we apply the previously presented approach to the case study and evaluate the results. Therefore, we first analyze the feasibility of our approach before discussing its impacts on maintenance, data integrity, and further development. Finally, we evaluate the overall advantages and drawbacks of our approach compared to the traditionally used ones from Section 3.2. To ensure validity of the evaluation, we rely on interviews of the ViT Development Team, which is experienced in applying global DB schemas to SPLs, because the case study previously used a global schema. Furthermore, this team tested the variable schema approach for more than a year. Finally, we strengthen our conclusions with specific measurements if possible.

### 5.1 Feasibility of the Approach

We want to evaluate whether the variable DB schema approach is manageable in practice. Thus we also have to evaluate the modeling process, the result of the modeling (features), and the variant generation process. As stated in Section 4, the modeling process, which includes the decomposition of the global schema into features is laborious. Furthermore, the decomposition procedure is not generally automatable, because of open research challenges and thus needs manual recall. However, according to interviews the

modeling is still manageable. Moreover, the modeling effort is an investment for the automated generation of a tailored DB schema variant. A customer simply chooses the desired features and the DB schema (and the client) are generated automatically. Additionally, the size of the features scales in the case study (see Figure 1), because the size on client and on DB side seems to correlate. This effect does not change significantly over time. As visualized in Table 1, even when comparing two major releases such as v1.6 and v1.7 the correlation exists.

**Table 1.** Correlation of size on DB and client side

Features v1.6	Rank	Size DB/client	Attributes	Relations	Features v1.7	Rank	Size DB/client	Attributes	Relations
ViT	1	1	88	16	Archiving	1	4	124	17
ViT-SPO	2	2	80	14	ViT-SPO	2	1	105	16
BVW	4	3	75	12	ViT	3	2	88	16
ViT Square	3	4	65	12	BVW	4	5	75	12
ViT Basic	5	5	63	12	ViT Square	5	3	66	12
...					...				
LDAP	24	21	1	1	LDAP	30	26	1	1

**Additional Challenges Due to Feature Interactions.** We identified two additional challenges of the variable schema approach: Redundantly defined schema elements and changing feature models. Redundant schema elements increase the size of a feature on database side. Thus, the features were extracted from global schema, two features that can be present in one variant cannot contain different models of the same schema element, which would produce an error during composition. But, when modeling the features without a previously used schema this challenge has to be faced, which rises also in view integration. The second challenge results because of the combination of optional features. The schema variant might need additional schema elements when a customer chooses two optional features to generate a variant. These schema elements are not part of the schema variant when only one of them is chosen. This effect is also known on client side (e.g., glue code), but to the best of our knowledge it has not been identified at DB schema level.

In the case study, there is an optional *Archiving* feature that archives data of different *optional* features, such as *ViT basic* or *BVW*. Thus, these optional features are not part of every variant. Therefore, the archiving feature does not need the tables to archive the *BVW* data if the *BVW* feature is not included into the variant. To minimize the complexity of the schema variant, these unneeded relations should not be included. Hence, we move the additionally needed schema elements into *derivatives*, which are included automatically when both optional features are part of the variant [15]. Particularly, the composition mechanism includes *Archiving.BVW* (Figure 6) into the schema variant, when a customer selects *BVW* and *Archiving*. Using Derivatives raises the expressiveness of the model and decreases the size of the *Archiving* feature in some variants dramatically, by making the modeling process more complex.

**Trade-Off.** According to our interviews, the trade-off between the drawbacks of our approach, such as additional modeling overhead, and the advantages like automated variant generation are beneficial only when handling multiple variants of an application. But the specific costs, such as modeling the variable schema, related to the amount

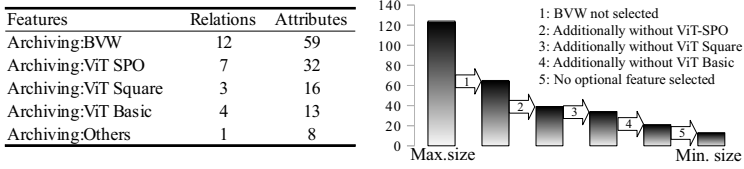


Fig. 6. Derivatives: Size of the Archiving features in different feature combinations

of time saved (e.g., for manually tailoring a variant or additional bug fixing if using a global schema) to the customers needs, is imprecise for the following reasons. The complete costs for alternative solutions are unknown, because these approaches have not been implemented and the previous costs can hardly be taken into account because the tasks are highly different. Moreover, there a lot of fine-grained costs where the specific amount of time is unknown. For instance for bug fixing in the variable-schema approach, we do not know the amount of time, as only the time from reporting the bug until fixing it is known. After all, the numbers we have and interviews with the developers suggest that for the *ViT-Manager* the variable-schema approach is beneficial when there are at least five (different) variants used productively.

### 5.2 Improving Maintenance, Data Integrity, and Future Development

Improving the *maintenance* has two aspect. First, the *effort estimation* to maintain the DB schema or the data inside shall be improved. When using a global schema, it is complex and thus hard to understand. Hence, the effort estimation is often imprecise, because of possible side effects. Furthermore, the maintenance process itself shall be supported in a positive way. This issue is highly related to reducing the *variants schema complexity*. Here we argue that not having unnecessary schema elements in the variant schema is a suitable way to reduce its complexity. The effort to maintain a schema variant becomes more predictable, because we have a mapping of a particular function to schema elements (features). Thus, the estimation of possible side effects becomes easier.

The complexity of a schema variant is minimized, because it contains only necessary schema elements. Therefore, the understandability of the DB schema variants rises and is additionally supported by the mapping of schema elements to features. When comparing the complexity reduction of the schema variants in different versions (Figure 7), the benefits of the variable schema become even more obvious for the case study. The minimum configuration (no optional feature selected) of v1.6 covered about the half of the maximum configurations schema elements (all optional features selected). Whereas, the difference in v1.7.6 is about one fourth. Furthermore, the DB schema of the minimum configuration remains nearly stable. This is, because many new *optional* features have been added to the case study, but the core functions remained roughly unchanged, which is common in SPLs. Thus, we conclude that the benefit of our approach increases over time, when new optional features are added to the SPL.

According to our interviews, after introducing the variable schema the number of support request slightly increased for a quarter of a year, especially because of problems related to the db schema. After this, there was significant decrease of support request



even if introducing new features. We hypothesize that one of reason therefore is the variable schema, but we cannot say to what extend.

**Strong Benefit: Improving Data Integrity.** One of the strongest point supporting the variable schema approach, is the benefit w.r.t. data integrity. Using the variable schema allows to reintroduce the integrity constraints that had to be dropped (see Section 3.2), because of the global schema approach. Additionally, integrity constraints that were encoded on client side, can now be added to the DB schema. As a result the variable schema approach *ensures integrity* of the data and is therefore beneficial here. Furthermore, tricks as inserting dummy values to circumvent Not Null constraint for attributes not used in this variant are not necessary any more. Therefore, the variable schema approach is beneficial for *data quality* as well. Moreover, the rules from Section 4.2 guarantee that we can only introduce valid integrity constraints and that additional restrictions of the variability of the SPL (introduced by foreign keys referencing from an optional feature to a different optional feature) are visible in the Feature Model.

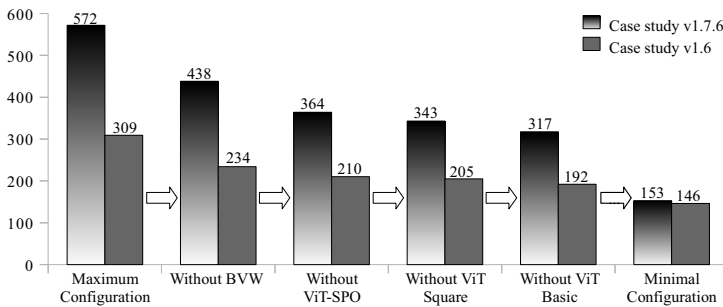


Fig. 7. Complexity reduction of the schema variants in different versions of the case study

**Further Development.** Extending the case study with new features becomes easier and is more predictable in the variable DB schema approach. First, a software engineer can simply add new features to the composition process. Furthermore, the mapping between features and schema elements is helpful when designing new features or extending existing ones. This was highly useful, when designing the *Archiving* feature and its *derivatives*, because for every feature was know, which schema elements are necessary and thus have to be archived. Challenges arise when modifying schema elements used in different features in preserving the model's consistency. Additionally, we see naming problems when adding new features, which also arise in view integration. However, we are confident, that this problem can be solved with adequate tool support.

### 5.3 Comparison to Existing Approaches

In Table 2 we resketch the result of currently used approaches of Section 3 and compare them to the variable schema approach. The scoring of variable schema is based on the

discussion in Section 3.2. The variable schema has strong benefits due to the complexity reduction of schema variants and the improvement of data integrity (also helping to improve data quality). Furthermore, the model expressiveness increases, which allows conflicting schema elements and changing feature models that is not possible with a global schema approach. By contrast, the global schema approach and frameworks have benefits w.r.t. modeling complexity. The variable schema needs a previously used global schema, which has to be decomposed into features. Alternatively, the modeling of features instead of views, including all derivatives is also more complex than modeling one global schema. This is the trade-off, our approach needs to improve maintenance and further development.

**Table 2.** Evaluation of the variable schema approach compared to currently used approaches

	Modeling		Schema Variants		
	Complexity	Expressiveness	Completeness	Data Integrity	Schema Complexity
Global DB Schema*	+	--	++	-	-
View Approach*	--	-	++	+/-	--
Framework Solutions	+/-	+/-	++	-	+
<b>Variable Schema</b>	-	+	++	++	++

++ very good, + good, 0 neutral, - unhandy, -- very unhandy \*Approach only possible if global schema exists.

## 6 Related Work

There has been little work published in literature that deals with tailoring DB schema in SPLs or similar concepts. In [23], Ye et al. discover the need for tailoring software systems at different levels and of different granularity, which includes the DB schema. In their analysis of an industrial case study, they discover the use of schema element overloading in a global schema, which creates a hard to maintain DB schema. Furthermore, service engineers add additionally required schema elements manually. However, this approach is only possible in case studies with very little variability at DB schema level.

In his PhD Thesis [16], MAHNKE proposed a component approach that is similar to the framework solution described in Section 3.2. Unfortunately, his ABLE-SQL requires an extension of the SQL:1999 standard, which is not necessary in our variable schema approach. Whereas view-based approaches, such as [18][22], use view integration strategies or view tailoring [5]. In Section 3.2 we also discovered limitations of these approaches because they need a global schema and build views on top of the global schema. Thus, the complexity of the schema variants even increases.

Recently, DYRESON et al. [10] presented an aspect-oriented approach to integrate optional features into XML, which the authors adopted to the relational data model in [9]. This approach needs a change of the query processing to evaluate the aspects, which is not necessary using the variable schema, because our approach is totally transparent to the underlying DB system.

In [17], RASHID identified the lack of variability, but concentrates on evolutionary changes of the DB schema in his aspect-oriented framework approach and not on composing different tailor-made schema variants. Furthermore, model-merging approaches

suggest more complex mechanism (e.g., Comparison phase to identify equivalent elements) than our approach and are thus more expressive [12][8][19]. However, superimposing Feature Structure Trees is less complex and sufficient for our purposes.

Finally, our approach uses composition (physical separation of concerns) [13] as the schema elements of each feature are kept in their own file (refer to Figure 3) and need to be composed to form a particular variant. Alternatively, it may be possible to use an annotative approach (virtual separation of concerns) as there are solutions such as CIDE [14] and the concept generally is language independent as well. However, as we needed to integrate new features during the evolution of the case study, we applied the compositional approach. As a result, we can simply integrate new features into the composition process or delete existing ones without changing respective annotations in the virtually separated schema.

## 7 Conclusion and Future Perspectives

In this paper, we have shown the feasibility of building IS variants with tailored DB schemas in SPLs using a variable schema. First, we analyzed the limits of existing approaches. Second, we discussed the relationship of features on client and database side. Third we showed how to model a variable schema and use the superimposition composition mechanism that generates a particular IS variant having a tailored schema automatically. The approach has been applied to a case study to show advantages and disadvantages compared to existing approaches and its influence over time. The variable schema has significant advantages, such as reducing the complexity of DB schema variants reduction and increased expressiveness. Whereas, the modeling is much more complex compared to existing approaches.

In future, we have to solve several challenges when decomposing a previously used DB schema into features. Furthermore, a better tool support is highly needed to apply our approach to large-scale case studies and to verify the results conducted from the ViT-Manager case study. Thus, we want to integrate our approach into existing solutions (e.g., [1]) to tailor source code or to visualize the relationship between features and respective implementation artifacts such as [11]. As result, we will furthermore support the holistic design of IS. Better tool support additionally includes research for multi-lingual variability (e.g., SQL and Java). Finally, we will analyze how to upgrade one variant to a different one efficiently (i.e., schema and data). Especially we will analyze, which benefits and (new) challenges arise in DB schema evolution.

**Acknowledgements.** This paper has been funded in part by the German Federal Ministry of Education and Science (BMBF) through the Research Program under Contract FKZ: 13N10818 and 13N10819. We would also thank Martin Kuhlemann and Sandro Schulze for their support for this paper.

## References

1. Apel, S., Kästner, C., Lengauer, C.: Featurehouse: Language-independent, automated software composition. In: Proc. Int'l Conf. on Software Engineering, pp. 221–231. IEEE (2009)

2. Apel, S., Lengauer, C.: Superimposition: A Language-Independent Approach to Software Composition. In: Pautasso, C., Tanter, É. (eds.) SC 2008. LNCS, vol. 4954, pp. 20–35. Springer, Heidelberg (2008)
3. Batini, C., Lenzerini, M., Navathe, S.: A comparative analysis of methodologies for database schema integration. *ACM Computing Surveys* 18, 323–364 (1986)
4. Batory, D., Sarvela, J., Rauschmayer, A.: Scaling step-wise refinement. *IEEE Transactions on Software Engineering* 30(6), 355–371 (2004)
5. Bolchini, C., Quintarelli, E., Rossato, R.: Relational Data Tailoring Through View Composition. In: Parent, C., Schewe, K.-D., Storey, V.C., Thalheim, B. (eds.) ER 2007. LNCS, vol. 4801, pp. 149–164. Springer, Heidelberg (2007)
6. Clements, P., Northrop, L.: *Software product lines*. Addison-Wesley (2001)
7. Czarnecki, K., Eisenecker, U.: *Generative programming: Methods, tools, and applications*. Addison-Wesley (2000)
8. Kolovos, D.S., Paige, R.F., Polack, F.A.C.: Merging Models with the Epsilon Merging Language (EML). In: Wang, J., Whittle, J., Harel, D., Reggio, G. (eds.) MoDELS 2006. LNCS, vol. 4199, pp. 215–229. Springer, Heidelberg (2006)
9. Dyreson, C., Florez, O.: Data aspects in a relational database. In: *Proc. Int’l Conf. on Information and Knowledge Management*, pp. 1373–1376. ACM (2010)
10. Dyreson, C., Snodgrass, R., Currim, F., Currim, S., Joshi, S.: Weaving temporal and reliability aspects into a schema tapestry. *Data Knowl. Eng.* 63, 752–773 (2007)
11. Heidenreich, F., Kopcsek, J., Wende, C.: Featuremapper: Mapping features to models. In: *Comp. Proc. Int’l. Conf. on Software Engineering*, pp. 943–944. ACM (2008)
12. Jossic, A., et al.: Model integration with model weaving: a case study in system architecture. In: *Proc. Int’l. Conf. Systems Engineering and Modeling*, pp. 79–84. IEEE (2007)
13. Kästner, C., Apel, S., Kuhlemann, M.: Granularity in software product lines. In: *Proc. Int’l Conf. on Software Engineering*, pp. 311–320. ACM (2008)
14. Kästner, C.: Cide: Decomposing legacy applications into features. In: *Demonstration at Proc. Int’l. Software Product Line Conf.*, pp. 149–150 (2007)
15. Liu, J., Batory, D., Lengauer, C.: Feature-oriented refactoring of legacy applications. In: *Proc. Int’l Conf. on Software Engineering*, pp. 112–121. ACM (2006)
16. Mahnke, W.: Towards a modular, object-relational schema design. In: *Doctoral Consortium at Proc. Int’l. Advanced Information Systems Engineering*, pp. 61–71. Springer (2002)
17. Rashid, A.: A framework for customisable schema evolution in object-oriented databases. In: *Proc. Int’l. Symp. on Database Engineering and Applications*, pp. 342–346. IEEE (2003)
18. Sabetzadeh, M., Easterbrook, S.: View merging in the presence of incompleteness and inconsistency. *Requir. Eng.* 11(3), 174–193 (2006)
19. Sabetzadeh, M., Nejati, S., Liaskos, S., Easterbrook, S., Chechik, M.: Consistency checking of conceptual models via model merging. In: *Proc. Int’l Conf. on Requirements Engineering*, pp. 221–230. Springer, Heidelberg (2007)
20. Schäler, M., Leich, T., Siegmund, N., Kästner, C., Saake, G.: Generierung maßgeschneiderter Relationenschemata in Softwareproduktlinien mittels Superimposition. In: *Proc. GI-Fachtagung Datenbanksysteme für Business, Technologie und Web*, pp. 414–534. GI (2011)
21. Siegmund, N., Kästner, C., Rosenmüller, M., Heidenreich, F., Apel, S., Saake, G.: Bridging the gap between variability in client application and database schema. In: *Proc. GI-Fachtagung Datenbanksysteme für Business, Technologie und Web*, pp. 297–306. GI (2009)
22. Spaccapietra, S., Parent, C.: View integration: A step forward in solving structural conflicts. *IEEE Trans. on Knowl. and Data Eng.* 6(2), 258–274 (1994)
23. Ye, P., Peng, X., Xue, Y., Jarzabek, S.: A case study of variation mechanism in an industrial product line. In: *Proc. Int’l Conf. on Software Reuse*, pp. 126–136. Springer (2009)

# Evolutionary Search-Based Test Generation for Software Product Line Feature Models

Faezeh Ensan<sup>1</sup>, Ebrahim Bagheri<sup>2</sup>, and Dragan Gašević<sup>2</sup>

<sup>1</sup> University of British Columbia

<sup>2</sup> Athabasca University

faezeh.ensan@ubc.ca, {ebagheri, dragang}@athabascau.ca

**Abstract.** Product line-based software engineering is a paradigm that models the commonalities and variabilities of different applications of a given domain of interest within a unique framework and enhances rapid and low cost development of new applications based on reuse engineering principles. Despite the numerous advantages of software product lines, it is quite challenging to comprehensively test them. This is due to the fact that a product line can potentially represent many different applications; therefore, testing a single product line requires the test of its various applications. Theoretically, a product line with  $n$  software features can be a source for the development of  $2^n$  application. This requires the test of  $2^n$  applications if a brute-force comprehensive testing strategy is adopted. In this paper, we propose an evolutionary testing approach based on Genetic Algorithms to explore the configuration space of a software product line feature model in order to automatically generate test suites. We will show through the use of several publicly-available product line feature models that the proposed approach is able to generate test suites of  $O(n)$  size complexity as opposed to  $O(2^n)$  while at the same time form a suitable tradeoff balance between error coverage and feature coverage in its generated test suites.

**Keywords:** Software product lines, Feature models, Evolutionary testing.

## 1 Introduction

Large and complex domains are a potential venue for the development of many different software applications. These applications can share a lot of similarities due to the fact that they have been developed for the same target domain and also have differences based on the nature of the specific problem that they are trying to solve within the target domain. The concept of software product lines is amongst the widely used means for capturing and handling these inherent *commonalities* and *variabilities* of the many applications of a target domain [5,22]. Within the realm of software product lines, these similarities and differences are viewed in terms of the core competencies and functionalities, referred to as *features*, provided by each of the applications [14,11]. Therefore, a product line for a domain is a model of that domain such that it formalizes the existence of and the interaction between all of the possible features of the domain.

Software product lines are often represented through *feature models*, which are tree-like structures whose nodes are the domain features and the edges are the interaction

between the features. Each feature model is an abstract representation of the possible applications of a target domain; therefore, it is possible to derive new applications from a feature model by simply selecting a set of most desirable features from the feature model. Since a feature model is a generic representation of all possible applications of a domain, the selection of a set of features from the feature model yields a specific application. This process is referred to as *feature model configuration*. It is clear that the selection of different features from the feature model results in different feature model configurations and hence different software applications. For this reason, it is reasonable to say that a single feature model can be configured in different ways to form numerous domain-dependent applications.

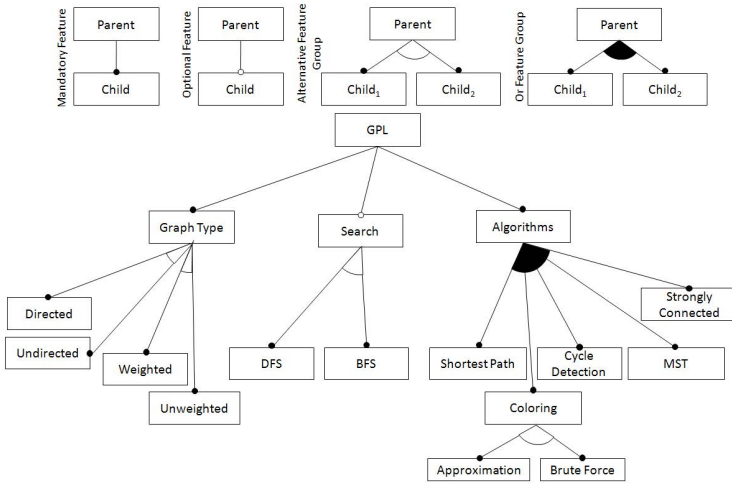
As a matter of fact, the number of possible configurations of a feature model increases *exponentially* with the size of the feature model, i.e., a feature model with more features is a potential for many more configurations [13]. This observation leads to the main concern, namely, the time and effort required for testing all of the possible applications of a software product line. Outside the context of software product lines, testing often involves the analysis of one single application that needs to be fully evaluated. However, the process of testing a software product line and ensuring that most if not all of its errors<sup>1</sup> are detected requires comprehensive analysis of all of its potential products, i.e., as opposed to testing a single application, testing a product line requires testing multiple applications and is therefore much more time consuming and costly. Assuming that a feature model contains  $n$  features and that each application configured from that feature model takes  $O(m)$  to be tested, a complete test of that feature model would in the worst case take  $O(2^n \times m)$ , which is impractical in terms of the required resources to generate all of the tests and also the time needed for performing the tests.

In many cases, an acceptable tradeoff between error coverage and the number of generated tests would need to be found such that while a high error coverage is reached, the number of tests that are generated for the product line are kept as small as possible. In this paper, we benefit from Genetic Algorithms in order to select a smaller set of applications from among all possible applications of a feature model. These selected applications will be then comprehensively tested instead of testing all of the possible applications of the product line. Throughout this paper, a *test suite* is a restricted collection of applications derived from a feature model in order to reduce the test space; hence a test suite consists of several applications from all possible applications to be tested independently. We refer to each member of the test suite as a *test*. The members of a test suite, i.e. the tests, are the ‘selected’ applications that are to be tested instead of testing all of the application space. Therefore, simply put, our goal is to select a set of applications referred to as the test suite where each individual application in the test suite is called a test. It should be noted that in reality the testing of each test would require a set of test cases. The generation of the test cases for each application can be done using conventional test case generation techniques for single software applications and is outside the scope of this paper.

To this end, we look at *search-based test generation* to explore the feasible test space and find appropriate test suites for a given software product line. More specifically, we

---

<sup>1</sup> An error is a bug, fault or alike in the implementation of a feature provided in the domain engineering phase.



**Fig. 1.** Graph product line feature model

employ Genetic Algorithms for this purpose. Genetic Algorithms allow us to start with a randomly generated set of tests for a product line and gradually improve the quality of the tests by evolving them in a controlled process. In this paper, we will discuss how a test generation process can be defined based on Genetic Algorithms for software product lines. This will include the process of designing appropriate *chromosome* representations of software product line configurations and the definition of a suitable *fitness function*. It will be shown through our experimentations that the proposed strategy is able to generate small but efficient test suites for software product lines.

## 2 Background

### 2.1 Feature Models

Feature models are one of the most important modeling means for representing aspects of a software product line. In this paper, we focus on feature models to deal with the details of a product line. In a feature model, features are hierarchically organized and are typically classified as: *Mandatory*, the feature must be included in the description of its parent feature; *Optional*, the feature may or may not be included in its parent description given the situation; *Alternative feature group*, one and only one of the features from the feature group can be included in the parent description; *Or feature group*, one or more features from the feature group can be included in the description of the parent feature.

Figure 1 depicts the graphical notation of the feature relationships. The tree structure of feature models falls short at fully representing the complete set of mutual interdependencies of features; therefore, additional constraints are often added to feature models and are referred to as *Integrity Constraints (IC)*. The two most widely used integrity constraints are: *Includes*, the presence of a given feature (set of features) requires the

*existence* of another feature (set of features); *Excludes*, the presence of a given feature (set of features) requires the *elimination* of another feature (set of features).

The Graph Product Line (GPL) depicted in Figure 1 is the typical feature model in the software product line community that covers the classical set of applications of graphs in the domain of Computer Science. As it can be seen, GPL consists of three main features: 1) Graph Type: features for defining the structural representation of a graph; 2) Search: traversal algorithms in the form of features that allow for the navigation of a graph; 3) Algorithms: other useful algorithms that manipulate or analyze a given graph. Clearly, not all possible configurations of the features of GPL produce valid graph programs. For instance, a configuration of GPL that checks if a graph is strongly connected cannot be implemented on an undirected graph structure. Such restrictions are expressed using integrity constraints. Two examples of these constraints are: Cycle Detection EXCLUDES BFS and Strongly Connected INCLUDES DFS.

These integrity constraints and the structure of the feature model ensure that correct product configurations are derived from a feature model. For instance, the feature model in Figure 1 can be configured in 308 different ways; hence, having the potential to produce 308 domain-dependent applications. As mentioned earlier, a comprehensive strategy for testing feature models is to test all of its possible configurations. In other words, each possible configuration is a *test* for evaluating the feature model. Taking this approach for testing the simple graph product line feature model shown in Figure 1 would require 308 different applications to be tested (308 tests to be evaluated).

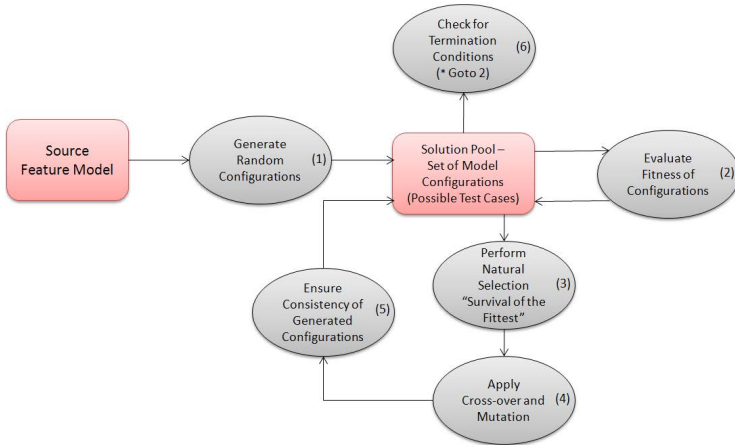
## 2.2 Metaheuristic Search

The generation of tests is often a part of the responsibilities of the test engineer in industrial settings. However, this process is too costly, labor-intensive and lengthy for large applications. For this reason, *metaheuristic search techniques* have been favored by many for the automatic generation of appropriate tests. According to McMinn [15], metaheuristic search techniques are high-level frameworks which utilize heuristics to find rather optimal solutions to combinatorial problems at an acceptable computational cost. Several metaheuristic techniques have been used in software test generation, including *Hill climbing*, which is a local search algorithm; *Simulated annealing*, which is a probabilistic extension of Hill climbing; and *Evolutionary algorithms*.

The application of evolutionary algorithms such as Genetic Algorithms to software test generation is referred to as *evolutionary testing* [15]. Evolutionary testing approaches have proven to be effective in generating test input data for specific execution paths or certain program structures [27,4,30]. Here, we will be exploiting Genetic Algorithms for generating efficient tests for software product line feature models.

Evolutionary algorithms [6] are inspired by *biological evolution*, which includes reproduction, mutation, recombination, and selection; hence, they focus on evolution as a search strategy. The main idea behind this class of metaheuristic search techniques is that given an initial set of possibly random solutions for an optimization problem, an evolutionary algorithm will be able to find a near optimal solution by gradually mutating and recombining these existing solutions into newer solutions such that a measure of fitness is improved in the process. Genetic Algorithms [10] are one of the widely used forms of evolutionary algorithms. They operate by forming a correspondence between





**Fig. 2.** The overview of our proposed approach

the genetic structure of chromosomes and the encoding of solutions in the optimization problem. With this correspondence, since solutions of the optimization problem are represented as chromosomes, natural biological evolution operators such as *mutation* and *cross-over* can be used to generate new chromosomes (i.e., new solutions).

Essentially, the process of generating new chromosomes based on the previous chromosomes is repeated until an acceptable set of answers are developed. The suitability of each chromosome (solution) for the given optimization problem is evaluated by a *fitness function*. In the Genetic Algorithm process, each round of generating new chromosomes is referred to as a *generation*. The important factor that drives Genetic Algorithms towards a near optimal solution is the role of *natural selection* in each generation, i.e., in each generation weaker chromosomes die (weaker solutions are deleted), and stronger chromosomes survive (solutions with higher fitness are kept) and are moved onto the next generation. With this *survival of the fittest* strategy, Genetic Algorithms often converge to a (near) optimal solution for optimization problems.

### 3 Test Suite Generation

#### 3.1 Approach Overview

The main objective of our work is to generate small but efficient test suites for software product line feature models. We employ Genetic Algorithms to search the possible configuration space of a feature model ( $2^n$  possible feature model configurations) in order to select a limited number of the most effective and covering configurations to serve as the tests in our test suite. The outline of our approach is depicted in Figure 2 as follows:

In the first step (1), a set of feature model configurations are randomly generated from the feature model without considering any extra constraints on the feature model other than the fact that the configurations must be valid. These configurations serve as the initial population for Genetic Algorithm. Once the initial population is formed, each

of the configurations (aka solutions) is evaluated in step (2) to see how *fit* it is. We will formally define the fitness function in the next section. Given the fitness of the configurations, step (3) discards the weaker configurations, while the rest of the configurations are preserved to be used as seeds for the mutation and cross-over operators in step (4). It is important to mention that since the new population is generated based on the previous configurations, the new configurations in the population are not necessarily always valid. Therefore, step (5) evaluates the validity of the generated configurations from step (4) and only accepts valid ones<sup>2</sup>. Step (2)-(5) are repeated until the termination condition (introduced later) is satisfied. At the end of this process, the final population of the Genetic Algorithm is considered to be the generated *test suite* and each of the configurations in the population is a *test* in the test suite that can be used as a sample application to be tested for testing the software product line.

### 3.2 Approach Realization

The representation of possible solutions in a Genetic Algorithm are called chromosomes. Each chromosome is composed of smaller building blocks called genes. In each optimization round of a Genetic Algorithm, a collection of chromosomes are formulated and evolved in order to guide the optimization process. As we will show later, once the algorithm ends, one or several of the chromosomes will represent the possible solutions. In our settings, for a feature model of size  $n$  (feature model with  $n$  features), we create binary chromosomes of size  $n$ . This means that the chromosomes of our Genetic Algorithm are in the form of an array of length  $n$ , whose elements can either be 0 or 1. Each gene of the chromosomes shows whether the corresponding feature is selected or not. For instance, assume that a feature model with four features is being tested. A possible chromosome such as  $\langle 0, 1, 1, 0 \rangle$  would represent a configuration of that feature model where the second and third features of the feature model are present in the configuration, while the first and last features are not included. Also, the size of the chromosomes is  $n$  because the largest configuration that can be developed could at most contain all of the features, and since the chromosomes represent the possible configurations, we generate chromosomes with  $n$  genes.

Each of the chromosomes represents a specific feature model configuration and is therefore considered to be a potential test. Further, the Genetic Algorithm population is viewed as the test suite. With this assumption, the size of the test suites is dependent on the population size selected for the Genetic Algorithm. Some techniques have already been proposed including our own strategy [1] for dynamically configuring the optimal population size for a Genetic Algorithm; however, in this paper, we have chosen to evaluate different population sizes because we are interested in evaluating *feature coverage* as well as *error coverage*. A dynamic approach to determining the population size would have prevented us from studying the tradeoff between feature and error coverage.

In a previous study [2], we have already explored some of the structural properties of feature models and their important characteristics. Two of the most important distinguishing aspects of feature models is their ability to 1) capture variability in design and 2) represent feature interdependencies through integrity constraints. For this reason, we

<sup>2</sup> This is computationally inexpensive using FaMa. <http://www.isa.us.es/fama/>

base the Genetic Algorithm fitness function on these two important aspects: variability and integrity constraints. To develop a concrete measurable stance for these aspects we benefit from two metrics for each feature model configuration: 1) *variability coverage* ( $\mathcal{VC}$ ) and 2) *cyclomatic complexity* ( $\mathcal{CC}$ ). In order to compute variability coverage of a product, we count the number of variation points that had to be bounded in order to derive that product from the feature model; in other words, variability coverage is the count of number of bounded variation points for a given product. Also, cyclomatic complexity is the number of distinct cycles that can be found in the feature model, which can be shown to be equivalent to the number of integrity constraints on the feature model. The cyclomatic complexity of each product is the number of integrity constraints that are enforced in the product derivation process of a given product. We define the fitness function ( $\mathfrak{F}$ ) as follows:

$$\mathfrak{F}(c) = \|(\mathcal{VC}(c), \mathcal{CC}(c))\| = \sqrt{\mathcal{VC}(c)^2 + \mathcal{CC}(c)^2}. \quad (1)$$

where  $c$  is the chromosome (derived application) for which the fitness is calculated.

Following the *survival of the fittest* strategy, fitter chromosomes survive while the weaker ones are discarded to make room for new chromosomes. We employ the *roulette-wheel selection* method to perform the natural selection process. This method does not necessarily remove all of the weakest chromosomes but associates each chromosome with a removal probability which is inversely proportional to its fitness value. This way, weaker chromosomes also have a chance although slim for survival.

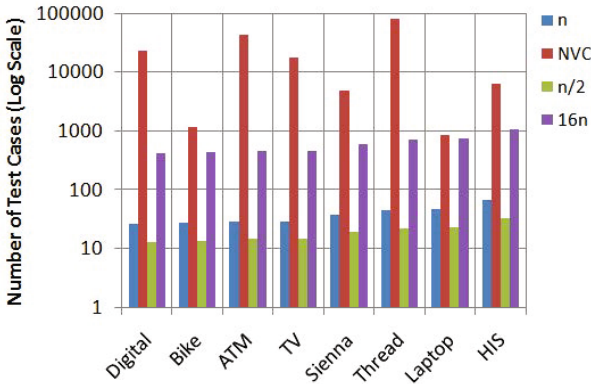
The new chromosomes are generated based on the mutation and cross-over operators. Given the structure of the chromosomes, mutation causes a feature that is not present in the previous chromosome to be included in the configuration or vice-versa (a random gene be flipped in the chromosome); therefore, mutation is quite useful in creating new configurations that are only slightly different from previous configurations (chromosomes), but have the possibility of covering more errors. However, unlike mutation, cross-over is a more radical operator for our case since it rotates the values of the genes around a central pivot gene in two chromosomes. The resulting chromosomes are radically different from the original chromosomes since the values of many genes could possibly change in this process. This operator is useful for avoiding the Genetic Algorithm from getting trapped in a local extremum that could lead to inefficient tests.

It is important to mention that the new chromosomes that are developed based on mutation and cross-over are not necessarily valid feature model configurations. Since we are interested in making sure that the generated tests are valid feature model configurations, the generated chromosomes that are invalid configurations are discarded and are not added as a part of the new generation.

In terms of the termination conditions for the Genetic Algorithm, two main stoppage criteria were defined: First, a maximum number of generations is defined that ensures that the algorithm will always terminate after a certain amount of time. Second, the algorithm would terminate if the difference between the average fitness values of two generations does not exceed a predefined threshold. While the former condition guarantees the termination of the algorithm, the latter would ensure that the algorithm will terminate once it is not providing any further optimizations. Once the algorithm terminates, the available chromosomes in the population form the generated tests for the

**Table 1.** The feature models used in the experiments and their characteristics

Feature Model	NF	CTCR	NVC
Digital Video System	26	23%	22,680
Bicycle	27	14%	1,152
ATM Software	29	0	43,008
TV-Series Shirt	29	27%	21,984
Sienna	38	26%	2,520
Thread	44	0	80,658
Dell Laptop	46	80%	853
HIS	67	11%	6,400

**Fig. 3.** The number of generated tests in each test suite

given product line. In the following section, we evaluate the quality of the generated tests for several SPLOT feature models in terms of both feature and error coverage.

## 4 Evaluation

### 4.1 Objects of Analysis

For the purpose of our experiments, we have selected a set of feature models that are publicly available through the Software Product Line Online Tools (SPLOT) website [18]. SPLOT's primary goal is to facilitate the transition process from research to practice and therefore provides the means for both researchers and practitioners to share and contribute their software product line feature models in an open and free manner. The feature models in this online repository are expressed in SXFM format.

The feature models that were used in our experiments along with three of their important metrics are shown in Table 1. The NF, CTCR, and NVC metrics denote the number of features in the feature model, the ratio of the number of distinct features in the integrity constraints to the number of feature model features, and the number of valid configurations of the feature model, respectively. Feature models with a CTCR

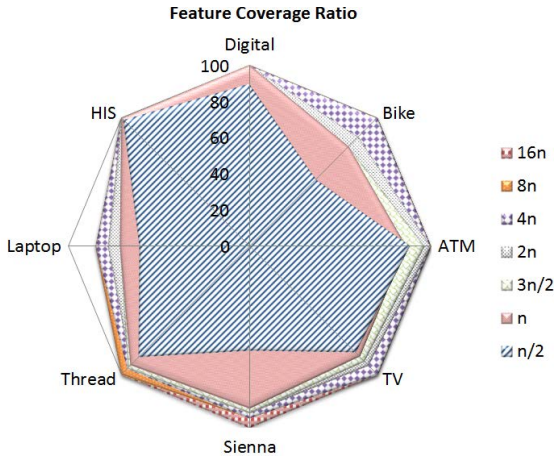


Fig. 4. Feature coverage based on test suite size

value of zero are those that do not consist of any additional integrity constraints. For the purpose of calculating the NVC of each feature model, the binary decision diagram technique proposed by Mendoca et al. [17] was employed. We also benefited from the FaMa toolkit for validating the feature model configurations. FaMa is a useful framework for the automated analysis of feature models. It consists of some of the most commonly used logical representations and configuration techniques for feature models.

## 4.2 Experiment Design

In order to evaluate the effectiveness of the proposed approach, we have performed our experiments on the feature models introduced in Table 1 using an automated software program. The program is able to parse feature models defined in SXFM and convert them to the required Binary chromosome format to be used with the Genetic Algorithm implementation provided in JGAP [16]. The program will create the representation of the software product line feature model configurations as chromosomes. This program also interacts with the FaMa toolkit to validate each of the chromosomes that are developed in the Genetic Algorithm. This interaction will ensure that all of the chromosomes that are available in the Genetic Algorithm population are valid feature model configurations and are hence appropriate tests for the product line.

Now in practice, once a feature model is configured and a specific application is derived, the selected features will be replaced by suitable software components, Web Services or software programs that are able to fulfil the requirements of that feature. The replacement of features with appropriate implementation details is often done manually by engineers using proprietary software; therefore, information on them is not publicly available. In view of this issue and to be able to evaluate the efficiency of our coverage criteria and their corresponding test suite generators, we developed an *error generation simulator*. The lack of publicly available datasets has already been pointed out in [82]

and several authors and tool suites (such as FaMa [3] and 3-CNF model generator [18]) have already been using model generation techniques to test their work.

The simulator considers the three introduced metrics shown in Table 1 for each feature model, i.e., NF, CTCR, and NVC, and generates errors for the feature model correspondingly. Feature models with more features, higher ratio of CTCR and a higher number of valid configurations will contain more errors generated by the simulator. The generated errors are in one of the following categories:

1. *Errors in individual features*: The simulator will select a number of features from the feature model proportional to NF and NVC. These selected individual features will be considered to contain errors and will need to be detected by the generated test suites;
2. *Errors in repulsive features*: These errors will be generated in the form of  $n$ -tuples, where each tuple contains a set of 2 or more features. The idea behind these errors is that a configuration will contain error due to the interactions between the features if the  $n$  features in the  $n$ -tuple are all in that configuration. These errors are proportional to NF and CTCR, i.e., more errors in repulsive features would be generated for a feature model with higher values for NF and CTCR.
3. *Errors in attracting features*: For this type of error, a pair of features are selected and an error occurs if one of the features in the pair is present in the configuration and the other is not. Similarly, the number of errors in attracting features is dependent on NF and CTCR.

Further details and code for the error generator is available for download at <http://ebagheri.athabascau.ca/splt/splt.zip>. Interested researchers are encouraged to use it for replication studies. The error generation simulator will create a set of errors for each feature model in the form of the above three error categories, which are then used to test how well the test suite generation technique is able to detect the generated errors for each of the feature models. The results of the evaluation based on the generated errors are reported in the following.

### 4.3 Results and Analysis

Our experiments were performed on a dedicated 2.8GHz Pentium IV Personal Computer with 2GB of memory, running a Windows XP operating system along with Java 6.0. In order to be able to study the tradeoff between feature and error coverage of our approach, we have performed multiple experiments in each of which we have selected a different population size for the Genetic Algorithm. As discussed earlier, since the population size of the Genetic Algorithm represents the size of the generated test suite, we are interested to see if and to what extent the size of the test suite impacts error coverage. For each of the feature models shown in Table 1, we have generated test suites based on the Genetic Algorithm with population sizes of ( $n$  being the number of features in the feature model):  $\frac{n}{2}$ ,  $n$ ,  $\frac{3n}{2}$ ,  $2n$ ,  $4n$ ,  $8n$ , and  $16n$ . This means that, e.g., for the Thread feature model, we generate test suites of sizes: 22, 44, 66, 88, 176, 352, and 704. As seen, the largest generated test suite only contains 704 tests, whereas if the total number of possible tests were to be tested, 80,658 configurations would have been

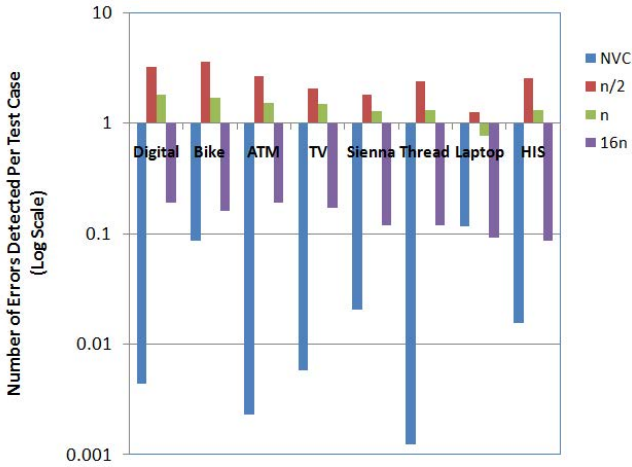


Fig. 5. Error detected per test

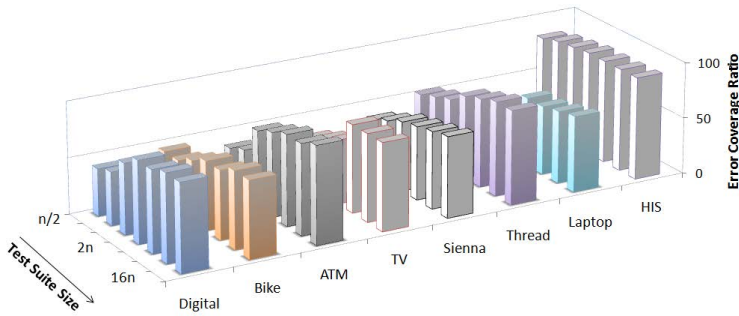


Fig. 6. Error coverage by different test suite sizes

needed to be considered. Figure 3 shows a comparison of the number of tests in each test suite generated by our approach with the possible set of tests for the feature model (NVC) in log scale. It can be seen that even for the test suites with  $16n$  tests, the size is smaller than the test suites generated by NVC.

Given the reduction in the size of the test suite, the important questions that need to be studied are: Q1) does the reduced test suite size provide sufficient feature coverage, i.e., do the test suites provide tests that test each feature of the feature model at least once? Q2) how would a smaller test suite impact error coverage, i.e., do smaller test suites lack the required precision to identify the possible errors?

With regards to Q1, it is clear that test suites based on NVC have a complete feature coverage. In other words, it is guaranteed that they will test all of the features of the feature model. However, in our approach since the size of the test suites are much smaller, full feature coverage is not always ensured. As the size of the test suite grows larger, higher feature coverage is achieved, which is shown in Figure 4. Despite the

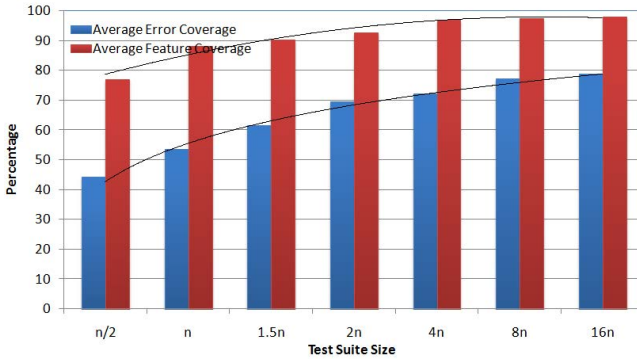
fact that test suite size impacts feature coverage, an important observation can be made based on Figure 4, which is that the increase of test suite size after a certain amount does not significantly impact feature coverage. As seen in the figure, for the feature models in our experiments, a test suite size of  $4n$  has been able to provide good feature coverage and larger test suite sizes have not been able to significantly improve this feature coverage compared to  $4n$ . So, as a matter of tradeoff between test suite size and feature coverage, one could argue that the advantages of a four times smaller test suite ( $4n$  vs  $16n$ ) in terms of time and cost of performing tests outweighs a 4 – 5% increase in feature coverage. Yet, the decision about this tradeoff is dependent on the domain being tested and the test engineers preferences.

Now regarding the impact of test suite size on error coverage (Q2), it is important to first study how efficient each of the tests of a test suite are. For this purpose, we compute the *average number of errors detected per test* in a generated test suite. The results are shown in Figure 5 in log scale. It is interesting to observe that as the size of the test suites increases, the effectiveness of the tests in identifying errors decreases significantly. This indicates that as the size of the test suite increases and more tests are considered, only marginal improvements in terms of error coverage is achieved. To put this observation in context, we would need to examine the trend of error coverage improvement given different test suite sizes. Figure 6 depicts this trend for the different objects of analysis and for different test suite sizes. It can be seen in this figure that the increase in error coverage occurs up until test suites with a size of  $2n - 4n$ , and further increase in test suite size only has a minor impact on the increase of error coverage. It seems that increasing the size of test suites is not necessarily a better strategy for testing software product lines, since although larger test suites increase the cost associated with testing a product line, they only provide marginal benefits wrt feature and error coverage.

Figure 7 shows the average feature and error coverage over all objects of study for different test suite sizes. Besides showing the fact that increasing the size of the test suite beyond a certain point is not necessarily a desirable choice, this figure also exposes an implicit relationship between feature coverage and error coverage, i.e., a test suite with a higher feature coverage is more likely to have a higher error coverage. This is further shown in Figure 8 where each of the points in the figure is a representative of a test suite generated for one of the objects of analysis with a defined test suite size. Although this correlation between feature and error coverage is expected, the degree of impact of feature coverage on error coverage is attractive in that it shows that even test suites with complete feature coverage (100%) do not necessarily identify all of the errors in a feature model. This can be explained as follows: According to Section 4.2, errors can be in three different classes. A complete feature coverage will only ensure that *Errors in individual features* are covered. The other two classes are not guaranteed to be fully covered unless test suites based on NVC are comprehensively generated and tested, which is both cost and computational wise infeasible. Therefore, reaching a tradeoff between the size of the test suite and the error and feature coverage as achieved in Figures 6 and 7 seems to be a viable strategy.

We believe that the our approach for generating test suites has been successful in terms of its ability to cover both existing errors and features given small test suite sizes. As an example, our approach has been able to reach ~80% error coverage (Figure 6) and





**Fig. 7.** Impact of test suite size on error and feature coverage

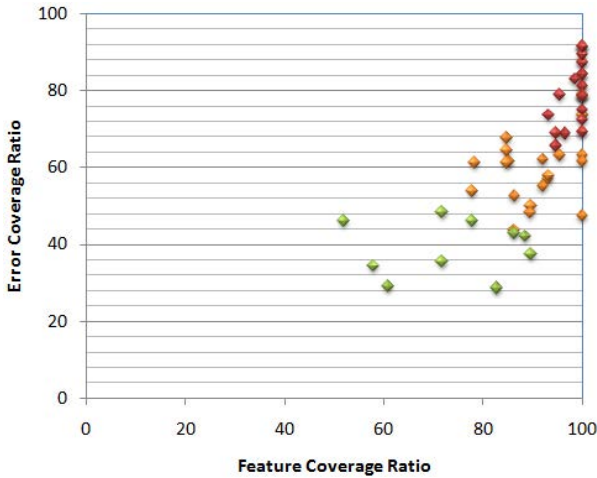
~90% feature coverage (Figure 4) with only a test suite of size  $4n = 704$  tests for the Thread feature model. This can be considered a rather efficient tradeoff if we consider the fact that a 100% feature and error coverage would only be achieved if a test suite of size 80, 658 tests is evaluated, which requires much higher testing costs and resources.

## 5 Related Work

As software product lines become more popular for facilitating reuse and enhancing efficiency and production time [22], issues related to quality engineering become of more importance. Some authors already believe that software product line development processes still lack appropriate testing techniques that can address commonality and variability inherent in the product lines [23][19]. Although generic software testing approaches can be used on products of the product line, an entire product line is much harder to be tested using these techniques due to the numerosity of possible configurations of the models and the complex feature interactions that may exist [24]. Many of the techniques for software product line testing have focused on ensuring quality within the process of software product line development and planning [9][20][7]. However, the focus of our work is on the development of appropriate test suite generation strategies for product line feature models.

There are only a limited number of approaches that directly address the issue of test generation in software product lines. Our approach is significantly different from the related work in that we employ an evolutionary approach for searching the test space, which has not been explored in the area of software product lines. Our approach is mainly guided by a fitness function that centers on variability and cyclomatic complexity of the feature model. The main benefits of our approach compared to other approaches can be seen as being very computationally efficient as it does not perform complex optimization, provides a tradeoff between feature and error coverage and develops tests that focus on specifically evaluating variability and integrity constraints.

In contrast, some related techniques employ automatic analysis based on SAT solvers [12] such as Alloy. For instance, Uzuncaova et al. propose a hybrid approach by



**Fig. 8.** Correlation between error and feature coverage (hotter colors show more desirable tests)

combining methods from software product lines and specification-based testing using Alloy [29]. In their approach, each product is defined as a composition of features represented as Alloy formulae. The Alloy analyzer is then used to incrementally generate tests over partial specifications of the products. This approach is an improvement over previous work that generated tests in a single pass execution of Alloy over complete specifications [28].

In a different work, Perrouin et al. employ *T-wise test generation* [21]. Their approach attempts to address the large combinatorial number of required tests for a software product line by only generating test sets that cover all possible  $T$  feature interactions. For this, they devise strategies to disintegrate  $T$ -wise combinations into smaller manipulable subsets, which are then used in a developed toolset over Alloy for generating the tests.

The authors of [13] base their work on the assumption that although the number of product line configurations are exponential in the number of available features but the features are often *behavior-irrelevant*, i.e., they augment but do not change the behavior of a system. According to this assumption, many of the tests become overlapping and the smaller test sets will be redundant; therefore, the authors are able to design a static program analysis method to find the behavior-irrelevant features and hence reduce the size of the test space. Other work which reduce the test space mainly focus on the use of user requirements to identify the most important set of features that need to be tested [25]. In such approaches, it is the end users' needs that drive the test generation process and not the feature interactions.

Less relevant to our line of work in this paper is the proposal by Segura et al [26]. In their paper, the authors develop a set of *metamorphic* relations between the input feature models and their product and use this as a way to generate appropriate tests. However, the tests generated in this approach are formed in such a way to test automated feature model configuration programs and not software product line feature models.

## 6 Concluding Remarks

In this paper, we have proposed a search-based testing approach based on the evolutionary Genetic Algorithms to automatically generate test suites for software product lines. The main issue with testing product lines is the exponential number of configurations that need to be tested. We have exploited the exploration capabilities of Genetic Algorithms to search the *configuration space* of a feature model and to find the subset that provides suitable error and feature coverage. We have performed experiments using some models provided by the SPLOT repository and have shown that our proposed test suite generation approach is able to develop test suites that provide a reasonable tradeoff balance between error and feature coverage. This is achieved by only generating test suites with a size complexity of  $O(n)$  as opposed to  $O(2^n)$ . We are currently evaluating our test suite generation approach on larger synthetic feature models that are generated by the SPLOT and FaMa toolkits. The results of these experiments can further show the performance of the generated test suites with regards to error and feature coverage. Also we are interested in comparing our test generation technique with some other similar approaches in the literature. Currently, an implementation of the proposed work by other researchers is not publicly available. We are releasing the implementation of our work at <http://ebagheri.athabascau.ca/splt/gasplt.zip> in hopes of future comparative studies.

## References

1. Bagheri, E., Deldari, H.: Dejong function optimization by means of a parallel approach to fuzzified genetic algorithm. In: IEEE Symposium on Computers and Communications, pp. 675–680 (2006)
2. Bagheri, E., Gasevic, D.: Assessing the maintainability of software product line feature models using structural metrics. *Software Quality Journal* 19(3), 579–612 (2011)
3. Benavides, D., Segura, S., Trinidad, P., Ruiz-Cortes, A.: FAMA: Tooling a framework for the automated analysis of feature models. In: Proceeding of the First International Workshop on Variability Modelling of Software-intensive Systems (VAMOS), pp. 129–134 (2007)
4. Buehler, O., Wegener, J.: Evolutionary functional testing of an automated parking system. In: International Conference on Computer, Communication and Control Technologies (CCCT 2003). Citeseer (2003)
5. Clements, P., Northrop, L.M.: *Software product lines* (2003), [http://www.sei.cmu.edu/programs/pls/sw-product-lines\\_05\\_03.pdf](http://www.sei.cmu.edu/programs/pls/sw-product-lines_05_03.pdf) (visited June 2009)
6. Coello, C., Lamont, G., Van Veldhuizen, D.: *Evolutionary algorithms for solving multi-objective problems*. Springer-Verlag New York Inc. (2007)
7. Cohen, M., Dwyer, M., Shi, J.: Coverage and adequacy in software product line testing. In: Proceedings of the ISSTA 2006 Workshop on Role of Software Architecture for Testing and Analysis, p. 63. ACM (2006)
8. Cohen, M.B., Dwyer, M.B., Shi, J.: Coverage and adequacy in software product line testing. In: Proceedings of the ISSTA 2006 Workshop on Role of Software Architecture for Testing and Analysis, ROSATEA 2006, pp. 53–63. ACM, New York (2006)
9. Dallal, J., Sorenson, P.: Testing software assets of framework-based product families during application engineering stage. *Journal of Software* 3(5), 11 (2008)

10. Goldberg, D.: Genetic Algorithms in Search and Optimization (1989)
11. Kang, K., Lee, J., Donohoe, P.: Feature-oriented product line engineering. *IEEE Software* 19(4), 58–65 (2002)
12. Khurshid, S., Marinov, D.: TestEra: Specification-based testing of Java programs using SAT. *Automated Software Engineering* 11(4), 403–434 (2004)
13. Kim, C., Batory, D., Khurshid, S.: Reducing Combinatorics in Testing Product Lines (Technical Report), UTexas (2010)
14. Lee, K., Kang, K., Lee, J.: Concepts and Guidelines of Feature Modeling for Product Line Software Engineering. In: Gacek, C. (ed.) *ICSR 2002. LNCS*, vol. 2319, pp. 62–77. Springer, Heidelberg (2002)
15. McMinn, P.: Search-based software test data generation: A survey. *Software Testing, Verification and Reliability* 14(2), 105–156 (2004)
16. Meffert, K., Rotstan, N., Knowles, C., Sangiorgi, U.: JGAP—Java Genetic Algorithms and Genetic Programming Package, <http://jgap.sf.net>
17. Mendonca, M., Wasowski, A., Czarnecki, K., Cowan, D.: Efficient compilation techniques for large scale feature models. In: *Proceedings of the 7th International Conference on Generative Programming and Component Engineering*, pp. 13–22. ACM, New York (2008)
18. Mendonca, M., Branco, M., Cowan, D.: S.p.l.o.t.: software product lines online tools. In: *OOPSLA 2009: Proceeding of the 24th ACM SIGPLAN Conference Companion on Object Oriented Programming Systems Languages and Applications*, pp. 761–762 (2009)
19. Nebut, C., Fleurey, F., Le Traon, Y., Jézéquel, J.-M.: A Requirement-Based Approach to Test Product Families. In: van der Linden, F.J. (ed.) *PFE 2003. LNCS*, vol. 3014, pp. 198–210. Springer, Heidelberg (2004)
20. Olimpiew, E., Gomaa, H.: Model-based testing for applications derived from software product lines. In: *Proceedings of the 1st International Workshop on Advances in Model-based Testing*, p. 7. ACM (2005)
21. Perrouin, G., Sen, S., Klein, J., Baudry, B., Le Traon, Y.: Automated and Scalable T-wise Test Case Generation Strategies for Software Product Lines. In: *ICST 2010*, pp. 459–468 (2010)
22. Pohl, K., Böckle, G., Van Der Linden, F.: *Software Product Line Engineering: Foundations, Principles, and Techniques*. Springer (2005)
23. Pohl, K., Metzger, A.: Software product line testing. *Communications of the ACM* 49(12), 81 (2006)
24. Reis, S., Metzger, A., Pohl, K.: Integration Testing in Software Product Line Engineering: A Model-Based Technique. In: Dwyer, M.B., Lopes, A. (eds.) *FASE 2007. LNCS*, vol. 4422, pp. 321–335. Springer, Heidelberg (2007)
25. Scheidemann, K.: Optimizing the selection of representative configurations in verification of evolving product lines of distributed embedded systems, *SPLC 2006* (2006)
26. Segura, S., Hierons, R., Benavides, D., Ruiz-Cortés, A.: Automated test data generation on the analyses of feature models: A metamorphic testing approach. In: *ICST 2010*, pp. 35–44 (2010)
27. Tracey, N., Clark, J., Mander, K.: Automated program flaw finding using simulated annealing. In: *Proceedings of the 1998 ACM SIGSOFT International Symposium on Software Testing and Analysis*, p. 81. ACM (1998)
28. Uzuncaova, E., Garcia, D., Khurshid, S., Batory, D.S.: A specification-based approach to testing software product lines. In: *ESEC/SIGSOFT FSE*, pp. 525–528 (2007)
29. Uzuncaova, E., Khurshid, S., Batory, D.S.: Incremental test generation for software product lines. *IEEE Trans. Software Eng.* 36(3), 309–322 (2010)
30. Watkins, A.: The automatic generation of test data using genetic algorithms. In: *Proceedings of the 4th Software Quality Conference*, vol. 2, pp. 300–309 (1995)

# Feature Model Differences

Mathieu Acher<sup>1</sup>, Patrick Heymans<sup>1,2</sup>, Philippe Collet<sup>3</sup>, Clément Quinton<sup>2</sup>,  
Philippe Lahire<sup>3</sup>, and Philippe Merle<sup>2</sup>

<sup>1</sup> PreCISE Research Centre, University of Namur, Belgium  
`{mac,phe}@info.fundp.ac.be`

<sup>2</sup> INRIA Lille-Nord Europe, LIFL CNRS UMR 8022, University of Lille 1, France  
`{clement.quinton,philippe.merle}@inria.fr`

<sup>3</sup> I3S – CNRS UMR 6070 University of Nice Sophia Antipolis, France  
`{collet,lahire}@i3s.unice.fr`

**Abstract.** Feature models are a widespread means to represent commonality and variability in software product lines. As is the case for other kinds of models, computing and managing feature model differences is useful in various real-world situations. In this paper, we propose a set of novel differencing techniques that combine syntactic and semantic mechanisms, and automatically produce meaningful differences. Practitioners can exploit our results in various ways: to understand, manipulate, visualize and reason about differences. They can also combine them with existing feature model composition and decomposition operators. The proposed automations rely on satisfiability algorithms. They come with a dedicated language and a comprehensive environment. We illustrate and evaluate the practical usage of our techniques through a case study dealing with a configurable component framework.

**Keywords:** Feature Models, Model Differences, Model Management.

## 1 Introduction

*Software product line* (SPL) engineering aims at generating software variants tailored to the needs of particular customers or market segments [24]. SPL principles, formalisms and techniques are gaining more and more attention in different application domains to efficiently produce and maintain multiple similar software products. Central to SPL engineering is the modeling and management of variability: exploiting what variants have in common and managing what varies among them. In this context, *feature models* (FMs) are widely used to model the variability of a system in terms of mandatory, optional and exclusive features as well as propositional constraints over the features [25,28]. The primary purpose and semantics of FMs is to characterize the combinations of features (called configurations) supported by a system. A number of formalizations (e.g., [25,10]), automated reasoning operations [7] and tools (e.g., [20,4,27]) have been developed to address this issue. The formalism of FMs is now at the core of many generative or model-based approaches [9,24,23,11].

When managing the variability of an SPL, reasoning about the *differences* of two FMs is a prime concern: for instance, when an SPL, and therefore its FM, evolve over time. Even small edits to an FM, like moving a feature from one branch to another, can unintentionally change the set of valid feature combinations. Understanding the impact of the evolution of an FM, and thus the differences between two versions, is known to be impractical to determine manually [28]. Consequently tool support is required to assist practitioners in computing and understanding FM differences. In practice, the need to support FM differences has been observed in different domains and for different purposes.

*Evolution of an FM.* In [11,23], the authors report that evolution support becomes particularly important for engineering SPLs and other variability-intensive systems. They propose model-driven support at the feature level, using FM concepts [23]. Lotufo *et al.* study the evolution of the Linux kernel variability model [16]. They identify edit operations applied in practice and new automation challenges, including the detection of edits that break existing configurations. Differencing techniques are thus needed, for example, to identify what are the added and removed configurations. In [2], we developed an automated procedure to extract the FM of FraSCAti, a large component and plugin-based system. The handling of FM differences was needed to compare the automatically extracted FM with one that was elaborated manually by the main developer of FraSCAti. It is also needed to understand and validate the evolution of the FMs for different versions of FraSCAti.

*Management of a product line (PL) offering.* Two kinds of variability are usually distinguished in SPL engineering [24,21]: software variability, hidden from customers, as opposed to PL variability, visible to them. An important property of an SPL is *realizability*, that is, whether the set of products that the PL management decides to offer is fully covered by the set of products that the software platform allows to build. Symmetrically, the *usefulness* property is interesting for a product manager to identify unused flexibility of the software platform. In this case, product managers and software engineers need to precisely understand what are the products supported by the platform but not offered to customers (it can be on purpose and justified by future market extensions). The differencing information is then exploited by product managers and software engineers to validate or evolve the FMs documenting the two kinds of variability. In the development of a video surveillance SPL and medical imaging workflows, we observed similar differentiation needs [6].

Until now, the problem of FM differences has neither been recognized nor comprehensively addressed by existing approaches. Model-based approaches mostly rely on syntactic mechanisms, basing their heuristics on the names and structure of model elements [22]. While showing some success, there are serious limitations. Models that are syntactically very similar may actually have very different semantics (intended meaning), and vice versa, models that describe the same system may have very different syntactic representations [19,17,13]. This observation also applies to FMs [27]. As a result, a list of syntactic differences, although accurate and useful, may not be able to reveal the real and

meaningful implications these differences have on the models involved. Some voices are calling for more *semantic differencing* [19,13] and such techniques have recently emerged for specific modeling formalisms [17,18]. In the feature modeling community, Benavides *et al.* do not report any differencing support in their survey, despite the impressive research effort on FM automations [7]. Only a few recent works have specifically considered the problem of FM differences at the syntactic [26] or at the semantic level [28,13] but have limitations to compute and present exploitable differences. In previous work, we developed a set of semantic techniques for FMs [3,2,4,6] but not yet for differences.

In this paper we first present and illustrate the problem of FM differences (Section 2). We develop a set of syntactic and semantic techniques to compute, reason about and present differences (Section 3). Thanks to these techniques, a practitioner can understand differences in a fine-grained way, visualize and manage a model of differences, augment an existing FM with the differences or compute the differences only on some parts of the two FMs. The techniques are automated using satisfiability algorithms and come with a dedicated language (Section 3.4). We also report on a practical usage and evaluation of the differencing techniques through a case study (Section 4). We discuss related work (Section 5) and conclude the paper (Section 6).

## 2 The Problem of Feature Model Differences

### 2.1 Background

FMs hierarchically structure application features into multiple levels of increasing detail. When decomposing a feature into subfeatures, the subfeatures may be optional or mandatory or may form *Xor*- or *Or*-groups (see Figure 1(a) for a visual representation of an FM). The terms FM and *feature diagram* are employed in the literature, usually to denote the same concept. In this paper, we consider that a feature diagram (see Definition 1) includes a feature hierarchy (tree), a set of feature groups, as well as human readable constraints (implies, excludes). The formalism of FMs considered is among the most popular in use.

**Definition 1 (Feature Diagram).** *A feature diagram  $FD = \langle G, E_{MAND}, G_{XOR}, G_{OR}, I, EX \rangle$  is defined as follows:*

- $G = (\mathcal{F}, E, r)$  is a rooted, labeled tree where  $\mathcal{F}$  is a finite set of features,  $E \subseteq \mathcal{F} \times \mathcal{F}$  is a finite set of edges and  $r \in \mathcal{F}$  is the root feature ;
- $E_{MAND} \subseteq E$  is a set of edges that define mandatory features with their parents ;
- $G_{XOR} \subseteq \mathcal{P}(\mathcal{F}) \times \mathcal{F}$  and  $G_{OR} \subseteq \mathcal{P}(\mathcal{F}) \times \mathcal{F}$  define feature groups and are sets of pairs of child features together with their common parent feature ;
- a set of implies constraints  $I$  whose form is  $A \Rightarrow B$ , a set of excludes constraints  $EX$  whose form is  $A \Rightarrow \neg B$  ( $A \in \mathcal{F}$  and  $B \in \mathcal{F}$ ).

Features that are neither mandatory features nor involved in a feature group are optional features. A parent feature can have several feature groups but a feature must belong to only one feature group. It should be noted that a feature diagram,

as defined above, is not expressively complete with respect to propositional logics. Similar to [27], we thus consider that an FM is composed of a feature diagram *plus* a propositional formula  $\psi_{cst}$  (see Definition 2).

**Definition 2 (Feature Model).** *An FM is a tuple  $\langle FD, \psi_{cst} \rangle$  where  $FD$  is a feature diagram and  $\psi_{cst}$  is a propositional formula over the set of features  $\mathcal{F}$ .*

Not all combinations of features (*configurations*) are authorized by an FM. A *valid* (or *legal*) configuration is obtained by selecting features in a manner that respects the following rules: *i*) If a feature is selected, its parent must also be selected; *ii*) If a parent is selected, the following features must also be selected - all the mandatory subfeatures, exactly one subfeature in each of its Alternative groups, and at least one of its subfeatures in each of its Or groups; *iii*) propositional constraints must hold. An FM thus defines a set of valid feature configurations (see Definition 3).

**Definition 3 (Configuration Semantics).** *A configuration of an FM  $fm_1$  is defined as a set of selected features.  $\llbracket fm_1 \rrbracket$  denotes the set of valid configurations of  $fm_1$  and is a set of sets of features.*

An FM is usually encoded as a propositional formula, denoted  $\phi$ , and defined over a set of Boolean variables, where each variable corresponds to a feature [10]. The translation to propositional logic is well known and off-the-shelf SAT solvers or *binary decisions diagrams* (BDDs) can be used to automatically reason about properties of an FM and its configurations [7,25,10].

## 2.2 Diff: A Running Example

We now illustrate with a simple example, extracted from [13], the problem of FM differences. Let us consider *applet<sub>1</sub>* (see Figure 1(a)) and *applet<sub>2</sub>* (see Figure 1(b)). Obviously, the two FMs differ, for example, feature *init* is a child feature of *mustOverride* in *applet<sub>1</sub>* whereas feature *init* is a child feature of the root feature *applet* in *applet<sub>2</sub>*. This difference may occur in many scenarios already described in the introduction: the FM *applet<sub>1</sub>* has evolved over time, leading to *applet<sub>2</sub>*; the two FMs have been reverse engineered from two existing frameworks ; *applet<sub>1</sub>* is a requirements model specified by a customer whereas *applet<sub>2</sub>* is the actual implementation model supported by an application programming interface, etc. A few questions arise: How do these two FMs, *applet<sub>1</sub>* and *applet<sub>2</sub>*, differ? Are they equivalent? If not, what is the actual difference? Several applications of FM differences are conceivable. First, the difference may serve as a debugging information. For example, the four configurations satisfying *applet<sub>1</sub>* but not *applet<sub>2</sub>*  $\{\{applet, destroy, init, mustOverride, stop\}, \{applet, init, mustOverride, stop\}, \{applet, init, mustOverride\}, \{applet, destroy, init, mustOverride\}\}$  can be considered as a specification error. Second, examples of configurations allowed by *applet<sub>1</sub>* but not in *applet<sub>2</sub>* might be used to expand configurations of *applet<sub>2</sub>*. Based on the analysis of differences, a practitioner may correct the error previously identified and relax



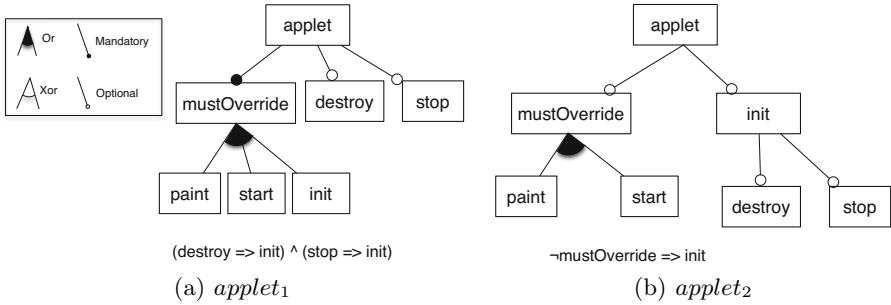


Fig. 1. Two example FMs extracted from [13]

some variability constraints and edit *applet<sub>2</sub>* (e.g., features *paint* and *start* become optional features and are no longer forming an Or-group). The example shows that a crucial issue for a practitioner, being modeler, product line manager, software engineer or architect, is to precisely understand and reason about differences of two FMs. Based on this information, a practitioner can validate the differences and/or evolve the FMs.

### 3 Set of Operators for Differencing of Feature Models

Let  $fm_1 = \langle FD_1, \psi_{cst_1} \rangle$  with  $FD_1 = \langle G_1 = (\mathcal{F}_1, E_1, r_1), E_{MAND_1}, G_{XOR_1}, G_{OR_1}, I_1, EX_1 \rangle$  and  $fm_2 = \langle FD_2, \psi_{cst_2} \rangle$  with  $FD_2 = \langle G_2 = (\mathcal{F}_2, E_2, r_2), E_{MAND_2}, G_{XOR_2}, G_{OR_2}, I_2, EX_2 \rangle$  be two FMs. In the remainder of this section, we consider the difference (*diff* for short) between  $fm_1$  and  $fm_2$  – in this particular order. We will use the example of Figure 1 to illustrate our contributions (i.e.,  $fm_1 = applet_1$  and  $fm_2 = applet_2$ ).

**Rationale and Overview.** Several techniques for the diff of FMs can be considered. Roughly, the diff between  $fm_1$  and  $fm_2$  is the set of elements in  $fm_1$  but not in  $fm_2$ . From a syntactical perspective, the elements to be considered in the diff may be features, feature hierarchies, feature groups or implies / excludes. We present *syntactic differencing* techniques in Section 3.1. Though the syntactic diff might be useful, we believe that a semantic diff for FMs should also be developed and possibly be combined with syntactic differencing. Many researchers share this vision (e.g., see [28,19,13]) and semantic differences have already been developed specifically for other kinds of models (class diagrams [17] and activity diagrams [18]). We present *semantic differencing* techniques in Section 3.2. Rather than directly computing and reasoning about differences, it can be useful to focus on the common parts of two FMs. Another developed technique is to provide the means to focus on a specific part, typically smaller, of the two FMs in order to facilitate the understanding of their differences. We describe a set of syntactic and semantic *composition* and *decomposition* techniques in Section 3.3 that comes in complement to the differencing techniques.

### 3.1 Syntactic Diff

A general approach to model differencing is to concentrate on matching between model elements using different heuristics related to their names and structure and on finding and presenting differences at a concrete or abstract syntactic level. Matching algorithms are out of the scope of this paper, surveys of different approaches can be found in [12,15]. We assume that features are identified by an unique label (i.e., name) in an FM and that two features of two FMs match if and only if they have the same name [1].

In terms of feature modeling, elements of interest are features, variability information (mandatory features, feature groups, and propositional constraints) and feature hierarchy (see Definition 1 and 2). We thus consider the diff of these model elements.

*Diff of features.*  $\mathcal{F}_{diff}$  is the set of features that are in  $fm_1$  but not in  $fm_2$ , i.e.,  $\mathcal{F}_{diff} = \mathcal{F}_1 \setminus \mathcal{F}_2$ . In the example of Figure 1,  $\mathcal{F}_{diff} = \emptyset$ .

*Diff of feature hierarchies.* Several techniques can be considered (e.g., tree edit distance [8]), including the computation of  $E_{diff}$  the set of edges modeling parent-child relationships in  $fm_1$  but not in  $fm_2$ . Formally:  $E_{diff} = E_1 \setminus E_2$ . In the example of Figure 1,  $E_{diff} = \{\{mustOverride, init\}, \{applet, destroy\}, \{applet, stop\}\}$ .

*Diff of mandatory features.* A syntactic diff of mandatory features produces  $E_{MAND_{diff}} = E_{MAND_1} \setminus E_{MAND_2}$ . In the example of Figure 1,  $E_{MAND_{diff}} = \{\{applet, mustOverride\}\}$ , showing that the feature `mustOverride` is mandatory in  $fm_1$ , which is not the case in  $fm_2$ .

*Diff of feature groups.* It is useful to determine feature groups (Xor and Or) that are in  $fm_1$  but not in  $fm_2$ , including  $G_{XOR_{diff}} = G_{XOR_1} \setminus G_{XOR_2}$  and  $G_{OR_{diff}} = G_{OR_1} \setminus G_{OR_2}$ . We consider that two feature groups are equal if and only if their parent features match and their child features match. In the example of Figure 1,  $G_{XOR_{diff}} = \emptyset$  and  $G_{OR_{diff}} = \{\{\{init, start, paint\}, mustOverride\}\}$

*Diff of implies and excludes.* A syntactic diff of implies (resp. excludes) constraints produces  $I_{diff}$  (resp.  $EX_{diff}$ ) so that  $I_{diff} = I_1 \setminus I_2$  (resp.  $EX_{diff} = EX_1 \setminus EX_2$ ). In Figure 1,  $I_{diff} = \{\{destroy \Rightarrow init\}, \{stop \Rightarrow init\}\}$  and  $EX_{diff} = \emptyset$ . It should be noted that  $I_{diff}$  is not semantically correct in this example: the features `destroy` and `stop` do imply the feature `init` in  $fm_2$ , owing to the parent-child relationships in  $fm_2$ .

### 3.2 Semantic Diff

Syntactic differences are useful for the example of Figure 1. Using the list of differences, a modeler can identify that the feature `init` has been moved or that the feature `mustOverride` becomes a mandatory feature in  $fm_2$ . However, a practitioner rather wants to understand the difference between the two FMs in terms of *configuration semantics* (i.e., in terms of sets of configurations). For this purpose, the list of syntactic differences fails to produce some differences, among

<sup>1</sup> This assumption is shared by [26,28,13] and holds for all the case studies presented in the introduction. The problem of *FM matching* is discussed in Section 4.2.

others: *i*) the four configurations authorized by  $fm_1$  but not by  $fm_2$  are not identified ; *ii*)  $I_{diff}$  does not report that  $stop \Rightarrow mustOverride$  holds in  $fm_1$  but not in  $fm_2$ . With this information, a practitioner could learn that the feature  $stop$  does not imply the selection of the feature  $mustOverride$  in  $fm_2$  whereas it is the case in  $fm_1$ .

To raise the limits of a syntactic diff, we address semantically the list of differences. We translate  $fm_1$  and  $fm_2$  into two formula  $\phi_1$  and  $\phi_2$ . Nevertheless, performing at the level of abstraction of Boolean variables may produce unexploitable results. Stated differently, a practitioner wants to understand differences in terms of feature modeling concepts rather than in terms of a propositional formula. As a result, we take care of producing meaningful information based on the analysis of the two formula.

**Diff of Information Extracted from the Two Formula.** A first general strategy consists in analyzing separately each formula and then performs the differences of the information produced.

*Diff of binary implication (resp. exclusion) graphs.* We consider a binary implication graph of an FM and its propositional formula  $\phi$  as a directed graph  $BIG = (V_{imp}, E_{imp})$  formally defined as follows:

$$V_{imp} = \mathcal{F} \quad E_{imp} = \{(f_i, f_j) \mid \phi \wedge f_i \Rightarrow f_j\} \tag{1}$$

Each binary, directed edge from feature  $f_i$  to feature  $f_j$  represents a binary implication. Based on the analysis of  $\phi_1$  and  $\phi_2$ , we can produce  $BIG_1$  and  $BIG_2$  and then compute  $BIG_{diff} = BIG_1 \setminus BIG_2$ . It is then straightforward to compute the set of binary implications expressed in  $fm_1$  but not in  $fm_2$ . In the example of Figure 1,  $E_{impl_{diff}} = \{\{destroy \Rightarrow mustOverride\}, \{applet \Rightarrow mustOverride\}, \{stop \Rightarrow mustOverride\}, \{init \Rightarrow mustOverride\}\}$ . As we support arbitrary propositional constraints in an FM, it should be noted that  $BIG_{diff}$  cannot be produced syntactically in the general case. Furthermore, the binary implication graph structure, reified from the propositional formula, has the advantage of exposing an information than can be directly translated in terms of feature modeling (i.e., either as a binary implication between a child feature and a parent feature or simply as an implies constraint). As a more general and powerful technique, the computation of  $BIG_{diff}$  should be used in favour of the syntactic diff of implies constraints.

*Diff of binary exclusion graphs.* Similarly, we can compute the set of binary exclusions expressed in  $fm_1$  but not in  $fm_2$ . We consider a binary exclusion graph of an FM and its formula  $\phi$  as an undirected graph, denoted  $BEG$ , consisting of vertices being features and edges denoting a mutual exclusion between two features  $f_i$  and  $f_j$  such that its formula  $\phi$  entails  $f_i \Rightarrow \neg f_j$ . Then,  $BEG_{diff} = BEG_1 \setminus BEG_2$ . In the example of Figure 1, the set of edges of  $BEG_{diff}$  is empty.

*Diff of cliques in implication and exclusion graphs.* We extend the previous technique to n-ary bi-implications and n-ary mutual exclusions. A n-ary bi-implication involves  $n$  features such that  $f_i \Rightarrow f_j$  for any  $i, j = 1 \dots n$ . It can

be obtained by computing cliques in *BIG*. (A clique in the implication graph is a subgraph in which any two vertices are connected by an edge). A  $n$ -ary mutual exclusion involves  $n$  features  $f_1, \dots, f_n$  and is detected if there exists a clique between  $f_1, \dots, f_n$  in *BEG*. (A clique in the exclusion graph requires each member to have an exclusion to every other member.) For the purpose of conciseness (no set of features is subsumed by other), we compute *maximal* cliques in *BIG* and *BEG*. In the example of Figure 1, we detect that features *applet* and *mustOverride* are bi-implied. We do not learn additional difference for this specific example as a syntactic technique already produces such information. Nevertheless the semantic technique is particularly suitable when there are complex cross-tree constraints in an FM.

*Semantic diff of feature groups.* Reasoning techniques can be performed on  $\phi$  to detect candidate feature groups (Xor and Or-groups). It is based on an important property: several FMs can represent the same set of configurations while having different hierarchies and feature groups. As a result some feature groups are not syntactically restituted in a feature diagram<sup>2</sup>. For example, in  $fm_1$ , there are two candidate Or-groups  $\{\{init, start, paint\}, mustOverride\}$ ,  $\{\{init, start, paint\}, applet\}$ , but only  $\{\{init, start, paint\}, applet\}$  is included in  $G_{OR_1}$ . To compute candidate feature groups, we rely on techniques exposed in [10,27] that perform over a propositional formula.

**Reasoning about the Two Formula.** A second general strategy consists in producing relevant information based on the logical combinations of the two formula. We first describe two existing techniques [28,13] relevant for FM differences. *Relationship between two FMs.* Thüm et al. [28] reason on the nature of FM edits, for example, when  $fm_1$  is edited (e.g., some features are moved, added, or removed), giving  $fm_2$ . They provide a classification (see Definition 4) and an efficient algorithm to compute the kind of relationship between two FMs. In case the relationship is not a refactoring, the authors propose a technique to generate an example of configuration authorized in one but not in another.

**Definition 4 (Edits).**  $fm_1$  is a specialization of  $fm_2$  if  $\llbracket fm_1 \rrbracket \subset \llbracket fm_2 \rrbracket$  ;  $fm_1$  is a generalization of  $fm_2$  if  $\llbracket fm_1 \rrbracket \supset \llbracket fm_2 \rrbracket$  ;  $fm_1$  is a refactoring of  $fm_2$  if  $\llbracket fm_1 \rrbracket = \llbracket fm_2 \rrbracket$  ;  $fm_1$  is an arbitrary edit of  $fm_2$  in other cases.

*Quotient.* In [13], an algorithm is presented that takes as input two formula  $\phi_1$  and  $\phi_2$  in *conjunctive normal form* (CNF) – FMs are easily converted to CNF. The algorithm finds for the quotient (i.e., difference) all clauses in  $\phi_1$  which are not entailed by  $\phi_2$  through the satisfiability checks of  $\phi_2 \wedge \neg c$  ( $c$  being a clause of  $\phi_1$ ). As recognized, this is clearly an over-approximation of

<sup>2</sup> In  $fm_2$ , the features *applet*, *mustOverride* and *init* are semantically forming an Or-group. This is not syntactically restituted in the feature diagram (see Figure 1(b)) and is considered as an *anomaly* in the literature [7]. This example, extracted from [13], can be seen as an additional argument in favour of a diff performing at the semantic level.

the difference, but might fail at maximality. In the example of Figure 1, the quotient is  $\{\{init \Rightarrow mustOverride\}, \{applet \Rightarrow mustOverride\}\}$

*Diff of Formula.* The two previous techniques fail to comprehensively represent the difference of the two configuration sets. To raise the limitations, we develop a diff operator, noted  $\oplus \setminus$ , that takes as input two FMs and produces a diff FM (i.e.,  $fm_{diff} = fm_1 \oplus \setminus fm_2$ ).  $fm_{diff}$  is depicted in Figure 2(a). The following defines the semantics of this operator:

$$\llbracket fm_1 \rrbracket \setminus \llbracket fm_2 \rrbracket = \{x \in \llbracket fm_1 \rrbracket \mid x \notin \llbracket fm_2 \rrbracket\} = \llbracket fm_{diff} \rrbracket \tag{M1}$$

Computing the diff formula that encodes  $\llbracket fm_{diff} \rrbracket$  is as follows:

$$\phi_{diff} = (\phi_1 \wedge not(\mathcal{F}_2 \setminus \mathcal{F}_1)) \wedge \neg(\phi_2 \wedge not(\mathcal{F}_1 \setminus \mathcal{F}_2))$$

*not* is a function that, given a non-empty set of features, returns the Boolean conjunction of all negated variables corresponding to features:

$$not(\{f_1, f_2, \dots, f_n\}) = \bigwedge_{i=1..n} \neg f_i$$

The presence of negated variables is needed since we need to emulate the deselection of features that are in  $fm_1$  (resp.  $fm_2$ ) but not in  $fm_2$  (resp.  $fm_1$ ). Otherwise, two features, say  $f_1 \in \mathcal{F}_1$  and  $f_2 \in \mathcal{F}_2$  such that  $f_1 \neq f_2$  (i.e.,  $f_1$  does not match  $f_2$ ), can be combined to form a configuration, thereby violating the configuration semantics of Definition M1. An important property of  $\phi_{diff}$  is that each valid assignment (true/false values assigned to variables) corresponds to a valid configuration of  $\llbracket fm_{diff} \rrbracket$ . With regards to maximality, it thus outperforms the quotient technique.

The connection between the characterization of edits (see Definition 4) and the diff operator is expressed by Lemma 1 (according to set theory,  $\llbracket fm_1 \rrbracket \subseteq \llbracket fm_2 \rrbracket$  is equivalent to  $\llbracket fm_1 \rrbracket \setminus \llbracket fm_2 \rrbracket = \emptyset$ ).

**Lemma 1 (Diff and Specialization/Refactoring).**  *$fm_1$  is a specialization or a refactoring of  $fm_2$  if  $(fm_1 \oplus \setminus fm_2)$  characterizes no valid configurations.*

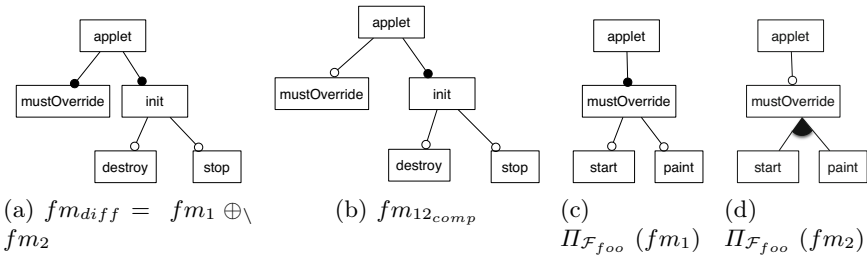
As a result, the satisfiability of  $\phi_{diff}$  can be checked to determine the kind of relationship between  $fm_1$  and  $fm_2$ . In the example of Figure 1,  $fm_1$  is an arbitrary edit of  $fm_2$  since  $\llbracket fm_1 \rrbracket \setminus \llbracket fm_2 \rrbracket \neq \emptyset$  and  $\llbracket fm_2 \rrbracket \setminus \llbracket fm_1 \rrbracket \neq \emptyset$ .

*From formula to an FM.* Though the diff formula is useful for *reasoning*, we cannot render the formula "as is". Producing a complete FM (including the feature hierarchy, feature groups, etc.) is needed typically when an FM is visualized, serialized to a given format or when syntactic differencing techniques are applied. In this case, we need to transform the diff formula  $\phi_{diff}$  as an FM. As stated above, several FMs can represent the same set of configurations [28,7,27]. Intuitively, we want to maximize the parent-child relations that occur in the two input FMs. Furthermore not all feature hierarchies can be chosen (e.g., a hierarchy with *start* as a parent feature of *mustOverride* is too logically restrictive). The

problem of choosing a hierarchy from amongst a set of hierarchies can be formulated as a minimum spanning tree problem over the binary implication graph of  $\phi_{diff}$ . Based on the formula and the hierarchy, propositional logic techniques are applied to synthesize a complete FM (see [4]).

### 3.3 Composition and Decomposition of FMs

*Computing the commonality and the union.* It is interesting for a practitioner to determine the common properties of two FMs (rather than the differences). Syntactical techniques can be naturally applied (e.g., to determine common features). In previous work [3], we propose a semantic operator, called *merge*, that computes an FM whose set of configurations is the intersection of the two sets of configurations. Using the merged FM, a practitioner can enumerate or count the common set of configurations or simply visualize it (see below for more details). In the example of Figure 1, there are 15 common configurations. Another variant of the merge operator, denoted  $\oplus_{\cup}$ , can compute an FM representing the *union* of the two sets of configurations.



**Fig. 2.** Diff FM, merge in union mode of two FMs, two slice FMs

*Complementing FMs.* The need to complement an existing FM, say  $fm_2$ , typically occurs when  $fm_2$  is an underapproximation of  $fm_1$  whereas it should not be the case. Therefore the differencing techniques are not only useful to *detect* an underapproximation but also to *compute* it and then *integrate* it in another model. Interestingly, we can combine the diff operator and the merge operators (in intersection or union mode). A possible application is, for instance, to compute an FM representing both configurations included in  $fm_2$  but not in  $fm_1$  and configurations included in  $fm_1$  but not in  $fm_2$ . It can be formalized in set theory and therefore automated using the techniques previously described:  $fm_{12_{comp}} = (fm_2 \oplus_{\setminus} fm_1) \oplus_{\cup} (fm_1 \oplus_{\setminus} fm_2)$ . Such an FM can be used by practitioners to develop products not yet supported by existing solutions (i.e., neither by  $fm_1$  nor  $fm_2$ ).  $fm_{12_{comp}}$  is depicted in Figure 2(b).

*Decomposing the problem of FM differences.* Understanding and managing FM differences is a manual process and can quickly become difficult for a practitioner when a large number of features and constraints are involved. We thus

propose to decompose the problem into subproblems. Intuitively, it is easier to focus on some parts of the two FMs rather than considering the two FMs in their entire form. In [5], we propose a semantic operator, called *slice*, that produces a projection of a FM (a slice) with respect to a set of selected features (slicing criterion). We define slicing as a unary operation on FM, denoted  $\Pi_{\mathcal{F}_{slice}}(fm)$  where  $\mathcal{F}_{slice} = \{ft_1, ft_2, \dots, ft_n\} \subseteq \mathcal{F}$ . The result of the slicing operation is a new FM,  $fm_{slice}$ , semantically defined in terms of configuration set:  $\llbracket fm_{slice} \rrbracket = \{x \cap \mathcal{F}_{slice} \mid x \in \llbracket fm \rrbracket\}$  (called the *projected* set of configurations). In the context of FM differences, the slice operator can be applied on  $fm_1$  and  $fm_2$  using the same slicing criterion. For example, we apply the slice on  $fm_1$  (the resulting FM is depicted in Figure 2(c)) and on  $fm_2$  (the resulting FM is depicted in Figure 2(d)) using the slicing criterion  $\mathcal{F}_{foo} = \{applet, mustOverride, start, paint\}$ . The two slice FMs can then be analyzed using the differencing techniques exposed throughout the section.

### 3.4 Tool Support: Language and Environment

We rely on FAMILIAR (for *FeAture Model scrIpt Language for manIpulation and Automatic Reasoning*) a domain-specific language for FMs [4]. The language already includes facilities for composing/decomposing FMs, editing FMs (e.g., renaming and removal of features), reasoning about FMs (e.g., validity, comparison of FMs) and their configurations (e.g., counting or enumerating the configurations in an FM). FMs and other types (configuration, set, etc.) are manipulated using variables. Compared to [4], we extend the language and integrate the differencing techniques developed in the paper through the form of operations over FMs (quotient, computation of candidate feature groups and implication / exclusion graphs, etc.). Basic operators to perform the union, intersection or difference of variables are also provided. Two reasoning back-ends (SAT solvers using SAT4J and BDDs using JavaBDD) are internally used and perform over propositional formula to implement the semantic operators. An important property of the language is that operations can be sequentially executed while properties of the variables can be observed. Hence, complex management scenarios can be applied using FAMILIAR environment. A practitioner can decompose the two FMs, then apply some techniques to understand local differences, edit the FMs, and reiterate the process. At the end, the FMs including their differences can be serialized in different formats and be used to pilot configuration or a generative approach.

## 4 Evaluation

### 4.1 Complexity

Performing differences at the semantic level, though more powerful, has a cost. The computation of *BIG* and *BEG* heavily depends on *satisfiability* checks of implications and exclusions. In practice, the computation of *BIG* and *BEG*

scales for thousands of features and can be realized using SAT solvers or BDDs [10,27]. Due to transitivity of implication, maximal cliques (see Section 3.2) are actually strongly connected components in *BIG*, which can be found efficiently by graph traversal. Furthermore, the computation of cliques and Xor-groups scales for thousands of features [27]. To the best of our knowledge, the computation of Or-groups has only been implemented using BDDs. We rely on BDDs to synthesize a complete FM from a formula. (SAT solvers can be used but in this case we do not reconstitute Or-groups). The cost of feature diagram construction is polynomial regarding the size of the BDD [10]. We reuse the heuristics developed in [20] to reduce the size of the BDD – they scale up to 2000 features.

SAT solvers require a formula to be in CNF. Converting  $\phi_{diff}$  into CNF requires  $\phi_2$  to be negated (see Equation M1). As argued in [28], an exponential explosion of clauses occurs when  $\phi_2$  is negated, even for a small number of features. To avoid explosion, we rely on BDDs for computing and reasoning about  $\phi_{diff}$ , since computing the disjunction, conjunction and negation of BDDs can be performed in at most polynomial time with respect to their sizes.

## 4.2 Applying Differencing Techniques: Co-evolution of FMs

**Case Study.** In [2], we presented a process for reverse engineering the FM of FraSCAti, a large and highly configurable component and plugin-based system. The overall challenge is to derive an FM so that its scope is not too large (otherwise some unsafe compositions are authorized) or too narrow (otherwise it is a symptom of unused flexibility). On the one hand, the FM produced by an automated procedure may not be an accurate representation, typically when FraSCAti artefacts do not correctly document the variability of the system. On the other hand, a software architect, while manually elaborating an intentional variability model of FraSCAti, may forget to specify some features or constraints. In order to manage (e.g., understand) the differences between the two FMs, we applied the techniques presented in the paper and the tool support.

**Results and Lessons Learned.** We now report *what* techniques have been used and *how* they helped to manage differences between the FMs. Further details and material (e.g., FAMILIAR scripts) about the case study are available in [1].

*Implications.* We first observed that the FMs involved have the following properties: an average of  $\approx 50$  features and  $\approx 10^6$  configurations per FMs, and a large number of cross-tree constraints (implies). Therefore we made an extensive use of the diff between binary implication graphs. It allowed one to identify dozens of implies constraints expressed in one FM but not in another (and vice-versa). Implies constraints are very important in the FraSCAti case study. First, the software architect elaborates the FM and specifies an important number of binary implications. Second, the extraction procedure combines different sources of information, including plugin dependencies. These dependencies are essentially expressed through implies constraints. The diff between binary implication graphs has the merit of reifying the differences of the two FMs *in terms*



of implies constraints. It is then easier for a software architect to understand the impact of the difference: it is either an implication unintentionally not specified or an implication not documented by plugin dependencies. Compared to a syntactic diff, the major advantage of the structure of binary implication graph is the ability to derive *transitive* implications. The method of quotient produces some disjunctive clauses that can be transformed into implications. Nevertheless, we observed many times that the method suffers from a lack of completeness regarding the diff of implies constraints.

*Or-groups vs Optional.* We use a syntactic diff for computing feature groups expressed in one FM but not in another. A semantic diff for computing candidate feature groups, though more powerful in theory, does not produce additional information in this specific case study. The difference between feature groups concerns *Or-groups*. Indeed some features were modeled as optional features in the FM of the software architect whereas corresponding features formed an Or-group in the FM produced by the automated procedure. This difference, though subtle, occurs for three Or-Groups and has to be identified and managed (since in one case it is possible to not select any feature).

*Decomposition.* The compared FMs did not necessary have the same set of features (e.g., some features are added when a new version of FraSCAti is released). The features included in one FM but not in another disturb the management of differences. Intuitively, such features produce new configurations but we were mostly interested by the evolution of the common subset. Therefore we made an extensive use of the *decomposition* operator (i.e., slice, see Section 3.3) by focusing only on features commonly shared by the two FMs. We then applied the differencing techniques on the decomposed FMs. In particular, it made possible to determine if the common subset has correctly evolved and that no configuration has been broken (i.e., the refactoring or generalization property holds, see Definition 4).

*Interactive process.* Managing differences is not a simple one step-process. Once differences have been identified and understood, we edited FMs accordingly and reiterated the process until having satisfying FMs. Therefore the process is rather incremental and interactive. Automation and reproducibility of the operations are indeed crucial.

**Opportunities for Future Work.** The extracted FM and the FM elaborated by the software architect use different names for features that are actually similar. To avoid unexploitable differencing results, some predirectives were needed and consist in manually renaming features. More automated support seems desirable and *matching* techniques already integrated in model-based tools (e.g., see [12,15]) are good candidates. At the current state of the research, it is difficult to assess the significance of the FM matching problem in practice. Many techniques exposed in this paper can be combined on demand to manage differences. We observed that some information produced are sometimes redundant (e.g., the method of quotient detects a binary implication, already detected by the diff of binary implication graphs). To reduce the cognitive process of a practitioner, an

interesting perspective is to *summarize* the differences (by aggregating information produced by the differencing techniques so that there is no redundancy). We leave it as future work. Another direction for future work is to provide *guidelines* and a *methodological* proposal that could help non-experts to apply all these operators in practice. The usability of the approach (e.g., whether the produced information is understandable enough) should be *evaluated* accordingly using other case studies.

## 5 Related Work

Model differencing has attracted research efforts in recent years, including the development of tools (e.g., see [22,15,19,19,17,18]). The bibliography [22] compiles about 300 publications in this field. Existing approaches mainly focus on *syntactical* differences. As argued in [19,13], models (e.g., FMs) that are syntactically very similar may induce very different semantics and a list of differences should be best addressed *semantically*. Maoz *et al.* define a semantic diff as an operator that takes as input two models and outputs a set of diff witnesses, i.e., instances of one model that are not instances of the other [19]. In our context, instances are configurations and the set of witnesses is finite and can be enumerated if needs be. Fahrenberg *et al.* propose an alternative definition of a semantic diff and argue that a difference between models should be a model [13]. One contribution of our work is precisely to compute a diff FM.

Recently, Maoz *et al.* tackled the problem of semantic model differencing, specifically for class and activity diagrams [17,18]. They defined and implemented two versions of semantic diff operator, *cddiff* and *addiff*. The *cddiff* operator [18] takes as inputs two class diagrams and computes diff witnesses using Alloy Analyzer, a solver for first-order logic. For the *addiff* operator, they presented algorithms that take as input activity diagrams [17]. These two contributions, as ours, are specific to a given formalism and its associated semantics. A few works consider semantic diff between programs, e.g., Jackson and Ladd summarize the semantic diff between two procedures in terms of observable input-output behaviors [14]. We focus on model comparison and not on program comparison.

In the field of feature modeling, Benavides *et al.* [7] survey a set of operations and techniques proposed for automated analysis of FMs. No automated techniques have been reported to reason about or compute differences. A notable exception is the algorithm described in [28], which classifies the evolution of an FM via modifications (see Section 3.2). The algorithm can be used in the context of FM differences but have two limitations. First, the kind of relationship between two FMs does not help to precisely *understand* the impact of a change, e.g., what implies or excludes constraints have been removed and added. Second, the technique does not compute *all* added and removed configurations. In [13], the authors illustrate their vision and theory of semantic model differences using FMs. They propose an algorithm to compute the *quotient* (see Section 3.2). As recognized in [13], the quotient is an approximation of the differences between two FMs whereas the diff FM is not. Another limitation is that the quotient

is a set of disjunctive clauses that are difficult to understand for a practitioner (see Section 4.2). In practice, an additional step seems necessary to transform these clauses into a more readable and manageable information, closer to FM constructs. Segura et al. propose a catalog of rules for merging FMs (union and intersection) [26]. They present syntactic mechanisms (see a comparison in [3]) and no diff operator is considered. The works exposed in [23,11] developed a set of operators to make evolve FMs but no differencing technique is proposed to control the evolution of the FMs.

## 6 Conclusion

Feature models (FMs) are widely used to compactly represent the valid combinations of features (i.e., configurations) supported by a given system. In several application domains and contexts (e.g., software evolution), differences between two FMs should be managed, for example, to identify what are the configurations of an FM that are not included in another. We presented a set of techniques to understand, compute and reason about such differences. The techniques perform at the semantic level (i.e., in terms of sets of configurations) and present relevant information reified from the analysis of propositional formula. A practitioner can detect a difference (e.g., for the purpose of debugging), compute the differences as an FM and then integrate the differences into an existing FM. The tool-supported techniques overcome limitations of earlier attempts and are proved to be essential in a case study.

As future work, we plan to evaluate further the practicality and usefulness of the proposed solution. We hope these insights can contribute to a methodology that guide practitioners in managing FM differences.

**Acknowledgement.** This work was supported by the FNRS, the University of Namur, the FP7 Marie-Curie COFUND program, the Interuniversity Attraction Poles Programme, Belgian Science Policy (MoVES), the BNP, the french ANR SALTY project (<https://salty.unice.fr>) under contract ANR-09-SEGI-012, and the French Ministry of Higher Education and Research, Nord-Pas de Calais Regional Council and FEDER through the CPER-CIA 2007-2013.

## References

1. <https://nyx.unice.fr/projects/familiar/wiki/DiffFMs>
2. Acher, M., Cleve, A., Collet, P., Merle, P., Duchien, L., Lahire, P.: Reverse Engineering Architectural Feature Models. In: Crnkovic, I., Gruhn, V., Book, M. (eds.) ECSA 2011. LNCS, vol. 6903, pp. 220–235. Springer, Heidelberg (2011)
3. Acher, M., Collet, P., Lahire, P., France, R.: Comparing Approaches to Implement Feature Model Composition. In: Kühne, T., Selic, B., Gervais, M.-P., Terrier, F. (eds.) ECMFA 2010. LNCS, vol. 6138, pp. 3–19. Springer, Heidelberg (2010)
4. Acher, M., Collet, P., Lahire, P., France, R.: A Domain-Specific Language for Managing Feature Models. In: Proc. of SAC 2011, pp. 1333–1340. ACM (2011)

5. Acher, M., Collet, P., Lahire, P., France, R.: Slicing Feature Models. In: Proc. of ASE 2011, pp. 424–427. ACM (2011)
6. Acher, M., Collet, P., Lahire, P., Gaignard, A., France, R., Montagnat, J.: Composing Multiple Variability Artifacts to Assemble Coherent Workflows. *Software Quality Journal (Special issue on Quality Engineering for SPLs)* (2011)
7. Benavides, D., Segura, S., Cortés, A.R.: Automated analysis of feature models 20 years later: A literature review. *Inf. Syst.* 35(6), 615–636 (2010)
8. Bille, P.: A survey on tree edit distance and related problems. *Theor. Comput. Sci.* 337(1-3), 217–239 (2005)
9. Czarnecki, K., Eisenecker, U.: *Generative Programming: Methods, Tools, and Applications*. Addison-Wesley (2000)
10. Czarnecki, K., Wąsowski, A.: Feature diagrams and logics: There and back again. In: Proc. of SPLC 2007, pp. 23–34 (2007)
11. Dhungana, D., Grünbacher, P., Rabiser, R., Neumayer, T.: Structuring the modeling space and supporting evolution in software product line engineering. *Journal of Systems and Software* 83(7), 1108–1122 (2010)
12. Euzenat, J., Shvaiko, P.: *Ontology matching*. Springer, Heidelberg (2007)
13. Fahrenberg, U., Legay, A., Wąsowski, A.: Vision Paper: Make a Difference! (Semantically). In: Whittle, J., Clark, T., Kühne, T. (eds.) *MODELS 2011*. LNCS, vol. 6981, pp. 490–500. Springer, Heidelberg (2011)
14. Jackson, D., Ladd, D.A.: Semantic diff.: A tool for summarizing the effects of modifications. In: Proc. of ICSM 1994, pp. 243–252 (1994)
15. Kolovos, D.S., Di Ruscio, D., Pierantonio, A., Paige, R.F.: Different models for model matching: An analysis of approaches to support model differencing. In: Proc. of CVSM 2009 (ICSE Workshop), pp. 1–6 (May 2009)
16. Lotufo, R., She, S., Berger, T., Czarnecki, K., Wąsowski, A.: Evolution of the Linux Kernel Variability Model. In: Bosch, J., Lee, J. (eds.) *SPLC 2010*. LNCS, vol. 6287, pp. 136–150. Springer, Heidelberg (2010)
17. Maoz, S., Ringert, J.O., Rumpe, B.: Addiff: semantic differencing for activity diagrams. In: Proc. of ESEC/FSE 2011, pp. 179–189. ACM (2011)
18. Maoz, S., Ringert, J.O., Rumpe, B.: CDDiff: Semantic Differencing for Class Diagrams. In: Mezini, M. (ed.) *ECOOP 2011*. LNCS, vol. 6813, pp. 230–254. Springer, Heidelberg (2011)
19. Maoz, S., Ringert, J.O., Rumpe, B.: A Manifesto for Semantic Model Differencing. In: Dingel, J., Solberg, A. (eds.) *MODELS 2010*. LNCS, vol. 6627, pp. 194–203. Springer, Heidelberg (2011)
20. Mendonca, M., Wąsowski, A., Czarnecki, K., Cowan, D.: Efficient compilation techniques for large scale feature models. In: Proc. of GPCE 2008, pp. 13–22. ACM (2008)
21. Metzger, A., Pohl, K., Heymans, P., Schobbens, P.-Y., Saval, G.: Disambiguating the documentation of variability in software product lines: A separation of concerns, formalization and automated analysis. In: Proc. of RE 2007, pp. 243–253 (2007)
22. Bibliography on Comparison and Versioning of Software Models, <http://pi.informatik.uni-siegen.de/CVSM>
23. Pleuss, A., Botterweck, G., Dhungana, D., Polzer, A., Kowalewski, S.: Model-driven support for product line evolution on feature level. *Journal of Systems and Software* (2011)
24. Pohl, K., Böckle, G., van der Linden, F.J.: *Software Product Line Engineering: Foundations, Principles and Techniques*. Springer (2005)
25. Schobbens, P.-Y., Heymans, P., Trigaux, J.-C., Bontemps, Y.: Generic semantics of feature diagrams. *Computer Networks* 51(2), 456–479 (2007)

26. Segura, S., Benavides, D., Ruiz-Cortés, A., Trinidad, P.: Automated Merging of Feature Models Using Graph Transformations. In: Lämmel, R., Visser, J., Saraiva, J. (eds.) GTTSE 2007. LNCS, vol. 5235, pp. 489–505. Springer, Heidelberg (2008)
27. She, S., Lotufo, R., Berger, T., Wąsowski, A., Czarnecki, K.: Reverse engineering feature models. In: Proc. of ICSE 2011, pp. 461–470. ACM (2011)
28. Thüm, T., Batory, D., Kästner, C.: Reasoning about edits to feature models. In: Proc. of ICSE 2009, pp. 254–264. ACM/IEEE (2009)

# Wiki Refactoring as Mind Map Reshaping

Gorka Puente and Oscar Díaz

Onekin Research Group, University of the Basque Country (UPV/EHU),  
San Sebastián, Spain

{gorka.puente,oscar.diaz}@ehu.es

**Abstract.** Wikis’ organic growth inevitably leads to wiki degradation and the need for regular wiki refactoring. So far, wiki refactoring is a manual, time-consuming and error-prone activity. We strive to ease wiki refactoring by using mind maps as a graphical representation of the wiki structure, and mind map manipulations as a way to express refactoring. This paper (*i*) defines the semantics of common refactoring operations based on *Wikipedia* best practices, (*ii*) advocates for the use of mind maps as a visualization of wikis for refactoring, and (*iii*) introduces a DSL for wiki refactoring built on top of *FreeMind*, a mind mapping tool. Thus, wikis are depicted as *FreeMind* maps, and map manipulations are interpreted as refactoring operations over the wiki. The rationales for the use of a DSL are based not only on reliability grounds but also on facilitating end-user participation.

**Keywords:** Wiki, refactoring, DSL, mind maps, end-user.

## 1 Introduction

Wikis are becoming mainstream for knowledge formation and sharing [14]. Consubstantial to knowledge formation is exploration, tentative guessing and trial-and-error practices. That is, knowledge formation goes together with regular knowledge revision. In a wiki setting, this knowledge (i.e., content) and its structure evolve with its supporting community (a.k.a. the wiki’s Organic Principle [4]). In practice, this ends up in large structures of articles and categories which constantly need manual refactoring. Refactoring is a disciplined technique for restructuring an existing body of code, altering its internal structure without changing its external behaviour [8]. Our premise is that for wikis, this “external behaviour” (i.e., the invariant to be kept during refactoring) is the wiki content and authorship. Wiki refactoring can change the wiki’s internal structure for the sake of navigability, accessibility or comprehension, but the content (and its authorship) should be kept immutable.

Unfortunately, wiki engines (e.g., *MediaWiki* [1]) are thought for spurring contributions (certainly the cornerstone of this approach) but overlook refactoring. As a result, wiki refactoring is far from trivial. For instance, merging/splitting

---

<sup>1</sup> [www.mediawiki.org](http://www.mediawiki.org) (accessed 19-Mar-12).

two wiki articles requires of at least five interactions in *MediaWiki*. In other words, the semantics of refactoring is not natively supported by the wiki engine. The implications are twofold. First, refactoring is left to the user interpretation. Different users can face the same refactoring problem with different strategies. Although best practices are textually documented<sup>2</sup> [12], the wiki engine does not ensure coherence among the refactoring strategies used throughout the wiki lifespan. Second, the engine does not ensure refactoring reliability. Refactoring operations behave like database transactions in the sense that they comprise a sequence of wiki interactions that (i) should be performed in an all-or-nothing manner, and (ii) should move the wiki to a consistent state (i.e., wiki content must be preserved). This operational semantics is certainly not supported in current wiki engines but on the minds of the wiki users. Consequently, users are left unassisted with the cumbersome task of refactoring.

We advocate for wiki refactoring to follow the main wiki hallmarks [4], namely:

- *Open*, which implies lowering the barriers for layman participation. This tenet entails refactoring to be conducted with minimal disturbance (i.e., reducing “accidental complexity”) and in terms closer to the user. This calls for the introduction of Domain-Specific Languages (DSLs) [13] that help users to conduct refactoring in high-level terms.
- *Observable*, which requires wikis to track changes as well as providing pervasive peer-review mechanisms. To counteract potential misbehaviour, the community can detect and reverse malicious editions. Refactoring should also be observable. Although refactoring keeps wiki content changeless, it can rearrange the very same content along a different set of articles and categories. Such rearrangement should be traceable while preserving authorship attribution.

This work contributes to the area of wiki refactoring by (i) identifying main wiki refactoring constructs, (ii) providing the operational semantics for these constructs, and (iii) introducing a graphical DSL for these constructs, i.e., *WikiWhirl*. *WikiWhirl* does not achieve anything that cannot be obtained by directly interacting through the *MediaWiki* front-end. The difference stems from productivity (how long does it take?), accessibility (who can do the refactoring?) and reliability (are authorship preserved?). *WikiWhirl* is available to download at [www.onekin.org/wikiwhirl](http://www.onekin.org/wikiwhirl).

The paper starts by highlighting the differences between wiki and database refactoring (Section 2). Next, we provide a refactoring session as a motivating scenario (Section 3). We introduce the abstract syntax for *WikiWhirl* (i.e., the definition of the concepts of the language and their relationships) and its concrete syntax in Section 4 and Section 5, respectively. Related work and some conclusions end the paper.

---

<sup>2</sup> [http://en.wikipedia.org/wiki/Wikipedia\\_guidelines](http://en.wikipedia.org/wiki/Wikipedia_guidelines) (accessed 19-Mar-12).

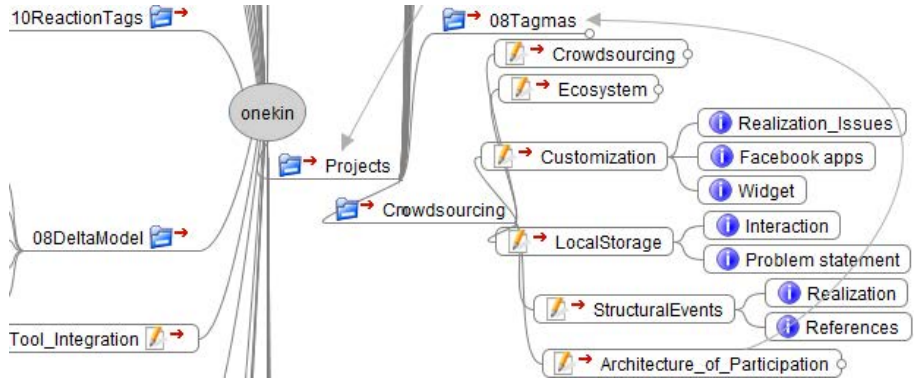


Fig. 1. The *Onekin* wiki depicted as a mind map

## 2 Wiki Refactoring *versus* Database Refactoring

At first sight, wiki refactoring does not look a big deal. After all, Ward Cunningham defines wikis as "the simplest online database that could possibly work"<sup>3</sup>. And database refactoring is a long lasting research topic [1]. However, the fact that wikis are databases does not mean that wiki refactoring equates with database refactoring. Differences stem from the “what”, the “how” and the “who” conducts the refactoring.

**What.** Database refactoring mainly focuses on the database schema. At most, tuples might be subject to co-evolution, i.e., a migration process from the old schema to the new schema. By contrast, wiki refactoring is not about the schema of the database that keeps the wiki content. These tables are set by the wiki engine (e.g., *MediaWiki*): all *MediaWiki* wikis use the very same database schema. The wiki is not the schema but the data, i.e., the tuples. Therefore, wiki refactoring entails changing the data, not the schema.

**How.** Logical independence is a solid principle of database operation whereby changes in the database schema should minimally disturb client applications. A similar issue arises when facing API evolution, leading to the so-called backward compatibility. In general, this notion of "logical independence-ness" rises from any *share* resource that evolves: let it be a database schema, a component or a software library. Wikis are shared resources. The question is then what can be affected by wikis' evolution. While applications are impacted by changes in the database schema, wikis impact end users in their double role of readers and authors. Changing the wiki structure (as result of a refactoring) should cause minimal interference on these activities (i.e., reading and authoring). Hence, we introduce two dimensions of independence for wikis:

<sup>3</sup> <http://wiki.org/wiki.cgi?WhatIsWiki> (accessed 19-Mar-12).

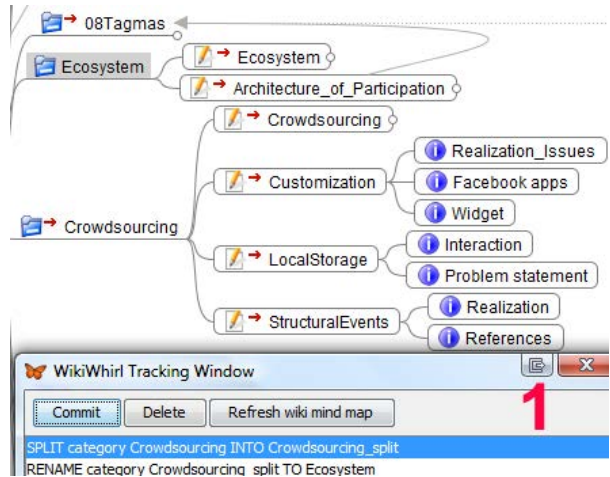


- *Readership independence.* Refactoring does not alter the content but how this content is distributed among articles or categories. Wiki readers should be informed of where content has been moved to. In addition, wiki articles are Web resources users can bookmark. Hence, readership independence also includes the ability for a wiki to preserve URL addresses upon evolving articles/categories.
- *Authorship independence.* Acknowledging the authorship has been reported as a main motivator of contributions [2], and it is one of the directives of *Wikipedia*. Wiki refactoring must preserve authorship.

**Who.** The complexity and criticality of DBMSs require of dedicated users: the database administrators. By contrast, wikis promote openness and accessibility. This is realized as minimizing the skills to operate upon wikis. Unfortunately, wiki refactoring is not technically obvious. Not only is it time consuming but also error prone. Defective refactoring can compromise wiki coherence (e.g., no common understanding of how to conduct article merge) as well as authorship and readership independence. As a result, those that know *what to refactor* (i.e., the domain experts that know which articles are better merged) might ignore *how to refactor* (i.e., the operative that goes to properly merge two articles). Means are needed for end users to refactor their own body of knowledge by themselves. Next, we provide an example of such a refactoring process.

### 3 A Refactoring Session

A wiki is basically a graph of articles and categories (i.e., keywords for article characterization). For the time being, let us depict this graph in a radial way, where the center is the wiki's title, the leaves are the wiki's articles, and categories are in-between nodes. Fig. 1 provides an example for the wiki of the *Onekin* research group. This wiki documents ongoing research projects. Projects are reflected as wiki categories (e.g., *Crowdsourcing*) whereas project issues become wiki articles (e.g., *Ecosystem*).



**Fig. 2.** Split operation during a refactoring session

For its very nature, research projects are conceived and developed on the go. Wikis perfectly fit this way of working. For instance, gains in understanding what *Crowdsourcing* is about, can impact the wiki as follows (Figs. 2, 3 and 4):

1. **Splitting** *Crowdsourcing*. Rationale: the issue of *Ecosystem* has evolved into a new matter on its own,
2. **Merging** articles *LocalStorage* & *StructureEvent* into a *Realization* article. Rationale: wide overlap between these issues,
3. **Moving** the section *Realization\_Issues* from *Customization* to the *Realization* article. Rationale: *Realization* should frame all realization concerns no matter where they arise.

These scenarios should be conducted through the wiki’s front-end. However, wikis’ front-end favours easy contribution rather than refactoring. Moving or merging implies moving back and forth between the affected wiki articles. In addition, the user should care for preserving authorship: if you move a section, authorship should be moved as well by leaving appropriate references. This entails to handle redirects, talk pages and recent changes. In short, the previous splitting-merging-moving process may take a long time in *MediaWiki*.

Notice however, that it is not only a question of efficiency but also of reliability and coherence. Some refactoring operations might involve more than one interaction on *MediaWiki*. The user should ensure such set is handled as “a transaction” in the sense of atomicity and coherent meaning. In addition, the process should care for authorship and readership independence.

We advocate for abstracting the terms in which refactoring is conducted. That is, moving from low-level UI interactions to high-level domain-specific operations. To this end, we investigate the use of Domain-Specific Languages (DSLs). This entails systematizing a set of refactoring operations and their semantics (the DSL’s abstract syntax), and nailing down a way to specify them (the DSL’s concrete syntax). The output is the *WikiWhirl* DSL.

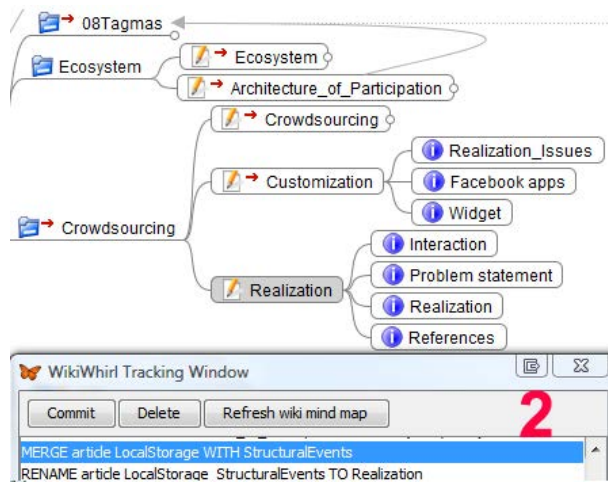


Fig. 3. Merge operation during a refactoring session

## 4 WikiWhirl’s Abstract Syntax

The abstract syntax describes the concepts of the language, the relationships among them, and the structuring rules that constrain the model elements and their combinations in order to respect the domain rules [19]. This is expressed as the DSL metamodel. Fig. 5 depicts the *WikiWhirl* metamodel. A *WikiWhirl* process (*WW\_Process*) includes: (i) a *Wiki* model, and (ii) a set of *WikiWhirl* refactoring operations (*WW\_Operations*) on the *Wiki* model.

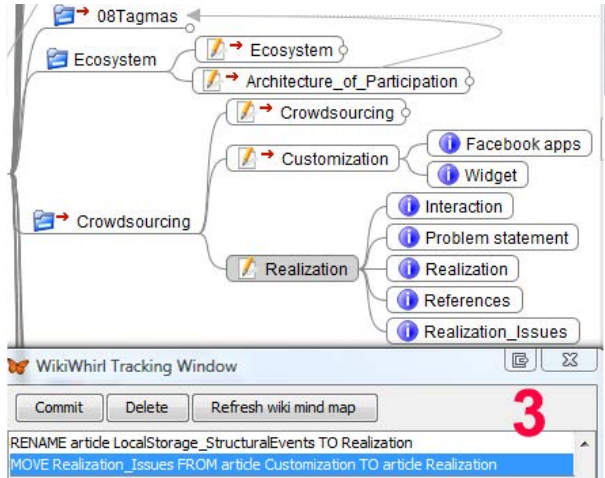


Fig. 4. Move operation during a refactoring session

**The Wiki Model.** It conceives the wiki as an aggregate of resources (i.e., *categories*, *articles* and *sections*). Wiki content is supported through *sections*<sup>4</sup>. Notice that sections are abstractions over existing wikis. Wiki engines do not support sections as independent artefacts but embedded as part of the wikitext of pages. However, we promote sections as full-fledged class elements due to its importance during refactoring. Categories play a double role: as pages, they describe what the category is about; as tags, they are used to organize and locate articles along the wiki. Adding a category to an article creates a link that permits easy navigation from this page to pages in that category.

**Operations.** Table 1 summarizes the *WikiWhirl* refactoring operations. There are two main reasons to include an operation. First, the operation is natively supported by most wiki engines (e.g., *create*, *categorize*). Second, the operation as such is not supported but documented as part of *Wikipedia*’s good practices (e.g., *merge*, *split*)<sup>5</sup>. The table shows the expected frequency and productivity gains. The latter is measure as the number of clicks required to conduct the operation through *MediaWiki*. For example, *drop* an article involves one click on the delete button and one click on the confirmation button.

However, metamodels only capture the structure but not the operational semantics of these operations, i.e., the impact of the operation on the *Wiki model*. This is important not only for the user to know what to expect about the enactment of these operations but also to assess the abstraction effort made by these operations. Such semantics is scattered and textually described in different

<sup>4</sup> Only first level sections are considered (`'== sectionName =='` in *MediaWiki*).

<sup>5</sup> [http://en.wikipedia.org/wiki/Wikipedia\\_guidelines](http://en.wikipedia.org/wiki/Wikipedia_guidelines) (accessed 19-Mar-12).

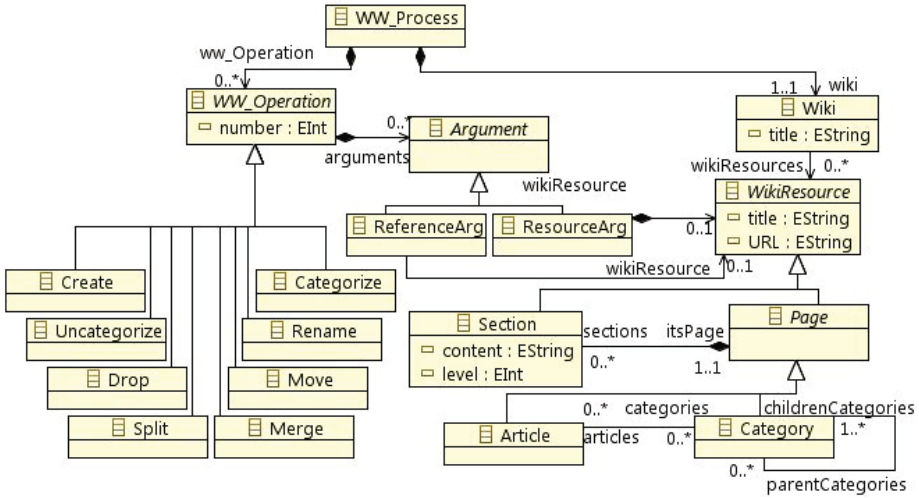


Fig. 5. WikiWhirl metamodel (abstract syntax)

Wikipedia recommendations. In addition, we should indicate also the impact on authorship and readership independence. This requires outlining other ancillary wiki artefacts that, although removed from the refactoring abstraction, they do play an important role in ensuring the independence principles that refactoring operations should obey. That is, these artefacts are transparent for the user during refactoring, but they need to be considered during the implementation of the refactoring operations. These artefacts include:

- *Talk pages* (a.k.a. discussion pages) hold discussions about the content of the associated page without interfering with content editing. Talk pages might be used to publicize refactoring changes on the associated articles.
- *Redirect pages* send the reader to another page. These can handle alternative syntactic representations of the same concept whereby no matter the reference used they are all redirected to the same page. During refactoring, articles' content can be moved to different places so the original article vanishes. Redirection avoids dangling references to the removed article so that existing references are dynamically redirected to the new location.
- *Recent changes* page keeps a trace of the most recent edits made to the wiki. Using this page, users can monitor and review the work of other users, allowing mistakes correction and vandalism elimination. This page can also be used to trace refactoring changes so that the rest of the community is informed about who, when and how conducts the refactoring. A brief explanation, called edit summary, can be added to each edit.

Next subsections focus on *splitting* and *merging*.

**Table 1.** *WikiWhirl* operations: expected frequency & #clicks in *MediaWiki* & impact on authorship and readership independence

<i>WikiWhirl</i> refactoring operation	Frequency <sup>6</sup>	# clicks	Authorship independence		Readership independence	
			Recent changes	Talk page	Summary section	Redirect page
Create	High	3	✓			
Categorize	High	2	✓			
Uncategorize	Medium	2	✓			
Rename <sup>7</sup>	Medium	2	✓	✓		✓
Drop	Medium	2	✓			
Split	Low	6	✓	✓	✓	
Merge	Low	9	✓	✓	✓	✓
Move	Medium	5	✓	✓	✓	✓

### 4.1 Article Split (*sourceArticle*, *newArticle*)

Article split is a refactoring process documented by *Wikipedia*<sup>8</sup>. The two main reasons for article split are size and content relevance. If either the article becomes too large or its content seems to diverge between different purposes, then a split may be considered. The *newArticle* is categorized along the lines of the *sourceArticle*.

**Authorship Independence.** It is a requirement of *Wikipedia*'s licensing that attribution is given to the original author(s), and deletion of that content should be avoided. We follow this recommendation by (i) preventing content deletion and (ii), introducing in the associated talk pages a note such as “*Section sectionName from sourceArticle was copied into newArticle at timestamp*”. In addition, the *recentChanges* page of the *newArticle* is to include a summary, noting the origin of this article (e.g., “*split content from [[article name]]*”). Likewise, the *recentChanges* page of the *sourceArticle* should also include another note indicating that it has been subject to split (e.g., “*split content to [[article name]]*”). This permits users to follow the content trail and to protect against the article subsequently being deleted and the history of the new page eradicated.

**Readership Independence.** If a section is split from the *sourceArticle*, a summary section should be left in this article. At the top of the section, it should contain a link to the newly created page. Category split is similar to article split. The difference stems from children (articles or categories) of the *sourceCategory*

<sup>6</sup> <http://en.wikipedia.org/wiki/Wikipedia:Statistics>

<sup>7</sup> Category rename is not possible in *MediaWiki*. (i) Create a new category, (ii) copy the content, (iii) change the category tag manually on every page that link to that category, and (iv) redirect the old to the new category. This is automatic in *WikiWhirl*.

<sup>8</sup> <http://en.wikipedia.org/wiki/Wikipedia:Splitting> (accessed 19-Mar-12).

might need to be re-allocated into the *newCategory*. That is, some articles might undertake a *categorize* operation.

## 4.2 Article Merge (*articleToMerge1*, *articleToMerge2*)

Article split is a refactoring process documented by *Wikipedia*<sup>9</sup>. Reasons to merge an article include the unnecessary duplication of content, significant overlap with the topic of another page, and minimal content that could be covered in or requires the context of a page on a broader topic. This operation creates a new article whose content is that of the merged articles, and its title results from concatenating “*articleToMerge1\_ articleToMerge2*”.

**Authorship Independence.** A comment in the edit summary must be made in the *articlesToMerge* as to where they have being merged to, and must be noted in the *targetArticle*’s edit summary where the content from other pages are being merged from. For our sample scenario, Fig. 6 shows the outcome of merging *LocalStorage* and *StructuralEvents* into *Realization*.

**Readership Independence.** Merging should always leave a *redirect* from the *mergedArticles* to the *targetArticle*.

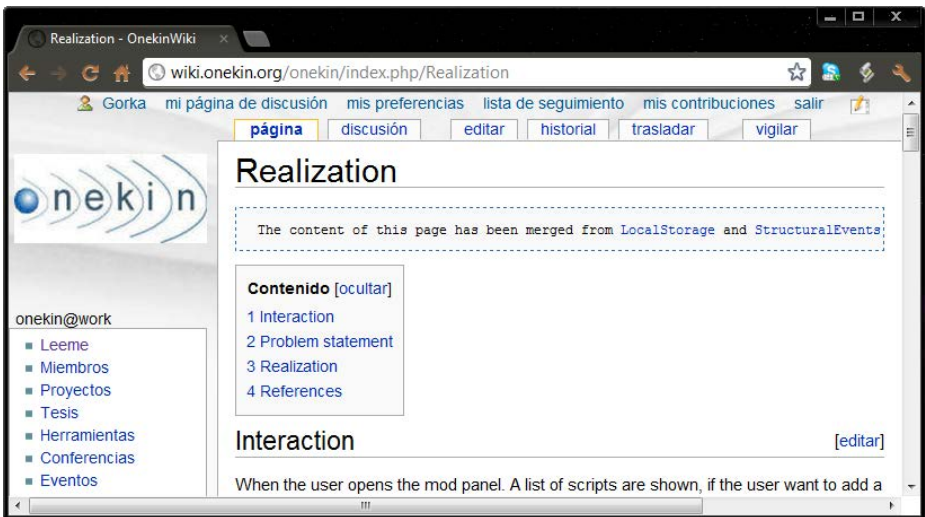


Fig. 6. “*Realization*” article after merging. The warning at the top is automatically generated as part of the merging semantics.

## 5 WikiWhirl’s Concrete Syntax

Broadly, the concrete syntax is a mapping between the metamodel concepts (i.e., abstract syntax) and their textual or visual representation. While the abstract

<sup>9</sup> <http://en.wikipedia.org/wiki/Wikipedia:Merging> (accessed 19-Mar-12).

syntax addresses expressiveness, the concrete syntax cares for the DSL usability and learnability. Should these goals be better served by a textual or a graphical representation? If graphical, which kind of notation? As highlighted by Lengler and Eppler [10], there might not be only one appropriate visualization for a body of knowledge. The selection of the visualization depends on the requirements to meet: visualize author collaboration [20], editions number, network density, page connections<sup>10</sup>, single page relationships (e.g., *Annoki* [17]), wiki structure (e.g. *WikiNavMap* [18]), etc. In addition, the expressiveness of these representations for the matter at hand should be balanced against learnability.

Therefore, we need to look into not only the characteristics of the object to be displayed (e.g., size) but also how this object is to be manipulated. The object to be displayed is a graph, where nodes stand for pages while edges denote relationships between these pages. Specifically, we focus on corporate wikis which have an estimate of up to 1500 nodes [16]. The second criterion is manipulation, i.e., the process of refactoring a wiki. Two approaches co-exist. In the bottom-up approach, the user knows the subject of refactoring (i.e., you know which article/category needs to be refactored), and next, a larger view might be required to set this subject into a larger context. By contrast, the top-down approach starts with a global view of the wiki, and next, the user looks for “bad smells” (e.g., too deep category hierarchies with few articles may indicate too much structure [12]). This way of working calls for agile visualizations that permit to define “views” over existing wiki graphs as well as collapse or extend these views as we gain understanding about the refactoring needed.

A main premise of this work is that refactoring is a process of gradual understanding of the wiki structure. Even if you detect “bad smells”, the strategy to follow will in most cases, require some “refactoring reasoning” before taking the decision. Indeed, it not clear whether “refactoring recipes” (i.e., patterns) like those available for software would make sense for wikis. Wiki articles are not code but knowledge, better said, “knowledge drafts”. It is this *embryonic-like* nature of wiki content together with its *organic growth* what suggests the parallelism with mind maps. According to its proponents, mind maps mimic the way the brain works: “Your brain does not think linearly or sequentially like a computer; it thinks multilaterally, radiantly. When you create a mind map the branches grow outwards from the central image to form another level or sub-branches, encouraging you to create more ideas out of each thought you add” [3].

Drawing on these insights, we select mind maps as the visual representation for wiki refactoring. Notice that this requires moving from graphs to a tree-like representation. Rather than developing our own visual representation, we capitalize on an existing one: *FreeMind*<sup>11</sup>. With over 6,000 daily downloads, it is one of the most popular mind mapping tools. Fig. 1 shows a *FreeMind* map that stands for a wiki. The main advantage rests on the easiness to play around as you gain understanding about the wiki (e.g., nodes are easily moved

<sup>10</sup> <http://sonivis.org> (accessed 19-Mar-12).

<sup>11</sup> <http://freemind.sourceforge.net> (accessed 19-Mar-12).

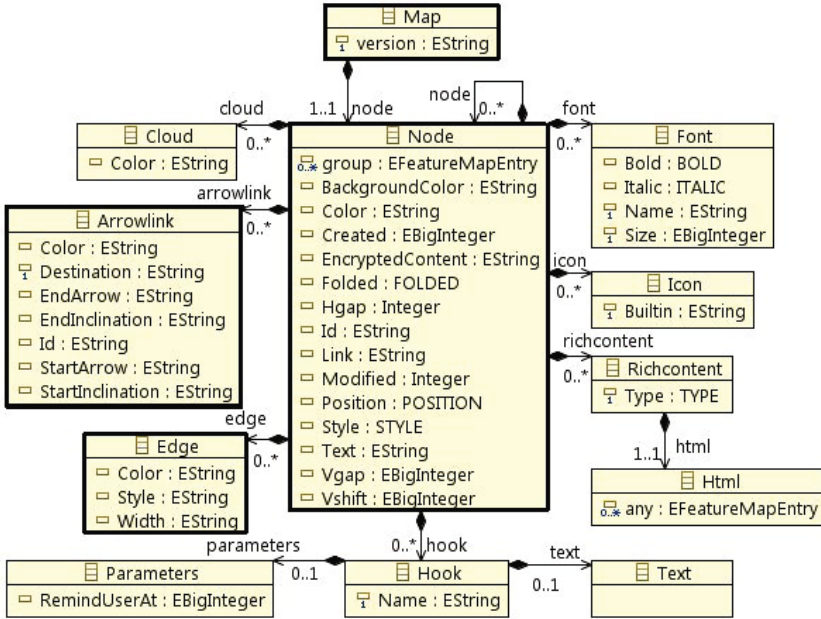


Fig. 7. *FreeMind* metamodel




around; branches collapsed, etc.). Betting for an existing tool not only speeds up development but learnability as well on the hope that the target audience (i.e., employees in corporate wikis) may be familiar with *FreeMind*. The rest of this section introduces *WikiWhirl* as a visual language on top of *FreeMind*, i.e., how the wiki model and its operations are realized in *FreeMind*. This still leaves open two questions: (i) how to *import* the wiki into *FreeMind*, and (ii) how to *export* the map changes back to the wiki. This is the topic of Section 5.3.

### 5.1 Mapping Wiki Models to Mind Maps




A *WikiWhirl* model is to be realized as a compliant *FreeMind* map. A mapping is then set between the *WikiWhirl*'s metamodel (Fig. 5) and their visual counterparts, i.e., the *FreeMind* metamodel. The latter is depicted in Fig. 7. A **Map** is a compound of **Nodes**. Nodes have *Text* as the title and might hold a *Link* to an external resource (local or remote) as well as a set of properties mainly for rendering concerns. For instance, the *Style* property can be *fork* or *bubble* and determines the look of the node as a tagged line or a bubble, respectively. Next, nodes are basically arranged in a tree-like way. A central node serves as the tree root. Tree structures are constructed using **Edges**. An Edge is a connector that relates a node with its parent. Additionally, **Arrowlinks** are also connectors



**Table 2.** *WikiWhirl-to-FreeMind* mapping

WikiWhirl primitives	FreeMind primitives
“Wiki” class	Root node
“WikiResource” class (see subclasses)	Node
“title” property	node title: Text
“URL” property	Link
“Category” class	Node with “folder” icon 
first “parentCategories” reference	edge
rest “parentCategories” references	arrow link
“Article” class	Node with “edit” icon 
first “categories” reference	edge
rest “categories” reference	arrow link
“Section” class	Node with “info” icon 
“itsPage” reference	edge

but in this case, the connection is between two arbitrary nodes. Finally, **Icons**<sup>12</sup> and **Fonts** can be associated with nodes in an attempt to reflect the underlying semantics of the node. Of course, this semantics resides in the users’ head.

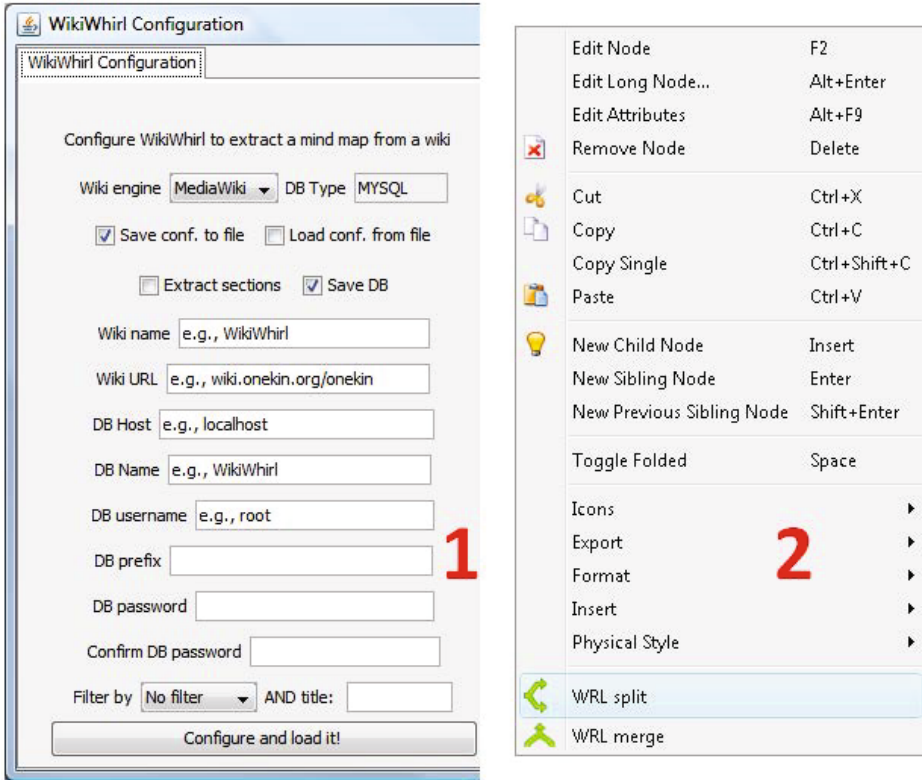
Next, we define the *WikiWhirl-to-FreeMind* correspondence (see Table 2). *WikiResources* are mapped as *nodes*<sup>13</sup>. *WikiResources* are classified as categories, articles and sections. This categorization is represented through *FreeMind* icons: the “folder” icon , the “edit” icon  and the “info” icon  denotes category, article and section nodes, respectively. Next, *WikiWhirl* relationships. Since some relationships are M:N, the first relationship instance is represented through an *edge* (the links that support the tree-like structure) while the rest of the relationship instances are denoted through *arrowLinks*. Fig. 1 illustrates this situation. Article “*Architecture of Participation*” belongs to two categories; the relationship with “*Crowdsourcing*” is denoted by an *edge*, while that of “*08Tagmas*” is depicted as an *arrowLink*.

## 5.2 Mapping Refactoring Operations to Manipulations on Mind Maps

*FreeMind* is now turned into a refactoring tool for wikis. This basically entails two aspects. First, traditional actions in *FreeMind* are re-interpreted in terms of the refactoring operations (e.g., “node removal” becomes “article drop”). Second, we extend *FreeMind* to cater for refactoring specifics: aspects that do not rise

<sup>12</sup> *FreeMind* provides a fixed set of icons. In the last version, users can introduce their own icons, although it is not recommended for interoperability reasons.

<sup>13</sup> Interesting enough, this representation provides a “site map” of the wiki, i.e., the map accounts for a global view of the wiki where users may navigate to any page just by clicking on a node. *FreeMind* mind maps can be inlayed in *MediaWiki* using the *FreeMind* extension [www.mediawiki.org/wiki/Extension:FreeMind](http://www.mediawiki.org/wiki/Extension:FreeMind)



**Fig. 8.** Turning *FreeMind* into a wiki refactoring tool. (i) The configuration menu, and (ii) the popup menu (right mouse click).

during the handling of traditional mind maps. This implies (see Fig. 8): (i) a way to *import mind maps* from wikis. This is achieved through the configuration menu. This menu sets the configuration service and the load mode. The former indicates the wiki database configuration parameters. The load mode permits to filter sections out of the refactoring session for the sake of efficiency; (ii) *inhibiting the removal of nodes* that stand for sections. Sections can only be moved, but never removed since they are content containers (the wiki invariant). Likewise, article/categories can only be deleted if they contain no section; (iii) introducing “*merge*” and “*split*” as operations on nodes. For this purpose, the right-mouse-click menu is extended with these two options; and (iv), introducing a *tracking window* to trace what the user is graphically doing (Fig. 11). The trace is described in terms of *WikiWhirl* operations. The window offers three buttons: the “*Delete*” button removes the highlighted operation from the trace; the “*Commit*” button enacts the refactoring process, enduring the changes in the wiki database, and the “*Refresh wiki mind map*” button restores an initial database dump and regenerates the mind map.

### 5.3 Importing and Exporting *WikiWhirl* Expressions

A *WikiWhirl* expression is a wiki model (i.e., a mind map) plus a sequence of operations over that wiki. A trivial expression is that with no operations (i.e., an empty sequence) but just a wiki model. A refactoring session:

- Starts with a trivial expression that is gradually enriched as the user operates over the wiki model. It is not expected users to directly provide this initial expression, i.e., the wiki model. Rather, wiki maps should be harvested from existing wiki databases. That is, we import a trivial *WikiWhirl* expression from the wiki database. For this purpose, we use *Schemol* [7], a transformation language that declaratively specify how class models can be obtained from tables.
- Ends by exporting the “sequence of operations” as obtained from *FreeMind* into an SQL script. This process is threefold. First, the refactoring trace is mapped to a *WikiWhirl* model using the EMF persistence framework<sup>14</sup>. Second, the *WikiWhirl* model is transformed to a SQL script. This model-to-text transformation is realized through *MOFScript*<sup>15</sup>. Finally, the *SQL* script is executed; which updates the *MediaWiki* database, i.e., the wiki project.

## 6 Related Work

This work is related with wiki visualization and management. As for the former, a recent survey [11] highlights that wikis usually exhibit poor structure. On account of this, distinct tools try to improve the user experience by providing improved textual or graphical representations, or even combinations of both. Hirsch et al. [9] define a *Visual Wiki* as a combination of a visual and a textual representation of a wiki. Two of the proposed prototypes, *Thinkbase* and *Thinkpedia*, represent two wikis: *FreeBase* (a semantic wiki) and *Wikipedia*. The main differences with our work are: (i) they represent the semantic relationships and not the wiki structure (i.e., hierarchy of categories and articles), and (ii) their aim is the visual navigation and exploration whereas our is wiki refactoring.

As for wiki management, in a previous work [5], we also resorted to *FreeMind* as a convenient means to engage users for wiki scaffolding. The aim was to depict a blueprint of the wiki as a mind map, and next, generate the wiki installation. Now the problem is the other way around. *WikiWhirl* first extracts the wiki structure from the wiki database; next depicts the mind map counterpart, which can then be modified by the users; finally, the resulting mind map is transformed into a set of refactoring *MediaWiki* directives. Only this last step keeps some resemblance with our previous work on wiki scaffolding.

As far as automating repetitive tasks, bots<sup>16</sup> are the most common mechanism. Wikis have many routinely tasks to be done and many of them are

<sup>14</sup> <http://eclipse.org/modeling/emf/> (accessed 19-Mar-12).

<sup>15</sup> <http://eclipse.org/gmt/mofscript/> (accessed 19-Mar-12).

<sup>16</sup> <http://en.wikipedia.org/wiki/Wikipedia:Bots> (accessed 19-Mar-12).

too cumbersome to be performed by users (e.g., mass edits or check copyright violations). Bots are tools that take care of articles maintenance. However, other cases are more dubious, and automatic correction is inappropriate. An assisted approach seems to be more appropriate when it needs the supervision of a human eye. This is the rationale behind [6], where ballots are used to detect inconsistencies and inform the users.

Rosenfeld et al. [15] propose a strategy for semantic wiki evolution based on software refactoring. They identify “bad smells” and the refactoring pattern counterparts. They introduce six *semantic* refactoring operations (e.g., move annotation: change the subject of an annotation to another) and four bad smells (e.g., concept too categorized: it belongs to many categories). The differences with our work stem from (i) the focus (semantic resources *vs.* wiki structure), and (ii) the approach (template-based description *vs.* graphical DSL).

## 7 Conclusions

We aim to pave the road for an assisted refactoring for wikis. So far, wiki refactoring is a cumbersome, lengthy activity whose *modus operandi* does not match the accessible and friendly way of editing wiki articles. Such difficulty puts the layman off. But, it is the layman (no *tech-savvy* people) who writes the article, knows the wiki content, and detects refactoring opportunities. This paper strives to abstract from low-level wiki interactions to domain-specific constructs that permit easily and reliably express refactoring processes to the layman. This vision is realized into *WikiWhirl*, a DSL built on top of *FreeMind*: (i) wikis are imported as mind maps, (ii) users perform refactoring operations as re-arrangements of mind map nodes, and (iii) this re-arrangement denote refactoring operations that can be saved into the wiki database while preserving authorship and readership. Future work includes to collect empirical evidence about the usefulness of *WikiWhirl*, and to extend *WikiWhirl* to other wiki engines.

**Acknowledgments.** We wish to thank Javier Luis Cánovas Izquierdo of AtlanMod Research group, for his support with *Schemol*. This work is co-supported by the Spanish Ministry of Education, and the European Social Fund under contract TIN2011-23839 (*Scriptongue*). Puente has a doctoral grant from the Spanish Ministry of Science & Education.

## References

1. Ambler, S.W., Sadalage, P.J.: Refactoring Databases: Evolutionary Database Design. Addison Wesley Professional (2006)
2. Arazy, O., Stroulia, E., Ruecker, S., Arias, C., Fiorentino, C., Ganev, V., Yau, T.: Recognizing Contributions in Wikis: Authorship Categories, Algorithms, and Visualizations. Journal of the American Society for Information Science and Technology (JASIST), 1166–1179 (2010)

3. Buzan, T., Griffiths, C.: *Mind Maps for Business*. BBC active (2010)
4. Cunningham, W.: *Design Principles of Wiki: How can so Little do so Much?* In: *Int. Sym. Wikis (WikiSym)*, pp. 13–14 (2006)
5. Díaz, O., Puente, G.: *A DSL for Corporate Wiki Initialization*. In: Mouratidis, H., Rolland, C. (eds.) *CAiSE 2011*. LNCS, vol. 6741, pp. 237–251. Springer, Heidelberg (2011)
6. Díaz, O., Puente, G., Arellano, C.: *Wiki Refactoring: an Assisted Approach based on Ballots*. In: *Proceedings of the 7th International Symposium on Wikis and Open Collaboration, WikiSym 2011*, pp. 195–196. ACM (2011)
7. Díaz, O., Puente, G., Izquierdo, J.L.C., Molina, J.G.: *Harvesting Models from Web 2.0 Databases*. *Software and Systems Modeling*, 1–20 (2011), doi:10.1007/s10270-011-0194-z
8. Fowler, M.: *Refactoring: Improving the Design of Existing Code*. Addison-Wesley, Boston (1999)
9. Hirsch, C., Hosking, J.G., Grundy, J.C., Chaffe, T.: *ThinkFree: Using a Visual Wiki for IT Knowledge Management in a Tertiary Institution*. In: *Int. Sym. Wikis, WikiSym* (2010)
10. Lengler, R., Eppler, M.J.: *Towards a Periodic Table of Visualization Methods for Management*. In: *Conference on Graphics and Visualization in Engineering*, pp. 1–6 (2007)
11. Lykourantzou, I., Dagka, F., Papadaki, K., Lepouras, G., Vassilakis, C.: *Wikis in Enterprise Settings: a Survey*. In: *Enterprise Information Systems*, pp. 1–53 (2011)
12. Mader, S.: *Wikipatterns: A Practical Guide to Improving Productivity and Collaboration in your Organization*. John Wiley & Sons Inc., Wiley (2008)
13. Mernik, M., Heering, J., Sloane, A.M.: *When and How to Develop Domain-Specific Languages*. *ACM Computing Surveys* 37(4), 316–344 (2005)
14. Raman, M.: *Wiki Technology as A "Free" Collaborative Tool within an Organizational Setting*. *IS Management* 23, 59–66 (2006)
15. Rosenfeld, M., Fernández, A., Díaz, A.: *Semantic Wiki Refactoring. A Strategy to Assist Semantic Wiki Evolution*. In: *5th Workshop on Semantic Wikis Linking Data and People, SemWiki 2010* (2010)
16. Stein, K., Blaschke, S.: *Corporate Wikis: A Comparative Analysis of Structures and Dynamics*. In: *Wissensmanagement*, pp. 77–86 (2009)
17. Tansey, B., Stroulia, E.: *Annoki: A MediaWiki-based Collaboration Platform*. In: *Int. Conf. on Software Engineering, ICSE* (2010)
18. Ullman, A.J., Kay, J.: *WikiNavMap: a Visualisation to Supplement Team-based Wikis*. In: *Human Factors in Computing Systems (CHI)*, pp. 2711–2716 (2007)
19. Vallecillo, A.: *On the Combination of Domain Specific Modeling Languages*. In: Kühne, T., Selic, B., Gervais, M.-P., Terrier, F. (eds.) *ECMFA 2010*. LNCS, vol. 6138, pp. 305–320. Springer, Heidelberg (2010)
20. Viégas, F.B., Wattenberg, M., Dave, K.: *Studying Cooperation and Conflict between Authors with History Flow Visualizations*. In: *Conference on Human Factors in Computing Systems (CHI)*, pp. 575–582 (2004)

# Purpose Driven Competency Planning for Enterprise Modeling Projects

Janis Stirna<sup>1</sup> and Anne Persson<sup>2</sup>

<sup>1</sup> Department of Computer and Systems Sciences, Stockholm University, Forum 100,  
SE-1644 0, Kista, Sweden

js@dsv.su.se

<sup>2</sup> University of Skövde, Informatics Research Centre, P.O. Box 408,  
SE-541 28 Skövde, Sweden

anne.persson@his.se

**Abstract.** Much of the success of projects using Enterprise Modeling (EM) depends more on the quality of the process of modeling rather than on the method used. One important influence on the quality of the modeling process is the competency level of the experts responsible for the EM approach. Each EM project is, however, specific depending on the purpose of modeling, such as developing the business, ensuring the quality of business operations, and using EM as a problem solving tool. The objective of this paper is to discuss the core competency needs for the EM practitioner and to relate those needs to different purposes of EM.

**Keywords:** Enterprise Modeling, modeling practitioner, competence profile.

## 1 Introduction

Enterprise Modeling (EM) is a process where an integrated and negotiated model describing different aspects of an enterprise is created for the *purposes* of (1) developing the business, (2) ensuring the quality of business operations, and (3) using EM as a problem solving tool [1]. In [1 and 2] we have argued that EM usage is heavily influenced by a large number of situational factors, one of which is the intention behind its use. Knowledge about the purpose of a particular EM endeavor is essential when making decisions about which modeling language, way of working, tool support etc. is appropriate. It is important to bear in mind that organizations do not use EM methods only for the sake of using methods. They want to solve a particular business problem and EM is only one of several instruments in that problem solving process.

In [2] we have analyzed what requirements the different purposes of EM put on the EM project in terms of input models and documentation, models to be developed, EM language, EM process, EM tool, as well as enterprise model quality. EM usually is organized in the form of a project or it is a part of a larger, e.g. organizational or information system (IS) development, project.

Much of the success of projects using Enterprise Modeling depends more on the quality of the process of modeling rather than on the method used. One important

influence on the quality of the modeling process is the competency level of the experts responsible for the EM approach. In [3] we defined a set of *core competencies* for the modeling practitioner and related these competencies to a detailed stereotype EM project process. However, since each EM project is specific depending on the purpose of modeling, this paper analyses how the purpose of EM influences competency requirements. Hence, we formulated the following research questions:

1. What are the relationships between the identified core competencies and the purposes of EM?
2. How can the core competencies be refined, if at all, based on these relationships?

The research focuses only on an enterprise modeling practitioner who is responsible for carrying out part of or the whole EM project process towards effectively achieving its goals.

The remainder of the paper is organized as follows. Section 2 discusses the concept of EM competency. In section 3 the research approach is presented. The main purposes of EM are outlined in section 4 while section 5 discusses the core competencies of the EM practitioner. In section 6 the core competencies are analyzed in relation to the purposes of EM and a refinement of the competencies is proposed. The paper ends with a discussion (section 7), related work (section 8), and some concluding remarks in section 9.

## 2 Competency – A Critical Resource to Achieve the Goals of EM

The concept of competency is complex and can be defined in a number of ways. In [3] we argue that for the purposes of analyzing competency issues in EM we have to concentrate on four main aspects:

- 1) *Knowledge* - a person's factual knowledge about a specific subject matter, as a result of e.g. education.
- 2) *Skills* - a person's ability to actually use the knowledge to achieve goals.
- 3) *Individual properties* - a wide range of personal characteristics e.g. ability to communicate, intelligence, flexibility, integrity, ability to co-operate, courage etc.
- 4) *Willingness to contribute competency* - a person's attitude towards actually contributing her/his knowledge and skills to the achievement of goals other than her/his own.

In this paper we particularly target the competency of the modeling practitioner. In the following we give an overview of the key roles in EM.

*Domain experts* provide knowledge about different aspects of some organization in its current or perceived future state. They are responsible for the correctness and relevance of that knowledge in the context of the EM project. Most EM projects also have a *project or problem "owner"* – someone who is in charge of the problem domain, allocates resources, and has the authority to implement the decisions and designs developed by the EM project.

Creation of the Enterprise Models using a participative approach needs to be supported by a *modeling practitioner*. The professional EM practitioner may take on a number of sub-roles, e.g. EM project leader, facilitator of a modeling session, and tool expert. As *project leader*, the modeling practitioner negotiates the goals for the modeling project and plans the modeling process together with the project or problem owner. A *facilitator* moderates each modeling session. In a session there can be more than one facilitator and also a *tool expert*. A larger modeling project will typically have several facilitators and tool experts forming a *modeling practitioner team*, which is headed by project leader. The team leader should be an experienced facilitator.

[4] summarizes that the main responsibility of the modeling practitioner is that (1) the models produced have good enough quality to accomplish the project goals; (2) that the chosen EM method is suitable for modeling the problem at hand and that (3) the method is effectively used to accomplish the project goals. This means not only to use the method's notation in a reasonable way but to also construct and to run a modeling process that makes the best of available resources, e.g. the knowledge and abilities of domain experts. The modeling practitioner is also responsible for making sure that the project resources are used in a way that enables the modeling project to be completed on time and in such a way that the goals are achieved.

The main challenge of the EM process is to ensure that the quality of its outcome is fit for the intended use. Potential outcomes of EM are, e.g., models, decisions and enhanced knowledge among those involved in the EM process. Since the EM process is highly intellectual, it is dependent on the competency of its participants. If the proper EM competency is not available in the EM process, the effects of EM will not appear. In this paper we focus on the competency of the EM practitioner.

### 3 Research Approach

The research contribution of this paper is based on a number of research efforts carried out since beginning of the 1990-ies and their resulting publications:

- Development of the Enterprise Knowledge Development (EKD) EM method [5],
- Extensive field work applying versions of EKD to a variety of problems,
- Interview studies involving experienced EM consultants and method developers.

The most influential application cases were, for the most part, carried out within international research projects financed by the European Commission. An overview of the cases is given in Table 1. The applications that contributed to this paper took place in the years 1993-2008. Their processes and their outcome were observed and analyzed. Collected data and experiences from method development, field work and interviews were analyzed. Two interview studies focusing on the intentional and situational factors that influence participatory EM and EM tool usage [1, 4, 7] were also carried out. In addition, EKD and its earlier versions have also been used in a number of smaller problem solving and organizational design cases at organizations such as e.g. Strömma AB (Sweden), Ericsson (Sweden), Livani District (Latvia), Riga Technical University (Latvia), University of Skövde (Sweden) and RRC College (Latvia).



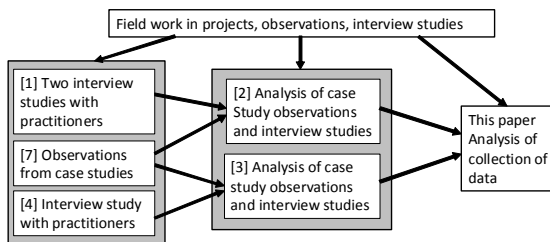
**Table 1.** Overview of main application cases

Organization	Domain	Period	Problems addressed
British Aerospace, UK	Aircraft development and production	1992-1994	Requirements Engineering
Telia AB, Sweden	Telecommunications industry	1996	Requirements validation Project definition
Volvo Cars AB, Sweden	Car manufacturing	1994-1997	Requirements engineering
Vattenfall AB, Sweden	Electrical power industry	1996-1999	Change management, Process development, Competence management
Riga City Council, Latvia	Public administration	2001-2003	Development of vision and supporting processes for knowledge management
Verbundplan GmbH, Austria	Electrical power industry	2001-2003	Development of vision and supporting processes for knowledge management
Skaraborg Hospital, Sweden	Health care	2004-2007	Capturing knowledge assets and development of a knowledge map of a knowledge repository.
SYSteam AB, Sweden	Management consulting	2008	Development of a vision for an employee knowledge management portal

Empirical data from modeling activities of the above mentioned types were documented as written notes and analyzed. Interviews with practitioners were transcribed and analyzed using Grounded Theory [6] data analysis. The data and analyses have then been used as input for a series of argumentative syntheses targeting:

1. Requirements on EM related to the purposes of using EM, reported in [2],
2. Core competencies of an EM practitioner, reported in [3], and
3. The relationship between core competencies and the purposes of EM, *reported in this paper.*

The goal of this series of analyses is to establish a line of research that addresses different aspects of EM from a purpose and situational perspective. Figure 1 illustrates the research process and resulting publications that lead up to this paper.

**Fig. 1.** The research process leading to this paper

## 4 Purpose of EM Projects

In [1 and 2] we have argued that EM projects usually have the following purposes:

- *To develop the business.* This entails, e.g., developing business vision, strategies, redesigning business operations, developing the supporting information systems, etc. Business development is one of the most common purposes of EM. It frequently involves change management – determining how to achieve visions and objectives from the current state in organizations. Business process orientation is a specific case of business development – the organization wants to restructure/redesign its business operations.
- *To ensure the quality of the business operations.* This purpose primarily focuses on two issues: 1) sharing the knowledge about the business, its vision, and the way it operates, and 2) ensuring the acceptance of business decisions through committing the stakeholders to the decisions made. A motivation to adopt EM is to ensure the quality of operations. Two important success factors for ensuring quality are that stakeholders understand the business and that they accept/are committed to business decisions. Recently, organizations have taken an increased interest in Knowledge Management (KM), which concerns creating, maintaining and disseminating organizational knowledge between stakeholders. Sharing business knowledge becomes instrumental when organizations merge or collaborate in carrying out a business process. One aspect of this is terminology. EM has a role to play here as it aims to create a multifaceted “map” of the business as a common platform for communicating between stakeholders. One KM perspective is keeping employees informed with regard to how the business is carried out. Most modern organizations consider that the commitment of stakeholders to carry out business decisions is a critical success factor for achieving high quality business operations. Differences in opinion about the business must hence be resolved, requiring that communication between stakeholders be stimulated. EM, particularly using a participative approach, can be effective to obtain such commitment.
- *To use EM as a problem solving tool.* EM is here only used for supporting the discussion among a group of stakeholders trying to analyze a specific problem at hand. In some cases making an EM activity is helpful when capturing, delimiting, and analyzing the initial problem situation and deciding on a course of action. In such cases EM is mostly used as a problem solving and communication tool. The enterprise model created during this type of modeling is used for documenting the discussion and the decisions made. The main characteristics of this purpose are that the company does not intend to use the models for further development work and that the modeling activity has been planned to be only a single iteration. In some cases this situation changes into one of the other EM purposes because the organization sees EM as beneficial or the problem turns out to be more complex than initially thought and more effort is needed for its solution.

In [3] we analyze the different purposes in order to formulate requirements for: input models and documentation needed for the project, enterprise models to be developed, EM language needed to document the modeling product, the EM process, EM tool to support the project, as well as factors of model quality that the project should consider. These requirements are described in Table 2 in relation to the different EM purposes.

**Table 2.** Requirements on EM [3]

Purpose of EM	Input models and documentation	Models to be developed	EM language requirements	EM process requirements	EM tool requirements	Model quality requirements
<i>Develop the business</i>						
Develop visions and strategy	Existing models and other business "blueprints"	Business oriented models, e.g. Goal Model (GM), Concept Model (CM), Business Proc. Model (BPM), Actor Model (AM) ,inter-model links	Notation that domain stakeholders understand	Participatory	Plastic wall, simple documenting tools	Understandability, correctness, simplicity, flexibility
Design/ Redesign the business	Vision and strategy models and other kinds of business "blueprints"	Business oriented models, e.g. as above as well as inter-model links	Established notation that domain stakeholders understand	Participatory involving multiple stakeholder groups	Plastic wall, EM tools that makes it possible to seamlessly move to requirements analysis and IS design	Completeness, correctness, flexibility, integration, understandability, usability
Develop IS	Business oriented models	IS architecture models as well as links with business oriented models	Enough formality and precision to allow modeling of complex facts	Partly participatory and partly analyst driven	Plastic wall, EM tools or CASE tools depending on the development approach	Completeness, correctness, flexibility, integration, usability
<i>Ensure the quality of business operations</i>						
Ensure acceptance for business decisions	Various types of business "blueprints" (e.g. Balanced Scorecard)	Business oriented models (GM, CM, BPM, ARM, BRM) as well as inter-model links	Notation that domain stakeholders understand	Participatory involving knowledge bearers and users	Plastic wall, simple tools, tools for presentation of models to a wider audience (e.g. web-based tools)	Completeness, correctness, integration, simplicity, understandability, usability
Maintain and share knowledge about the business	Business models (GM, CM, BPM, ARM, BRM), inter-model links	"Cleaned" models that make sense to a wider audience	Simple and intuitive modeling language	Partly participatory, partly analyst driven	EM tools with web interface	Correctness, integration, understandability, usability
<i>Use EM as a business problem solving tool</i>						
To analyze and solve a specific problem or task	Initial problem statement and other relevant documentation	Business oriented models (GM, CM, BPM, ARM, BRM) & inter-model links	Notation that domain stakeholders understand	Participatory involving multiple stakeholder groups	Plastic wall, simple documenting tools	Correctness, flexibility, understandability

## 5 Core Competencies in Enterprise Modeling Projects

Previous research [4] has identified three levels of EM practitioner competency:

- *Ability to model*, more specifically, to construct an Enterprise Model which is syntactically correct according to the used EM language and in a reasonable way reflects the domain and problem in question.
- *Ability to facilitate modeling sessions* – being able to lead a group of domain experts in creating/refining an Enterprise Model and doing it in such a way that the group’s knowledge and abilities work together to create a high quality model.
- *Ability to lead EM projects* towards fulfilling their goals and making the best of the project resources.

We claim that in order to target the main challenge of the EM process, which is to produce an EM outcome that is fit for its intended purpose, we need to consider a set of essential core competencies. In [3] we have analyzed a generic EM process in order to identify essential competencies related to its activities. The process contained the main activities depicted in Table 3.

**Table 3.** Activities in EM (adapted from [3])

---

Define scope and objectives of the modeling project
Plan for project activities and resources
Plan for modeling session
Gather and analyze background information
Interview modeling participants
Prepare modeling session
Conduct modeling session
Write meeting minutes
Analyze and refine models
Present the results to stakeholders

The identified competencies were found to fall into two categories:

- 1) *Competencies related to modeling*
  - ability to model, including assessing and improving model quality according to the EM purpose and
  - ability to facilitate participatory modeling sessions.
- 2) *Competencies related to setting up and managing EM projects*
  - ability to select an appropriate EM approach and tailor it in order to fit the situation at hand
  - ability to interview involved domain experts
  - ability to define a relevant problem
  - ability to define requirements on the results
  - ability to establish a modeling project
  - to adjust a presentation of project results and issues related to them to various stakeholders
  - ability to navigate between the wishes of various stakeholders while upholding the EM project goal
  - ability to assess the impact of the modeling result and the modeling process in the organization

In the following, these core competencies will form the baseline for discussing more specific competencies that we claim are necessary to fulfill the different purposes of EM. Note that the competences only relate to the purposes described in Section 4 and EM activities listed in table 3.

## 6 Purpose Driven Refinement of Competencies in EM Projects

EM practitioners need the core abilities to model and to facilitate modeling sessions to effectively fulfill any of the EM purposes. Hence we will only discuss these abilities further *if an additional sub-set of abilities are specifically needed*. The same holds also for competencies related to setting up and managing EM projects. In the following sections we will discuss each of the EM purposes with respect to the more specific competency needed to fulfill this purpose.

At this stage of research we cannot claim to be complete in any way. Rather we see this as another step towards fully developed competency profiles for EM practitioners at different levels of experience and skill.

### 6.1 Develop Visions and Strategy

*Ability to model, including assessing and improving model quality according to the EM purpose.* In addition to the core abilities to model and to facilitate modeling sessions this EM purpose also requires specific modeling abilities to model on a high level of abstraction where initially the enterprise model is not internally connected and may appear to be consisting of “small islands”. The main challenge here is to guide the modeling work towards a certain direction. This might be hard because a part of strategy development is to allow the group to explore different options to a certain degree. The facilitator, however, needs to have the ability to see a certain “path” in the models and steer the modeling participants from drifting off course, i.e. discussing peripheral problems and defining goals that are plainly unrealistic. This situation requires the practitioner to deal with a large degree of uncertainty while demonstrating confidence to the participants in the group.

Regarding competencies related to setting up and managing modeling projects the following specifics should be paid attention to:

*Ability to select an appropriate EM approach and tailor it in order to fit the situation at hand.* For strategy development there exist a number of suitable development approaches that the modeling practitioner needs to be reasonably knowledgeable about. The modeling participants might often be familiar with a specific modeling language and notation. In this case this language should be preferred instead of introducing a new one. Importing a strategy development approach into EM in most cases implies defining new “inter-model” links with the existing components in the enterprise model and/or defining synonyms among modeling components. This requires deep knowledge about the meta-models and intentions of the involved methods.

*Ability to define a relevant problem.* The ability to define a relevant problem goes hand in hand with the ability to facilitate. The problem might be defined relatively

vaguely, e.g. to find the real problem. Nevertheless the EM practitioner should have the ability to define at least general boundaries for it.

*Ability to navigate between the wishes of various stakeholders while upholding the EM project goal.* Deciding on the direction of an organization influences an organization more than short-term decisions. Therefore, the risk of having to deal with various hidden agendas from stakeholders is imminent. Also, these stakeholders often have an influential position in the organization. This means that the modeling practitioner needs to be listening and diplomatic while demonstrating that she/he is the person in charge of the EM project. Taking this role requires experience and knowledge about how organizational cultures function as well as patience, an agreeable personality and a firm but pedagogical way of communicating.

*Ability to assess the impact of the modeling result and the modeling process in the organization.* Being able to assess the impact of a vision or a strategy requires some experience from both successful and unsuccessful processes of implementing strategies, since the degree of uncertainty can be quite high.

## **6.2 Design/Redesign the Business**

*Ability to model, including assessing and improving model quality according to the EM purpose.* To fulfill this purpose, the EM practitioner should be able to assess that the models have enough quality to be possible to implement under the conditions that exist or will exist in the business at hand. This requires some previous experience from being involved in e.g. implementing processes in an organization. One important aspect here is to be able to assess not only the practical implications of change but also the cultural implication of change.

*Ability to facilitate participatory modeling sessions.* When designing/redesigning the business one of the main challenges is to avoid polishing the current way of thinking and working. The EM practitioner should be able to support the creativity of the group while maintaining upholding a critical view on the resulting models.

Specific competencies related to setting up and managing EM projects are as follows:

*Ability to interview involved domain experts.* Following the previous discussion, the EM practitioner should be able to assess the ability of potential modeling participants to think creatively and out of the box and to make sure that some people in the group are also “critical thinkers”.

*Ability to define requirements on the results.* Often projects like this requires that different types of models are developed. The EM practitioner should, therefore, be able to define how the whole project sticks together and how each model contributes to bigger picture. Changes in the target area of the project could also influence other areas of the business. This means that the requirements on the result must be related to an even bigger picture that constitutes the surrounding organization and sometimes partner organizations. Requirements must be defined such that the result can be implemented afterwards. Putting this complicated “puzzle” together requires a high level of experience and skill from the EM practitioner.

*Ability to establish a modeling project.* An important aspect of establishing an EM project with this purpose is to negotiate authority for change. Both the EM project leader (EM practitioner) and the modeling group/s involved must feel that they are authorized to make design decisions that take into account the goals and constraints of the project. This means that the EM practitioner must be able to identify which authorities are needed, identify the involved decision-makers and negotiate the proper authorities. Sometimes it happens that decision-makers go back on what they have approved, and then the EM practitioner will need to be able to either negotiate maintained authority or re-define the scope of the project.

*Ability to navigate between the wishes of various stakeholders while upholding the EM project goal.* All processes that involve change will cause different kinds of resistance in the organization. Therefore, the risk of having to deal with various hidden agendas from stakeholders is imminent. This means that the modeling practitioner needs to be listening and diplomatic while demonstrating that she/he is the person in charge of the EM project. Taking this role requires experience and knowledge about how organizational cultures function as well as patience, an agreeable personality and a firm but pedagogical way of communicating.

### 6.3 Develop Information Systems

*Ability to model.* The specifics in relation to this purpose primarily focus on assessing and improving model quality according to the EM purpose of developing an IS. More specifically, the enterprise model should be created in such a way that it is possible to implement in a system. This might require increasing model formality. In this context the EM practitioner should understand the use of the models in the IS development project, e.g. how Concepts Model can be used as input for developing a database schema. Ultimately, the EM practitioner should have some experience from IS development projects, preferably from an operative point of view.

*Ability to facilitate participatory modeling sessions.* This purpose requires the ability to guide the modeling effort in such a way that a balance between the IT and business aspects is ensured. If one of the aspects dominates the other, the resulting solution risks being unsuitable for the organization.

Regarding competencies related to setting up and managing EM projects the following specifics should be paid attention to:

*Ability to select an appropriate EM approach and tailor it in order to fit or be docked to the IS development process and its specific development methodology and (often) tools.* In principle many modeling languages and tools can be used for this purpose and most often modeling language of one sub-model in the enterprise model can be replaced with another modeling language as long as the inter-model links remain in tact. E.g. EKD Goals Model notation in principle can be replaced with i\* notation [8], and Concepts Model changed to UML Class Diagram [9]. The modeling practitioner should be able to perform such a method engineering task.

*Ability to define requirements on the results.* Modeling practitioner should be able to assess how are models used in the IS development process and when is the model complete enough to proceed to IS development activities.

*Ability to establish a modeling project.* This EM purpose means that EM is used in a IS development project and perhaps not seen as a project in itself, but rather a set of intertwined activities. The modeling practitioner should be able understand the IS development methodology and design EM steps in a fitting way.

#### **6.4 Ensure Acceptance of Business Decisions**

For the EM purpose the main success criteria is that the decisions made during modeling and reflected in the model are accepted and taken up by the organization.

*Ability to facilitate participatory modeling sessions.* The facilitator should make sure that the group reaches consensus and real decisions that can be implemented in the organization. Furthermore, the facilitator should also make sure that the participants perceive the enterprise model as documentation of the decisions and the decisions as real allocated to real people for implementation.

Concerning competencies related to setting up and managing EM projects, the following are of relevance:

*Ability to select an appropriate EM approach and tailor it in order to fit the situation at hand.* All kinds of modeling tricks can be necessary, including breaking the methodology rules, modeling notation and using unconventional approaches. The modeling practitioner should be able to assess the impact of such actions on the modeling results and the process.

*Ability to interview involved domain experts.* This EM purpose might be associated with hidden agendas that need to be uncovered. Finding out how decisions are made in the organization and how they are implemented also is crucial. The modeling practitioner should have good listening skills because these issues can seldom be addressed with a straight question.

*Ability to define requirements on the results.* In this case the acceptance and the consensus could be seen more important than the model quality. Hence, it might sometimes be purposeful accepting low model quality if the group agrees with the decisions made.

*Ability to navigate between the wishes of various stakeholders while upholding the EM project goal.* This is particularly important because the project usually covers broader group of stakeholders than is possible to involve in modeling directly.

#### **6.5 Maintain and Share Knowledge about the Business**

For this purpose it is important to understand that sometimes existing models are used as input and sometimes models are created for the purpose of depicting how the business is carried out. In the latter case the modeling is about capturing the current state of affairs and ways of working.

*Ability to model, including assessing and improving model quality according to the EM purpose.* The models that are created for this purpose have a specific target, which is to convey a message and also to instruct a diverse group of stakeholders.



This means that the understandability of models is essential, which in this case is a critical aspect of model quality together with correctness. For the EM practitioner this means that the ability to become knowledgeable about the characteristics and needs of the target groups is important. Listening and communication skills are essential here.

The following specifics should be paid attention to concerning competencies related to setting up and managing EM projects:

*Ability to select an appropriate EM approach and tailor it in order to fit the situation at hand.* In this case, the modeling language should either be familiar to the stakeholders or be simple enough to understand without prior knowledge of the language. The EM practitioner should have enough knowledge about modeling languages to be able to balance quality aspects such as correctness of the models in terms of state of affairs of the business with the quality aspect of understandability.

*Ability to define requirements on the results.* This ability relates, again, to the understandability of models. Also, in order to be able to properly define the requirements, the EM practitioner should have some knowledge relating to organizational learning. This is further discussed under the ability to assess the impact of the modeling result below. One important aspect of this ability is setting up for maintenance of models, because the business changes and the models should change accordingly. This means that the EM practitioners needs to be knowledgeable about different tools that can be used to document and maintain models as well as be able to set up a feasible maintenance process. More on what is needed to keep models “live” is available in [7 and 10] and on using Active Knowledge Models in organizations see [11].

*Ability to adjust a presentation of project results and issues related to them to various stakeholders.* Presentation of the resulting models is one of the key issues for this purpose. Sometimes the models themselves may not be the best way to present the organizational knowledge and some alternative ways of presentation must be devised. Providing background information and connecting models to this can be one potential approach. Another approach could be to develop simulations based on the models. The ability to understand the conditions under which the models/information makes sense to the intended stakeholder groups is essential here. Pedagogical knowledge is also helpful, together with knowledge about how different media can enhance the message that is being conveyed

*Ability to assess the impact of the modeling result and the modeling process in the organization.* The ultimate desired impact of fulfilling this purpose is that organizational learning is created. To lead a modeling project with this purpose the EM practitioner should preferably have some knowledge and experience from fields that address organizational learning, such as e.g. Knowledge Management.

## **6.6 Use EM to Analyze and Solve a Specific Business Problem**

EM projects with this purpose are usually quite short and compact in time, e.g. they should be done within a week. More about what happens in early phases of EM is available in [12]. For this purpose the modeling competencies do not have specific areas or concerns, but competencies related to setting up and managing EM projects have the following specific issues to consider:

*Ability to select an appropriate EM approach and tailor it in order to fit the situation at hand.* In this case simple EM languages and notations should be preferred because there will be no time to familiarize the participants with the modeling approach. Also, using the approaches that the organization already has should be preferred. The EM practitioner should be able to find out what the organization has in terms of existing approaches and assess how it can be used in an EM project.

*Ability to interview involved domain experts.* Interviewing needs in this context to follow a tight schedule, since time for planning is usually very restricted. The interviews should, however, not be done in a haphazard way, i.e. the EM practitioner should be able to follow a predefined interview script and keep the schedule.

*Ability to define a relevant problem.* Even in short projects such as these the problems put forward by the domain experts can be quite large and many. The modeling practitioner should be able to find the relevant problem/s in a cluster of problems.

*Ability to define requirements on the results.* In this situation there are often very strict constraints in terms of time and other resources. E.g., there may only be time for one modeling session. This requires the modeling practitioner to be realistic in terms of what can be achieved and to communicate this to the problem owner, especially in cases, where the resource constraints put at risk achieving the expected result.

*Ability to assess the impact of the modeling result and the modeling process in the organization.* Since the EM project happens so quickly in the organization the difficulty is to make sure that someone will actually take up the result and implement it. This needs to be addressed in the project negotiation and planning phase, for example, by inviting modeling participants that can support implementation.

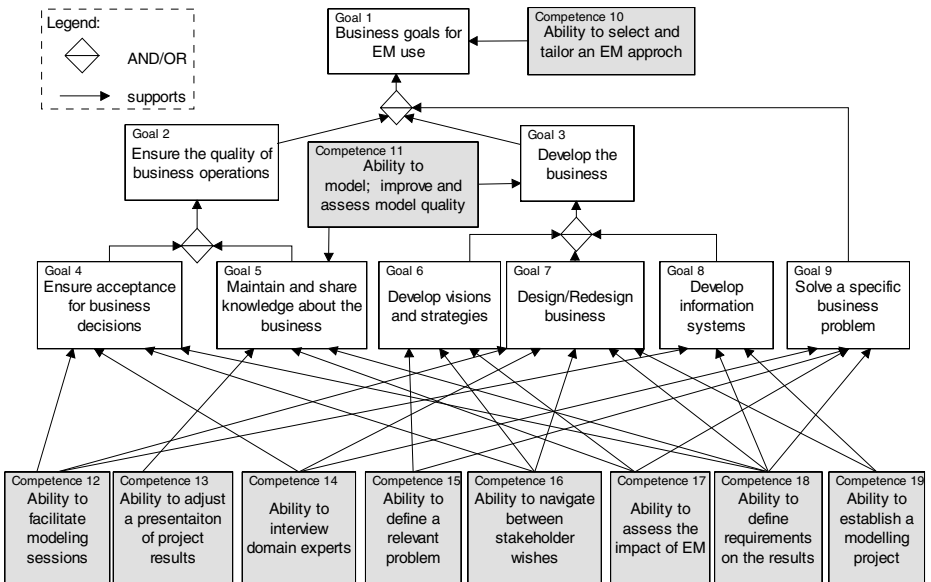


Fig. 2. EM goals aligned with supporting competencies (shaded)

## 6.7 Discussion

Relationships between the EM goals and competencies discussed in the previous section are shown in figure 2. In the paper we have only focused on an EM practitioner who is responsible for managing part of or the whole EM project process towards effectively achieving its goals. Looking at the refined result we see that a few of the core competencies support more or less all purposes in a specific way:

- Ability to model, including assessing and improving model quality according to the EM purpose.
- Ability to select an appropriate EM approach and tailor it to fit the situation at hand.
- Ability to define requirements on the results
- Ability to assess the impact of the modeling result and the modeling process in the organization.
- Ability to navigate between the wishes of various stakeholders while upholding the EM project goal.

## 7 Related Work

The current state of the art is that scholars in EM formulate a set of guidelines and various dos and don'ts for conducting the EM or business modeling process (c.f., for example, 11, 13 and 14). Most of these recommendations are centered on the modeling activities and relatively few guidelines address the preparatory stages of the modeling process. The skills formulated in this paper are needed in order to successfully apply the guidelines given in the literature,

Beyond EM and Requirements Engineering there also are other relevant areas that need to be considered when planning for EM projects such as general competences and soft skills of project management, competences specific for IT projects (c.f. e.g. 15 and 16) and consulting (see [17]). [18] presents a set of general characteristics of agile modelers which can also be seen as relevant skill-sets to EM projects.

Facilitation is a general technique used in group processes for a wide variety of purposes not only in EM (see further e.g. [19] and International Association for Facilitators (IAF) <http://www.iaf-world.org>). This ability is very much based on knowledge about the effects of modeling, the principles of human communication and socialization (especially in groups), as well as the conditions of human learning and problem solving (cognition).

In this paper we have been inspired by modeling of organizational tasks related to their necessary competencies that we performed during the MAPPER project [12]. However a more ontology driven approach as, for instance, presented in [20] should be used in the future to develop more fine-grained competency indicators and values.

## 8 Concluding Remarks and Future Work

The area of EM is to a large extent practice driven. Scientific theories targeting the practice of EM are lacking. This paper, together with [2] and [3], are exploratory

attempts to provide building blocks for theories addressing the preconditions of successful EM.

The objective of this paper is to further refine a set of previously defined core competencies for the EM practitioner. The refinement is based on the competency needs that are related to different purposes of EM.

As a limitation we would like to point out that the current set of competences only address the EM process to be carried out for different purposes. Hence it should be seen as a subset of competences for business consultants, requirements analysis, and project managers. Furthermore, the competences are derived from applying EM in a participatory way in an organizational context. Also, the process of collecting and analyzing data from a large number of sources could potentially be a threat to validity. We have tried to minimize this risk by being clear about how intermediate results were synthesized. The current competence profile should be seen as a proposal to be tested and validated in future research and practice.

The refinements emphasize that successful management of modeling projects, large or small, is heavily dependent on experience, not only from modeling but also from a broader set of areas such as implementation of strategies, change management, and systems development. In earlier papers we have concluded that this is not something for the novice. Hence, there needs to be at least two competency profiles for the EM practitioner: 1) The beginner EM practitioner and 2) the expert EM practitioner. We also believe that these profiles need to define the kind of roles that these two profiles can take on, both in the actual modeling and in the management of EM projects. The definition of these two profiles is needed to construct effective training, both at university level as well as in-practice training/mentoring to become an expert EM practitioner. One argument for this is that it is difficult to know which competency should be developed at which stage of maturity in the EM expert. E.g., confusing the novice with issues that only a more experienced EM practitioner can understand is not particularly effective. Looking at EM education and training, particularly in the university context, it is mostly focused on the ability to model. Often it is assumed that with that ability comes, implicitly, the ability to facilitate modeling sessions and particularly to manage EM projects. Based on our research we find that this is clearly not the case.

Finally, the fact that the issue of implementing the EM result is evident in several cases also emphasizes that an important characteristic of successful EM is that the modeling result is actually used.

## References

1. Persson, A., Stirna, J.: An explorative study into the influence of business goals on the practical use of Enterprise Modelling methods and tools. In: Tenth International Conference on Information Systems Development (ISD 2001), Royal Holloway, University of London, September 5-7 (2001)
2. Bubenko Jr., J.A., Persson, A., Stirna, J.: An Intentional Perspective on Enterprise Modeling. In: Salinesi, C., Nurcan, S., Souveyet, C., Ralyté, J. (eds.) *An Intentional Perspective on Enterprise Modeling*. Springer (2010) ISBN 978-3-642-12543-0

3. Persson, A., Stirna, J.: Towards Defining a Competence Profile for the Enterprise Modeling Practitioner. In: van Bommel, P., Hoppenbrouwers, S., Overbeek, S., Proper, E., Barjis, J. (eds.) PoEM 2010. LNBIP, vol. 68, pp. 232–245. Springer, Heidelberg (2010) ISBN 978-3-642-16781-2
4. Persson, A.: The Practice of Participatory Enterprise Modelling – a Competency Perspective. In: Johannesson, P., Söderström, E. (eds.) Information Systems Engineering - from Data Analysis to Process Networks, pp. 129–157. Idea Group Inc. (2008) ISBN-13: 978-1-59904-567-2
5. Bubenko Jr., J.A., Persson, A., Stirna, J.: User Guide of the Knowledge Management Approach Using Enterprise Knowledge Patterns. Deliverable D3, IST Programme project Hypermedia and Pattern Based Knowledge Management for Smart Organisations, project no. IST-2000-28401, Royal Institute of Technology, Sweden (2001)
6. Glaser, B.G., Strauss, A.L.: The Discovery of Grounded Theory: Strategies for Qualitative Research. Weidenfeld and Nicolson, London (1967)
7. Stirna, J., Persson, A.: An Enterprise Modeling Approach to Support Creativity and Quality in Information Systems and Business Development. In: Halpin, T., Krogstie, J., Proper, E. (eds.) Innovations in Information Systems Modeling: Methods and Best Practices. IGI Global (2008) ISBN 978-1-60566-278-7
8. Yu, E.S.K., Mylopoulos, J.: From E-R to “A-R” - Modelling Strategic Actor Relationships for Business Process Reengineering. In: Loucopoulos, P. (ed.) ER 1994. LNCS, vol. 881, pp. 548–565. Springer, Heidelberg (1994)
9. Object Management Group (OMG), Unified Modeling Language (UML) 2.0 (2005)
10. Wesenberg, H.: Enterprise Modeling in an Agile World. In: Johannesson, P., Krogstie, J., Opdahl, A.L. (eds.) PoEM 2011. LNBIP, vol. 92, pp. 126–130. Springer, Heidelberg (2011)
11. Lillehagen, F., Krogstie, J.: Active Knowledge Modeling of Enterprises. Springer (2008) ISBN: 978-3-540-79415-8
12. Sandkuhl, K., Lillehagen, F.M.: The Early Phases of Enterprise Knowledge Modelling: Practices and Experiences from Scaffolding and Scoping. In: Stirna, J., Persson, A. (eds.) PoEM 2008. LNBIP, vol. 15, pp. 1–14. Springer, Heidelberg (2008) ISBN 978-3-540-89217-5
13. Bridgeland, D.M., Zahavi, R.: Business Modeling: A Practical Guide to Realizing Business Value. Morgan Kaufmann (2009) ISBN: 978-0-12-374151-6
14. Lankhorst, M., et al.: Enterprise Architecture at Work: Modelling, Communication and Analysis, 2nd edn. Springer (2009) ISBN 978-3-642-01309-6
15. Suikki, R., Tromstedt, R., Haapasalo, H.: Project management competence development framework in turbulent business environment. *Technovation* 26, 723–738 (2006)
16. Gillard, S.: Soft Skills and Technical Expertise of Effective Project Managers. *Issues in Informing Science and Information Technology* 6 (2009)
17. Kaarst-Brown, M.L.: Five symbolic roles of the external consultant – Integrating change, power and symbolism. *Journal of Organizational Change Management* 12(6), 540–561 (1999)
18. Ambler, S.: Agile modeling: Effective practices for extreme programming and the unified process, 1st edn. John Wiley & Sons Inc. (2002) ISBN 978-0471202820
19. Zavala, A., Hass, B.H.: The Art and Power of Facilitation: Running Powerful Meetings, Management Concepts, Inc., Vienna, USA (2008) ISBN 9781567262124
20. Albertsen, T., Sandkuhl, K., Seigerroth, U., Tarasov, V.: The Practice of Competence Modelling. In: van Bommel, P., Hoppenbrouwers, S., Overbeek, S., Proper, E., Barjis, J. (eds.) PoEM 2010. LNBIP, vol. 68, pp. 106–120. Springer, Heidelberg (2010)

# Work Experience in PAIS – Concepts, Measurements and Potentials

Sonja Kabicher-Fuchs and Stefanie Rinderle-Ma

University of Vienna  
Faculty of Computer Science  
1010 Vienna, Austria

{sonja.kabicher-fuchs,stefanie.rinderle-ma}@univie.ac.at

**Abstract.** Process-Aware Information Systems consider various characteristics of resources, such as capabilities, as a driver for allocating task to humans. Work experience has been discussed as a possible variable of history-based allocation. However, work experience has been considered in a limited extend, reducing the perspective on measurements to single aspects such as years of working in an organization, ore amount of performed tasks. Further, the allocation has mainly been oriented towards the best possible fit of humans to the requirements of the task and the process. This contribution is a first step towards an human-centric work experience allocation. It concentrates on the question how experience collected by individuals working with business processes may be measured in PAIS. A collection of work experience measurements at various organizational levels is provided. The measurement collection resulted from a literature review of PAIS theory, selected psychological literature and an qualitative analysis of job offers.

**Keywords:** Human-centric allocation, work experiences, experience-based allocation.

## 1 Introduction

Process-Aware Information Systems (PAIS), offer various advantages to organizations such as increased quality of output, shorter cycle times and faster feedback. However, PAIS have received criticism as well, mainly referring to their potential to support a too rigid or mechanistic work. Over the last years, there have been several works addressing various aspects of human orientation in PAIS, such as the interaction of automated and human workflows, like BPEL4People [2][4], human interactions [33][14], flexibility in workflow enactment [9], strategic resource allocation for longer-term skill acquisition and diversification in organizations [1], understandability of business processes [20][19], and the human-oriented tuning of functionalities in such systems [38], which contributed toward the growing attention on humans and the effects on their performance, satisfaction, and motivation by working with such systems.

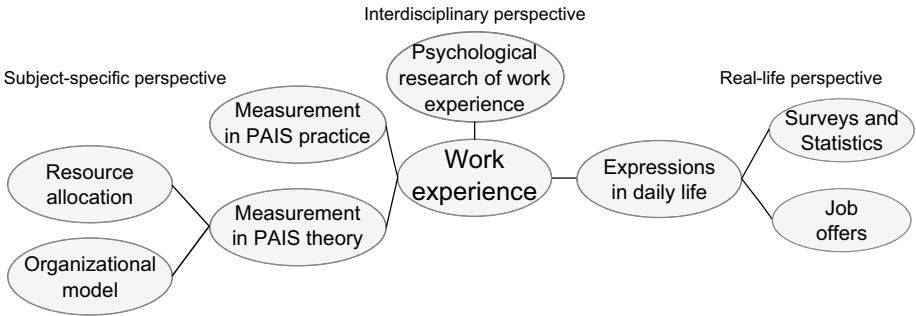
Human work experience has been considered particularly in capability-, and history-based allocation as one of the attributes according to which the assignment of humans to tasks were guided. However, work experience has mainly been considered as a simple count or measurement in PAIS so far, such as the indication of previous work experience in the field in years (capability) or amount of execution tasks (history). However, work experience is more complex than simple measures such as ‘more executions = more experience’ [39]. In this work we aim to support an understanding of work experience as a complex, multidimensional construct. Our study comprises a literature review of PAIS theory led by the question how resources in PAIS are described, an analysis of job offers that should provide an understanding of the common practice of expressing and measuring work experience in daily life, and an insight into the construct work experience as illustrated in selected psychological theory. Further, this contribution aims at discussing the potentials of work experience as the driver for human-centric allocation in PAIS.

## 2 Classification and Literature Study

An optimal performance of processes and their tasks is essential for the organizations’ competitiveness, however it has been shown that job satisfaction and experience might increase performance as well. Our goal is to support a shift of perspective on the allocation in PAIS from a process-oriented to a human-centric one. In a *process-centric view*, task allocation is oriented according to the best possible fit of humans to task requirements. In a *human-centric view* the center of interest is on humans working with a PAIS. Allocation is led by humans’ development and considers individuals’ goals of development in an organization and development strategies (such as specialization and generalization). A human-centric allocation should ensure at least a comparable quality of process performance as provided by an process-oriented allocation. We argue, that human-centric allocation can be strongly supported by work experience as a driver for the alignment of tasks to human requirements.

Towards a human-centric allocation we want to provide an understanding of work experience as a complex construct with various facets. The underlying study was guided by the question *whether and how can work experience of humans be measured* (a) in the field of PAIS, e.g., to use this information for the allocation of tasks to humans, (b) in *work psychology* to provide an interdisciplinary view on work experience, and (c) in *daily life* to provide an overview of common practice of describing work experience. Figure 1 illustrates sources that were used to find out more about work experience and its measurements. The source categorizations resulted from applying Brainstorming [23] in a small group of researchers to the topic *Where to find work experience measurements?*

*‘Work Experience’ in Process-Aware Information Systems.* A literature review was conducted based on the goal to find out *to what degree work experience of humans has been considered in PAIS theory*, for example, as a supportive



**Fig. 1.** Sources for work experience measurements as a result of Brainstorming

factor for the allocation of business process tasks to humans. As a more general question, we also wanted to find out *how individuals have been described in PAIS theory*, and even more general, *how resources have been specified in PAIS theory*.

*‘Work Experience’ in Daily Life.* The labor market in general, and job offers in particular appeared to be an appropriate source for gathering information about work experience as handled in practice. Furthermore, selected surveys and studies were used to illustrate common practice in capturing information about work experience.

*‘Work Experience’ in Psychology.* To bring perspectives from other disciplines into our field of view, *psychological research on experience*, and particularly on work experience was considered in this study.

### 2.1 ‘Work Experience’ in PAIS Literature - A Review

A literature review was conducted to capture information about how *resources* in general, and persons and their work experiences in particular are described in PAIS theory. The literature review was conducted in four phases: broad and deep search, selection of matches, coding, and analysis.

Five broad and two deep searches were conducted by means of Google Scholar using different search terms as illustrated in Figure 2. As Google Scholar sorts the hits according to their relevance, the first 100 hits of each of the broad searches 2, 4 and 5 were considered in the list of results. The broad searches resulted in a total of 443 hits, the deep search in a total of 164 hits (Information Systems: 11 hits, Data and Knowledge Engineering: 11 hits, MIS Quarterly: 7 hits, Business Process Management Journal: 32 hits, BPM: 31 hits, CAISE: 72 hits). To sum up, the broad and deep searches resulted in a total of 607 hits.

The text material was further selected according to text availability and duplicate reduction. Further, keywords were used to automatically highlight relevant text segments in the text material. Examples of keywords were *resource*, *human*, *actor*, *role*, *experience*, *capability*. The highlighted text passages offered



Search Terms of the Broad and Deep Search				
Iteration	All words	Exact phrase	With at least one of the words	Hits
Broad Search 1	Resource, allocation, capabilities, experience, workflow	Process aware information systems	Skills, competencies, attitudes	61
Broad Search 2	Organizational, resources, perspective	Process aware information systems		100/ 593
Broad Search 3	Organizational, model, workflow	Process aware information systems	Capabilities, skills, competencies, experience	82
Broad Search 4	resource allocation, organizational model	Workflow system, Workflow management system	Capabilities, skills, competencies, experience	100/279
Broad Search 5	Actor, human agent, resources, human resources, work distribution	Workflow system, Workflow management system	Capabilities, skills, competencies, experience	100/233
Deep search 1	Resource, organizational, perspective		Capabilities, skills, competencies, experience	
Deep search 2	Single search: Resource view, resource perspective, resource allocation, organizational perspective, work experience			164

**Fig. 2.** Search terms of the searches via Google Scholar

a basis to manually categorize contributions according to their relevance to the leading literature review questions. 142 contributions included text phrases relevant for the further analysis. The selected contributions<sup>1</sup> were manually coded by one researcher according to predefined rules, the categories were elaborated inductively. One broad category was *resources* in which text passages were summarized that included a description of particularly ‘humans’ as resources. This category could be subdivided into further 19 subcategories, as illustrated in Figure 3. We summarize that a resource may be of different types, e.g. human or non-human and a member of various resource classes that may be based on capabilities, such as functional requirements, or the structure of the organization. A resource may be an entity, or item, object, service, actor, agent, participant, person, subject, employee, user, individual, worker, workflow participant which is able to perform a task. A resource may have attributes, features and particularities (such as an identifier, name, description, availability/absence status, costs), one or more roles, capabilities, a position (function, job), skills, relations, experience, privileges, constraints, responsibilities, and behavior. A resource may belong to organizational units (also including e.g. branches, divisions), teams or work groups, and an organization. The task of a resource is doing work, such as performing tasks, or projects. A resource may use other (e.g. non-human) resources.

Further, we were interested in the explanations of particular terms such as *capabilities, features, competencies, skills, qualification and experience* which were partly used to describe resources, and often mentioned as being important for the composition of roles. We experienced during the analysis, that in most of the contributions the terms used were not further explained in detail, e.g., ‘*capabilities may include qualifications and skills.*’. It could occur that similar terms were used to express different meanings. Some examples will be presented in the following.

<sup>1</sup> Literature list available under <http://cs.univie.ac.at/research/projects/projekt/infproj/1026/>

Category	Subcategory	Cases	Category	Subcategory	Cases
RESOURCE_ MAY BE	OF TYPES	51	RESOURCES_ MAY BELONG TO	PRIVILEGES	3
	ENTITY	38		CONSTRAINTS	2
	MEMBER OF RESOURCE CLASSES	11		RESPONSIBILITIES	2
RESOURCE_ MAY HAVE	ATTRIBUTES, FEATURES	26		BEHAVIOR	2
	ROLE(S)	36		ORG UNITS	24
	CAPABILITIES	17	TEAM_ WORK GROUP	10	
	POSITION	11	ORGANIZATION	5	
	SKILLS	6			
	RELATIONS	5	RESOURCE TASK	DOING WORK	16
	EXPERIENCE	4	RESOURCE_ MAY USE	RESOURCE(S)	2

**Fig. 3.** Categorization of text phrases referring to resources

*Capabilities* have been used to describe both human and non-human resources. Referring to the former, capabilities were synonymously used for *attributes* (3 cases) and were understood as direct property of the resource (3 cases), and qualities that persons possess (2 cases). Capabilities were argued to be used to clarify the suitability or ability of performing work items (7 cases). Capabilities were considered as skills (8 cases), qualifications (7 cases), as well as specific responsibilities (4 cases) and previous work experience (4 cases) (both subsumed as ‘job-related or personal attributes’ (4 cases)), functional requirements (3 cases), and equipment (1 case). Capabilities might be recorded (2 cases), such as in the organizational model (1 case), and might have several metrics, such as name, type, and values (4 cases). Capabilities might be considered for work distribution (2 cases), and evaluated during runtime (1 case).

The use and the textual description for the term *feature* in order to describe resources was found in just a few contributions: ‘*which further describe specific characteristics that they [individual resources] may possess that could be of interest when allocating work items.*’ [30, p. 4] [31, p. 221].

*Competencies* seemed to be understood in two different ways: some contributions considered competencies as separated from the resource, e.g. connected to the position (such as privileges) (2 cases), whereas others considered them as attributes of the person and strongly connected to a task (2 cases). Referring to the latter, competencies have been understood as, for example, the combination of knowledge, skills and behavior utilized to improve performance (1 case). There seem to be different types of competencies, such as generic, organic and changing ones, or in-stock (previously acquired), in-use (currently put in practice), and in-making (target competencies) ones (1 case). Competencies have been considered for evaluating and monitoring persons (3 cases), and the trade-off between the transparency of competencies and user privacy has been mentioned as an emerging challenge (1 case). Competency management has been considered as support of competency identification, assessment, acquisition, and usage (1 case).

*Skills* were considered as the ability of being able to fulfill a function or role (2 cases), capability (1 case), and particular knowledge (1 case). Skills were argued to be measurable (1 case), and changing over time (e.g. due to development) (2 cases). A skill was considered as a service a person might offer, a direct property of a person (1 case), and as part of a person’s skills set (3 cases).

*Qualifications* were explained as ‘expressed as the roles they [human and non-human resources in a business] can fulfill.’ in [10, p. 51]. Qualifications were considered as direct property of the person (2 cases). [32] distinguish among capabilities (including qualification and previous work experience) and abilities to undertake certain tasks (such as *licenses held, trade certifications*) [32, p. 185]. Qualifications were described by means of a name, description, comment, and condition (1 case).

*Experience* was understood in several ways, such as *who has the most experience with this type of work item* [32, p. 187], *who has had the least numbers of failures when tackling similar tasks* [30, p. 19] [32], or was expressed as *familiarity* (how familiar is a resource with performing a work item [16, 8, p. 910] described the *actor experience* as experience of an actor (human or software) in performing a specific job. [8] and [24, p. 308] distinguished a actors’s (human or software) level of experience, such as novice, expert and guru. Still referring to experience, [24, p. 308] argues that *typically, an activity will be allocated to actors with the highest level of expertise available*, bringing another term, *expertise*, into the field of view. As a measurement of experience, the *number of sibling work items the resource has already performed*, including the best past execution were mentioned. [6] describes an *experience index calculation* which was divided into 3 steps: the level of activity performance of each performer was summed for each degree of activities, then each sum was multiplied per a coefficient/weight which expressed the complexity degree and the sum of all these multiplications was finally normalized by using a value associated to an expert scale (discipline advisor - very expert specialist - expert specialist - senior specialist - basic specialist) [6, p.307]. A more simple experience measure was proposed by [39], indicating that *more executions of an activity = more experience*. Other statements referring to work experience addressed the work experience measurement in years, e.g., ‘How many years have you been in the IS profession’ [36]. As a specification of individual characteristics, [32] mentioned *previous jobs*, which can also be seen as previous work experience. [21] mentioned *the amount of work done in a PAIS*, and *the time worked with the PAIS* in the context of internal validity of their study, however these factors may also be considered as measurements for work experience. History-based allocation was mentioned as a resource pattern that provide information that may be seen as analogue to human work experience with PAIS [30].

*Lessons Learned.* The results of the literature review of PAIS theory indicated no evidence that a human-centric allocation based on work experience has already been proposed in the field of resource allocation in PAIS. Further we could identify a mix of terms used for describing characteristics of humans considered as resources in PAIS. The various terms seem to be used intuitively rather than following a clear and separated understanding of the respective terms. We recognized that often terms were just mentioned without further explaining their understanding by the author and offered a basis for different interpretations by the reader. Further, several graphical illustrations were not completely described and explained which aggravated finding a common understanding of terms. We

focused on explicit explanations of the terms in textual form which reduced the relevant text material compared to the initially found literature.

### 2.2 ‘Work Experience’ in Daily Life

In order to find out more about ‘work experience’, e.g., how it is expressed and proven in daily life, a qualitative content analysis of job offers was conducted. The job offers were collected from the online career network *jobpilot.de*. A total of 83 job offers (without duplicates) were selected which resulted from searching for ‘*permanent positions*’, ‘*career level: experienced*’, ‘*worldwide job offers*’, ‘*occupational fields: strategic management, information technology*’, ‘*all specifications, all branches, all regions, ‘job offers of the last 4 weeks*’. The number of employers could not be identified, as several job offers were announced in an anonymous way via recruitment agencies. Altogether, the material for analysis comprised 24,516 words. Categories were elaborated inductively and are illustrated as results of the qualitative analysis of job offers in Figure 4.

Category	Subcategory	Count*	Example
Work experience	Industry/Employment experience	16,9% (14/14)	'They are looking for an engineer who is CCNA-CCNP level with 3-5 years of industry experience.'
	Experience in the field	43,4% (53/36)	'To be qualified for this role, this individual should have over 7 years of experience within development and architecture.'
	Functional experience	8,4% (7/7)	'Strong experience in a database administration role...'. '...preferably bying a qualified accountant with at least 5 years of functional experience.'
	Hands-on experience	56,6% (77/47)	'2-3 years of experience with C# and ASP.NET and the SDLC.'
	Generic experience	7,2% (7/6)	'This individual should also have lead experience.' 'Language skills...'
Proof	Time (years)	43,4% (45/36)	'10-12+ years experience in relevant software or internet service industry with a service operational background.'
	Track record	22,9% (20/19)	'1 year project management experience with digital agency, a proven track record in delivering various digital advertising projects...'
	Work knowledge and skills	80,7% (110/67)	'Experience in writing or executing test cases.'
* % of job offers (count of the statement / number of job offers where statement was found)			

Fig. 4. Categories of ‘work experience’ resulting from job offer content analysis

The qualitative content analysis included the search and coding of statements that referred to *experience*. Five sub-categories were elaborated to reflect the meaning of *experience* in the statements. Some statements referred to past employment experience which were subsumed into the subcategory *industry/employment experience*. Statements, that referred to experience in a broader area, such as ‘internet technologies’ were subsumed to *experience in the field*. More specific statements which included explicit functions, jobs, or roles were

aggregated into the subcategory *functional experience*. The sub-category *hands-on experience* included statements that focused on experience from practicing, e.g., developing software in Java. In the final subcategory *generic experience* statements were collected that refer to work experiences that can be seen as transferable among different subjects, disciplines and hence interpreted as more generic experiences, such as ‘experience with leadership and budget responsibilities’. This subcategory was included to highlight that there were not exclusively subject-specific (fundamental to the subject, discipline) work experiences stated in the analyzed job offers. The second category focused on the *proof of work experience*. Three subcategories were derived from the underlying job offer text material: *time* (typically expressed in years), *track record* (including records such as projects, employers and functions), and *work knowledge and skills*. Many of the statements referred to work knowledge and skills (e.g., ‘experience working with version control systems’, financial experience’). All these statements were summed up to one subcategory as they had one aspect in common: independent of the further supportive records - finally it is the individual who needs to proof the experience, e.g., by demonstration.

The qualitative content analysis of selected job offers helped us to understand, that there seemed to be different perspectives on work experience and that work experience cannot be captured in its complexity by one simple measure. The search in surveys and scales supported this perception. Traditional measurements of work experience in, for example, earning studies were to deduct the years of completed schooling from the individuals’ age (in years) at the start of a specific period in order to receive the years of accumulated experience. Further work experience measurements have been, for example, the time spent in the labor force, time employed, and time since school graduation as mentioned above [25]. In addition to the general perspective on work experience, there were also measurements of hands-on experiences, for example in business process modeling studies which noticed modeling experiences (e.g., levels such as novices and experts) as a factor influencing, e.g., task performance [27]. Often the participants of the surveys were asked to self-assess their level of experience, to express their modeling experience in time (e.g., number of years experience in process modeling overall, number of months experience in a particular process modeling grammar), or in the number of process models created [28].

### 2.3 ‘Work Experience’ in Psychology - An Initial Touch

In the following section, our intention was not to provide a holistic insight into psychology research on work experience but rather to initially sense the construct ‘work experience’ as discussed in psychology. Considered were contributions of the Journal ‘Personnel Psychology’ including research around people at work. A search via ProQuest by using the search terms ‘work experience’ resulted in 6 hits. In the following we will summarize these contributions by illustrating the model of work experience as proposed by [35] (based on [26]), and providing an overview of work experience components, as well as quantitative and qualitative work experience measurements.

*Understanding of work experience.* [35] suggest to consider work experience as a ‘*multidimensional, multilevel, and temporarily dynamic construct*’ [35, p. 326] and describe a model of work experience that includes three major components of work experience: the quantitative, qualitative, and the interaction component. The *quantitative component* includes in general two measurement methods, time-based and amount-based measurements. Explicit quantitative measurements are (citations were taken from [26] [35]): time on a task [26], time on the job/position (job tenure) (e.g., [17] [5]), time spend in an organization (organizational tenure) (e.g., [18]), number of times a task has been performed (activity level, task frequency) (e.g., [15] [37] [7]), number of jobs held in an organization [26] [35], and number of employers. The advantage of the amount measurements (e.g., number of times a task has been performed) is that they imply information about qualities that affect work experience, such as the opportunity to perform and practice particular tasks. The *qualitative component* of work experience includes aspects (such as variety, challenge, and complexity) that will differ in their relevance for different domains. Explicit qualitative measurements are: variety of tasks, breadth (number of different tasks), and responsibilities performed in a job, types of challenges encountered in an assignment, task type (difficulty/criticality/complexity of the task performed) [26], job complexity [26], type of organization [26], opportunities to develop new knowledge and skills through training (see also [35]), working with a highly supportive mentor (see also [35]), recency of tasks [7], and supervisory tasks [7]. The *interaction component* considers the interaction between the qualitative and quantitative components of work experience. The interaction may be reflected in ‘density’ which intends to capture the level of intensity of experiences. A scenario that illustrates a high-density experience is an individual in a one year assignment who faces many challenging situations while another individual in a similar assignment faces just a few challenging conditions. Another interaction component is ‘timing’. How an individual is influenced by an experience depends on when the experience occurs during a career. Experiences can be sequenced in ways that maximize motivational and performance outcomes [35, p. 330]. The *level of of specificity* determines how specific is the measure of experience in question. Experience measurement can be specified in different levels such as the task-, job-, and organizational level [26], and the work group level (measurements may be the number of different work groups and the type of teams such as cross-functional problem-solving teams) [35, p. 330]. Work experience can lead to the development of knowledge and skills, motivation, and attitudes and values that factor into performance and other organization-valued outcomes’ [35, p. 334]. For example, the number of times a task has been performed can enhance proficiency by honing skills through practice. Direct outcomes may be increased work knowledge and skills, motivation, and work-related values and attitudes. Indirect outcomes of work experience may be performance, and participation in developmental activities [35].

### 3 ‘Work Experience’ as Critical Factor in PAIS

According to the psychological theory there is a recognizable correlation between work experience and job performance. From our point of view, humans as critical ‘resources’ in PAIS have received too little attention in PAIS theory so far.

Work experience can be considered as twofold: it may be understood as (a) the experience a person has gained from and in performing a business activity and task; or (b) the perception of one’s work. Referring to the latter, performing a business activity or task may be experienced as, for example, stressful, pleasant, challenging, or dull. How work is experienced affects personal and organizational outcomes [3,22]. Qualitative measurements of work perception may be placed on different time levels [3, p. 532], e.g. immediately, asking individuals what they are doing now and how they feel about it; short-termed, asking individuals after a short period of time about their experiences, e.g., after a working day; or long-termed, asking individuals to recollect or reconstruct experience over an extended period of time. The perception of a person’s work may be positively or negatively influenced by the task allocation strategy in PAIS. In the following, we put our focus on the former understanding of work experience, the experiences an employee has gained by performing business activities and tasks.

The description of humans in PAIS is typically based on the concept of *roles*. A role may be seen as a group of humans with specific capabilities and privileges. While privileges are assigned to an organizational position, capabilities are considered as a direct property of the person. We understand work experience as a specification of capabilities. As illustrated in Figure 5 we subdivide work experiences into *previous work experiences* and *process work experiences*.

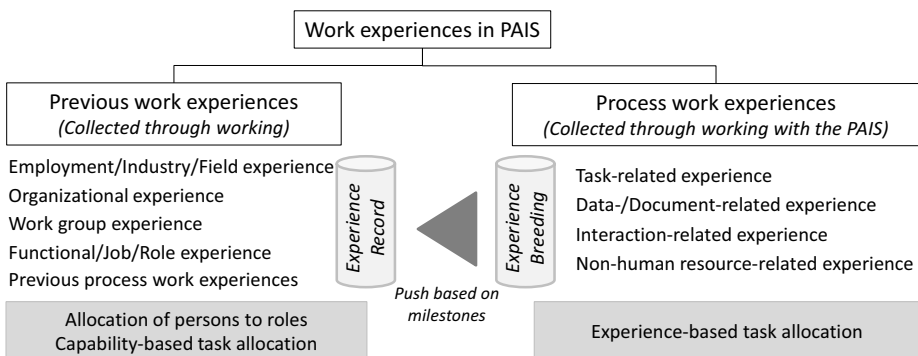


Fig. 5. Work experiences in PAIS

*Previous work experiences* to be considered in PAIS may include work experiences that were gathered, on the one hand, *before* a person started to work in the current organization, job, or role, and, on the other hand, during working in

an organization up to a particular milestone. As illustrated in Figure 5 we subdivide the previous work experiences according to the levels of specificity of 35 (as discussed in Section 2) into employment/industry/field experience (occupation level), organizational experience (organization level), work group experience (work group level), functional/job/role experience (job level), and previous process work experience (task level). Measurements about previous work experiences were subdivided into the measurement modes *time* and *amount* (quantitative measurements), as well as *type* (qualitative measurements) according to 26 35 and are illustrated in Figure 6. Information about previous work experiences is particularly relevant for the *allocation of persons to roles* and *capability-based task allocation in PAIS* and needs to be recorded adequately in a database (*Experience Record*). The information about previous work experiences can improve a more fine-grained allocation of persons to roles, such as the allocation of a new employee to a role based on start values derived from CV and the job interview, or the (more objective) suggestion of potential internal employees for the reoccupation of a post. A capability-based task allocation in general, and a previous work experience-based task allocation in particular makes sense if previous work experiences can be used to describe in more detail what business activities and tasks require in order to be adequately performed. Typical work experience information already used is, for example, the number of employers, the type of employers, or previous work experience in years (e.g. 10 years work experience in Airbus servicing 30, p. 17) as described in the capability-based allocation resource pattern 30.

Previous Work Experiences Measurements in PAIS			
Labels of Previous Work Experiences	Level of Measurements	Measurements in PAIS	
		Quantitative Measurements	Qualitative Measurements
Employment/ Industry/Field experience	Occupation	Time employed	Type of field
		Time employed in a field	
Organizational experience	Organization	No. of organizations	Type of organization
		Time spent in an organization (seniority)	
Work group experience	Work group	No. of work groups	Type of work group
		Time spent in a work group	
Functional / Job / Role experience	Job, Role	No. of functions/ jobs/ roles performed	Type of function/ job / role
		Time in a role	Type of responsibilities
Previous process work experiences	Process/Task	See Fig. 7: Collection of process work experiences measurements	

Fig. 6. Collection of previous work experience measurements

The greatest potential of work experiences measurements in PAIS we see in the measurements located at the task level. We called the work experiences measured at task level *process work experiences*. Process work experiences are gained from working in processes with the PAIS. We subdivided the process work experiences into task-related, data-/document related, interaction-related, and non-human resource-related experiences. The process work experience measurements were subdivided into quantitative (time, amount) and qualitative (type, and (self-, and peer-) evaluations) ones, as illustrated in Figure 7. Additionally



to the measurement modes time, amount and type we argue, that work experience measurements that refer to the *quality of tasks results* need to be considered in PAIS. Such measurements can be of quantitative and qualitative nature. For example, Andy has performed the rough-grained task ‘writing a research paper’ 100 times so far. However, the number of times the task has been performed does not provide insights into the quality of the task results. In order to get a picture of the quality of the resulted research papers, we need specific measurements, for example, depending on the type of the research papers (such as journal article, conference paper, workshop paper, book section), the impact factor of the journal, organization-internal rankings of publication organs, the number of citations (e.g. without self-citations), and feedback of the reviewers could be considered. We suggest that process work experiences are collected by various measurements during a period of time up to a particular milestone (*Experience Breeding*) and then pushed into the *Experience Record* database in which information about previous work experience is stored. Milestones could be, for example, experience levels to be reached (e.g. newbie, valuable, special force, magister), and dates (e.g. filling of a post). While the milestone is not reached, work experience values are collected and aggregated in the Experience Breeding database. If the milestone is reached, the aggregated values are pushed to the Experience Record database.

The information collected about process work experiences is highly relevant for *process experiences-based task allocation in PAIS* as it provides more fine-grained characteristics of employees performing in the same role. The fine-grained information about process work experiences can improve task allocation to humans, e.g., in the following ways: (a) breeding of process work experiences up to a particular level, (b) support of work experience-building strategies (such as the support of specialists and generalists), and (c) mentoring, e.g. to support exchange and the sharing of experiences across levels.

Process Work Experiences Measurements in PAIS		
Labels of Process Work Experiences	Measurements in PAIS	
	Quantitative Measurements	Qualitative Measurements
Task-related experience	No. of times performing a task	Type of task (difficulty, complexity, criticality)
	Time on a task (task completion time)	Types of challenges encountered in the task
	No. of task types (variety)	Actor evaluation
	No. of sibling tasks performed	
	No. of task success/failure	
Task-result quality measurements		
Data-/Document related experience	No. of times working on data/a document	Type of data/document
	Time working on data/a document	Data-/Document evaluation
	No. of times working in an interaction	Type of interaction (e.g. teams, customers)
	Time in an interaction	Persons interacted with
		Interaction evaluation
Non-human resource-related experience	No. of times working with a resource	Type of non-human resources
	Time working with a resource	

Fig. 7. Collection of process work experience measurements

## 4 Discussion

*‘Work Experiences’-Based Allocation in PAIS.* ‘Work experience’-based allocation can be understood as a combination of capability-based [30] and history-based allocation [30]. In order to be able to consider work experiences for allocating tasks to humans in PAIS, the details captured and maintained for ‘human resources’ need to be extended, or the information from previous execution history of humans needs to be extracted from workflow logs to use it in the allocation process [30, p. 19]. Referring to the latter, it needs to be determined what information has to be logged whereby the design of the process logs will be affected. A concrete description and guideline for work experience-based allocation and the illustration of log design that captures work experience will be presented in future work. Work experience-based allocation has a potential to support various allocation best practices as presented in [29], such as case manager, case assignment, customer teams, flexible assignment and specialist-generalist assignment. For example, information about work experience can be used to identify the most experienced case manager in the organization in order to, e.g., guide critical cases or escalations. Customer teams may be composed based on the individuals’ work experience. Furthermore, work experience may be used as a factor for identifying specialists and generalist, but also to lead individuals to a particular specialization or generalization level. Work experience can not only be used to identify individuals who have a high level of work experience, but also to find these individuals who need to build up work experience (e.g., novices). These individuals can be supported by, for example, providing a mentor for a task or a case, offering ‘how-to’ video streams, or exemplary output of the activity as a guideline. Highly experienced individuals may be suggested as mentors, or as experts in critical process instances. Future research will address human-centric functionalities in PAIS which are based on work experience. A challenge of considering work experience information may be the trade off between work experience transparency and the privacy of the users. To deal with this data in a sensitive way, appropriate access control need to be considered (e.g., restricted access to anonymised data).

*Benefit of Work Experience Information from PAIS for Daily Life.* Information of work experience which we suggest to be gathered and used in PAIS for the allocation of tasks to humans may provide also benefits from a more general point of view. The information of an individual’s work experiences may be considered and prepared as a portfolio providing the individual with a detailed documentation of his or her work at the particular employer. This portfolio may be understood as a detailed track record, or in other words, a kind of proof of work experience collected in an organization (compare with Figure 4). Furthermore, the transparency of work experience of humans in the organization may serve the organization and the individuals as a information basis on which decisions concerning formal learning and training for specifying or broadening individuals’ knowledge, skills and competencies may be taken. An important stream of

research and work in this context concentrates on competency and skills standards and specifications in order to make particularly competencies and skills usable and reusable across education, work and the labor market in a ‘lifelong’ perspective. Specifications and standards are, for example, the IMS Reusable Definition of Competency or Educational Objective Specification (RDCEO) [13], the IEEE Data Model for Reusable Competency Definitions (RCD) [12], HR-XML (human resources XML) [11], or SIFA (Skills Framework for the Information Age) [34]. These information sources should also be considered particularly when a combination of competencies/skills and work experiences are considered in PAIS.

*Collecting Data for Work Experience Research.* There are several ways of looking at how ‘work experience’ can be considered in PAIS and to what outcomes it should lead. In our point of view PAIS seem to have enormous potentials to provide a fruitful context for collecting data critical to find out more about the construct ‘work experience’. In general, PAIS may be used to find out under which conditions experience leads to a desired outcome.

## 5 Conclusion

The main goal of this work was to focus on work experience from various perspectives to perceive the construct work experience with its various facets, to bring some transparency to its measurements, and to discuss potentials its potentials as a possible individual attribute in PAIS. A literature review was conducted in PAIS theory which concentrated on the descriptions of resources, in particular human ‘resources’ in the context of PAIS. The text material analyzed partly lacked term explanations used to describe resources, particularly when individual attributes directly connected to a particular person (such as qualifications, skills, competencies, experiences) were mentioned. The woolly manner of expressions of individual attributes indicates a necessity for clarification including explanation and distinction between the terms and attributes used. The analysis of job offers showed that there were several ways to express and measure work experience in daily life. Various facets of work experience were as well reflected in the multidimensional understanding of work experience as illustrated in psychological literature. The result of the study was a collection of work experience measurement that can be considered in PAIS. The better understanding of work experience and the collection of work experience measurements for PAIS provide the basic step for further work. The potentials of work experience as one of the individual attributes describing and considering humans in PAIS are particularly seen in a finer and more value adding allocation of humans to tasks from the perspective of the individual (e.g., allocation led by the goal to build-up experiences). This contribution can be considered as a first step towards providing a holistic solution of integrating work experience into PAIS.

## References

1. Agarwal, S., Kenkre, S., Pandit, V., Sengupta, B.: Studying the evolution of skill profiles in distributed, specialization driven service delivery systems through work orchestration. In: Proceedings of the 2011 Annual SRII Global Conference, SRII 2011, pp. 201–213. IEEE Computer Society (2011)
2. Agrawal, A., Amend, M., Ford, M., Keller, C., Kloppmann, M., König, D., Leymann, F., Mueller, R., Pfau, G., Plösser, K., Rangaswamy, R., Rickayzen, A., Rowley, M., Schmidt, P., Trickovic, I., Yiu, A., Zeller, M.: Web services human task, version 1.0. Tech. rep., Active Endpoints Inc., Adobe Systems Inc., BEA Systems Inc., International Business Machines Corporation, Oracle Inc. and SAP AG (2007)
3. Alliger, G.M., Williams, K.J.: Using signal-contingent experience sampling methodology to study work in the field: a discussion and illustration examining task perceptions and mood. *Personnel Psychology* 46 (1993)
4. Ayachitula, N., Buco, M., Diao, Y., Maheswaran, S., Pavuluri, R., Shwartz, L., Ward, C.: It service management automation - a hybrid methodology to integrate and orchestrate collaborative human centric and automation centric workflows. In: IEEE Int'l Conf. on Services Computing (SSC), pp. 574–581 (July 2007)
5. Borman, W.C., Hanson, M.A., Oppler, S.H., Pulakos, E.D., White, L.A.: Role of supervisory experience in supervisory performance. *Journal of Applied Psychology* 78, 443–449 (1993)
6. Corallo, A., Lazoi, M., Margherita, A., Scalvenzi, M.: Optimizing competence management processes: A case study in the aerospace industry. *Business Process Management Journal* 16(2), 297–314 (2007)
7. DuBois, D., McKee, A.S.: Facets of work experience. In: Ninth Annual Conference of the Society for Industrial and Roganizational Psychology (1994)
8. Fakas, G., Karakostas, B.: A workflow management system based on intelligent collaborative objects. *Information and Software Technology* 41(13), 907–915 (1999)
9. Faustmann, G.: Configuration for adaptation a human-centered approach to flexible workflow enactment. *Computer Supported Cooperative Work* 9, 413–434 (2000)
10. Hall, J.: D2.1 - visp workflow technologies - functional analysis and comparison. Tech. rep., Project no. FP6-027178 (2006)
11. HR-XML Consortium: HR-XML, <http://www.hr-xml.org/>
12. IEEE Standard for Learning Technology - Data Model for Reusable Competence Definitions. IEEE Computer Society (2008-2012)
13. IMS Global Learning Consortium: IMS Reusable Definition of Competency or Educational Objective Specification, <http://www.imsglobal.org/competencies/>
14. Kim, S., Godbole, A.: A perspective on the design of human-centered collaboration systems. In: IEEE Int'l Conf. on Information Reuse Integration, pp. 212–217 (August 2009)
15. Lance, C.E., Hedge, J.W., Alley, W.E.: Joint relationships of task proficiency with aptitude, experience, and task difficulty: A cross-level interactional study. *Human Performance* 2, 249–272 (1989)
16. de Leoni, M., van der Aalst, W., ter Hofstede, A.: Visual support for work assignment in process-aware information systems: Framework, formalization, operationalisation. Tech. rep., PM Center Report BPM-08-06, BPMcenter.org (2008)
17. McDaniel, M.A., Schmidt, F.L., Hunter, J.E.: Job experience correlates of job performance. *Journal of Applied Psychology* 73(2), 327–330 (1988)

18. McEnrue, M.P.: Length of experience and the performance of managers in the establishment phase of their careers. *Academy of Management Journal* 31, 175–185 (1988)
19. Melcher, J., Mendling, J., Reijers, H.A., Seese, D.: On Measuring the Understandability of Process Models. In: Rinderle-Ma, S., Sadiq, S., Leymann, F. (eds.) *BPM 2009. LNBIP*, vol. 43, pp. 465–476. Springer, Heidelberg (2010)
20. Mendling, J., Reijers, H., Cardoso, J.: What Makes Process Models Understandable? In: Alonso, G., Dadam, P., Rosemann, M. (eds.) *BPM 2007. LNCS*, vol. 4714, pp. 48–63. Springer, Heidelberg (2007)
21. Mutschler, B.: Modeling and simulating causal dependencies on process-aware information systems from a cost perspective. Ph.D. thesis, Univ. of Twente, Enschede (January 2008)
22. Nakatumba, J., van der Aalst, W.M.P.: Analyzing Resource Behavior Using Process Mining. In: Rinderle-Ma, S., Sadiq, S., Leymann, F. (eds.) *BPM 2009. LNBIP*, vol. 43, pp. 69–80. Springer, Heidelberg (2010)
23. Osborn, A.F.: *Applied Imagination*. Scibner, New York (1957)
24. Papadopoulos, G., Fakas, G.: Component-Based Development of Dynamic Workflow Systems Using the Coordination Paradigm. In: Malyshkin, V. (ed.) *PaCT 2003. LNCS*, vol. 2763, pp. 304–315. Springer, Heidelberg (2003)
25. Plantes, M.K.: The effect of work experience on young men’s earnings. Tech. rep., Institute for Research on Poverty (1979)
26. Quinones, M., Ford, J.K., Teachout, M.S.: The relationship between work experience and job performance: a conceptual and meta-analytic review. *Personnel Psychology* 48(4), 887–910 (1995)
27. Recker, J.: Continued use of process modeling grammars: the impact of individual difference factors. *European Journal of Information Systems* 19, 76–92 (2010)
28. Recker, J.: Understanding process modelling grammar continuance: a study of the consequences of representational capabilities. Ph.D. thesis, Queensland University of Technology (2008)
29. Reijers, H.: Process design and redesign. In: *Process-Aware Information Systems*, pp. 207–234. Wiley Interscience (2005)
30. Russel, N., ter Hofstede, A., Edmond, D., van der Aalst, W.: Workflow resource patterns - beta working paper series, wp 127. Tech. rep., Eindhoven University of Technology, Eindhoven (2004)
31. Russell, N., van der Aalst, W.M.P., ter Hofstede, A.H.M., Edmond, D.: Workflow Resource Patterns: Identification, Representation and Tool Support. In: Pastor, Ó., Falcão e Cunha, J. (eds.) *CAISE 2005. LNCS*, vol. 3520, pp. 216–232. Springer, Heidelberg (2005)
32. Russell, N.C.: Foundations of process-aware information systems. Ph.D. thesis, Queensland University of Technology (2007)
33. Schall, D., Gombotz, R., Dorn, C., Dustdar, S.: Human interactions in dynamic environments through mobile web services. In: *IEEE Int’l Conf. on Web Services (ICWS)*, pp. 912–919 (2007)
34. SFIA Foundation Ltd.: SFIA - Skills Framework for the Information Age (2003-2012), <http://www.sfia.org.uk/>
35. Tesluk, E., Jacobs, R.R.: Toward an integrated model of work experience. *Personnel Psychology* 51(2), 321–355 (1998)

36. Truman, G., Baroudi, J.: Gender differences in the information systems managerial ranks: An assessment of potential discriminatory practices. *MIS Quarterly* 18(2), 129–142 (1994)
37. Vance, R.L., Coovert, M.D., MacCallum, R.C., Hedge, J.W.: Construct models of task performance. *Journal of Applied Psychology* 74, 447–455 (1989)
38. Vanderfeesten, I.T.P., Reijers, H.A.: A Human-Oriented Tuning of Workflow Management Systems. In: van der Aalst, W.M.P., Benatallah, B., Casati, F., Curbera, F. (eds.) *BPM 2005*. LNCS, vol. 3649, pp. 80–95. Springer, Heidelberg (2005)
39. Zur Muehlen, M.: Organizational management in workflow applications - issues and perspectives. *Information Technology and Management* 5 (2004)

# Ontological Foundations for Conceptual Modeling with Applications

Giancarlo Guizzardi

Ontology and Conceptual Modeling Research Group (NEMO)  
Computer Science Department,  
Federal University of Espírito Santo (UFES), Brazil  
gguizzardi@inf.ufes.br

**Abstract.** The main objective of this tutorial is to introduce researchers to the theory and practice of advanced conceptual modeling through the application of a new emerging discipline named Ontology-Driven Conceptual Modeling. In this discipline, theories coming from areas such as Formal Ontology in philosophy, but also Cognitive Science, Philosophical Logics and Linguistics are employed to derive engineering tools (e.g., modeling languages, methodologies, design patterns, model compilers and simulators) for improving the theory and practice of Conceptual Modeling. In particular, here, the expressiveness and relevance of these theories and derived tools are demonstrated through their application to solve some classical and recurrent conceptual modeling problems concerning the well-founded representation of: classification and taxonomic structures, part-whole relations, intrinsic and relational properties, formal and material associations, association specialization, attribute value spaces and roles.

**Keywords:** Ontological Foundations, Ontology-Driven Conceptual Modeling, Unified Foundational Ontology (UFO), OntoUML.

## 1 Introduction

The main objective of this tutorial is to introduce researchers to the theory and practice of advanced conceptual modeling through the application of a new emerging discipline named Ontology-Driven Conceptual Modeling.

Conceptual Modeling is a discipline of great importance to several areas in Computer Science. Its main objective is concerned with identifying, analyzing and describing the essential concepts and constraints of a universe of discourse, with the help of a (diagrammatic) modeling language that is based on a set of basic modeling concepts (forming a metamodel). In this tutorial, we show how Conceptual Modeling Languages can be evaluated and (re)designed with the purpose of improving their *Ontological Adequacy*.

In simple terms, Ontological Adequacy is a measure of how truthful the models produced using a modeling language are to the situations in the reality they are supposed to represent (*Domain Appropriateness*), and how easy it is for users to use these models for communicating, domain learning and problem-solving (*Comprehensibility Appropriateness*) [1].

The tutorial starts by briefly discussing a systematic evaluation method for comparing a metamodel of the concepts underlying a language to a *Reference Ontology* of the corresponding domain in reality. The in this presentation is on general conceptual modeling languages (as opposed to domain specific ones). Hence, the reference ontology employed here is a Foundational (or Upper-level) Ontology. Moreover, since, the tutorial focuses on structural modeling aspects (as opposed to dynamic ones), this foundational ontology is an *Ontology of Endurants* addressing issues such as categories of object types and taxonomic structures, intrinsic properties, modes and attribute value spaces, formal and material relations, association specialization, as well as different and subtle aspects regarding the representation of conceptual part-whole relations.

The foundational ontology which is adopted in this tutorial (termed *UFO – Unified Foundational Ontology*) has been developed by adapting and extending a number of theories coming, primarily, from formal ontology in philosophy, but also from cognitive science, philosophical logics and linguistics. Once developed, every sub-theory of the ontology has been used for the creation of a number of methodological tools including: (a) an ontologically well-founded version of UML 2.0 (latter dubbed *OntoUML*) with an explicitly defined metamodel, an expressive formal characterization and a number of associated methodological guidelines; (b) a set of Ontological Design Patterns, including Model Construction Patterns, Model Analysis Patterns and Model Transformation Patterns; (c) Computational Tools for Model Creation and Verification, but also Model Validation via Visual Simulation.

The expressiveness and relevance of these engineering tools are shown throughout the presentation to solve some classical and recurrent conceptual modeling problems. In particular, the tutorial discusses a number of examples of the successful application of these tools in complex domains and industrial application scenarios.

**Acknowledgement.** Research supported by FAPES (grant #52272362).

## References

1. Benevides, A.B., et al.: Validating modal aspects of OntoUML conceptual models using automatically generated visual world structures. *Journal of Universal Computer Science, Special Issue on Evolving Theories of Conceptual Modeling* 16/20 (2010)
2. Benevides, A.B., Guizzardi, G.: A Model-Based Tool for Conceptual Modeling and Domain Ontology Engineering in OntoUML. In: Filipe, J., Cordeiro, J. (eds.) *Enterprise Information Systems. LNBIP*, vol. 24, pp. 528–538. Springer, Heidelberg (2009) ISSN 1865-1356
3. Costal, D., Gómez, C., Guizzardi, G.: Formal Semantics and Ontological Analysis for Understanding Subsetting, Specialization and Redefinition of Associations in UML. In: Jeusfeld, M., Delcambre, L., Ling, T.-W. (eds.) *ER 2011. LNCS*, vol. 6998, pp. 189–203. Springer, Heidelberg (2011)
4. Guizzardi, G., Lopes, M., Baião, F., Falbo, R.: On the importance of truly ontological representation languages. *International Journal of Information Systems Modeling and Design (IJISMD)* (2010) ISSN: 1947-8186
5. Guizzardi, G.: *Ontological Foundations for Structural Conceptual Models*. Universal Press, The Netherlands (2005) ISBN 1381-3617
6. Guizzardi, G.: *Theoretical Foundations and Engineering Tools for Building Ontologies as Reference Conceptual Models*. *Semantic Web Journal* (2010)
7. Zamborlini, V., Guizzardi, G.: On the representation of temporally changing information in OWL. In: *Proceedings of the 5th International Workshop on Vocabularies, Ontologies and Rules for the Enterprise (VORTE 2010)*, Vitória, Brazil (2010)



# Designing Technical Action Research and Generalizing from Real-World Cases

Roel Wieringa

Department of Electrical Engineering, Mathematics, and Computer Science,  
University of Twente, The Netherlands  
<http://www.ewi.utwente.nl/~roelw>

**Abstract.** This tutorial presents a sound methodology for technical action research, which consist of testing a new artifact by using it to solve a real problem. Such a test would be useless if we could not generalize from it, and the tutorial introduces architectural inference as a way of supporting generalizations by technical action research.

**Keywords:** Case studies, Action research, Technology validation, Generalization.

## 1 Introduction

A *case study* is an empirical investigation of an instance of a phenomenon in its natural context. For example, a researcher could investigate what the causes of delay are in ERP implementations by investigating one particular ERP implementation project. In some case studies, a researcher may investigate multiple cases, but because each individual case study requires considerable resources from the researcher, multiple case studies rarely involve more than four individual cases.

In an *observational case study*, the researcher tries to minimize his or her influence on the case. The researcher collects documents, interviews people, and observes their behavior without trying to change what happens in the case. For example, in an observational case study of a running ERP implementation project, the researcher will not intervene if he or she observes that the project will run over time. At the other extreme is the *action case study*, in which the aim of the researcher is to improve something in the case. For example, the researcher may join the ERP implementation project and help monitor its progress, signal problems, and provide knowledge about methods to speed up progress.

One particular kind of action case study is *technical action research* (TAR), which is an action case in which the researcher tests a technique by using it in practice [1]. For example, the researcher may have developed a new technique for effort estimation for ERP implementations. In a TAR project, the researcher joins an ERP implementation project and then uses this technique to estimate the effort of this project. The goal is both to help the project and to learn something about the technique.

In a TAR project, the researcher plays three roles, that must be kept separate:

- Technique developer. The researcher has developed some technique X.
- Client helper. The researcher helps some stakeholder, called the client, and uses X to do this.
- Empirical researcher. The researcher investigates the effects and utility of X in practice.

For example, the researcher (1) may have developed some new effort estimation technique, (2) use this technique in an ERP implementation project, and then (3) draws lessons learned from this use: Is it usable? Is it easier to use than alternative techniques? Does it give accurate results?

The answers to these questions would be of merely anecdotal value if the researcher could not generalize from them. In a TAR project, as in all other case studies, statistical inference is not applicable as a generalization technique, and the researcher must use case-based inference. In *case-based inference*, the researcher reasons from an observed case to an unobserved case by pointing out relevant similarities between the cases. For example, if an effort estimation technique has been effective in one project done for a large bank, it is plausible that it may be effective also in other projects done for large banks, because these other projects are probably similar to the current one.

But how similar must these projects be for this inference to be plausible? Not just any similarity is relevant. The relevance of the similarities must be based on a theory, and the inference is as plausible as this theory is. An effective way of case-based inference is to use *architectural theories*, which are theories that represent the case in terms of components with capabilities and interactions, and then identify mechanisms of interaction that are reproducible from case to case, and that explain the observations made in the case [2]. For example, suppose the effectiveness of an effort estimation technique in one case can be explained by the fact that the technique itself is based on a number of project factors that could be influenced by the project manager, then it is plausible that in all projects where these factors are present, and can be manipulated by the project manager, the same technique will be effective too. This generalization is based on the architecture of the projects (project factors, project manager) and mechanisms in the projects (influence of project manager on project factors).

## References

1. Wieringa, R., Morali, A.: Technical Action Research as a Validation Method in Information Systems Design Science. In: Peffers, K., Rothenberger, M., Kuechler, B. (eds.) DESRIST 2012. LNCS, vol. 7286, pp. 220–238. Springer, Heidelberg (2012)
2. Wieringa, R., Daneva, M., Condori-Fernandez, N.: The structure of design theories, and an analysis of their use in software engineering experiments. In: International Symposium on Empirical Software Engineering and Measurement (ESEM), pp. 295–304. IEEE Computer Society (September 2011)

# Improvisational Theater for Information Systems: An Agile, Experience-Based, Prototyping Technique

Martin Mahaux and Patrick Heymans

PReCISE Research Centre, Faculty of Computer Science, University of Namur  
Grandgagnage, 21, 5000 Namur, Belgium  
{martin.mahaux,patrick.heyman}@fundp.ac.be

**Abstract.** Collaborative creativity is key to innovative software development. It is however not easy to master, and few techniques have focused on those aspects. Games have recently started to receive serious attention to fill this gap. In this context, this tutorial proposes participants a fun and refreshing learning moment. Through actually playing improvisational theatre (improv) games in group, participants will learn new ways to generate scenarios through a collaborative, cheap, rapid, experience-based, design technique.

## 1 Introduction

According to many commentators, creativity is the new key economic activity. This holds for the development of Information Systems (IS) as much as in many other sectors of the economy, and so we must look for ways of bringing more creativity into IS development [1]. Developing IS can be seen as a fundamentally collaborative and creative process, from discovering requirements until development, through architecture. While these concerns are still largely considered in isolation, the success of agile paradigms forces us to see them as a whole, putting an even stronger emphasis on collaboration and creativity. While collaboration and creativity may yield wonderful results, they remain difficult to achieve. IS engineers and researchers are currently lacking skills in these areas, and training is hard to find. Moreover, adapted collaborative creativity techniques are lacking too. This mini-tutorial intends to show participants how to fill this gap by building upon results on using improv for design (e.g. [3,4,5,6,7]). These show that improv supports *team-based innovation*. To achieve this, it provides techniques for improving communication between stakeholders, increasing mutual understanding as well as open-mindedness, creativity, empathy, self and mutual confidence, agility. . . It also helps generating and testing ideas, by exploiting the notion of story and story-telling. We argue that this cheap and rapid story-making technique can effectively be considered an experience-based design technique: the focus is on the experience around the product, not on the product. By exaggerating the characteristics of group creativity (emergence, intersubjectivity, unpredictability, communication [8]), improv eases its understanding. It also

provides a framework for different types of creativity: exploration, combination, transformation. Finally, improv offers a more collaborative and positive model for conflict resolution, in that it uses conflict as the source of creativity.

## 2 Outline

During the **introduction (10')**, we first briefly define creativity, motivate the need for it, and then expose some theoretical bases that establish the link between improv and creativity. Then comes an **awareness exercise (10')** to raise group awareness and open the participants' mind for the remainder of the session. The **warm-up exercise (10')** will help people remove their tie or sweat-shirt, and get rid of some of their auto-censorship reflexes. We can now switch to the main activity, the **creative session (60')**. We will pretend that we are a group of stakeholders (including problem owners, developers, experts) willing to invent a new software product. We will use guided improvisations as an experience-based technique, inventing user stories on the fly, and commenting them in order to determine what our product should or should not do. So, while some play out scenarios they invent on the fly guided by the improv' coach, others watch and comment, take note of ideas, scenarios, requirements, feasibility problems and alternative solutions. Then we all debrief.

This work is sponsored by the Walloon Region under the European Regional Development Fund (ERDF).

## References

1. Jones, S., Lynch, P., Maiden, N.A.M., Lindstaedt, S.N.: Use and influence of creative ideas and requirements for a work-integrated learning system, pp. 289–294 (2008)
2. Mahaux, M., Mavin, A., Heymans, P.: Choose Your Creativity: Why and How Creativity in Requirements Engineering Means Different Things to Different People. In: Regnell, B., Damian, D. (eds.) REFSQ 2012. LNCS, vol. 7195, pp. 101–116. Springer, Heidelberg (2012)
3. Brandt, E., Grunnet, C.: Evoking the future: Drama and props in user centered design. In: Proceedings of Participatory Design Conference (PDC 2000), pp. 11–20 (2000)
4. Iacucci, G., Kuutti, K.: Everyday life as a stage in creating and performing scenarios for wireless devices. *Personal and Ubiquitous Computing* 6(4), 299–306 (2002)
5. Sorby, I., Melby, L., Seland, G.: Using scenarios and drama improvisation for identifying and analysing requirements for mobile electronic patient records. In: *Requirements Engineering for Sociotechnical Systems*, pp. 266–283.
6. Boess, S.: Making role playing work in design. In: *Design and Semantics of Form and Movement*, p. 117 (2006)
7. Mahaux, M., Maiden, N.A.M.: Theater improvisers know the requirements game. *IEEE Software* 25(5), 68–69 (2008)
8. Sawyer, R.K.: *Group Creativity: Music, Theater, Collaboration*. LEA (January 2009)

# Full Model-Driven Practice: From Requirements to Code Generation

Óscar Pastor and Sergio España

PROS Research Centre, Universitat Politècnica de València  
{opastor, sergio.espana}@pros.upv.es

**Abstract.** A crucial success factor in information systems development is the alignment of the system with business goals, business semantics and business processes. Developers should be freed from programming concerns and be able to concentrate on these alignment problems. Model-driven system development (MDD) does not only provide a structured and systematic approach to systems development, but also offers developers the possibility of using model-transformation technologies to derive models of a lower abstraction level that can be further refined, and even generating software code automatically. This tutorial shows how to successfully integrate business process modelling (BPM), requirements engineering (RE) and object-oriented conceptual modelling with the objective of leveraging MDD capabilities. Participants work with state of the art modelling methods and code generation tools to explore different ways to match an information system with business requirements.

**Keywords:** model-driven development, requirements engineering, business process modelling, model transformation, model-driven architecture.

## 1 Introduction

Model-driven development (MDD) is, no doubt, an active area of research and innovation nowadays. Within the information systems research community, MDD methods for software development are being proposed. MDD first covered the (platform independent and platform specific) conceptual-modelling stage, paving the way to industrial tools that support modelling and code generation (e.g. Integranova [1]). Recently, (computation independent) model-driven requirements engineering (RE) academic approaches start to appear. Although, for the present, not many of them are mature enough to be applied under conditions of practice, it is a matter of time that many industrial methods and tools are available for model-driven RE.

## 2 Overview of the Tutorial

The tutorial presents the principles, concepts and common practices of MDD, introduces an MDD strategy covering the full cycle from requirements engineering to model compilation.

The goals of the tutorial are the following:

- To be able to elicit and specify the requirements of an information system, using MDD-suitable modelling languages.

- To learn to create the object-oriented conceptual model of the computerised information system, including the following abilities:
  - To systematically derive an initial platform-independent conceptual model from the computation-independent requirements model.
  - To complete the conceptual model in order to specify the software system considering both its static and dynamic aspects.
- To compile the conceptual model using state-of-the-art technology in order to automatically generate fully-functional source code.

The concepts and strategies are general enough to be able to apply them in many MDD scenarios. However, to operationalise them into a concrete approach, a specific method is applied: the integration of Communication Analysis and the OO-Method.

Communication Analysis is a business process modelling and RE method that proposes undertaking information system analysis from a communicational perspective [2]. As a highlighted feature, business process models are complemented with message structures that describe the new and meaningful information that is conveyed to the information system in each business activity (referred to as communicative events) [3].

Recently, Communication Analysis has been integrated into the OO-Method, an object-oriented MDD framework [4] that is UML-compliant. The OO-Method conceptual model consists of four complementary models that cover all the relevant aspects of the information system so as to allow for automatic code generation.

A model transformation strategy has been defined to derive, from Communication Analysis requirements models, initial versions of OO-Method conceptual models that can already be compiled to automatically generate software code [5,6]. This way, the MDD method that results from the bottom-up integration of CA and the OO Method covers the software development life-cycle from RE to code generation.

In short, this tutorial intends to provide insights on MDD that are useful to both researchers (so they can apply them in their proposals) and practitioners (so they are aware of what is coming and can anticipate the evolution of MDD technologies).

## References

1. CARE Technologies. IntegranovaModel Execution System, <http://www.care-t.com>
2. España, S., González, A., Pastor, Ó.: Communication Analysis: A Requirements Engineering Method for Information Systems. In: van Eck, P., Gordijn, J., Wieringa, R. (eds.) CAiSE 2009. LNCS, vol. 5565, pp. 530–545. Springer, Heidelberg (2009)
3. González, A., Ruiz, M., España, S., Pastor, Ó.: Message Structures: a modelling technique for information systems analysis and design. In: WER 2011 (2011), <http://arxiv.org/abs/1101.5341>
4. Pastor, O., Molina, J.C.: Model-Driven Architecture in practice: a software production environment based on conceptual modeling. Springer, New York (2007)
5. González, A., España, S., Ruiz, M., Pastor, Ó.: Systematic Derivation of Class Diagrams from Communication-Oriented Business Process Models. In: Halpin, T., Nurcan, S., Krogstie, J., Soffer, P., Proper, E., Schmidt, R., Bider, I. (eds.) BPMDS 2011 and EMMSAD 2011. LNBP, vol. 81, pp. 246–260. Springer, Heidelberg (2011)
6. España, S.: Methodological integration of Communication Analysis into a Model-Driven software development framework. PhD at Universitat Politècnica de València, Spain (2011)

# Author Index

- Aalst, Wil M.P. van der 222, 270  
Aanen, Steven S. 334  
Acher, Mathieu 629  
Akkermans, Hans 95, 518  
Ali, Raian 206  
Allen, Mary 518  
Angelov, Samuil 445  
Asadi, Mohsen 366  
Assaf, Ali 175  
Atzeni, Paolo 160
- Baesens, Bart 254  
Bagheri, Ebrahim 613  
Barone, Daniele 502  
Binder, Michael 398  
Bitar, Abdalla 175  
Boer, Victor de 518  
Bon, Anna 518  
Borgida, Alexander 382  
Bose, R.P. Jagadeesh Chandra 270  
Bošković, Marko 366  
Boyera, Stephane 518  
Broucke, Seppe K.L.M. vanden 254  
Bugiotti, Francesca 160
- Carpenter, Martin 111  
Castano, Silvana 486  
Charfi, Anis 80  
Collet, Philippe 629  
Colman, Alan 301  
Comuzzi, Marco 445  
Conte, Tobias 581
- Dalpiaz, Fabiano 206  
Daskalaki, Evangelia 286  
De Leenheer, Pieter 95, 518  
Díaz, Oscar 646  
Dorda, Wolfgang 398  
Duftschmid, Georg 398  
Dumas, Marlon 31  
Dunkl, Reinhold 398  
Dustdar, Schahram 460, 565
- Eder, Johann 144  
Engel, Robert 222
- Ensan, Faezeh 613  
Ernst, Neil A. 382  
España, Sergio 701
- Ferrara, Alfio 486  
Fräsincar, Flavius 334  
Front, Agnès 549  
Fröschl, Karl Anton 398
- Gall, Walter 398  
Gašević, Dragan 366, 613  
Ghaddar, Ali 175  
Giorgini, Paolo 206  
Gordijn, Jaap 95  
Gröner, Gerd 366  
Grossmann, Wilfried 398  
Guéret, Christophe 518  
Guerra, Esther 127  
Guizzardi, Giancarlo 318, 695  
Gyan, Nana Baah 518
- Han, Jun 301  
Harmankaya, Kaan 398  
Heymans, Patrick 629, 699  
Hronsky, Milan 398
- Indiono, Conrad 238  
Indulska, Marta 429  
Izdebski, Waldemar 533
- Juell-Skielse, Gustaf 190  
Jureta, Ivan J. 382
- Kabicher-Fuchs, Sonja 678  
Kabir, Muhammad Ashad 301  
Karagiannis, Dimitris 414  
Khazankin, Roman 460, 565  
Kontogiannis, Kostas 350  
Köpke, Julius 144  
Kurowski, Krzysztof 19
- Lahire, Philippe 629  
Lara, Juan de 127  
La Rosa, Marcello 31  
Lécué, Freddy 111  
Leich, Thomas 597  
Leopold, Henrik 64

- Liptchinsky, Vitaliy 565  
 Lönn, Carl-Mikael 190  
 Luckner, Marcin 533  
 Ly, Linh Thao 238  
  
 Mäesalu, Raul 31  
 Maggi, Fabrizio M. 270  
 Mahaux, Martin 699  
 Mangler, Jürgen 238  
 Mankovskii, Serge 350  
 Martin, Jochen 581  
 Martin, Marko 80  
 Mazarakis, Athanasios 581  
 Mehandjiev, Nikolay 111  
 Mendling, Jan 31, 64, 549  
 Merle, Philippe 629  
 Mezini, Mira 80  
 Missikoff, Michele 1  
 Mohabbati, Bardia 366  
 Montanelli, Stefano 486  
 Moser, Christoph 414  
 Mostashari, Arash 414  
 Moustafa, Ahmed 473  
 Mylopoulos, John 350, 382, 502  
  
 Nederstigt, Lennart J. 334  
  
 Pastor, Óscar 701  
 Persson, Anne 662  
 Petrusel, Razvan 47  
 Pichler, Christian 222  
 Plexousakis, Dimitris 286  
 Polyvyanyy, Artem 64  
 Priego-Roche, Luz María 549  
 Puente, Gorca 646  
  
 Quinton, Clément 629  
  
 Rabhi, Fethi A. 111  
 Razo-Zapata, Iván S. 95  
 Reijers, Hajo A. 31  
  
 Rieu, Dominique 549  
 Rinderle-Ma, Stefanie 238, 398, 678  
 Rinner, Christoph 398  
 Rosenmüller, Marko 597  
 Rossi, Luca 160  
  
 Saake, Gunter 597  
 Sadiq, Shazia 429  
 Sánchez-Cuadrado, Jesús 127  
 Schäler, Martin 597  
 Schall, Daniel 460  
 Schmeling, Benjamin 80  
 Semenenko, Nataliia 31  
 Silva Parreiras, Fernando 366  
 Stirna, Janis 662  
 Syed Abdullah, Norris 429  
  
 Tamzalit, Dalila 175  
 Thom, Lucinéia Heloisa 549  
 Topaloglou, Thodoros 502  
 Truong, Hong-Linh 565  
 Tuyt, Wendelien 518  
  
 Uppström, Elin 190  
  
 van Aart, Chris 518  
 Vandić, Damir 334  
 Vanthienen, Jan 254  
 Vonk, Jochem 445  
  
 Weber, Stefanie 398  
 Weerd, Jochen De 254  
 Werthner, Hannes 222  
 Wieringa, Roel 697  
 Wohed, Petia 190  
  
 Yu, Jian 301  
  
 Zapletal, Marco 222  
 Zawawy, Hamzeh 350  
 Zhang, Minjie 473