

Witold Abramowicz (Ed.)

LNBIP 87

Business Information Systems

14th International Conference, BIS 2011
Poznań, Poland, June 2011
Proceedings

 Springer

Lecture Notes in Business Information Processing

87

Series Editors

Wil van der Aalst

Eindhoven Technical University, The Netherlands

John Mylopoulos

University of Trento, Italy

Michael Rosemann

Queensland University of Technology, Brisbane, Qld, Australia

Michael J. Shaw

University of Illinois, Urbana-Champaign, IL, USA

Clemens Szyperski

Microsoft Research, Redmond, WA, USA

Witold Abramowicz (Ed.)

Business Information Systems

14th International Conference, BIS 2011
Poznań, Poland, June 15-17, 2011
Proceedings

Volume Editor

Witold Abramowicz
Poznań University of Economics
Department of Information Systems
Al. Niepodległości 10
61-875 Poznań
Poland
E-mail: w.abramowicz@kie.ue.poznan.pl

ISSN 1865-1348

e-ISSN 1865-1356

ISBN 978-3-642-21829-3

e-ISBN 978-3-642-21863-7

DOI 10.1007/978-3-642-21863-7

Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011929366

ACM Computing Classification (1998): J.1, H.4, H.3

© Springer-Verlag Berlin Heidelberg 2011

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

BIS 2011 held in Poznań, Poland, was the 14th in a series of international conferences on business information systems. The BIS conference series has been recognized by professionals from the very beginning as a forum for the exchange and dissemination of topical research in the development, implementation, application and improvement of computer systems for business processes.

The theme of the conference was “Towards Flexible, Personalized and Adaptive Business Applications.” While the authors observed the need to make processes more flexible, they also identified risks related to losing control over processes. Therefore, the proceedings start with papers on business rules and business process verification. The necessity for process adaptation may lead to process variants; hence the session on business process variants and composition. The part devoted to business processes was summarized with a business process improvement session. Adaptability of business applications also requires a change in data models. The biggest challenge is posed by data integration. Several authors researched the potential of the Web and their findings are included in the Internet science session. The modern enterprises session was about moving business applications and enterprises beyond current limitations. Finally, there were several niche papers that were devoted to specific BIS issues.

Altogether, a set of 25 papers illustrating these trends were selected for presentation during the main event, grouped into 8 sessions. The Program Committee consisted of almost 100 members, who carefully evaluated all the submitted papers.

June 2011

Witold Abramowicz

Conference Organization

BIS 2011 was organized by Poznań University of Economics, Department of Information Systems.

Local Organization

Elżbieta Bukowska	Poznań University of Economics, Poland
Szymon Łazaruk	Poznań University of Economics, Poland
Piotr Stolarski	Poznań University of Economics, Poland
Krzysztof Węcel (Chair)	Poznań University of Economics, Poland

Program Committee

Witold Abramowicz	Poznan Univeristy of Economics, Poland
Hamideh Afsarmanesh	University of Amsterdam, The Netherlands
Harith Alani	KMi, The Open University, UK
Dimitris Apostolou	University of Piraeus, Greece
Morad Benyoucef	University of Ottawa, Canada
Gerardo Canfora	University of Sannio, Italy
Jorge Cardoso	University of Coimbra, Portugal
Silvana Castano	University of Milan, Italy
Michelangelo Ceci	Università di Bari, Italy
Wojciech Cellary	Poznan University of Economics, Poland
Houn-Gee Chen	National Taiwan University, Taiwan
Dickson K.W. Chiu	Dickson Computer Systems, China
Oscar Corcho	Universidad Politécnica de Madrid, Spain
Bin Cui	Peking University, China
Zhihong Deng	Peking University, China
Tommaso Di Noia	Politecnico di Bari, Italy
Peter Dolog	Aalborg University, Denmark
Johann Eder	University of Vienna, Austria
Suzanne Embury	University of Manchester, UK
Dieter Fensel	University of Innsbruck, Austria
Bogdan Franczyk	University of Leipzig, Germany
Ulrich Frank	University of Duisburg-Essen, Germany
Flavius Frasincar	Erasmus University Rotterdam, The Netherlands
Johann-Christoph Freytag	Humboldt Universität zu Berlin, Germany
Marko Grobelnik	Jozef Stefan Institute, Slovenia
Francesco Guerra	Università di Modena e Reggio Emilia, Italy

Jon Atle Gulla	Norwegian University of Science and Technology
Hele-Mai Haav	Tallinn University of Technology, Estonia
Axel Hahn	Carl von Ossietzky University Oldenburg, Germany
Stephan Haller	SAP Research, Germany
Martin Hepp	Bundeswehr University Munich, Germany
Knut Hinkelmann	University of Applied Sciences Northwestern, Switzerland
Marta Indulska	The University of Queensland, Australia
Marijn Janssen	Delft University of Technology, The Netherlands
Adam Jatowt	Kyoto University, Japan
Pontus Johnson	Royal Institute of Technology, Sweden
Pawel Kalczyński	California State University, Fullerton, USA
Ralf Klischewski	German University in Cairo, Egypt
Marek Kowalkiewicz	SAP Research, Australia
Helmut Krcmar	Technische Universität München, Germany
Daniel Lemire	Université du Québec à Montréal, Canada
Maurizio Lenzerini	Sapienza Università di Roma, Italy
Frank Leymann	University of Stuttgart, Germany
Chengfei Liu	Swinburne University of Technology, Australia
Peter Lockemann	Universität Karlsruhe, Germany
Peter Loos	IWi at DFKI, Saarland University, Germany
Alexander Löser	Technische Universität Berlin, Germany
Qiang Ma	Kyoto University, Japan
Zakaria Maamar	Zayed University, UAE
Leszek Maciaszek	Macquarie University, Australia
Zaki Malik	Wayne State University, USA
Yannis Manolopoulos	Aristotle University of Thessaloniki, Greece
Wagner Meira	University of Rochester, USA
Günter Müller	University of Freiburg, Germany
Markus Nüttgens	Universität Hamburg, Germany
Andreas Oberweis	Universität Karlsruhe, Germany
Mitsunori Ogihara	University of Miami, USA
Marcin Paprzycki	Polish Academy of Sciences, Poland
Eric Paquet	National Research Council, USA
Carlos Pedrinaci	KMi, The Open University, UK
Vassilios Peristeras	DERI Galway, Ireland
Jaroslav Pokorný	Charles University of Prague, Czech Republic
Elke Pulvermueller	University of Osnabrück, Germany
Manjeet Rege	Rochester Institute of Technology, USA
Hajo A. Reijers	Eindhoven University of Technology, The Netherlands

Ulrich Reimer	University of Applied Sciences St. Gallen, Switzerland
Riccardo Rosati	Sapienza Università di Roma, Italy
Gustavo Rossi	University of La Plata, Argentina
Massimo Ruffolo	ICAR-CNR, Italy
Shazia Sadiq	The University of Queensland, Australia
Demetrios Sampson	University of Piraeus, Greece
Ismael Sanz	Universitat Jaume I, Spain
Juergen Sauer	Carl von Ossietzky University Oldenburg, Germany
Alexander Schill	TU Dresden, Germany
Gheorghe Cosmin Silaghi	Babes-Bolyai University, Romania
Elmar Sinz	University of Bamberg, Germany
Kilian Stoffel	University of Neuchâtel, Switzerland
Darijus Strasunskas	Norwegian University of Science and Technology
Vojtech Svatek	University of Economics in Prague, Czech Republic
Sergio Tessaris	Free University of Bozen-Bolzano, Italy
Bernhard Thalheim	Christian Albrechts University Kiel, Germany
Barbara Thoenssen	University of Applied Sciences Northwestern, Switzerland
Robert Tolksdorf	Freie Universität Berlin, Networked Information Systems, Germany
Vassileios Tsetsos	University of Athens, Greece
Olegas Vasilecas	Vilnius Gediminas Technical University, Lithuania
Herna Viktor	University of Ottawa, Canada
Johanna Voelker	University of Mannheim, Germany
Hans Weigand	Tilburg University, The Netherlands
Hannes Werthner	Vienna University of Technology, Austria
Mathias Weske	University of Potsdam, Germany
Roel Wieringa	University of Twente, The Netherlands
Krzysztof Węcel	Poznan University of Economics, Poland
Seiji Yamada	National Institute of Informatics, Japan
Yun Yang	Swinburne University of Technology, Australia
Qi Yu	Rochester Institute of Technology, USA
Slawomir Zadrozny	Polish Academy of Sciences, Poland
John Zeleznikow	Victoria University, Australia
Jozef Zurada	University of Louisville, USA

Additional Reviewers

A

Afrasiabi Rad, Amir
Alhosban, Amal

B

Buschle, Markus

D

Dali, Lorand

F

Franke, Ulrik

H

Hashmi, Khayyam
Heise, David

K

Kopecky, Jacek

L

Lozano, Esther

M

Mayer, Manuel

N

Netjes, Mariska
Nowak, Alexander
Närman, Pia

O

Oro, Ermelinda

S

Sagolzaei, Mahdi
Schaaf, Marc
Schwering, Benjamin
Shadi, Mahdieh
Smaizys, Aidas
Suárez-Figueroa, Mari Carmen
Svab-Zamazal, Ondrej

T

Thalhammer, Andreas

Y

Yongchareon, Sira

Table of Contents

Session 1

Business Rules

Probabilistic Model Checking of Constraints in a Supply Chain Business Process	1
<i>Tamara Mendt, Carsten Sinz, and Olga Tveretina</i>	
Framework for Business Process and Rule Integration: A Case of BPMN and SBVR	13
<i>Ran Cheng, Shazia Sadiq, and Marta Indulska</i>	
Flexible Workflows at Design- and Runtime Using BPMN2 Adaptation Patterns	25
<i>Markus Döhring, Birgit Zimmermann, and Lars Karg</i>	

Session 2

Business Process Verification

Behavioral Conformance of Artifact-Centric Process Models	37
<i>Dirk Fahland, Massimiliano de Leoni, Boudewijn F. van Dongen, and Wil M.P. van der Aalst</i>	
Framework for Business Process Verification	50
<i>Andreas Speck, Sören Witt, Sven Feja, Aneta Lotytc, and Elke Pulvermüller</i>	
A Query-Driven Approach for Checking the Semantic Correctness of Ontology-Based Process Representations	62
<i>Michael Fellmann, Oliver Thomas, and Bastian Busch</i>	

Session 3

Business Process Variants and Composition

Guaranteeing Soundness of Adaptive Business Processes Using ABIS ...	74
<i>Falko Koetter, Monika Weidmann, and Daniel Schleicher</i>	
Merging Business Process Variants	86
<i>Wassim Derguech and Sami Bhiri</i>	
Empirical Validation of MoDe4SLA; Approach for Managing Service Compositions	98
<i>Lianne Bodenstaff, Andreas Wombacher, and Manfred Reichert</i>	

Session 4

Business Process Improvement

Towards a Goal-Driven Approach for Business Process Improvement Using Process-Oriented Data Warehouse 111
Khurram Shahzad and Constantinos Giannoulis

Business Process Optimization Using Formalized Optimization Patterns 123
Florian Niedermann, Sylvia Radeschütz, and Bernhard Mitschang

Facilitating Business Process Improvement through Personalized Recommendation 136
Mukhammad Andri Setiawan, Shazia Sadiq, and Ryan Kirkman

Session 5

Data Modelling and Integration

System Architecture for Handling the Information Overload in Enterprise Information Aggregation Systems 148
Philipp Katz, Torsten Lunze, Marius Feldmann, Dirk Röhrborn and Alexander Schill

Automated Process Decision Making Based on Integrated Source Data 160
Florian Niedermann, Bernhard Maier, Sylvia Radeschütz, Holger Schwarz, and Bernhard Mitschang

Making Decision Process Knowledge Explicit Using the Decision Data Model 172
Razvan Petrusel, Irene Vanderfeesten, Cristina Claudia Dolean, and Daniel Mican

Session 6

Internet Science

Sentiment Lexicon Creation from Lexical Resources 185
Bas Heerschop, Alexander Hogenboom, and Flavius Frasinca

Matching Organizational Structure and Social Network Extracted from Email Communication 197
Radosław Michalski, Sebastian Palus, and Przemysław Kaziemko

Application Specific Communication Stack for Computationally Intensive Market Research Internet Information System 207
Peter Kurz and Andrzej Sikorski

Session 7**Modern Enterprises**

An Approach to the Semantization of ERP Systems	218
<i>Robert Andrei Buchmann, Radu Meza, and Delia Pulcher</i>	
Open IT for Business: Transforming Information System Infrastructure with a Commercial BPM Suite	230
<i>Sava Mintchev</i>	
The Logistics Service Engineering and Management Platform: Features, Architecture, Implementation	242
<i>Christopher Klinkmüller, Robert Kunkel, André Ludwig, and Bogdan Franczyk</i>	

Session 8**Specific BIS Issues**

Cooperative Semantic Document Management	254
<i>Joerg Leukel, Michael Schuele, Andreas Scheuermann, Dominic Ressel, and Wiltrud Kessler</i>	
Deploying an Agent Platform to Automate the IT Infrastructure Auditing Process	266
<i>Ana-Maria Ghiran, Gheorghe Cosmin Silaghi, and Nicolae Tomai</i>	
Modeling Support for Confidentiality and Integrity of Object Flows in Activity Models	278
<i>Bernhard Hoisl and Mark Strembeck</i>	
Inventory Management with Dynamic Bayesian Network Software Systems	290
<i>Mark Taylor and Charles Fox</i>	
Author Index	301

Probabilistic Model Checking of Constraints in a Supply Chain Business Process*

Tamara Mendt^{1,2,3}, Carsten Sinz¹, and Olga Tveretina¹

¹ Institute for Theoretical Computer Science,
Karlsruhe Institute of Technology, Germany

² SAP Research Center, Karlsruhe, Germany

³ Universidad Simón Bolívar, Venezuela

05-38546@usb.ve, carsten.sinz@kit.edu, olga@ira.uka.de

Abstract. Business process models represent corporate activities, their dependencies and relations, as far as they are needed to reach a specific company goal. In practice, they often exhibit stochastic behavior, e.g., to deal with uncertain information. In this paper, we consider the problem of verifying properties over business processes that deal with such uncertain information. We employ a probabilistic model checking algorithm for verification, and demonstrate the applicability of this approach by a case study. Modeling and verification is achieved using the model checking tool *PRISM*. Based on the results, general specifications for modeling business processes using a probabilistic model checker are identified. Also, the difference between declarative and procedural business process modeling approaches is discussed. We propose to combine declarative and procedural techniques, thereby gaining increased expressiveness in modeling business processes, but still maintain verification feasible.

Keywords: Supply chain business processes, compliance, probabilistic model checking, PRISM.

1 Introduction

Business processes (*BP*) are collections of coordinated activities or tasks that will lead to accomplish specific organizational goals. The correctness, effectiveness and efficiency of a business process are vital for any organization. To ensure the quality of the service or product resulting from a business process, each company works under a series of policies that must hold throughout the entire runtime of the process.

Since flaws in the design of a business process can mean significant losses for an organization, it is important to detect these flaws before a business process is put into practice [1]. This means companies need to focus on ensuring that their businesses are compliant with the regulations that govern them. Therefore, it is

* This work was supported in part by the “Concept for the Future” of Karlsruhe Institute of Technology within the framework of the German Excellence Initiative.

necessary to provide business process modeling software with tools that allow the business experts designing a process to specify and check constraints within their process model [2].

The applicability of model checking techniques to verify compliance of business processes has been studied since the mid ninties [3]. In first studies the model checking tool *SPIN* was used to verify properties expressed in temporal logics. Since then, model checking has been mainly used to ensure soundness and consistency in business process specifications [4] and to ensure compliance of *BPs* with security requirements such as authorization constraints [5]. Studies have also focused on making model checking techniques accesible to users lacking knowledge in formal methods [6] and more recently, on determining the requirements for a comprehensive support of semantic constraints [2]. However, the approaches taken so far generally assume that the behavior of the business process over time is predictable.

In this paper we consider the problem of verifying properties of business processes that contain uncertain information, e.g. variable control flow, or resource assignment following a probabilistic distribution. The goal of our research is to determine the feasibility of implementing probabilistic model checking in a business process design and, at the same time, set the theoretical basis necessary to implement this verification process. Modeling is made up of two phases: (1) modeling of the business process as a stochastic process, and (2) formalizing and checking consistency constraints of the model.

Probabilistic model checking is used as a tool to provide more precise feedback related to the compliance of business processes, i.e. the probability of constraints holding throughout the whole execution. By keeping a record of past runtime data, i.e. executed business process instances, the probability distribution of the random variables involved in the business process can be updated making the results obtained from the model checking process more reliable with each process execution.

To show the applicability of probabilistic model checking in a business process context, we have considered the problem of variable prediction in a supply chain.

The rest of the paper is organized as follows. Section 2 gives a short introduction to supply chain management systems. In Section 3 we present the basic (probabilistic) model checking notions, continuing with Section 4, where we discuss the probabilistic model checking tool *PRISM* which is used throughout our study. Section 5 presents our case study, and Section 6 shows how to model it using the tool *PRISM*. Finally we present the results of our experiments in Section 7 and conclude in Section 8.

2 Process-Driven Supply Chain Management

A supply chain can be defined as a sequence of business processes and information that provides a product or service from suppliers, through manufacturing and distribution to the ultimate customer. Supply chain management (*SCM*) involves coordinating and integrating these flows both within and among the organization.

In recent years the interest over *SCM* systems has grown considerably since companies have noticed that they must rely on effective supply chains or networks, to compete in the global market and networked economy. Because of this, the companies which build Enterprise Application Software are focusing on implementing software systems specifically designed for *SCM*. These software building companies must seek to include desirable and innovative features in *SCM* systems so that their final product will be preferred over that of competing enterprise application software building companies.

This study has emerged in the search to provide new features to *SCM* systems, hence the case studies have been designed in a supply chain context. It can however be extended to business processes applied to different areas.

3 Probabilistic Model Checking

Model checking is a technique to verify formally finite state systems [7]. The essential idea behind model checking: A model-checking tool accepts system requirements or design (called models) and a property (called specification) that the final system is expected to satisfy. The tool then outputs yes if the given model satisfies given specifications and generates a counterexample otherwise.

Whereas model-checking techniques focus on the absolute guarantee of correctness, in practice such rigid notions are difficult to guarantee. Instead, systems are subject to various phenomena of a stochastic nature making the correctness become less absolute. The key point of probabilistic model checking is the ability to combine probabilistic analysis and model checking in a single tool [8].

In the scope of this work the probabilistic systems that will be used are discrete time Markov chains (*DTMCs*) and Markov decision processes (*MDPs*). Formally, a *DTMC* can be defined by a tuple $M = (S, s_0, T, AP, L)$ where S is a set of states, $s_0 \subseteq S$ corresponds to the initial set of states, $T : S \times S \rightarrow [0, 1]$ is a transition relation such that $\forall s \in S, \sum_{s' \in S} T(s, s') = 1$, AP is a set of atomic propositions, and finally $L : S \rightarrow 2^{AP}$ is a labeling function.

Markov decision processes are generalizations of Markov chains which allow a system to present non-determinism. For each state there may exist more than one probability distribution for transitions to next states.

4 The PRISM Model Checker

PRISM (PRobabilistic and Symbolic Model checker) is a tool for the modeling and analysis of systems which exhibit probabilistic behavior [9]. *PRISM* was chosen among several probabilistic model checking tools because of its user friendly interface and overall benefits related to model checking times and expressiveness.

In *PRISM*, system modeling is accomplished using a module-based language which basically consists of state variable declarations and transitions between states. This tool only allows the use of boolean and integer variables and they can be declared as follows:

```
var1 : bool init false;
var2 : [0..10] init 1;
```

The first statement declares a boolean type variable which initially takes the value false. The second statement declares an integer type variable with initial value 1, and which can be assigned values between 0 and 10. The global state of a system is determined by the current value of the variables.

In our approach to *BP* modeling using *DTMCs* or *MDPs*, system states are defined by a task of the business process and the resource assignment at the moment of execution of the task. We use an integer value to represent the different process tasks and generally require a separate variable for each resource involved in the process as well.

State transition rules are specified using the guarded commands of the form:

```
<guard> -> <command>;
```

where guard is a predicate over system variables (representing a state or set of states) and <command> is the transition executed by the system if the guard command evaluates to true (if the current state matches a state of the set determined by the guard). If the transition must be chosen probabilistically, the discrete probability distribution is specified as follows:

```
<guard> -> p1:<command1> + p2:<command2> + ... + pN:<commandN>;
```

Transition represented by `commandi` is executed with probability `pi`, and the sum of the probabilities must equal 1.

The *PRISM* property specification is based on temporal logics *PCTL* and *PCTL**. Temporal logics are used to reason about the temporal order of state transitions in a system. Formulae in temporal logics include temporal operators such as “eventually” (F), “always” (G) and “next” (X) which are used respectively to indicate that an atomic proposition will eventually be true, always be true, become true in the next transition. *PCTL* and *PCTL** include a probabilistic operator P, used to reason about the probability of a system property holding. This operator can be used to determine whether the probability of a path property holding is between some bounds (qualitative approach), or to calculate the numerical value of the probability of the property holding (quantitative approach).

5 Variable Prediction in a Supply Chain

Forecasting is widely used in supply chain management to predict the outcome of variables given a process instance, and optimize the subprocesses involved in the supply chain. We use probabilistic model checking as a forecasting technique to determine the probability of a certain event happening or property holding throughout the process. This can provide *BP* modelers with information to help make design decisions and changes in a process considering the possible risks the process may be exposed to.

5.1 Case Study

The case study which has served as the basis for our study consists of a workflow including non-determinism. Since the nature of the business process is not entirely procedural or declarative, no formal business modeling language was used in the modeling of the process.

Company *X* produces and distributes frozen foods to different retailers. Every lot of a specific product must first go through packaging and then delivered to a retailer. In between products can be stored in warehouses.

The supply chain consists of three phases: packaging, storage and distribution. In each of these phases quality loss of the products can occur. Clearly it is in the interest of the company to keep the final quality as high as possible, hence it is important to be able to predict this outcome.

For each phase of the supply chain, the company has more than one execution option. Each phase is explained in detail as follows:

- **Packaging:** this is the technology of enclosing and protecting products for storage, distribution and sale.
 - *Simple packaging:* this is the cheapest alternative, however, it often lacks the capacity to fulfill packaging demand. Since this option takes place internally, it generally provides highest quality.
 - *Co-packaging:* another company is hired to execute the packaging process. The cost is higher, as well as the required time, but the quality maintains similar.
 - *Labeling:* this option is cheaper and generally less time consuming than co-packaging since the company providing the service has more flexibility in the execution of the process. However the quality of the products diminishes since the process is not necessarily the same as with the previous two options.
- **Storage:** The longer a frozen product remains in storage, the higher the probability is of its quality decreasing.
- **Delivery:** the regular delivery options requires more time and is more likely to have a negative effect on the quality of products. The express delivery is only used if it is very important to minimize the delivery time.

5.2 Supply Chain

Figure [1a](#) shows the basic schema of the supply chain of company *X*. Figure [1b](#) shows a more restricted model of the supply chain process, including constraints over the execution order between tasks.

For every phase in the supply chain, there are two variables related to quality loss: the percentage products which cannot be sold due to significant quality loss and an abstract measure related to the overall quality of a lot of products. Company *X* keeps record of the quality loss and time consumption which occur in each phase, making it possible to determine probability distributions for the

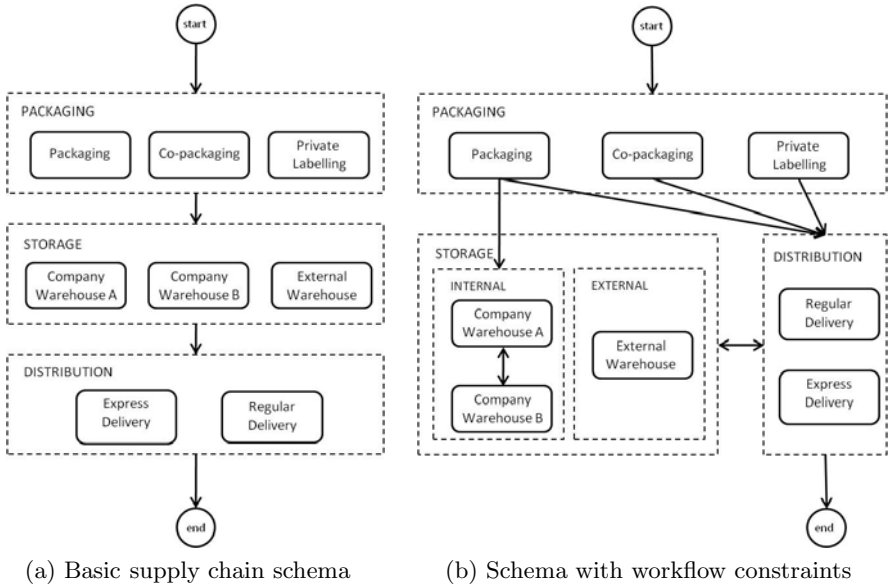


Fig. 1. Case study: phases of the supply chain: packaging, storage and distribution

resulting values of time and quality associated variables at the end of every phase.

Since Markov chains feature the memoryless property, variable dependency was avoided in the design of the use case scenarios, and the probability distributions for time consumption and quality loss in each phase are independent of the previous execution of the process. For the storage phase however, the quality related variables are dependent of the time a lot of products spends under storage.

The specified business process has an adhoc nature since the execution flow is not fixed in the design. Given that the variables are probabilistically distributed it is not possible to foresee the exact time the process will require or the final quality of the product even for a specific business process instance. Similar to constraint based workflow models [10], probabilistic model checking can be of use during design time and run time of the process:

- If a specific route instance has been specified for a certain lot of products: how long will the lot remain under way and what will the final quality measures of the products will be?
- Whether the probability of keeping quality throughout the whole process is one, independently of the route chosen.
- Whether new workflow constraints are consistent with each other (no deadlocks), and also that the presence of these constraints still ensures that the expected quality maintains over the company defined bounds.

6 Modeling the Case Study with PRISM

The system is represented as a Markov decision process (*MDP*) because the workflow is not fixed. The *MDP* consists of four modules:

- The first is the main module, or workflow module. An integer variable `task` determines the current task which is being carried out in the supply chain process, and transitions of this variable mark the workflow of the process.
- The remaining modules correspond to the time, quality loss and damaged goods percentage distributions associated with each task.

Since we are interested in obtaining total resource values (total time consumption, total damage percentage, etc), we have two options.

1. To use a variable which ranges from the minimum of the ranges of the value of the resource for each of the process phases to the sum of all the maximums.
2. To use formulas, which are expressions in *PRISM* composed of system variables, combined with operators (not including temporal modalities). These expressions are dynamic in the sense that if a variable is updated, the formulas including that variable will be updated simultaneously.

In our model we have used the first option for the variable `quality_loss`, and the second option for the variables `time` and `damage_percentage`. We have preferred the use of formulas for these last two resources because the values of these resources for each process phase are different, and not necessarily integer. Formulas allow us to associate integer variable values to non-integer values. Table [1](#) summarizes the variables used in the *MDP* model.

Probabilistic model checking in this study case was applied as a final time and quality prediction technique. The properties which we wished to check were modeled using *PCTL* and *PCTL** and have been classified in three classes. These properties were verified using *PRISM* and the results of the model checking process will be mentioned in the following section.

- **No path restrictions:** if the property must hold for *any* possible execution route, that is, for any allowed sequence of tasks, only information regarding the total time, and quality loss values is required. Properties of this class include:
 - $(G \text{ totaldamage} > x \ \& \ \text{totaltime} < y)$
 - $(G \text{ quality_loss} < x \ \& \ \text{totaltime} < y)$
 These properties state that throughout the entire process the percentage of undamaged goods must stay above a bound x , and the quality loss must stay below a given bound, respectively.
- **Task inclusion restrictions:** these properties present restrictions over which tasks are executed (or not executed) during the process. However, the execution order of these tasks is left unspecified. Two properties were defined in this class:
 - $(G \text{ totaldamage} < x \ \& \ \text{totaltime} < y) \ \& \ (F \ p1 \ \& \ s1 \ \& \ s2)$
 - $(G \text{ totalqual} < x \ \& \ d_1 != 2 \ \& \ d_2 != 2 \ \& \ d_3 != 2)$

Table 1. Case study: variables used for *MDP*

Identifier	Type / Range	Description
task	int:[0..4]	Represents the workflow of the model task=0 packaging phase task=1 storage phase, only company warehouses task=2 storage phase, only external warehouses task=3 delivery phase task=4 process has ended
p1; p2 p3; s1 s2; s3	bool	Boolean variables initialized with false which change to true if the packaging or storage option associated to them is used
aux	int:[0..3]	Auxiliary boolean variable which enforces that time is assigned before quality, for the storage phase. aux=0 not storage phase aux=1 storage phase, company warehouse A aux=2 company warehouse B aux=3 external warehouse
d_1 d_2 d_3	int:[0..2]	The delivery phase takes place at most three times during the process. Every time delivery is used, related time and quality information must be saved. There is a variable for each possible delivery execution. d_i=0 delivery phase has not happened i times d_i=1 normal delivery used for i-th delivery option d_i=2 express delivery used for i-th delivery option
damage_p damage_s1 damage_s2 damage_s3 damage_d_1 damage_d_2 damage_d_3	int:[1..2]	Variables associated to percentage of damaged goods for each process phase. Packaging takes place only once: one variable damage_p Storage: one variable for each storage option Delivery: one variable for each delivery execution
quality	int:[0..12]	Quality loss ranges from 0 to 2 for each phase option. Instead of one variable for each quality, one global quality loss variable is updated with each task.
time_p time_d_1 time_d_2 time_d_3	int:[1..2]	Variables associated to time for each process phase. Analogous to damage variables
time_s1 time_s2 time_s3	int:[1..3]	Variables associated to time for each process phase. Analogous to damage variables. Larger range.

The first specifies that packaging option $p1$ and storing options $s1$ and $s2$ are used. The second specifies that express delivery is never used, only normal delivery.

- **Task execution order restrictions:** these properties specify restrictions regarding the temporal dependencies between tasks. These restrictions over execution order do not include those which are explicitly specified in the workflow. Two properties were defined and verified in this class:
 - $(G \text{ totaldamage} > x) \ \& \ (G \text{ ("labelling" } \Rightarrow (X \text{ "express_delivery"})))$
 - $(G \text{ totaltime} < x) \ \& \ (G \text{ ("external warehouse" } \Rightarrow (X \ G \ !\text{"normal_delivery"})))$

The first property indicates that immediately after labelling, express delivery must be used. The second property states that after storing products in an external warehouse, no normal delivery can be used, only express delivery.

7 Experimental Results

We refer an interested reader to [11] for the full details of the experiments.

The large size of the *MDP* model is caused by 1) the requirement of including a time and damage variable for each storage and delivery option, and 2) the presence of non-determinism in the model. Depending on the path restrictions, it is possible to reduce non-determinism by explicitly including restrictions in the model, instead of representing them exclusively with *PCTL/PCTL** properties.

Table 2 presents the effect of the variable ranges over the size of the *MDP*. Reducing variable ranges reduces the size of the model significantly.

Table 2. Comparison between model sizes with different variable ranges

model	# states	# transitions	model size	construction time	relation with original
original	26333953	83277168	2,19302E+15	0,112	–
$ R(\text{quality_loss}) = 2$	17634849	46939712	8,27775E+14	0,069	2,65
$ R(\text{time_s}) = 2$	8464793	25885512	2,19116E+14	0,046	10,01
$ R(\text{damage_s}) = 1$	4571281	13385304	6,11880E+13	0,105	35,84
$ R(\text{quality_loss}) = 1$	2535017	4811992	1,21985E+13	0,064	179,78
$ R(\text{damage}) = 1$	391761	1091376	4,27559E+11	0,082	1936,05

Tables 3 and 4 show the effect of reducing non-determinism in the model. The first tables shows the comparison between model dimensions and the second compares model checking time and memory requirements for each model. The confidence used for the upper and lower bounds in the second table is 0.01 and the time is assumed to have a normal probability distribution. A tradeoff is made between formal semantics of constraint modeling and time and memory requirements of the model checking process.

The results obtained when running *PRISM* with the *DTMC* corresponding to a model with the entire path specified are depicted in Figure 2. Figure 2a is a

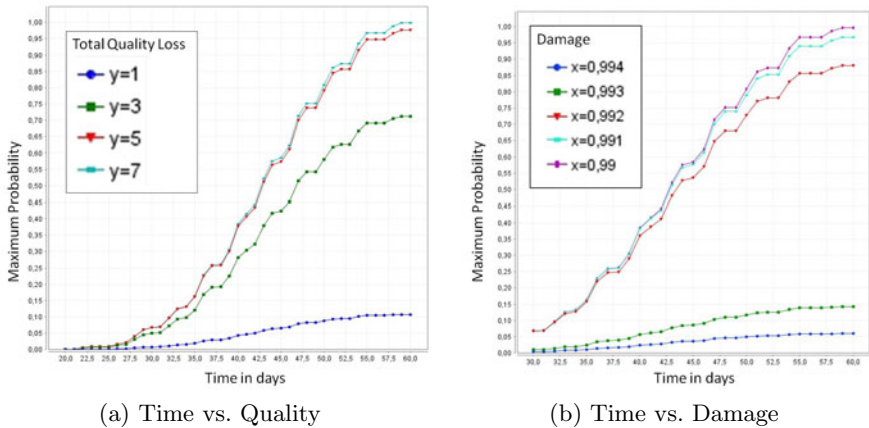
Table 3. Comparison between model sizes with different levels of non-determinism

model	# states	# transitions	model size	construction time	relation with original
original	26333953	83277168	2,1930E+15	0,112	–
simple packaging	8894633	27718904	2,4654E+14	0,096	8,89
copackaging	8719661	27779132	2,4222E+14	0,094	9,05
normal delivery	4549313	15519856	7,0604E+13	0,092	31,06
no non-determinism	901965	2760204	2,4896E+12	0,039	880,87

Table 4. Model checking time and memory requirements for models with different levels of non-determinism

property	original model		model without non-determinism	
	MTBDD	time(min)	MTBDD	time(min)
no path restrictions	664,3 MB	22,4 ± 0,4		
simple packaging	681,6 MB	1050 ± 20	235,0 MB	7,4 ± 0,1
copackaging	677,1 MB	518 ± 8	227,7 MB	5,2 ± 0,2
normal delivery only	662,6 MB	21,8 ± 0,5	124,4 MB	3,68 ± 0,06
complete path specified	666 MB	97 ± 1	17,8 MB	0,29 ± 0,01

graph of the probability of the property ($G \text{ totaltime} < x \ \& \ \text{quality_loss} < y$) holding for different values over the *DTMC*. The graph in Figure 2b is for the total time and total damage related variables. These graphs provide visualization of the applicability of probabilistic model checking and the power of the *PRISM*.

**Fig. 2.** Prediction results obtained from model checking properties over a DTMC

8 Conclusions and Future Work

We have shown that it is possible to model business processes through *DTMCs* and *MDPs* using *PRISM*. Even more, when the randomly distributed variables associated to a business process vary within a small range of values, verification of *PCTL* and *PCTL** properties over a *BP* is feasible in terms of time and memory requirements.

Although by definition, *DTMCs* and *MDPs* present the memoryless property, the inclusion of the conditional operator in the *PRISM* language and module concurrency make it possible to simulate dependency between the probability distribution of the transitions parting from a given state and the path followed by the system to reach that state. This is of particular interest for us, since business processes tend to present dependencies between the tasks which compose them. However, using this type of dependencies increases the size of a system model significantly.

It was shown that including non-determinism in the workflow of a model makes the size of the model grow exponentially. To maintain model checking feasible, the system model may need to be modified in terms of the *PCTL/PCTL** property. This however, makes automating system and property modeling quite challenging, since they cannot be considered separately.

This study is a first approach towards probabilistic model checking applied to business processes. The expressiveness provided by the temporal logics *PCTL* and *PCTL** will allow the extension of this study towards verifying compliance of other types of business process related constraints, as demonstrated in [11]. Future studies in this direction involve applying the modeling approach and specifications established in this first research to real world processes.

We have made a distinction between a procedural *BP* modeling approach and a declarative *BP* modeling approach. In the first approach the workflow is fully specified, leaving not many possibilities for process variants. Whereas in the second approach, *BP* workflows are defined through a pool of tasks and a set of constraints which determine the temporal dependencies between the tasks. The flexibility provided by declarative business process modeling helps avoid over specification of a business process, and allows a very large number of process variants for one *BP*. On the other hand, when treating highly constrained business process workflows, the declarative modeling approach is not recommended; because model checking procedures become expensive or even infeasible [12].

The standard Business Process Modeling Notation (*BPMN*) [13] takes a first approach to merging both modeling techniques, by including the specification of "ad hoc" subprocesses in a business process. We propose to follow this combination of procedural and declarative techniques in an attempt, not only to provide greater flexibility to designers of a business process, but to facilitate the incorporation of model checking tools in business process engines.

Integrating a *BPM* engine with the model checker *PRISM* is not trivial, but due to the simplicity of the *PRISM* language and interface, it is believed, that this integration is feasible. Future studies made in this direction will be directed to identifying the requirements for this type of integration such as determining formal semantics for *BP* designers to include the uncertain information, required

for modeling a probabilistic system, in the process design. Since automation of system translation is not trivial, we propose to study advanced techniques such as machine learning, as a tool for translating *BPs* to mathematical models.

Acknowledgements. The first author thanks Dr. Andreas Schaad and Dr. Phillip Miseldine from the Security and Trust Group at the SAP Research Center for their support.

References

1. Wynn, M.T., Verbeek, H.M.W., van der Aalst, W.M.P., Hofstede, T.A.H.M., Edmond, D.: Business process verification - finally a reality! *Business Process Management Journal* 15(1), 74–92 (2007)
2. Ly, L.T., Göser, K., Rinderle-Ma, S., Dadam, P.: Compliance of semantic constraints - a requirements analysis for process management systems. In: 1st Int'l Workshop on Governance, Risk and Compliance - Applications in Information Systems, Montpellier, France (2008)
3. Janssen, W., Mateescu, R., Mauw, S., Fennema, P., van der Stappen, P.: Model checking for managers. In: Dams, D.R., Gerth, R., Leue, S., Massink, M. (eds.) SPIN 1999. LNCS, vol. 1680, pp. 92–107. Springer, Heidelberg (1999)
4. Lu, R., Sadiq, S., Governatori, G., Yang, X.: Defining adaptation constraints for business process variants. In: Abramowicz, W. (ed.) *Business Information Systems. Lecture Notes in Business Information Processing*, vol. 21, pp. 145–156. Springer, Heidelberg (2009)
5. Schaad, A., Lotz, V., Sohr, K.: A model-checking approach to analysing organisational controls in a loan origination process. In: SACMAT 2006: Proceedings of the Eleventh ACM Symposium on Access Control Models and Technologies, pp. 139–149. ACM Press, New York (2006)
6. Janssen, W., Mateescu, R., Mauw, S., Springintveld, J.: Verifying business processes using spin. In: Proceedings of the 4th International SPIN Workshop, pp. 21–36 (1998)
7. Clarke, E.M., Emerson, E.A., Sistla, A.P.: Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Trans. Program. Lang. Syst.* 8(2), 244–263 (1986)
8. Kwiatkowska, M., Norman, G., Parker, D.: Stochastic model checking. In: Bernardo, M., Hillston, J. (eds.) SFM 2007. LNCS, vol. 4486, pp. 220–270. Springer, Heidelberg (2007)
9. PRISM website (2010), <http://www.prismmodelchecker.org>
10. Pesic, M., Schonenberg, M.H., Sidorova, N., van der Aalst, W.M.P.: Constraint-based workflow models: Change made easy. In: Chung, S. (ed.) OTM 2007, Part I. LNCS, vol. 4803, pp. 77–94. Springer, Heidelberg (2007)
11. Mendt, T.: Probabilistic model checking in a business process context. Master's thesis, Institute of Theoretical Informatics at the Karlsruhe Institute of Technology (2010)
12. van der Aalst, W.M.P., Pesic, M., Schonenberg, H.: Declarative workflows: Balancing between flexibility and support. *Computer Science - R&D* 23(2), 99–113 (2009)
13. OMG Business Process and Modeling Notation specification version 2.0 (June 2010)

Framework for Business Process and Rule Integration: A Case of BPMN and SBVR

Ran Cheng¹, Shazia Sadiq¹, and Marta Indulska²

¹ School of ITEE, The University of Queensland, St. Lucia QLD 4072 Australia
{kelvinrc, shazia}@itee.uq.edu.au

² UQ Business School, The University of Queensland, St. Lucia QLD 4072 Australia
m.indulska@business.uq.edu.au

Abstract. In the current heavily regulated business world, organisations struggle to establish a consistent view of their policies and operating procedures. These are generally captured through process modeling and business rule modeling, resulting in separate representations of rules and actual practice. The separation of the two models leads to increased risk of non-compliance, as well as reduced benefits from process improvement initiatives. So far, little guidance exists for organisations who struggle to consolidate their existing process models and business rules into one holistic model. In this paper we propose a preliminary framework that provides an overarching scope for the consolidation of existing business process models and business rule descriptions. The framework is supported by a collection of mapping methods (currently based on BPMN and SBVR) that assist business modelers in identifying inconsistencies between their existing business process models and business rule repositories, and thus helps to establish a coherent view of organizational policies and procedures.

Keywords: BPMN, SBVR, business process, business rule, integration.

1 Introduction

Business process modeling and business rule modeling languages are two approaches to modeling of organizational policies and procedures. Recent empirical research, however, indicates that neither approach in isolation is sufficient to represent all required details [1, 2]. Accordingly, [3] posited that the integration of the two approaches would be fruitful for organisations. Accordingly, the main aim of this paper is to take the first step towards developing a framework that facilitates a holistic view of essential techniques and methods that can be deployed to establish a consistent view of business operations using a combination of the outputs of business process and business rule modeling.

Integrating the outputs of the two modeling approaches is a challenging task. First, business process models tend to be visual in nature, with most of the relevant information represented graphically. Business rules, however, tend to be text-oriented. Thus, integration of the outputs of the two approaches requires an information exchange format with minimal information loss. Second, process models differ from

business rules fundamentally as they have different composing elements. Third, they are designed for different purposes - process models describe how things should happen, whereas business rules describe what should happen. Finally, the overlap and inconsistencies between business process models and business rules also presents a significant challenge. In particular, a set of criteria is required that helps a business analyst to resolve identified overlaps and inconsistencies in a satisfactory manner.

Accordingly, in this paper we propose an initial framework for process model and business rule integration. To explore the framework and explore its related methods, we select, for demonstration purposes, two popular standards for process and rule modeling - namely Business Process Modeling Notation 2.0 (BPMN) [4] and Semantics of Business Vocabulary and Business Rules (SBVR) [5]. We select BPMN because of its expressive representation ability as well as industry support. We select SBVR because it defines the vocabulary and rules for documenting business facts and business rules, and, because in combination with BPMN it provides the highest level of representational power [3].

In the remainder of this paper, we first present a brief exploration of related topics. We then present the proposed integration framework and its constituent methods. These methods have been deployed in a proof of concept tool suite, using BPMN and SBVR.

2 Related Work

The two predominant approaches for modeling organizational policies and procedures, namely business process modeling and business rule modeling [6], have been the focus of much research over the last few decades. This resulted in many proposals of modeling notations. For example, FlowMake [7], YAWL [8], and BPMN 2.0 [4] for business process modeling, to name a few; and E-C-A Based Business Rules [9], AgFlow [10], RuleSpeak [11] and SBVR [5] for business rule modeling, among others.

Since the introduction of rule modeling languages, researchers have considered the integration of process and rule models. [12] suggested business rule modeling should be merged with business process modeling to help with the temporal information control for information systems development. [9] proposed that ECA rules should be used to help integrate different process modeling and workflow languages. [13] proposed an initial procedure model for integrated process and rule modeling. All of these papers propose top-down approaches for integration and suggest that the integration should happen from the design stage. While we agree with this view, in reality organisations may already have existing separate process and rule models. Along this argument, [14] proposed an annotation approach to apply the compliance rules to business process modeling. [15] incorporated control objectives rules into process model via Formal Contract Language. [16] further supported the possibility of integration by proposing a methodology to translate rule-base business contract constraints into a business process.

Overall, while some research has been integrated business process modeling and business rule modeling, there are no guidelines to integrate business processes and

generic business rules from implementation level (bottom-up) and handle the existing model repositories and rule bases.

3 Integration Framework

Figure 1 depicts the two approaches discussed above, *viz.* top-down and bottom-up. The importance of an approach that provides consistency across process model and rule description by design cannot be underestimated. In Figure 1, the left side indicates the top-down approach of [13]. However, most organisations already have large process model repositories as well as rule bases that have been developed historically and/or may have been inherited during acquisitions and mergers. As such, the main problem that organizations are faced with is how to integrate existing process models and rules to ensure that business requirements are consistent across the two and that the consistency can be sustained. Accordingly, the right side of Figure 1 depicts the bottom-up approach.

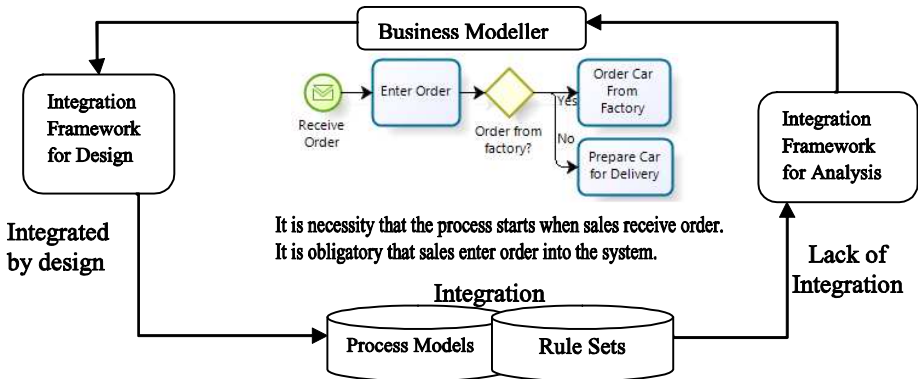


Fig. 1. Integration Framework for Design and Analysis

The bottom-up integration framework is built around a collection of mapping methods that provide distinct ways in which overlap and consistency (or lack of) between processes and rules can be studied. We observe two main aspects of the integration framework that can be found in most integration problems, namely, semantic and structural aspects:

Semantic Aspects: A prelude to studying the structural overlap/differences between process models and rule descriptions is to first identify and homogenize the semantics of the various elements used in the two models. For BPMN these include names of activities, roles, events, gateways, pools and lanes. For SBVR these include names, terms, verbs and keywords. It is clear that this is no trivial task and utilization of computational methods to overcome this challenge is incumbent upon the availability of a reference ontology that can provide the necessary relationships between common terms. In addition to establishing an agreement on the modeling labels, the semantic

aspects can also include issues relating to business goals, where there may be differences in the perceived goals of a process model and a rule set. In this paper, we do not consider the semantic aspects of the integration framework. Significant works that relate to term linkage and ontology development are already in existence [17-21].

Structural Aspects: It is clear that business and rule modeling notations have inherent structural differences. In order to study where these differences arise, we have identified and developed some key methods (see next section). These methods are not exhaustive but provide an initial set necessary to study the structural overlap and are demonstrated on BPMN and SBVR in this paper.

4 Approach

This research follows a Design Science [22] approach, where the artefact under development is the integration framework as well as the integration methods that are part of the framework. The notation agnostic framework was provided in previous section. In this section, we will present the integration method.

To begin the development of these methods an equivalence relationship had to be established between our two chosen notations, viz. BPMN and SBVR (see following section and Appendix 1) before any methods were developed. This process involved a construct-by-construct analysis of each of the two notations to determine where equivalences exist (details of which are omitted here due to page limitations). It was conducted by one researcher, discussed in a workshop-type setting, and then validated by another researcher so as to reduce the risk of bias. Once the set of construct equivalences were established, the researchers developed approaches for annotation and transformation of models. The methods then underwent a preliminary evaluation using a sample scenario.

Because BPMN is a visual process modeling approach while SBVR is a text based rule modeling language, a canonical intermediary is needed to bridge the gap between two representations. We adopt the XML Process Definition Language (XPDL) [23] as the intermediate format. XPDL is a format standardized by the Workflow Management Coalition (WfMC) for interchange of business process definitions between different workflow products. We select XPDL as the bridge between the BPMN and SBVR for two reasons. First, it fulfils the transformation from the BPMN graphic presentation into text representation. Second, it is standardized and well supported by many business process modeling tools.

Below we present two methods for mapping the structural aspects of the two representations, namely annotation-based and transformation-based methods. The former allows identifying the overlaps between two modeling approaches, while the latter allows performing the transformation between two modeling approaches. Before the methods are developed, however, it is important to first study the structural property of the respective modeling constructs. As per the methodology discussed above, Appendix 1 summarises the construct equivalence between the two chosen languages and use this equivalence in the subsequent method presentation.

The constructs presented have been extracted from the respective specifications of BPMN and SBVR. SBVR consists of Name, Term, Verb, Fact Type and Keywords [5]. Since Fact Type can be composed by Terms and Verbs, we remove Fact Type

from the SBVR constructs list. The keywords are further divided into Quantification keywords, Logical keywords, Modal keywords and other keywords, so the Business Rule can be composed from Name, Term, Verb and four different types of keywords.

In terms of BPMN constructs, [24] identified the most often used BPMN constructs. [25] further divided these common used BPMN into four groups: BPMN Common Core, BPMN Extended Core, BPMN Specialist Set and BPMN Overhead. We present the BPMN-SBVR mapping through a mapping table provided in Appendix 1. This table focuses on the most common BPMN constructs as identified in above research, namely Common and Extended Core.

In Appendix 1, the first column depicts the BPMN constructs. The second column is the XML tag used in XPD L for that corresponding BPMN construct. The last column depicts the instance of the corresponding SBVR SE construct and associated pattern used. This table is used to map BPMN constructs, XPD L elements and SBVR SE instances (or patterns) during annotation and transformation processes. During the annotation process, only three kinds of BPMN constructs can be annotated, which are Activities, Events and Gateways, thus annotation will only take place for the corresponding XML tags and the rest will be ignored. The annotatable constructs will be compared with SBVR statements to find a match, while non-annotatable constructs will bypass the matching process. During the transformation process, for each BPMN construct, either the corresponding SBVR SE instance or pattern is found. Then one or more connecting BPMN constructs can be automatically transformed into SBVR SE statement by organising the corresponding SBVR SE instance and pattern in order.

Connecting Objects are implicitly translated into SBVR SE constructs depending on the objects they are connected to. Artefacts Objects, Annotation and Group are used to visually organize the BPMN diagram and they are not really part of the process so they are not mapped to SBVR SE constructs. In contrast, SBVR SE modal keyword does not have straightforward corresponding BPMN construct. BPMN does not have moderation feature. BPMN is designed to express either the “happy path”, which means the way it should be to get a job finished; or the “backup path”, which means in case of problem, what should happen to continue the work. Either way, the pattern is used there is “If <condition> then <consequence>”. In other words, under the <condition>, it must be <consequence> and the modal keyword here is always “must”. At this stage, part from “must” or “It is obligatory”, there is no way to present other modal operation in BPMN. We consider this as a limitation of this work.

Pools and Lanes are two different graphic representations to describe the same concept – participants. During the transformation process, both Pools and Lanes are mapped to the same SBVR SE element <participant>. In BPMN, Pools and Lanes are both containers for entities. However, within XPD L, Pools and Lanes are used as containers only to present graphic information and they do not claim the ownership of inside entities, such as activities, events and gateways. Although virtually, events and gateways are organized by participant, the participant-entity relationship is carried by the coordination information not by the ownership information. Task is the only entity which can carry participant information. To associate a task with the corresponding Pool or Lane, a pair of elements (<participant> and <performer>) are used. For each <task>, there is a hidden element <performer>, which contains the ownership information. For instance, if <task> is performed within a <lane>, the <task> must have a <performer> attribute which matches to the <participant> of that <lane>.

We now present the procedures for the mapping methods. These procedures are further demonstrated in the next section through a scenario implemented in a proof of concept tool.

4.1 Annotation Based Mapping

In BPMN, Annotation can only be associated with Flow Objects, which including Events, Activities and Gateways [4]. The objective of this annotation method is to find out which BPMN constructs are indeed used for SBVR annotation and where the access points are in BPMN diagram to insert relative SBVR SE statement as annotation. The annotation process includes the following procedures:

Segmentation of SBVR SE statement: SBVR SE statements are composed of Name, Term, Verb and Keywords. Proper segmentation helps to find the right access point in BPMN. For example, the statement “It is a necessity that the process starts when sales receive order” is made up of the following segments: “It is a necessity that” (modal keyword), “process, sales” (term) and “starts, receive order” (verb).

After getting a list of segments from a statement, analyses need to be conducted to find the point of entry in BPMN. Using the example from last bullet point, all modal keywords are not mapped into BPMN and can be ignored. Regarding to other keyword, apart from “before/after” which can provide temporal information to help find BPMN point of entry, the rest of them can be ignored. “process” refers to the BPMN diagram not a particular BPMN element. So the segments left are “sales, starts and receive order”. After filtering, the point of entry for this statement in BPMN diagram is “message start event” “receive order”. An SBVR SE statement may have more than one point of entry in a BPMN diagram. It is acceptable to attach the statement to any of the BPMN elements, since the objective of this method is to find out the overlaps and differences not the best access point.

Note that “before” and “after” keywords can be used to indicate the temporal information. For simplicity, in this paper, all the SBVR statements only use keyword “after”. For example, this format will be used “Its necessity that operation A happens after operation B.” Using this convention, if there are more than one activities are found in a SBVR statement, we will assure the one comes first will be what we are looking for and the others will be used to express temporal information. In other word, the statement will be attached to the activity that appears first.

Algorithm 1: BPMN annotation

Input: BPMN diagram (XPDL file) and SBVR statements (text format)

Output: Annotated BPMN diagram (XPDL format)

Define ActivitySet A, StatementSet S

For each statement in S

 Let SmallestActivityIndex = -1, ActivityName = ""

 For each activity in A

 Let ActivityIndex = statement.indexOf(activity)

 If(ActivityIndex > 0 and smallestActivityIndex = -1), then

 smallestActivityIndex = ActivityIndex; and ActivityName = activity

 If(ActivityIndex > 0 and ActivityIndex < smallestActivityIndex), then

 smallestActivityIndex = ActivityIndex; and ActivityName = activity

 If (smallestActivityIndex > -1) then

 attach statement to Activity where Activity.name = AcitivityName.

4.2 Transformation Based Mapping

Based on appendix 1, the following procedure is used to transform BPMN (in XPDL) format into SBVR SE statements.

Algorithm 2: BPMN to SBVR SE transformation

Input: BPMN diagram (XPDL file)

Output: SBVR statements (text format)

Foreach activity in XPDL

 If activity.type = StartEvent

 Then output statement: "process starts on the condition of " +

 StartEvent.type

 If activity.type = EndEvent

 Then output statement: "process ends on the condition of " + EndEvent.type

 If activity.type = Gateway

 Then output statement: pre-condition + gateway.type + post-condition

 If activity.type = Task

 Then output statement

 activity.performer + "perform task" + activity.name + "after" + activity.source.

 activity.performer + "send message" + activity.message.name + "to" +

 activity.message.target

Foreach pool in XPDL

 If pool.message exist

 Then Output statement pool.name + "send message" + pool.message.name + "to" +

 pool.message.target.

As discussed in previous section, only "must" (or "It's obligatory that") modal keyword can be used in BPMN, thus we make all the statements obligatory.

5 Method Demonstration¹

Consider a Car Sales process, there are six parties involved: Customer, Sales person, Preparation person, Finance person, Factory and Lender. These six parties need to interact with each other to finish the process, from car order to car delivery. A BPMN model for the scenario is shown in Appendix 2, while SBVR rules are presented below:

1. It is necessity that the process starts when sales receive order.
2. It is obligatory that sales enter order into the system.
3. It is obligatory that the finance people arrange finance for the customer after sales enter order.
4. It is necessity that the sales check if they need order car from factory after they enter order.

¹ We have developed a proof of concept prototype tool to implement the mapping methods presented in this paper. The tool is developed using Java. It takes BPMN diagram in XPDL format, and automatically annotates and/or transforms it with/into SBVR SE. JDom was used as the XML parser. Below we provide the results obtained from the tool of the implemented methods based on the above scenario.

5. It is possible that the sales send factory message to check ship date.
6. It is possible that factory sends message to provide ship date.
7. It is obligatory that the sales confirm the ship date with the customer.
8. It is possible that if car is unavailable, the sales terminate the process.
9. It is necessity that preparation people prepare car for delivery after the sales confirm the ship date with the customer.
10. It is necessity that preparation people prepare car for delivery if no need to order car from factory.
11. It is obligatory that the finance people communicate with lender if finance is required.
12. It is possible that the process is terminated if finance is unavailable.
13. It is obligatory that finance people need to wait for both “Arrange Finance” and “Prepare car for delivery” ready to close the order and deliver the car.
14. It is obligatory that the sales notify the customer when deliver the car with temporary registration.
15. It is obligatory that finance people check customer’s credit before arrange finance.
16. It is necessity that finance people report to their manager after close and deliver the new car.
17. It is necessity that preparation people apply for temporary registration number after prepare car for delivery.

Applying the annotation process to the initial BPMN diagram, the annotated BPMN diagram is generated. By observing the Annotated BPMN diagram, we notice the following points. Firstly the SBVR SE statements for message flow are not annotated. Although in theory, these statements can be attached to the message flow in BPMN diagram, according to the BPMN specification, message flows cannot have annotation. Secondly, some of the SBVR SE statements used to describe this car sales scenario are not covered in the BPMN representation. However, our tool picked them up and annotated to the BPMN (the annotation shown in green colour). This shows that our method has the capability to find the missing spots in BPMN diagram.

With the transformation method presented in previous section, the following SBVR SE statements are automatically generated.

1. It is obligatory that Customer send message to start process.
2. It is obligatory that Customer send message Confirmation response to Sales.
3. It is obligatory that Factory send message Ship Date to Sales.
4. It is obligatory that Lender send message Loan response to Finance People.
5. It is obligatory that after close and deliver, finish the process, send message Deliver vehicle and temporary registration to Customer.
6. It is obligatory that Sales send message Confirmation request to Customer.
7. It is obligatory that Sales send message Factory order to Factory.
8. It is obligatory that Finance People send message Loan request to Lender.
9. It is obligatory that Sales perform task enter order after receive order.
10. It is obligatory that both if finance is unavailable? equals No and after task prepare car for delivery, execute task close and deliver.
11. It is obligatory that after order car from factory, execute car is unavailable?.
12. It is obligatory that if car is unavailable? equals Yes, terminate all process.

13. It is obligatory that if car is unavailable? equals No, execute task prepare car for delivery.
14. It is obligatory that after arrange finance, execute finance is unavailable?.
15. It is obligatory that if finance is unavailable? equals Yes, terminate all process .
16. It is obligatory that Finance People perform task arrange finance after enter order.
17. It is obligatory that after enter order, execute order from factory?.
18. It is obligatory that if order from factory? equals Yes, execute task order car from factory.
19. It is obligatory that if order from factory? equals No, execute task prepare car for delivery.

Most of the BPMN constructs (appendix 1) have corresponding SBVR constructs or combination of SBVR constructs. The transformation can be performed based on these overlaps without loss of information. However, some differences remain. Firstly, Connecting Objects (Sequence Flow, Message Flow and Association) only exist in BPMN but not in SBVR. Secondly, Artefacts Objects Annotation and Group only exist in BPMN but not in SBVR. Thirdly, although in this paper we do not consider the executable modeling which is new in BPMN 2.0, to our best knowledge, SBVR is not capable to adopt this new feature. Fourthly, very limited moderation operation can be used in BPMN.

6 Conclusion

We have proposed a bottom-up approach to process and rule integration to complement design approaches for integrated modeling of processes and rules. To this end we have developed two mapping methods within an overarching integration framework to support business analysts and modelers in studying the overlaps and differences between processes and rule repositories. The focus of this paper was not to analyse the respective expressive and notational capabilities of BPMN and SBVR. It can be observed that due to the inherent differences, there are a number of constructs (within the structural aspect) that cannot be mapped easily using either of the methods (annotation and transformation) presented above. However, we argue that the proposed methods provide 1) proof of feasibility of integration between process modeling and rule modeling; 2) implementable model which has been partially implemented (semantic aspects are not implemented yet). In the future, we plan to further explore the limitations we discussed in this paper. We will try to refine the method we proposed for annotation and transformation; develop the method to measure the difference between the BPMN and SBVR; add the semantic aspect and explore further the extent to which the automation of integration can be achieved.

References

1. Recker, J., Indulska, M., Rosemann, M., Green, P.: How good is BPMN really? Insights from Theory and Practice (2006)
2. Indulska, M., Recker, J., Rosemann, M., Green, P.: Business process modeling: Current issues and future challenges. Springer, Heidelberg (2009)

3. zur Muehlen, M., Indulska, M.: Modeling languages for business processes and business rules: A representational analysis. *Information Systems* 35(4), 379–390 (2010)
4. OMG. Business Process Model and Notation 2.0 Beta 1 Specification (2009), <http://www.omg.org/cgi-bin/doc?dtc?dtc/09-08-14> (cited June 1, 2010)
5. OMG. Semantics of Business Vocabulary and Business Rules (SBVR), v1.0 (2008), <http://www.omg.org/spec/SBVR/1.0/PDF> (cited June 2, 2010)
6. Lu, R., Sadiq, S.: *A survey of comparative business process modeling approaches*. Springer, Heidelberg (2007)
7. Sadiq, W., Orłowska, M.: On capturing process requirements of workflow based business information systems. In: 3rd International Conference on Business Information Systems, Poznan, Poland (1999)
8. Van Der Aalst, W., Ter Hofstede, A.: YAWL: yet another workflow language. *Information Systems* 30(4), 245–275 (2005)
9. Knolmayer, G., Endl, R., Pfahrer, M.: Modeling processes and workflows by business rules. In: *Business Process Management*, pp. 201–245 (2000)
10. Zeng, L., Ngu, A., Benatallah, B., O’Dell, M.: An agent-based approach for supporting cross-enterprise workflows. IEEE Computer Society, Los Alamitos (2001)
11. Ross, R., Lam, G.: RuleSpeak Sentence Templates: Developing Rules Statements Using Sentence Patterns. *Business Rule Solutions* (2001), <http://www.BRCommunity.com>
12. Krogstie, J., McBrien, P., Owens, R., Seltveit, A.: *Information systems development using a combination of process and rule based approaches*. Springer, Heidelberg (1991)
13. zur Muehlen, M., Indulska, M., Kittel, K.: *Towards integrated modeling of business processes and business rules* (2008)
14. Governatori, G., Hoffmann, J., Sadiq, S., Weber, I.: *Detecting regulatory compliance for business process models through semantic annotations*. Springer, Heidelberg (2009)
15. 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)
16. Milosevic, Z., Sadiq, S., Orłowska, M.: *Translating business contract into compliant business processes* (2006)
17. Nirenburg, S., Raskin, V.: *Ontological semantics*. MIT Press, Boston (2004)
18. Maedche, A., Staab, S.: Measuring similarity between ontologies. In: *Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web*, pp. 15–21 (2002)
19. Giunchiglia, F., Shvaiko, P.: Semantic matching. *The Knowledge Engineering Review* 18(03), 265–280 (2004)
20. Bilenko, M., Mooney, R., Cohen, W., Ravikumar, P., Fienberg, S.: Adaptive name matching in information integration. *IEEE Intelligent Systems* 18(5), 16–23 (2003)
21. Ullman, J.: Information integration using logical views. In: Afrati, F.N., Kolaitis, P.G. (eds.) *ICDT 1997*. LNCS, vol. 1186, pp. 19–40. Springer, Heidelberg (1996)
22. Hevner, A., March, S., Park, J., Ram, S.: Design science in information systems research. *Mis Quarterly* 28(1), 75–105 (2004)
23. WfMC. XML Process Definition Language Specification Version 2.1 (2008), <http://www.wfmc.org/xpdl.html> (cited April 3, 2010)
24. Muehlen, M., Recker, J.: *How much language is enough? Theoretical and practical use of the business process modeling notation*. Springer, Heidelberg (2008)
25. OPAALS, *Automatic Code Structure and Workflow Generation from Models*, OPAALS Project, European Community (2008)

Appendix 1. Proposed Mapping between BPMN, XPDL and SBVR SE

BPMN Construct	XPDL Element	SBVR SE Instance or Pattern
Pool and Lane	<Participant/>	<participant placeholder>
Task	<Task/>	<task placeholder>
Data-Based Exclusive Gateway	<Route/>	After <antecedent 1> or <antecedent 2> or...<antecedent N> then <consequent> with N arbitrary (as merge) After <antecedent> if <condition 1> then <consequent 1> with I generic index from 1 to arbitrary N(as split) After <antecedent 1> or <antecedent 2> or...<antecedent N> then <consequent> with N arbitrary (as merge) ² After <antecedent> if <condition 1> then <consequent 1> with I generic index from 1 to arbitrary n(as split) ³ After <antecedent 1> and <antecedent 2> and ... <antecedent N> then <consequent> with N arbitrary (as merge) After <antecedent> then <consequent 1> and <consequent 2>... and <consequent n> (as split) After <antecedent 1> then <consequent 1> and <consequent 2>... and <consequent n> (as split) N> then <consequent> (as merge) ³ After <antecedent> if <condition 1> then <consequence 1>, if <condition 2> then <consequence 2>... if <condition n> then <consequence n> (as split) ³ Start <task placeholder> After <message placeholder> is received, start <task placeholder> After <time placeholder>, start <task placeholder> If <condition>, then start <task placeholder> After <signal placeholder> received, start <task placeholder> End the process After end of the process, send <message placeholder> to <participant placeholder> After end of the process, broadcast <signal placeholder> Cancel the process On <error placeholder>, end the process. Terminate all the process. <file placeholder>
Event-Based Exclusive Gateway	<Route GatewayType="EventBasedXOR" />	
Parallel Gateway	<Route GatewayType="AND" />	
Inclusive Gateway	<Route GatewayType="OR" />	
None Start Event	<StartEvent Trigger="None" />	
Message Start Event	<StartEvent Trigger="Message" />	
Timer Start Event	<StartEvent Trigger="Timer" />	
Conditional Start Event	<StartEvent Trigger="Conditional" />	
Signal Start Event	<StartEvent Trigger="Signal" />	
None End Event	<EndEvent />	
Message End Event	<EndEvent Result="Message" />	
Signal End Event	<EndEvent Result="Signal" />	
Cancel End Event	<EndEvent Result="Cancel" />	
Error End Event	<EndEvent Result="Error" />	
Terminate End Event	<EndEvent Result="Terminate" />	
Data Object	<Artifact ArtifactType="DataObject" />	

Note: 1. In pattern <participant placeholder><verbPhrase><Task Placeholder>, Pool and Lane's names will replace the <participant placeholder> and Task's name will replace the <Task Placeholder>. 2. Compare to Inclusive Gateway, only one <antecedent> <consequent> pair will happen. 3. Compare to exclusive gateway, more than one <condition> <consequence> pairs can happen. The <antecedent> may vary, but the <consequence> always be the same.

Flexible Workflows at Design- and Runtime Using BPMN2 Adaptation Patterns

Markus Döhring, Birgit Zimmermann, and Lars Karg

SAP Research Darmstadt, Germany
{firstname.lastname}@sap.com

Abstract. Flexible workflow management systems facilitate the adaptation of workflows to changing data contexts. They are however usually not aligned with industry standards and often entail the peril of uncontrolled execution variant growth. Therefore we present a novel approach relying on BPMN2 and business rules for workflow adaptation at design- and runtime. For the specification of structural adaptations, we consolidate change-, exception- and time-patterns into one BPMN2 pattern catalogue as the main contribution of this paper.

Keywords: business process management, flexible workflows, workflow variants, business rules, change patterns, BPMN.

1 Introduction

Workflow management systems (WfMS) [1] are becoming an essential part of many industrial IT system landscapes [2]. For some domains, traditional WfMS have already been determined as unsuitable to cover prevalent requirements w.r.t. the flexibility of workflows [3]. The generic term *workflow flexibility* comprises a large variety of features discussed in industry and research. On a coarse granular level, two main flexibility types and their associated challenges can be identified. *Design time flexibility* aims at supporting the definition of multiple allowed execution traces depending on context conditions, i.e. workflow variants, *before* an instance has been started. Traditional WfMS typically only allow for a one-by-one modeling of variants and therefore foster the creation of redundant and hardly maintainable business logic. *Runtime flexibility* aims at changing the course of a workflow instance *after* it has been started in a way that it does not necessarily correspond to the underlying model anymore, for example due to unforeseen context changes. Existing “adaptive” WfMS exist, but entail the peril of potential uncontrolled variant growth and inconsistencies contradicting the original workflow paradigm.

The challenge can be subsumed under the research question: “How can we cope with variance and complexity in a WfMS by making just the degree of flexibility available which is required for a particular application scenario?”. A suitable combination of workflow, business rule and eventing concepts is considered as a potential solution to this question [4], because modeling problems can be decomposed this way. However, the increased expressiveness needs to be

guided [5] and flexibility mechanisms need to be easily available for different application scenarios. Therefore, the research objective addressed by this paper is to concretize on how different aspects of flexibility can be modularized and conveniently be made available to support specific application scenarios of WfMS, but without carrying it to excess by annulling the original intention of workflow [4]. Besides the copious special notations and workflow execution environments produced by researchers to tackle many of the above challenges, OMG’s BPMN2 specification [6] aims at unifying business and IT needs across industries in one notation. BPMN2 already contains broad facilities for event- and exception handling [7]. In parallel, notation- and implementation-independent behavioral patterns have been defined for several aspects of WfMS. Although other types of patterns exist in the domain of WfMS, change patterns [8], exception handling patterns [7] and time patterns [9] are considered most central for this work in terms of flexible control-flow specification. Accordingly, in Section 2 we present an approach for the context-dependent rule-based adaptation of BPMN2 workflow segments which covers both design- and runtime flexibility and can easily be adjusted to different flexibility needs. Second in Section 3, a large fraction of patterns from the above mentioned prior work is integrated and mapped to BPMN2 fragments as the main contribution of this work. They can be reused in the presented adaptation rules and hence combine pattern-based guidance in workflow flexibility with BPMN2 standard notation. A working prototypical implementation is discussed in Section 4. In Section 5 we discuss related approaches, before we conclude in Section 6.

2 Design- and Runtime BPMN Variant Creation

In this section, we present the foundation of our approach for covering design- and runtime flexibility. It consists of two main conceptual components, which relate to specifying which parts of a workflow can change at runtime and defining which adaptations need to be applied for a particular data-context of a workflow.

2.1 Adaptive Workflow Segments

Figure 1 contains a fictitious but realistic workflow for ship engine maintenance. In the upper segment of its parallel part, tasks for maintaining the ship’s cooling system are executed. In the lower part, engine startup tests and a subsequent lifetime analysis of the motor are conducted. For these tests, the engineers on the ship have to wait for an approval, since only few ships may run their engine in the harbor at the same time for environmental reasons. Then, spare parts which are permanently stored on the ships are inspected and eventually replaced, but only if the customer is solvent w.r.t. a positive credit report. If the service provision is canceled in some rare cases, it might be necessary to revoke even already replaced spare parts from the ship. Figure 1 also contains a list of data context variables for each workflow instance. Some of them are static, such as the ship

¹ In the most flexible workflow anything can happen at anytime.

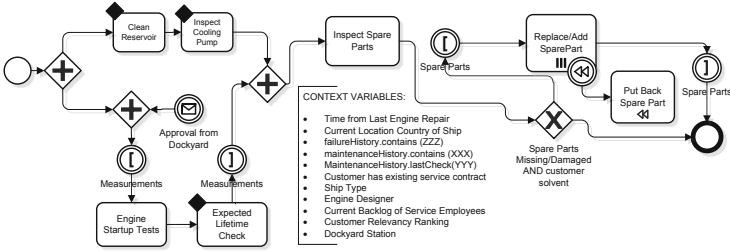


Fig. 1. Workflow for Ship Engine Maintenance Service Execution

type or its maintenance history, some of them may change during runtime, such as the backlog of service engineers. The maintenance workflow can be considered as a reference model, i.e. specifying the “regular” course of the workflow. In many situations, however, some adaptive segments of it need to be tailored to the concrete values of its context variables. We build upon the general framework of [10] for specifying *adaptive workflow segments*. These can be subject to structural adaptations at runtime. Analogously, we restrict an adaptive segment to be any block-structured partial graph of the eventually non-block-structured BPMN2 workflow, i.e. having only one entry and one exit point. This facilitates the modular definition of adaptation patterns we introduce in Section 3 and the checking of their consistent joint use. Figure 1 contains two alternative conventions²:

1. The adaptive segments can be marked with enclosing intermediate throw event nodes with opening or closing square brackets. $\textcircled{[}$ $\textcircled{]}$
2. An adaptive single task can be marked by a black diamond in its upper left corner, with same semantics as (1.), but being more space-saving. \blacklozenge

2.2 Application of Adaptation Patterns Defined in BPMN2

The execution semantics for adaptive segments are defined as follows: If an entry to an adaptive segment is signaled for a workflow instance, the context variables are evaluated and the segment eventually becomes subject to immediate adaptations. At each entry time to an adaptive segment, a consistent isolated variant segment is created. This “variant segment instance” does not change anymore even though a variable may change while the segment is executing. If so, the change is either ignored or more sophisticated checking and error resolution mechanisms have to be considered [10]. However, if the same adaptive segment is entered multiple times, e.g., via a cycle in the workflow, the execution semantics allow for the creation of multiple different variants of the same segment.

² Please note, that explicitly demarcating segments which can be subject to runtime adaptations is intended to restrict the uncontrolled growth of workflow execution variants at runtime. If required, full flexibility comparable to systems like ADEPT [3] or YAWL [12] can be emulated by marking each task as an adaptive segment.

The structural adaptations which can take place when entering an adaptive segment are defined in a pattern catalogue, where structural definitions also rely on BPMN2 semantics. Each pattern consists of a unique name and a description as well as an implicit input parameter $\langle AdaptationSegment \rangle$ relating to which workflow segment it is applied on. For some patterns, additional parameters relating to either BPMN2 model elements or element attributes are specified. Table 1 shows a simple example pattern for parallel insertion. Section 3 will elaborate on more sophisticated patterns as well as their interrelationships and correspondences to established pattern catalogues for flexible workflows.

Table 1. Basic Adaptation Pattern: Insert Element Parallel

Basic Adaptation Pattern BAP3: Insert Element Parallel		
<p>Description: When a segment of a workflow is entered, an additional element is activated concurrently. The finishing of the segment in terms of passing a token through its outgoing connector is delayed until the additional element has signaled completion.</p> <p>Example: In a maintenance workflow for a machinery, additional checks on the engine have to be performed while maintenance, for example if the engine stems from a specific engine manufacturer.</p> <p>Parameters: AdditionalElement(BPMN2:FlowNode)</p> <p>Constraints: BPMN2 syntactical constraints apply, e.g. a StartEvent may not be used in between sequence flows and is therefore excluded as a valid parameter.</p> <p>Reuses: -</p>		<p>Related Generic Patterns: AP1 8</p>
<p>Refines: BAP0</p>		

The connection between data-contexts and adaptation patterns can now be established by formulating business rules in an event-condition-action (ECA) format. The event consists in the entry event of an adaptive segment, the conditions constitute constraints on workflow context variables and the actions specify the application of patterns from the catalogue. They are at least parametrized with the adaptive segment belonging to the entry node of the rule’s triggering event. As such, the segment can simply be “wrapped” in an adaptation pattern by nesting the corresponding BPMN2 elements of a concerned workflow instance. A pseudo-syntax, where * stands for 0-n repetitions, can be defined as:

ON *entry-event* **IF** $\langle data-context \rangle$ **THEN APPLY** [$\langle pattern(segment, (parameter, value)^* \rangle$)*

Below, an example rule for the workflow in Figure 1 is presented. It extends its measurements phase to cover special occurrences in the maintenance history of the engine:

```
ON measurements_entry IF maintenanceHistory.contains(oilLeakageRepair)
THEN APPLY insert_parallel(segment='measurements_entry', additionalTask='integrityCheck')
```

2.3 Coverage of Flexibility Aspects at Design- and Runtime


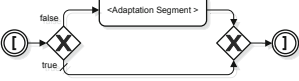


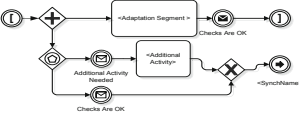


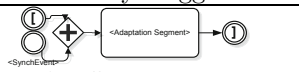

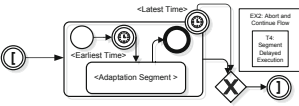


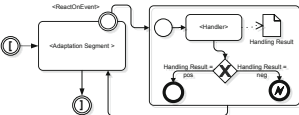
In summary, the introduction of explicit adaptive segments and adaptation rules to BPMN2 workflows provides a single integrated concept for realizing design- and runtime flexibility. At design-time, even a large number of workflow variants derived from one reference workflow can be conveniently managed. At runtime,

if the underlying implementation allows for dynamic changes of rulesets (as our prototype presented in Section 4 does), adaptations to workflow instances can be realized even after they have been started. By including an additional “instance selection” constraint in the condition part of a changed rule, it can be determined whether a change affects all instances resp. all instances started in the future (which is equal to a *permanent change* of the business logic) or only a single instance. With the latter concept, ad-hoc changes to workflows can be realized by letting users introduce single instance adaptation rules at runtime. Even more sophisticated flexibility concepts can be realized, like aspect orientation [11] by reusing adaptive segments of the same type within a model. Hierarchical variant models can be realized by restricting the allowed adaptation rulesets for a model to be a superset of another ruleset for the same model, similar to ripple-down rules [12]. However, adaptation rules alone do not yet provide sufficient guidance w.r.t. how the actual adaptations can be defined and applied for particular WfMS use cases, which is subject to the next chapter of this work.

3 Pattern Catalogue for Different Adaptation Purposes

The concepts presented in Section 2 provide a solid foundation for the coverage of the two flexibility types. It remains to be specified how concrete tailoring operations can be specified for invocation at workflow runtime. As introduced in Section 2.2, we express tailoring operations by parameterizable BPMN2 constructs. For the definition of patterns for tailoring operations, we aimed at covering and extending basic adaptation patterns (BAP) according to [8], time patterns (TP) [9] and exception handling patterns (EHP) [7]. Our catalogue currently supports 10/14 BAPs, 8/10 TP and 36/135 EHPs from the referenced collections and some additional patterns. Some patterns in turn are not realizable within the BPMN2 metamodel. We consolidated, modularized and aligned the patterns in a hierarchical manner and reused simple patterns for the assembly of more complex patterns wherever reasonable and useful. The resulting structure of our pattern catalogue is presented in Figure 2, having the most generic and simple patterns on the left hand side and the most dedicated and complex patterns on the right hand side. The patterns are divided into the categories *basic*, *exception handling* and *time* and eventually mapped to generic patterns from existing prior work. Abstract patterns are marked grey and have no direct correspondence in terms of a BPMN2 construct. They are rather designed for a more convenient understandability of the catalogue. Specialization arcs are informally used to indicate that a specific pattern can be conceived as the extension of another pattern in terms of inserting additional elements in its BPMN fragment graph or by introducing additional constraints w.r.t. their parameters or applicability. This is similar, although not completely equal to *projection inheritance* as introduced in [13]. Reuse arcs indicate the (partial) composition of a pattern from other patterns. An extract from our catalogue is presented in Table 2. Each pattern consists of a unique name, a description, a graphical representation in BPMN2 notation, an illustrative example as well as eventual

Table 2. Extract from Pattern Catalogue for BPMN2 Workflow Adaptation

Basic Adaptation Pattern BAP1: Skip Segment	
<p>Description: The adaptive segment is wrapped into an exclusive choice which always evaluates to true on the bypassing path.</p> <p>Example: Spare parts replacement in a maintenance workflow is only conducted at all if it is a regular customer and the utilization of service employees does not exceed a certain percentage.</p> <p>Parameters: -</p> <p>Constraints: -</p> <p>Reuses: - Refines: BAP0 Rel. Generic Patterns: AP2 </p>	
Basic Adaptation Pattern BAP16: Optional Spawn Synched Activity	
<p>Description: As long as the segment is in execution, an additional activity can be spawned off. The completion of the activity is waited for in a parallel or later part of the workflow. It can be signaled that an additional activity is not needed, so that the dependent parallel or later part of the workflow can continue. This signal is automatically given latest after the segment has completed.</p> <p>Example: Within some harbors it can be allowed that while measurements are conducted, it can be determined that additional spare parts need to be ordered. If so, those have to be available before spare parts replacement can begin.</p> <p>Parameters: AdditionalActivity(BPMN2:Activity), SynchName(BPMN2:LinkEventDefinition)</p> <p>Constraints: Can only be used pairwise with BAP17 or BAP18 with same adaptation conditions. The two concerned segments must not reside in different loop structures. BAP17 or BAP18 must be applied on a different adaptive segment within a parallel branch or after a synchronization point, such that its adaptive segment can be determined as a successor in the graph without any doubt.</p> <p>Reuses: - Refines: BAP15 Rel. Generic Patterns: AP11 , AP1 </p>	
Basic Adaptation Pattern BAP18: Segment Externally Triggered	
<p>Description: A segment eventually has to wait for a particular event before it can be started, e.g. the completion of another segment within the workflow.</p> <p>Example: In some harbors, before spare parts replacement can start, for efficiency reasons it has to be waited for a decision that no additional spare parts are needed or that additionally ordered spare parts are ready.</p> <p>Parameters: SynchEvent(BPMN2:EventDefinition)</p> <p>Constraints: <i>Should</i> only be used pairwise with patterns like BAP13 or BAP16 with same adaptation conditions. The two concerned segments must not reside in different loop structures. BAP13 or BAP16 must be applied on a different adaptive segment within a parallel branch or before a synchronization point, such that its adaptive segment can be determined as a predecessor in the graph without any doubt.</p> <p>Reuses: - Refines: BAP4 Rel. Generic Patterns: AP11 , AP1 </p>	
Time Adaptation Pattern TAP8: Validity Period	
<p>Description: The execution of a segment can only be conducted within a specific time interval. The interval can for example be re-occurring or can be relative to the point in time when the segment is entered.</p> <p>Example: In some harbors running on heavy utilization, measurements including engine startups have to start at 09am (due to noise pollution). If the ship's availability is classified as "urgent" and it is not possible to complete the measurements till 5pm same day, the measurements are skipped.</p> <p>Parameters: EarliestTime(BPMN2:TimerEventDefinition), LatestTime(BPMN2:TimerEventDefinition)</p> <p>Constraints: If (relative or absolute) points in time are used for time specification, EarliestTime has to be smaller than LatestTime.</p> <p>Reuses: EX2, T4 Refines: BAP0 Rel. Generic Patterns: TP7 </p>	
Exception Handling Adaptation Pattern EHP6: Abort, Try Resolve, Restart or Escalate	
<p>Description: During the execution of a segment, when a particular event occurs, the execution of the segment is interrupted. An error handler is called subsequently. If the error handler can resolve the problem, the segment is restarted. If not, the workflow is escalated.</p> <p>Example: For some specific engine types, tests have to be immediately aborted if abnormal values are measured. If the error source can be spotted and removed, the tests have to be repeated. If not, the situation has to be handled manually.</p> <p>Parameters: Handler(BPMN:Activity), ReactOnEvent(BPMN2:EventDefinition)</p> <p>Constraints: The error handling activity needs to provide a compatible data object which can be used by the gateway for deciding whether the problem is resolved or not.</p> <p>Reuses: - Refines: EX1 Rel. Generic Patterns: SFF-RCC-COM , SFR-CWC-COM </p>	

Scope and Nesting of Patterns. Due to the design of patterns from our catalogue as valid BPMN2 fragments combined with the design of our framework only allowing block-structures as adaptive workflow segments, syntactic errors³ caused by combining adaptations are already excluded on a conceptual level. As such, runtime errors like missing reference points for adaptations as potentially occurring in similar approaches [10] are excluded and patterns like BAP1 from Table 2 and BAP3 can be applied simultaneously without any syntactic problems. For traceability reasons, it is not possible to apply “adaptations on adaptations”, i.e. adaptations are only allowed to be applied on an entire workflow segment. Despite these restrictions, however, structural errors may still occur and have to be excluded as far as possible, which is subject to the remainder of this section. The component structure of tailoring operations also facilitates the construction of eventually domain-specific patterns by reuse. An illustrative example is pattern TAP8 from Table 2, which can directly be realized by the subsequent application of pattern TAP4 and EHP2.

Application and Parameter Constraints. To prevent structural errors resulting from runtime adaptation, for instance related to soundness violation in terms of deadlocks and livelocks [14], a set of additional constraints has to be fulfilled for the application of some adaptation patterns. The constraint for TAP8 assures that the workflow segment has a chance to execute at all and is not always skipped if relative points in time are used for interval specification. BAP16 and BAP18 carry a rather complex set of constraints, since they can only be used in combination to realize a dynamic control dependency insertion. Such a dependency insertion at runtime in turn can only be realized under specific structural conditions of the workflow graph [14]. Further constraints realize the specialization dependencies in Figure 2. As such, a *move segment* adaptation structurally corresponds to distinct *skip* and *insert* operations, however with the additional constraint that a segment which should be inserted within an adaptive segment must correspond to the block structure of another adaptive segment. We are currently working on a set of OCL constraints for our metamodel presented in [15] to enforce constraints of such type related to graph structure. EHP6 provides an example for a constraint concerning the data perspective of adaptations. Involving data issues for adaptations imposes some additional requirements especially on a design-time repository for reusable task- and event definitions and their associated input and output respectively payload. The constraint of EHP6 could for example be realized by an XPath expression on the XML Schema Definition of the data output associated to the handler activity parameter.

Ordering of Pattern Application. The rule-based adaptation approach presented in Section 2 allows for the specification of multiple patterns and their application order for one specific data context. Due to its partly declarative nature however, multiple adaptations resulting from different adaptation rules or human intervention may need to be applied simultaneously. Different problems may occur if

³ Syntactic errors relate to incorrect BPMN2 constructs, while structural errors relate to the specification of invalid behavior.

these adaptations are applied in an arbitrary order. One problem consists in the potential generation of unexpected or conflicting business logic. For example, if one rule defines the application of a skip pattern (BAP1) and one rule defines the application of an insert pattern (BAP2), the workflow runtime behavior depends on the application order of the patterns. Another problem consists in the generation of structural errors within the workflow, for example when incorrectly mixing loop (BAP14) and dependency (BAP16 & BAP18) adaptations as already discussed in Section 3. As an optional add-on to our approach, in order to prevent such structural problems resulting from incorrect adaptation order as well as to increase the understandability for the modeler, we propose the definition of a globally valid order for a set of adaptations depending on their type. Such an order can be defined by carefully considering the potential interrelations of pattern types which are simultaneously applied w.r.t. interfering execution semantics. It may make sense that a skip adaptation is always executed last, eventually “overriding” other adaptations. A loop adaptation is always executed before adaptations related to control dependencies, excluding the corresponding problems related to soundness violation. For loops, all loopback gateways should be inserted at the same place within the workflow, de-facto constructing a single loop which preserves all adaptations except synchronization and spawning configured when entering the adaptive segment. If rule conditions for adaptations should be reevaluated at loopback, this should be realized with a regular loop in the reference model. Of course, it is also possible to define different eventually use-case dependent global application orders. The examination of this issue as well as a more comprehensive definition of inter-pattern dependencies is subject to future work. Although other reasonable orderings may exist, based on the given considerations, one valid application ordering can be defined as follows:

```
Insert < EventHandler < TimePattern < WaitFor < Loop < Spawn < Synch < Skip
```

4 Prototypical Implementation

We have already developed a prototype based on jBoss Drools 5.1⁴ which supports the graphical modeling of adaptive segments and the specification of adaptation rules. Basic implementation details have been described in [4], therefore only an architectural overview is provided in Figure 3. For modeling and executing context- and event-aware workflows based on adaptation rules, we have integrated and extended different components of Drools, primarily its Eclipse-based modeling environment and its execution engine. Extensions relate to the ability to interpret the special intermediate event nodes for entering or leaving an adaptive workflow segment. Adaptation rules are currently specified in native Drools syntax, but are planned to be automatically generated from a generic metamodel for our approach as presented in [15]. Our prototype can interpret adaptation patterns defined in BPMN2 with some Drools-related technical constraints. It handles most of the basic and a large fraction of the time and

⁴ www.jboss.org/drools is an open platform for rules, workflow and event processing.

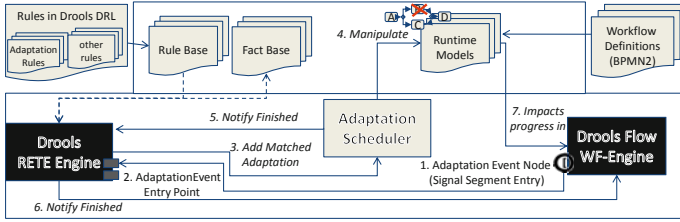


Fig. 3. Architecture for Runtime Variant Creation with jBoss Drools

exception handling adaptation patterns presented in this work on the execution level. Technically, a segment entry event is forwarded to the rule engine, which evaluates the data context. The adaptation actions of matching rules are first gathered in a scheduler, allowing for later advanced error checking or reordering of multiple adaptations. The adaptation handler then creates a local copy of the adaptive segment, applies the adaptations and notifies the rule and workflow engines after adaptations have been executed and the workflow may continue on the new variant segment instance.

5 Related Work

As a conceptual basis for process technology, e.g. to evaluate the expressiveness of workflow modeling languages, *workflow patterns* [16] provide a catalogue of control-flow dependencies like parallelism or choice between tasks in a workflow. Since then, additional patterns covering different dimensions of workflow modeling and execution have emerged. Exception handling patterns [7] deal with different general handling strategies for immediate reaction on unforeseen errors. Change patterns [8] specify concretely how adaptations of the process logic on workflow model and workflow instance level can be supported. Time patterns [9] capture the support of temporal features like complex task scheduling of a WfMS. The unification of these three dimensions into one BPMN2 pattern catalogue is a major contribution of this work.

The modularized definition and dynamic reuse of process fragments within a workflow is suggested in [11]. The authors motivate the need for modularization by cross-cutting concerns in BPMN and discuss the extension of workflows with aspects in an AO4BPMN syntax. In our work, cross-cutting concerns can easily be realized by re-using adaptation segment nodes of the same type multiple times in a workflow. However, we generalize the usage of BPMN fragments to establish effective variant management, allowing a 'reduction' of the reference model as well as context-dependent time-behavior and exception-handling. Especially we provide an extended set of corresponding adaptation patterns.

In the work of [17] and [18], the BPMN metamodel is extended to increase workflow flexibility. In [17], BPMN and a rule meta-model are weaved to form elements of higher expressiveness, like a rule-based gateway. Those can however not be directly employed for variant management. The authors of [18] focus on

versioning extensions for the BPMN model elements *task* and *process*. The work in [19] contains a sophisticated constraint network which allows the validation of variants resulting from ad-hoc task execution at runtime. Both [18] and [19] leave events aside, which is essentially required in our framework to realize context-dependent timing behavior and reactions on events.

With respect to workflow flexibility, a variety of systems have evolved. ADEPT [3] and YAWL [12] belong to the most prominent ones. However, these systems as a pure technological foundation usually rely on a proprietary notation and do not enclose guidance mechanisms for restricting flexibility to just the needed degree for a particular use case. For the latter reason, dedicated approaches for workflow variant management have been developed. Approaches like [20] propose the modeling of one large reference workflow in configurable YAWL (cYAWL), where parts of it may be faded out at runtime based on data constraints. As motivated in Section 1 for some scenarios, the initial modeling of all such variants may be infeasible and a factorization using a rule-based approach may be required. The authors of [10] focus on the configuration and management of workflows with the help of adaptation rules applying change operations on a reference workflow. Although we build upon their general underlying idea, we extend missing features primarily related to time and eventing, like context-specific error handling.

6 Conclusions

In this work, we have motivated the need for more conceptual guidance for design- and runtime flexibility for workflows based on the industry standard BPMN2 and business rules. Our solution approach relies on an extension of the BPMN2 metamodel for marking context-dependent adaptive workflow segments.

The main contribution of this work is the consolidation of change-, time- and exception handling patterns into one BPMN2 catalogue of adaptation patterns which can directly be used within the approach. The approach is customizable to restrict flexibility to the required degree, e.g. in terms of the set of adaptation pattern types which are allowed rule-based workflow adaptation. Some of the further multiple benefits are a more intuitive understanding of the adaptations by the modeler who is already familiar with BPMN, a better starting position for consistency checks of the overall variant model and especially a better modifiability and extensibility of the patterns themselves.

As next steps, we aim at providing a modeling environment for consistent rule-based application of patterns according to the underlying constraint set. Furthermore, regarding the applicability of our approach, modeling variants by rule-based and pattern-supported workflow adaptations is to be evaluated mainly within a marine service scenario of a globally operating engine manufacturer.

Acknowledgements

This work was developed in the project *Allianz Digitaler Warenfluss* (ADiWa) that is funded by the German Federal Ministry of Education and Research. Support code: 01IA08006.

References

1. Leymann, F., Roller, D.: *Production Workflow*. Prentice-Hall, Englewood Cliffs (1999)
2. Wolf, C., Harmon, P.: *The State of Business Process Management 2010*, BPTrends (2010), www.bptrends.com
3. Dadam, P., Reichert, M.: The ADEPT Project: A Decade of Research and Development for Robust and Flexible Process Support. *CSRD* 23(2), 81–97 (2009)
4. Döhring, M., Zimmermann, B., Godehardt, E.: Extended Workflow Flexibility using Rule-Based Adaptation Patterns with Eventing Semantics. In: *Informatik 2010 Service Science, Leipzig*. GI Lecture Notes in Informatics, pp. 195–200 (2010)
5. Döhring, M., Karg, L., Godehardt, E., Zimmermann, B.: The Convergence of Workflows, Business Rules and Complex Events. In: *ICEIS 2010, Funchal*, pp. 338–343 (2010)
6. OMG: *Business Process Model and Notation (BPMN) - Version 2.0 11-01-03* (2011)
7. Russell, N., van der Aalst, W.M.P., ter Hofstede, A.H.M.: Exception handling patterns in process-aware information systems. In: Martinez, F.H., Pohl, K. (eds.) *CAiSE 2006*. LNCS, vol. 4001, pp. 288–302. Springer, Heidelberg (2006)
8. Weber, B., Reichert, M., Rinderle-Ma, S.: Change Patterns and Change Support Features. *DKE* 66(3), 438–466 (2008)
9. Lanz, A., Weber, B., Reichert, M.: Workflow time patterns for process-aware information systems. In: Bider, I., Halpin, T., Krogstie, J., Nurcan, S., Proper, E., Schmidt, R., Ukor, R. (eds.) *BPMSD 2010 and EMMSAD 2010*. Lecture Notes in Business Information Processing, vol. 50, pp. 94–107. Springer, Heidelberg (2010)
10. Hallerbach, A., Bauer, T., Reichert, M.: Configuration and management of process variants. In: *International Handbook on BPM*, pp. 237–255. Springer, Heidelberg (2010)
11. Charfi, A., Müller, H., Mezini, M.: Aspect-oriented business process modeling with AO4BPMN. In: Kühne, T., Selic, B., Gervais, M.-P., Terrier, F. (eds.) *ECMFA 2010*. LNCS, vol. 6138, pp. 48–61. Springer, Heidelberg (2010)
12. Adams, M., ter Hofstede, A.H.M., Edmond, D., van der Aalst, W.M.P.: Worklets - A service-oriented implementation of dynamic flexibility in workflows. In: *CoopIS 2006, Montpellier*, pp. 291–308 (2006)
13. van der Aalst, W.M.P., Basten, A.A.: Inheritance of workflows. *TCS* 270(1-2), 125–203 (2002)
14. Rinderle, S., Reichert, M., Dadam, P.: Correctness Criteria for Dynamic Changes in Workflow Systems. *DKE* 50(1), 9–34 (2004)
15. Döhring, M., Zimmermann, B.: vBPMN - Event-Aware Workflow Variants by Weaving BPMN2 and Business Rules, In: *EMMSAD 2011, London* (accepted, 2011)
16. van der Aalst, W.M.P., ter Hofstede, A.H.M., Kiepuszewski, B., Barros, A.P.: *Workflow Patterns*. DAPD 14(1), 5–51 (2003)
17. Milanovic, M., Gasevic, D.: Towards a Language for Rule-Enhanced Business Process Modeling. In: *EDOC 2009, Auckland*, pp. 64–73. IEEE, Los Alamitos (2009)
18. Ben Said, I., Chaabane, M.A., Andonoff, E.: A model driven engineering approach for modelling versions of business processes using BPMN. In: Abramowicz, W., Tolksdorf, R. (eds.) *BIS 2010*. Lecture Notes in Business Information Processing, vol. 47, pp. 254–267. Springer, Heidelberg (2010)
19. Lu, R., Sadiq, S., Governatori, G.: On managing business processes variants. *DKE* 68(7), 642–664 (2009)
20. La Rosa, M., van der Aalst, W.M.P., Dumas, M., ter Hofstede, A.H.M.: Questionnaire-based variability modeling for system configuration. *SoSyM* 8(2), 251–274 (2008)

Behavioral Conformance of Artifact-Centric Process Models

Dirk Fahland, Massimiliano de Leoni, Boudewijn F. van Dongen, and
Wil M.P. van der Aalst

Eindhoven University of Technology, The Netherlands

{d.fahland,m.d.leoni,b.f.v.dongen,w.m.p.v.d.aalst}@tue.nl

Abstract. The use of process models in business information systems for analysis, execution, and improvement of processes assumes that the models describe reality. *Conformance checking* is a technique to validate how good a given process model describes recorded executions of the actual process. Recently, *artifacts* have been proposed as a paradigm to capture dynamic, and inter-organizational processes in a more natural way. In *artifact-centric processes*, several restrictions and assumptions of classical processes are dropped. This renders checking their conformance a more general problem. In this paper, we study the conformance problem of such processes. We show how to partition the problem into *behavioral conformance* of single artifacts and interaction conformance between artifacts, and solve behavioral conformance by a reduction to existing techniques.

Keywords: artifacts, process models, conformance.

1 Introduction

Process models have become an integral part of modern information systems where they are used to document, execute, monitor, and optimize processes. However, many studies show that models often deviate from reality (see. [1]). Hence, before a process model can reliably be used, it is important to know in advance to what extent the model conforms to reality.

Classical process modeling techniques assume monolithic processes where process instances can be considered in isolation. However, when looking at the data models of ERP products such as SAP Business Suite, Microsoft Dynamics AX, Oracle E-Business Suite, Exact Globe, Infor ERP, and Oracle JD Edwards EnterpriseOne, one can easily see that this assumption is *not* valid for real-life processes. There are one-to-many and many-to-many relationships between data *objects*, such as customers, orderlines, orders, deliveries, payments, etc. For example, an online shop may split its customers' *quotes* into several *orders*, one per supplier of the quoted items, s.t. each order contains items *for several customers*. Consequently, several customer cases synchronize on the same order at a supplier, and several supplier cases synchronize on the same quote of a customer. In consequence, we will not be able to identify a unique notion of a *process instance* by which we can trace and isolate executions of such a process, and classical modeling languages are no longer applicable [2-4].

The fabric of real-life processes cannot be straightjacketed into monolithic processes. Therefore, we need to address two problems:

- (1) Find a modeling language \mathcal{L} to express process executions where several cases of different objects overlap and synchronize;
- (2) The *conformance checking problem*: determine whether a process model M expressed in \mathcal{L} adequately describes actual executions of a dynamic and inter-organizational processes in reality — despite the absence of process instances.

The first problem is well-known [2–4] and several modeling languages have been proposed to solve it culminating in the stream of *artifact-centric process modeling* [2–6]. An *artifact instance* is an object that participates in the process. An *artifact* describes a class of similar objects, e.g., all *orders*, together with the *life cycle* of states and possible transitions that each of these objects follows in a process execution. An artifact-centric process model then describes how several artifact instances interact with each other in their respective life cycles. In this paper, we use *proclets* [2] to describe artifact-centric process models and to study and solve the second problem of conformance checking.

Conformance checking compares the behavior described by a process model M to process executions in an actual information system S . Classically S records all events of one execution in an isolated *case*; all cases together form a *log*. Existing conformance checking techniques then check to which degree a given process model can replay each case in the log [7–12]. Artifact-centric systems drop the assumption of an isolated case and a log. Here, S records events in a *database* \mathcal{D} [4]. Each event stored in \mathcal{D} is associated to a unique artifact instance. A complete case follows from an interplay of several artifact instances and several cases overlap on the same artifact instance. Existing conformance checkers cannot be applied in this setting.

In this paper, we investigate the conformance checking problem of artifacts. The problem decomposes into subproblems of significantly smaller size which we reduce to classical conformance checking problems. We contribute a technique to extract logs L_1, \dots, L_n of logs from a given database, one log for each artifact in the model. Each case of L_i contains all events associated to a specific instance of artifact i . Feeding L_1, \dots, L_n into existing conformance checkers [12] allows to check conformance of an artifact-centric process model w.r.t. artifact life-cycles as well as artifact interactions.

The paper is structured as follows. Section 2 presents the artifact-centric approach and *proclets* [2] as a light-weight formal model for artifacts. In Sect. 3, we state the *artifact conformance problem*. Section 4 introduces our techniques for reducing behavioral conformance and interaction conformance to classical process conformance; these techniques and conformance checkers are implemented in the Process Mining Toolkit ProM (available at www.processmining.org). The paper concludes with a discussion on related and future work.

2 The Artifact-Centric Approach

Artifacts emerged in the last years as an alternative approach for precisely describing dynamic, inter-organizational processes in a modular way [3–6]. In the following, we recall the key concepts of artifacts and present a simple formal model for artifact-centric processes that we will use in this paper.

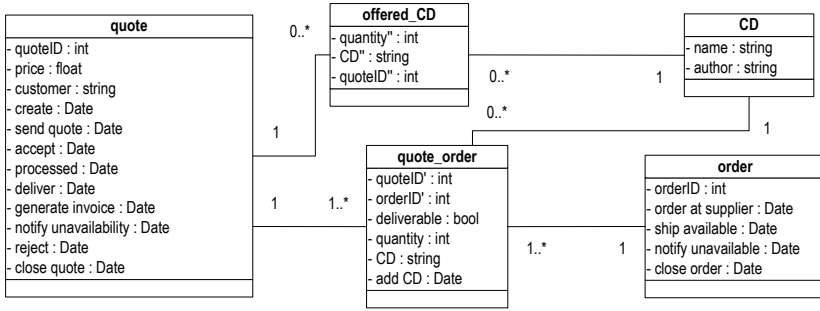


Fig. 1. Data model of a CD online shop's back-end process

Data objects and artifacts. Artifacts compose complex processes from small building blocks [3, 4]. The particular feature of artifacts is their foundation in the process' underlying *data model*. The approach assumes that any process materializes itself in the (data) objects that are involved in the process, for instance, a paper form, a CD, a customer's quote, or an electronic order; these objects have properties such as the values of the fields of a paper form, the processing state of an order, or the location of a package.

A data model describes the (1) *classes* of objects that are relevant in the process, (2) the relevant properties of these objects in terms of class *attributes*, and (3) the *relations* between the classes. A process execution *instantiates* new objects and changes their properties according to the process logic. Thereby, the relations between classes describe how many objects of one class are related to how many objects of another class.

An *artifact-centric process model* enriches the classes themselves with process logic restricting how objects may evolve during execution. More precisely, one *artifact* (1) encapsulates several classes of the data model, (2) provides *actions* that can update the classes' attributes, (3) defines a *life cycle*, and (4) exposes some of its actions via an *interface*. The artifact's life cycle describes when an *instance* of the artifact (i.e., a concrete object) is created, in which state of the instance which actions may occur to advance the instance to another state, and which *goal state* the instance has to reach to complete a case.

An example. As a running example for this paper, we consider the backend process of a CD online shop. The shop offers a large collection of CDs from different suppliers to its customers. The backend process is triggered by a customer's request for CDs. The shop then sends a *quote* of the offered CDs. If the customer accepts, the quote is split into several *orders*, one per CD supplier. Each order in turn handles all quotes for CDs from the same supplier. The order then is executed and the suppliers ship the CDs to the shop which distributes the different CDs from the different orders according to the original quotes. Some CDs may be unavailable at the supplier; in this case notifications are sent to the CD shop which forwards it to the customer. From an artifact perspective, this backend process is driven by the *quotes* and *orders*, their respective processing states, and their relations. The UML class diagram of Fig. 1 denotes the data model of our CD shop example.

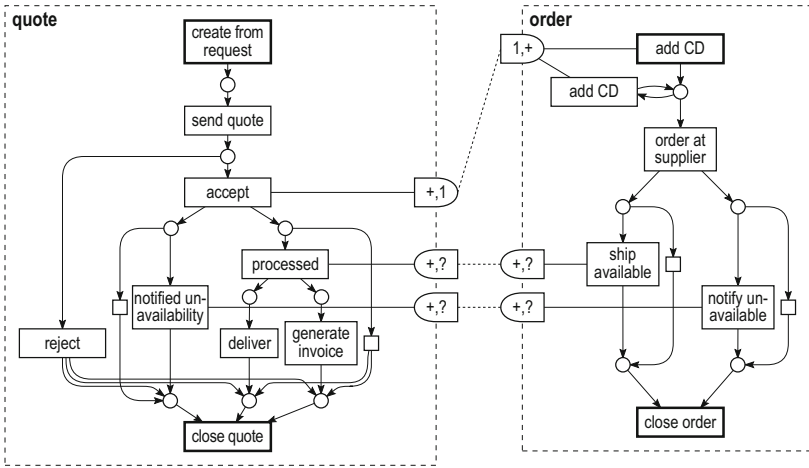


Fig. 2. A procelet system describing the back-end process of a CD online shop. A customer’s quote is split into several orders according to the suppliers of the CDs; an order at a supplier handles several quotes from different customers.

Describing processes by procelet systems. Procleets propose concepts for describing artifacts and their interactions [2]. A procelet $P = (N, ports)$ consists of a labeled Petri net, which describes the internal life cycle of one artifact, and a set of ports, through which P can communicate with other procleets [13]. Relations between several procleets are described in a procelet system $\mathcal{P} = (\{P_1, \dots, P_n\}, C)$ consisting of a set of procleets $\{P_1, \dots, P_n\}$ and a set C of channels. Each channel $(p, q) \in C$ connects two ports p and q of two procleets of \mathcal{P} . On one hand, procleets send and receive messages along these channels. On the other hand, the channels also reflects the relations between classes: annotations at the ports define how many instances of a procelet interact with how many instances of another procelet.

Figure 2 shows a procelet system of two procleets that model artifacts quote and order. Each half-round shape represents a port: the bow indicates the direction of communication. A dashed line between 2 ports denotes a channel of the system. Creation and termination of an artifact instance is expressed by a respective transition, drawn in bold lines in Fig. 2. Note that other modeling languages are likewise applicable to describe an artifact’s life cycle [3–6]. Procleets can be mapped to the data model of the process: for each procelet transition (e.g., add quote) exists a corresponding timestamp attribute that is set when the transition occurs (e.g., add quote of quote_order).

The decisive expressivity of procleets for describing artifacts comes from the annotations $1, ?, +$ that are inscribed in the ports [2]. The first annotation, called *cardinality*, specifies how many messages one procelet instance sends to (receives from) other instances when the attached transition occurs. The second annotation, called *multiplicity*, specifies how often this port is used in the lifetime of a procelet instance. For example, the port of accept has cardinality $+$ and multiplicity 1 denoting that a quote once sends out one or more messages on quoted CDs to multiple orders. Conversely, the process

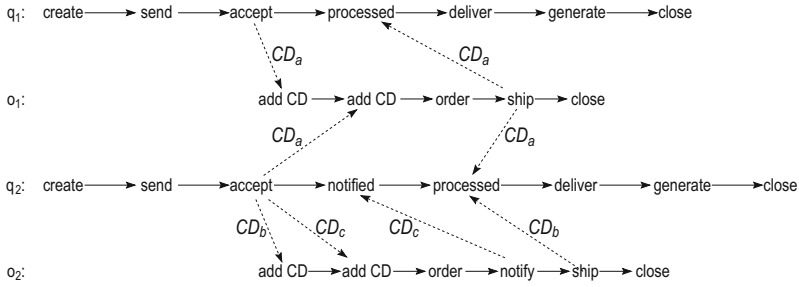


Fig. 3. An execution of the proclat system of Fig. 2 with two quote instances and two order instances

quote									
quotelD	create	send quote	accept	processed	notify	deliver	generate	reject	close quote
q_1	24-11,17:12	24-11,17:13	25-11,7:20	5-12,9:34	null	6-12,5:23	null	null	6-12,5:25
q_2	24-11,19:56	24-11,19:57	25-11,8:53	5-12,11:50	3-12,14:54	6-12,7:14	3-12,14:55	null	6-12,7:20

order				
orderID	ship	order	notify	close order
o_1	5-12,9:32	28-11,8:12	null	5-12,11:37
o_2	5-12,11:33	28-11,12:22	3-12,14:34	5-12,13:03

offered_CDs		
quotelD ^o	CD ^o	quantity ^o
q_1	a	2
q_2	a	1
q_2	b	3
q_2	c	1

CD				
name	author			
a	xyz			
b	zyx			
c	yxz			

quote_order				
quotelD ^o	orderID ^o	add CD	deliverable	CD
q_1	o_1	25-11,8:31	true	a
q_2	o_1	25-11,12:11	true	a
q_2	o_2	26-11,9:30	true	b
q_2	o_2	26-11,9:31	false	c

Fig. 4. Events of the run of Fig. 3 recorded in a database according to Fig. 1

repeatedly (+) adds one CD of a quote to an order. These constraints reflect the relation $1..*-1..*$ between quotes and orders denoted in Fig. 1.

The *semantics* of proclats generalizes the semantics of Petri nets by the ports. Basically, different proclat instances are distinguished by using instance identifiers as tokens. A transition at an output port *produces* as many messages (to other proclat instances) into the channel as specified by the port's cardinality. A transition at an input port waits for as many messages (from other proclat instances) as specified and consumes them. For example, Fig. 3 illustrates an execution of the proclat system of Fig. 2: one over CD_a and the other over CD_a , CD_b , and CD_c . CD_b and CD_c have the same supplier, CD_a has a different supplier. Hence, the quotes are split into two orders. In the run, CD_a and CD_b are available whereas CD_c is not, which leads to the behavior shown in Fig. 3 involving two quote instances and two order instances.

Operational semantics of proclats specify senders of messages to consume and recipients of produced messages [2]. For conformance checking, focusing on the number of produced and consumed messages is sufficient; see [13] for details. For example the run of Fig. 3 satisfies all cardinality and multiplicity constraints of the ports of Fig. 2 i.e., it *conforms* to the proclat system. A system that executes this process records timestamps of events in a database according to the data model of Fig. 1. The corresponding database tables could be populated as shown in Fig. 4. The question that we consider in the following is whether the model of Fig. 2 accurately describes the records of Fig. 4.

3 The Artifact Conformance Checking Problem

The problem of determining how accurately a process model describes the process implemented in an actual information system S is called *conformance checking problem* [7].

Classically, a system S executes a process in an isolated instance. The corresponding *observed* system execution is a sequence of events, called *case*, and a set of cases is a *log* L . The semantics of a formal process model M define the set of valid process executions in terms of sequences of M 's actions. Conformance of M to L can be characterized in several dimensions [7]. In the following, we consider only *fitness*. This is the most dominant conformance metric that describes to which degree a model M can replay all cases of a given log L , e.g., [12]. M fits L less, for instance, if M executes some actions in a different order than observed in L , or if L contains actions not described in M . Several conformance checking techniques for process models are available [7–12]. The more robust techniques, e.g., [12], find for each case $\rho \in L$ an execution ρ' of M that is as *similar* as possible to ρ ; the similarity of all ρ to their respective ρ' defines the fitness of M to L .

3.1 The Artifact Conformance Problem

We have seen in Sections 1 and 2 that many processes do not structure their executions into isolated instances. In the light of this observation, we identify the following *artifact conformance problem*. The system S records occurrences of a set Σ of actions in a *database* \mathcal{D} according to the system's data model. Each event is associated to a specific object, that is stored in \mathcal{D} . Let \mathcal{P} be a proclat system where each proclat transition maps to a timestamped attribute of \mathcal{D} (i.e., each proclat of \mathcal{P} describes an artifact of S). Can the proclats of \mathcal{P} be instantiated s.t. the life-cycles of all artifact instances and their interaction “replay” all events recorded in \mathcal{D} ? If not, to which degree does \mathcal{P} deviate from the behavior recorded in \mathcal{D} ?

3.2 Reducing Artifact Conformance to Existing Techniques

A naïve solution of the artifact conformance problem would replay all events of the database \mathcal{D} in the proclat system \mathcal{P} . Technically, this would mean to find the database \mathcal{D}' that can be replayed by \mathcal{P} and is as similar as possible to \mathcal{D} . In typical case studies we found the actual system S to record about 80,000 events of 40–60 distinct actions. Finding a conforming database \mathcal{D}' by replacing non-conforming events with conforming events defines a search space of 80,000⁶⁰ possible solutions. Even exploring only a small fraction of such a search space quickly turns out infeasible.

For this reason, we propose a compositional approach to check whether an proclat system \mathcal{P} fits \mathcal{D} . As we cannot employ the notion of a process instance to structure \mathcal{D} into smaller parts we partition the problem into checking conformance *within* proclats and *between* proclats.

Behavioral conformance. Each event in \mathcal{D} is associated to an object, and hence to an instance i of an artifact Ar described by a proclat P_{Ar} in the proclat system \mathcal{P} . All events associated to i together constitute the *artifact case* of i of Ar that describes

how i evolved along the life-cycle of Ar . It ignores how i interacts with other artifact instances. The *behavioral conformance* problem is to check whether the life cycle of P_{Ar} can replay each artifact case of Ar (i.e., each recorded artifact life cycle).

Interaction conformance. Completing a life cycle of an instance i of Ar also depends on other artifact instances, as discussed previously. Let J be the set of artifact instances with which i exchanges messages. All events of \mathcal{D} that send or receive messages and are associated to an instance in $\{i\} \cup J$ together constitute the *interaction case* of Ar . It contains all behavioral information regarding how i interacts with other instances. Procelet P_{Ar} fits the interaction case of instance i of Ar if the interaction case involves events of as many artifact instances as required by the ports of P_{Ar} . The *interaction conformance* problem is to check how good all procleets of \mathcal{P} fits all interactions cases that are stored in \mathcal{D} ; it describes how good the procelet interactions reflect the object relations in \mathcal{D} .

The behavioral conformance and the interaction conformance together yield the artifact conformance of the entire procelet system \mathcal{P} w.r.t. \mathcal{D} ; see [13] for a formal proof. Yet, either conformance can be checked per artifact case or per interaction action case, respectively, which significantly reduces the search space during checking.

4 Checking Behavioral Conformance of Artifacts

In the following, we first solve the *behavioral conformance problem* by reduction to classical process conformance. Assuming that events of artifacts Ar_1, \dots, Ar_n are recorded in a given database \mathcal{D} , we extract for each artifact Ar_i all artifact cases from \mathcal{D} into a log L_i . The logs L_1, \dots, L_n describe the internal life cycle behavior of the artifacts. These logs can then be used to check behavioral conformance of a procelet system w.r.t. \mathcal{D} in existing conformance checkers, as we show in Sect. 4.3. Moreover, the logs L_1, \dots, L_n can be leveraged to also express interaction between artifacts, which then allows to check interaction conformance with existing conformance checkers [13].

4.1 Extracting Logs from Databases

In the following, we provide a technique to extract logs from a relational database \mathcal{D} . We assume that \mathcal{D} recorded events of n different artifacts, and that each event is associated to a specific instance of an artifact. Our vehicle to extract logs from \mathcal{D} will be an *artifact view* on \mathcal{D} which specifies for each artifact of the system, the *types* of events occurring in this artifact. Each event type is characterized in terms of database attributes (of different tables) of \mathcal{D} which need to be related to each other according to the schema of \mathcal{D} . Using this characterization, we then extract events from \mathcal{D} by joining tables, and selecting and projecting entries according to the specified attributes. We first introduce some notion on databases and then present the details of this approach.

Preliminaries. We adopt notation from *Relational Algebra* [14]. A table $T \subseteq D_1 \times \dots \times D_m$ is a relation over domains D_i and has a *schema* $\mathcal{S}(T) = (A_1, \dots, A_m)$ defining for each *column* $1 \leq i \leq m$ an *attribute name* A_i . For each *entry* $t = (d_1, \dots, d_m) \in T$ and each column $1 \leq i \leq m$, let $t.A_i := d_i$. We write $\mathcal{A}(T) := \{A_1, \dots, A_m\}$ for the attributes of T , and for a set \mathcal{T} of tables, $\mathcal{A}(\mathcal{T}) := \bigcup_{T \in \mathcal{T}} \mathcal{A}(T)$.

A database $\mathcal{D} = (\mathcal{T}, K)$ is set \mathcal{T} of tables with corresponding schemata $\mathcal{S}(T), T \in \mathcal{T}$ s.t. their attributes are pairwise disjoint, and a key relation $K \subseteq (\mathcal{A}(\mathcal{T}) \times \mathcal{A}(\mathcal{T}))^{\mathbb{N}}$.

K expresses foreign-primary key relationships between the tables \mathcal{T} : we say that $((A_1, A'_1), \dots, (A_k, A'_k)) \in K$ relates $T \in \mathcal{T}$ to $T' \in \mathcal{T}$ iff the attributes $A_1, \dots, A_k \in \mathcal{A}(T)$ together are a *foreign key* of T pointing to the *primary key* $A'_1, \dots, A'_k \in \mathcal{A}(T')$ of T' . For instance, $(\text{quoteID}, \text{quoteID}') is a foreign-primary key relation from table quote to table quote_order of Fig. 4$

Relational algebra defines several operators [14] on tables. In the following, we use *projection*, *selection*, and the canonical crossproduct. For a table T and attributes $\{A_1, \dots, A_k\} \subseteq \mathcal{A}(T)$, the projection $Proj_{A_1, \dots, A_k} T$ restricts each entry $t \in T$ to the columns of the given attributes A_1, \dots, A_k . Please note that projection removes any duplicates: if there are two entries in $t_1, t_2 \in T$ that coincide on the values of the projected attributes A_1, \dots, A_k (i.e., $t_1.A_1 = t_2.A_1 \wedge \dots \wedge t_1.A_k = t_2.A_k$), after projecting, the entry obtained by projection t_2 is removed. Selection is a unary operation $Sel_{\varphi}(T)$ where φ is a boolean formula over atomic propositions $A = c$ and $A = A'$ where $A, A' \in \mathcal{A}(T)$ and c a constant; the result contains entry $t \in Sel_{\varphi}(T)$ iff $t \in T$ and t satisfies φ (as usual). We assume that each operation correspondingly produces the schema $\mathcal{S}(T')$ of the resulting table T' .

For a set $\mathcal{T}' = \{T_1, \dots, T_k\} \subseteq \mathcal{T}$ of tables, let $J_{K, \mathcal{T}'} := \{(A, A') \in K \mid k \in K, A, A' \in \mathcal{A}(T')\}$ denote the pairs of attributes that are involved in key relations between the tables in \mathcal{T}' . The $Join(\mathcal{T}', K) := Sel_{\varphi}(T_1 \times \dots \times T_k)$ with $\varphi := \bigwedge_{(A_i, A'_i) \in J_{K, \mathcal{T}'}} (A_i = A'_i)$ keeps from the cross-product of all tables \mathcal{T}' only those entries which coincide on all key relations.

With these notions at hand, we first introduce an *artifact view* on a database \mathcal{D} . It specifies for each artifact the types of events that are recorded in \mathcal{D} . Each event type is characterized by attributes of the database, defining in which instance an event occurred and when it occurred. We later use an artifact view to extract all events of an artifact and group them into cases, which yields a log.

Definition 1 (Artifact View). Let $\mathcal{D} = (\mathcal{T}, K)$ be a database. An artifact view $V = (\{\Sigma_1, \dots, \Sigma_n\}, Tab, Inst, TS)$ on \mathcal{D} is specified as follows:

- It defines n pairwise disjoint sets $\Sigma_1, \dots, \Sigma_n$ of event types (one set per artifact). Let $\Sigma := \bigcup_{i=1}^n \Sigma_i$.
- Function $Tab : \{\Sigma_1, \dots, \Sigma_n\} \rightarrow 2^{\mathcal{T}}$ specifies the set $Tab(\Sigma_i)$ of tables linked to each artifact $i = 1, \dots, n$.
- Function $Inst : \{\Sigma_1, \dots, \Sigma_n\} \rightarrow \mathcal{A}(\mathcal{T})$ specifies for each artifact $i = 1, \dots, n$ the attribute $Inst(\Sigma_i) = A_{id}^i \in \mathcal{A}(Tab(\Sigma_i))$ that uniquely identifies an instance of this artifact.
- Function $TS : \Sigma \rightarrow \mathcal{A}(\mathcal{T})$ specifies for each event type $a \in \Sigma$ the timestamp attribute $TS(a) = A_{TS} \in \mathcal{A}(Tab(\Sigma_i))$ that records when an event of type a occurred. Attributes $Inst(\Sigma_i)$ and $TS(a)$ must be connected through tables $T' \subseteq Tab(\Sigma_i)$.

Table 1. Artifact view for order

$Tab(\Sigma_{order}) = \{\text{quote_order}, \text{order}\}$,
 $Inst(\Sigma_{order}) = \text{orderID}$

event type $a \in \Sigma_{order}$	$TS(a)$
add CD	add CD
order at supplier	order
ship available	ship
notify unavailable	notify
close order	close order

Table 1 presents the artifact view for the artifact order of our running example on the database of Fig 4. The choice of the event types Σ_{order} , tables $Tab(\Sigma_{order})$, the instance identifier orderID and the corresponding time stamp attributes is straight forward.

After specifying an artifact view, an artifact log can be extracted fully automatically from a given database \mathcal{D} .

Definition 2 (Log Extraction). Let $\mathcal{D} = (T, K)$ be a database, let $V = (\{\Sigma_1, \dots, \Sigma_n\}, Tab, Inst, TS)$ be an artifact view on \mathcal{D} . The logs L_1, \dots, L_n are extracted from \mathcal{D} as follows. For each set $\Sigma_i, i = 1, \dots, n$ of event types:

1. Each event type $a \in \Sigma_i$ defines the event table
 $T_a = Proj_{Inst(\Sigma_i), TS(a)} Join(Tab(\Sigma_i), K)$.
2. Each entry $t = (id, ts) \in T_a$ identifies an event $e = (a, id, ts)$ of type a in instance id . Let \mathcal{E}_i be the set of all events of all event types $a \in \Sigma_i$.
3. For each instance $id \in \{id \mid (a, ts, id) \in T_a, a \in \Sigma_i\}$ of artifact $i \in \{1, \dots, n\}$, the set $\mathcal{E}_i|_{id} = \{(a, ts, id') \in \mathcal{E} \mid id = id'\}$ contains all events of instance id .
4. The artifact case $\rho_{id} = \langle a_1, a_2, \dots, a_n \rangle$ of instance id of artifact i orders events $\mathcal{E}_i|_{id}$ by their timestamp: $\mathcal{E}_i|_{id} = \{(a_1, id, ts_1), (a_2, id, ts_2), \dots, (a_n, id, ts_n)\}$ s.t. $ts_i < ts_{i+1}$, for all $1 \leq i < n$. The log L_i contains all artifact cases of artifact i .

Table 2. Intermediate table obtained by joining $Join(\{\text{quote_order}, \text{order}\}, K)$

quoteID'	...	orderID	add CD	ship	...
q_1	...	o_1	25-11,8:31	5-12,9:32	...
q_1	...	o_1	25-11,12:11	5-12,9:32	...
q_2	...	o_2	26-11,9:30	5-12,11:33	...
q_2	...	o_2	26-11,9:31	5-12,11:33	...

We illustrate the log extraction by our running example from Sect. 2. For the database of Fig. 4, we consider the artifact view on order as specified in Tab. 1. To extract events of order first join the tables order and quote_order on (orderID, orderID'), Tab. 2 shows parts of that table. To obtain events of type add CD, project this tables

onto $Inst(\Sigma_{order}) = \text{orderID}$ and timestamp attribute $TS(\text{add CD}) = \text{add CD}$, which yields four entries $(o_1, 25-11,8:31)$, $(o_1, 25-11,12:11)$, $(o_2, 26-11,9:30)$, and $(o_2, 26-11,9:31)$. For event ship available, the projection onto $Inst(\Sigma_{order}) = \text{orderID}$ and $TS(\text{ship available}) = \text{ship}$ yields two entries $(o_1, 5-12,9:32)$ and $(o_2, 5-12,11:33)$, duplicates are removed. Extracting all other events and grouping them by orderID yields two cases: $\rho_{o_1} = \langle \text{add CD}, \text{add CD}, \text{order}, \text{ship}, \text{close} \rangle$ and $\rho_{o_2} = \langle \text{add CD}, \text{add CD}, \text{order}, \text{notify}, \text{ship}, \text{close} \rangle$.

4.2 Checking Behavioral Conformance of Artifacts with Existing Techniques

With the notions of an artifact view (Def. 1) and automatic log extraction (Def. 2), we reduced the behavioral conformance problem to a classical setting: behavioral conformance of artifacts can be checked using existing conformance checkers.

Given a database \mathcal{D} and a proclat system $\mathcal{P} = (\{P_1, \dots, P_n\}, C)$ where each proclat P_i describes an artifact of the system, first define an artifact viewpoint V that specifies for each proclat P_i and each transition label a in P_i an event type $a \in \Sigma_i$ in terms of \mathcal{D} . Then extract the artifact logs L_1, \dots, L_n from \mathcal{D} using V .

Then check behavioral conformance of each proclat P_i w.r.t. \mathcal{D} by checking conformance of the Petri net that underlies P_i w.r.t. the log L_i , by ignoring the ports of P_i . A corresponding conformance checker [12] tries to replay each case ρ in L_i by firing transitions of P_i in the order given in ρ . If a transition cannot be fired, the checker searches for a log ρ' that is as similar to ρ as possible and that can be replayed in P_i . We implemented this approach: logs can be extracted using XESame [15], the Process Mining Toolkit ProM checks conformance of a proclat system and provides diagnostics on non-conformance per artifact case (Fig. 5).

The life cycle model of order of Fig. 2 conforms to the log L_{order} extracted from the database of Fig. 4 i.e., the two traces just presented. The conformance checker [12] will also report for $\text{orderID} = o_1$ that an “unobservable activity” occurred (to bypass notify). The cases for quote of Fig. 4 stored in Fig. 4 yield a different result. Here, the trace of $\text{quoteID} = q_1$ lacks an event for generate invoice and an “activity in the model that was not logged” is reported. The trace of $\text{quoteID} = q_2$ generates an invoice before the order is processed, so an “activity of the log that was not (yet) enabled” is reported.

4.3 Checking Interaction Conformance of Artifacts

We just showed how to check behavioral conformance artifacts, i.e., whether the internal life cycles of each artifact, described by a proclat, conform to the artifact cases stored in a database \mathcal{D} . Complete artifact conformance also requires to check conformance w.r.t. interactions between proclats. In the following, we sketch how to leverage the notions of a viewpoint (Def. 1) and of log extraction (Def. 2) to extract so called *instance aware logs*. Using instance aware logs, interaction conformance of artifacts can be checked again using existing techniques [13].

In an *instance-aware log*, an event $e = (a, id, SID, RID)$ not only describes that an event of type a occurred in instance id ; it also describes from which instances SID the event *consumed* a message, and for which instances RID the event *produced* a message. For instance, the *instance-aware cases* of artifact order of Fig. 3 are

$$\begin{aligned} \rho_{o1} &: \langle (\text{add CD}, o_1, [q_1], []), (\text{add CD}, o_1, [q_2], []), (\text{order at supplier}, o_1, [], []), \\ &\quad (\text{ship available}, o_1, [], [q_1, q_2]), (\text{close order}, o_1, [], []) \rangle \\ \rho_{o2} &: \langle (\text{add CD}, o_2, [q_2], []), (\text{add CD}, o_2, [q_2], []), (\text{order at supplier}, o_2, [], []), \\ &\quad (\text{notify unavailable}, o_2, [], [q_2]), (\text{ship available}, o_2, [], [q_2]), (\text{close order}, o_1, [], []) \rangle \end{aligned}$$

This information suffices to enrich each instance-aware case of an instance i with those events that produced a message for i or consumed a message from i . The resulting cases equivalently capture the interaction behavior that is stored in \mathcal{D} , and they can be fed to

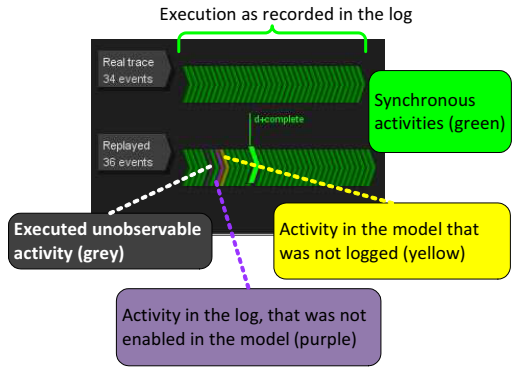


Fig. 5. Screenshot of ProM, showing conformance results of the proclat system of Fig. 2

existing conformance checkers [13]. To extract SID from \mathcal{D} , the artifact view (Def. 1) needs to be extended.

Events of type $a \in \Sigma_i$ may consume messages that were produced by a specific artifact. The attribute A_{id}^s that distinguishes the different instances of that artifact must be specified. The instance identifier $Inst(\Sigma_i)$ of the artifact of $a \in \Sigma_i$ and A_{id}^s must be connected by tables \mathcal{T}' of \mathcal{D} . Not every connection between $Inst(\Sigma_i)$ and A_{id}^s implies that a message was exchanged; a guard g over \mathcal{T}' specifies when this is the case. For instance, the set SID of ship available contains all identifiers of attribute `quoteID` when the guard `deliverable = true` evaluates to true.

The log extraction (Def. 2) needs to be extended correspondingly. For each entry $t = (id, ts) \in T_a$ in the event table of event type $a \in \Sigma_i$, extract values for SID as follows: joining all tables that connect attributes $Inst(\Sigma_i)$ and A_{id}^s , select from the result only the entries which satisfy $Inst(\Sigma_i) = id \wedge TS(a) = ts \wedge g$ (i.e., entries referring to t where also the guard g holds), and project the result onto A_{id}^s . The set RID of instances for which e produced a message is specified and extracted likewise. This procedure yields instance-aware logs L_1, \dots, L_n , one for each artifact in \mathcal{D} .

5 Conclusion

In this paper, we considered the problem of checking how a given process model *conforms* to executions of the actual process — under the realistic assumption that process executions are *not* structured into monolithic process instances. Rather, executions of most processes in reality are driven by their data objects which may participate in various, overlapping *cases*. Usually, the life cycle history of each objects that is involved in a process execution is recorded in a structured database. Likewise, the objects, their life cycles, and their interactions can be expressed in an artifact-centric process models, for instance using *proclats* [2].

In this setting, the *conformance problem* is to check how good a given proclat system describes all events recorded in the database. We decomposed this conformance problem in Sect. 3 into (1) the *behavioral conformance problem* on how good a proclat describes events of an artifact instance, and (2) the *interaction conformance problem* on how good the proclat system artifact interactions. Section 4 reduced behavioral conformance to classical conformance by extracting a classical process log for each artifact life cycle from the given database; technically, the log follows from a view on the database. The technique is likewise applicable for checking interaction conformance [13]; it is implemented in the Process Mining Toolkit ProM.

Related Work. Conformance checking, that is, comparing formal process models to actual process executions is a relatively new field that was studied first on monolithic processes with isolated process instances [16]. To the best of our knowledge, the conformance problem has not been studied yet for artifact-centric processes. Our approach currently only reduces artifact conformance to classical conformance. Yet, classical conformance checking knows several metrics which describe conformance differently [16].

The most advanced conformance metrics reflect that only parts of a trace are deviating [10, 17], and pinpoint where deviations occur [11], while taking into account that models may contain behavior that is unobservable by nature [12]. In particular the last

metric can be applied to several process modeling languages, including procllets used in Sect. 2 to describe artifacts.

Open Issues. This paper made a first step towards checking conformance of artifact-centric process models. Currently, we manually have to specify the artifact view on the database by identifying which tables relate to which artifact, and which attributes relate to which event. This can be cumbersome, as the relations between tables (expressed by foreign-primary key relations) need to be respected. A view is insensitive to adding further tables or attributes to the database, but sensitive to changes in the key relations. For this reason, automated techniques for checking *structural conformance* of a given procllet system to a database, and for *discovering* conformant artifact views for a given procllet system from a database would be required. Furthermore, metrics such as [12] need to be adapted to the artifact setting to describe the *degree* to which a process model describes observed executions. Finally, as artifact-centric processes are data-driven, also conformance of data-dependent guards to recorded process executions is an open issue.

Acknowledgements. The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 under grant agreement no. 257593 (ACSI).

References

1. Rozinat, A., Jong, I., Gunther, C., Aalst, W.: Conformance Analysis of ASML's Test Process. In: GRCIS 2009. CEUR-WS.org, vol. 459, pp. 1–15 (2009)
2. Aalst, W., Barthelmess, P., Ellis, C., Wainer, J.: Procllets: A Framework for Lightweight Interacting Workflow Processes. *Int. J. Cooperative Inf. Syst.* 10, 443–481 (2001)
3. Nigam, A., Caswell, N.: Business artifacts: An approach to operational specification. *IBM Systems Journal* 42, 428–445 (2003)
4. Cohn, D., Hull, R.: Business artifacts: A data-centric approach to modeling business operations and processes. *IEEE Data Eng. Bull.* 32, 3–9 (2009)
5. Fritz, C., Hull, R., Su, J.: Automatic construction of simple artifact-based business processes. In: ICDT 2009. ACM ICPS, vol. 361, pp. 225–238 (2009)
6. Lohmann, N., Wolf, K.: Artifact-centric choreographies. In: Maglio, P.P., Weske, M., Yang, J., Fantinato, M. (eds.) ICSOC 2010. LNCS, vol. 6470, pp. 32–46. Springer, Heidelberg (2010)
7. Rozinat, A., de Medeiros, A.K.A., Günther, C.W., Weijters, A.J.M.M.T., van der Aalst, W.M.P.: The need for a process mining evaluation framework in research and practice. In: ter Hofstede, A.H.M., Benatallah, B., Paik, H.-Y. (eds.) BPM Workshops 2007. LNCS, vol. 4928, pp. 84–89. Springer, Heidelberg (2008)
8. Greco, G., Guzzo, A., Pontieri, L., Sacca, D.: Discovering Expressive Process Models by Clustering Log Traces. *IEEE Trans. on Knowl. and Data Eng.* 18, 1010–1027 (2006)
9. Weijters, A., van der Aalst, W.: Rediscovering Workflow Models from Event-Based Data using Little Thumb. *Integrated Computer-Aided Engineering* 10, 151–162 (2003)
10. Medeiros, A., Weijters, A., van der Aalst, W.: Genetic Process Mining: An Experimental Evaluation. *Data Mining and Knowledge Discovery* 14, 245–304 (2007)
11. Rozinat, A., van der Aalst, W.: Conformance Checking of Processes Based on Monitoring Real Behavior. *Information Systems* 33, 64–95 (2008)
12. Adriansyah, A., Dongen, B., Aalst, W.: Towards Robust Conformance Checking. In: BPM 2010 Workshops (2010) (LNBIP to appear)

13. Fahland, D., de Leoni, M., van Dongen, B., van der Aalst, W.: Checking behavioral conformance of artifacts. BPM Center Report BPM-11-08 (2011), BPMcenter.org
14. Silberschatz, A., Korth, H.F., Sudarshan, S.: Database System Concepts, 4th edn. McGraw-Hill Book Company, New York (2001)
15. Verbeek, H., Buijs, J.C., van Dongen, B.F., van der Aalst, W.M.P.: ProM: The Process Mining Toolkit. In: BPM Demos 2010. CEUR-WS, vol. 615 (2010)
16. Rozinat, A., de Medeiros, A.A., Günther, C., Weijters, A., van der Aalst, W.: Towards an Evaluation Framework for Process Mining Algorithms (2007), BPM Center Report BPM-07-06
17. Weijters, A., van der Aalst, W., de Medeiros, A.A.: Process Mining with the Heuristics Miner-algorithm. Technical report, Eindhoven University of Technology, Eindhoven, BETA Working Paper Series, WP 166 (2006)

Framework for Business Process Verification

Andreas Speck¹, Sören Witt¹, Sven Feja¹, Aneta Lotyzc¹, and Elke Pulvermüller²

¹ Christian-Albrechts-University Kiel

Olshausenstrasse 40, 24098 Kiel, Germany

{`aspe,svfe,swi`}@informatik.uni-kiel.de, `aneta.lotyzc@email.uni-kiel.de`

² University of Osnabrueck

Albrechtstr. 28, 49076 Osnabrueck, Germany

`elke.pulvermueller@informatik.uni-osnabrueck.de`

Abstract. There are numerous concepts and tools for modeling business processes and several academic approaches to verify business processes. However, most modeling tools don't integrate the checking of the processes. The three-tier architecture of the **B**usiness **A**pplication **M**odeler (BAM) provides the graphical representation of business models and rules (presentation layer) as well as integrates a verification mechanism layer with an intermediate transformation layer.

The hierarchical architecture allows to extend the checking system with additional features supporting the use of the verifier in an industrial environment of commercial information system development. Examples for such add-ons are the improved graphical rule notation and an enhanced functionality of the verification system supporting the expressiveness in the business modeling domain.

Keywords: Business process models and rules, graphical rule notation (Graphical-CTL), model and rule transformation, extended model checking.

1 Introduction and Requirements

The behavior of business information systems is mostly described by business process models. These models are bridging the user's view and the implementation of commercial systems. To verify business processes created at analysis and design time is desirable. In a late state of the system development process the cost to repair incorrect business processes are extremely high. Therefore, it is reasonable to identify errors at design time. This may be considered as common sense in the software development [1].

The most interesting case is to verify the behavior and the temporal order of the activities of business process. Are the right functions in the process and are the functions in the correct order? Although these are crucial questions when business processes are designed there is little support by commercial business process modeling tools [2]. The existing tools (e.g. ARIS) end-up in purely syntactical checks.

1.1 Rules and Best Practices as Business Process Specifications

Rules and best practices are depending on the domain they occur. In the paper we present examples of the e-commerce domain [3] (however, other systems like ERP systems realize similar processes).

In the e-commerce systems domain we have specific rules based on experience and pattern which we may expect. Examples of rules are:

1. In any case the system has to provide a product presentation to the customer. This is actually the base presentation and therefore expected even if there is an empty presentation.
2. There must be at least one opportunity for the customer for searching a product.
Since the product search is essential for the customer this option has to be possible.
3. If the customer is of privileged type (e.g. *Centralized buyer*) then she or he should be offered a personalized offer.
In any case after a certain point after the successful login different types of personalized offers have to be presented.
4. In case a price alert is activated the price has to be checked until the threshold is met directly after the customer has to be informed.
The intention of the price alert functionality is to inform customers directly after the threshold is met.

Examples for specific patterns which may be expected are:

1. For customers there is – besides a simple search possibility – at least a complex search accepting multiple search items and both search types are then processed by one multi-purpose system function.
2. At an invoice there is an *Invoice order* and the *Announcing invoice* for the customer followed by the *Invoice commission*.

In many cases there are temporal dependencies. For example, when a functions occurs a specific time later another function is expected or should be possible. How these rules may be expressed in a formal way or the corresponding requirements we present in the following subsection.

1.2 Editing Process Models and Specifications

A verification system has to support both the modeling of the system which means the modeling of the business processes as well as the modeling of the rules (the specifications). Ideally, the editing of business process models and specifications may be performed with the same or similar editor or tool. The elements used in the business process model language should be reused and enriched in the specification notation. This means that the operators used to express the specification need to be combined with the business process notations. It is desirable that the operators are expressed by graphical symbols.

1.3 Checking and Error Detection

The main concern of the verification system is the checking of temporal dependencies. This may be the conclusion of the rules presented in subsection 1.1.1. Moreover, the verification should consider all types of elements of the specific business process notation in general and the elements which express the control flow in particular. It is also desirable to distinguish between the different element types and to focus only on specific element types when a model is checked.

Many checking concepts like model checking present a simple “OK” when the checked model fulfills the rule(s). In an error case model checking presents a single counter example. It is desirable that the counter example is presented in the business process model which is checked. Any further information about a successful checking or error case, e.g. the presentation of an error pattern leading to the problem would further improve the verification concept.

2 Architecture

Considering the requirements of the previous section 1.1 it is clear that a specific system like a model checker is not sufficient as verification system for business process models. Moreover, a checking system must consist of different subsystems being concerned with the representation of models and specifications (graphical presentations are most desirable), a generic intermediate model for business processes and specifications as well as transformation mechanisms and finally an enhanced checking system based on established verification concepts such as model checking.

As depicted in figure 1 our framework, therefore, comprises of three layers. The data exchange between the layers is kept simple.

- The business process model and temporal specification of the *graphical representation layer* are transformed in the *transformation layer* serving as input for the checking tool of the *verification layer*.

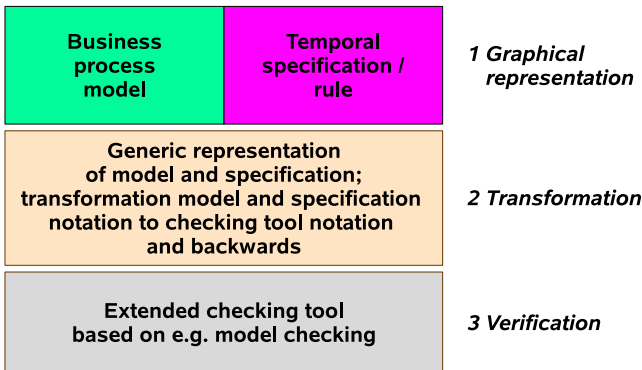


Fig. 1. Basic Architecture of the Business Process Checking Framework

- The result(s) of the verification (of the *verification layer*) are transformed backwards and in an error case are presented in the business process model.

In this architecture the different components (business process model editor, graphical rule editor, transformation and checking system) may flexibly be modified to specific requirements of the business process modeling.

2.1 Graphical Representation

According to the requirements presented in section 1 the graphical representation has to capture both the business process model as well as the specifications (or rules).

The system we implemented to elaborate the integrated approach is the **Business Application Modeler (BAM)** [4]. BAM is based on the Eclipse Graphical Editing Framework (GEF) [5]. As example for this paper we use EPCs (Event-driven Process Chains) as business process models of the ARIS modeling concept (Architecture of Integrated Information Systems). This is an arbitrary choice and as such just one model type for demonstration purposes of the framework. UML activity diagrams as well as BPMN models, for instance, could be supported similarly.

In figure 2 one business process model (a search functionality, e.g. of an e-commerce system) and two corresponding rules are displayed.

The rules notation is based on the specific business process notation, ARIS EPC models in our example. In order to express the temporal relationships we

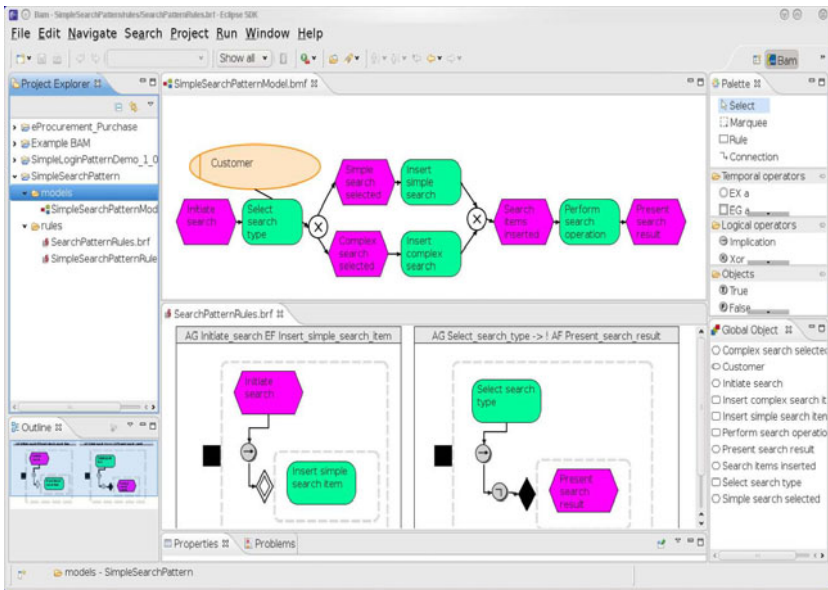


Fig. 2. Graphical Representation: Business Application Modeler

apply temporal logic operators. Our operators are based on CTL (Computational Tree Logic). This graphical notation is called G-CTL (Graphical Computational Tree Logic) [6]. The operators of the temporal logic are represented by graphical symbols and the textual notation is transferred to a graphical one.

Without going into detail about the specification (further information may be found in [7]) the meaning of the two rules is:

1. Rule (left window): Always globally it has to be true that if the event *Initiate search* has occurred (became true) implicitly exists in the future the function *Insert simple search item*.

Or in other words: When the event *Initiate search* occurred in the following of the process there must be at least one branch with the function *Insert simple search item*.

2. Rule (right window): Always globally it has to be true that if the function *Select search type* has occurred (became true) implicitly always in the future the event *Present search result* becomes true (or holds, respectively).

Or in other words: When the function *Select search type* occurred (which means that the search function is activated by the user) in the following of the process there must be on all branches the event *Present search result*. This means that at least an empty search result is presented but it has to become clear that there has been a search.

2.2 Transformation

In order to check the business process models these and the specifications have to be transformed into documents which may be processed by a checking tool. This document is a textual description of the behavior of a process instance. Hence the semantics of the process model needs to be applied and a temporal behavior must be assumed. Here, mainly the semantics regarding the switching of the operators is of interest. It determines the overall behavior of the process.

Examples for research approaches to formalize the semantics of EPCs and to develop algorithms are [8] and [9]. Here, especially the non-local semantics of the (*x*)*or*-join has been challenging. [9] provides an overview over several existing semantics. In general, these semantics differ in the assumptions concerning the processes model (e.g. forbidden constellations) on the one hand. On the other hand, the resulting process behavior is defined, e.g. if a process has deadlocks under certain conditions or not. The semantics should be chosen depending on, which of them matches the behavior of a real process instance (e.g. execution inside a SOA environment) best.

Another semantic aspect that needs to be considered, is the decision whether a process element conceptually consumes time or not. So one may assume that operators do not consume time, since they probably do not have a representation in the real process instance. Functions (or any other kind of modeled activity) consume time for sure. Whether events may consume time, is a matter of discussion. One may find arguments for both variants.

We are not going into detail or compare these semantics in this paper. Any of them can be calculated on the basis of the process model before anything is

passed to the model checker or another checking tool. The result of calculating the semantics may be a reachability problem in a graph, that shows how the process may evolve. In the context of an EPC the overall state of a process is defined by currently active model elements. We define each state as a set of sub-states. A sub-state is built around each concurrently active element, that consumes time. Therefore, an EPC sub-state could, for example, consist of a function to which a cluster is attached. It may exist concurrently beside another sub-state built around another function. All active sub-states define the current overall state of the process.

Erroneous results of this semantic processing may already allow to conclude about the validity of the process. If no errors occurred, the outcome is the extra information, which may be used to derive a state machine for a model checker from the process model. In order to check a process model using a model checker, the process model must be represented in the according input language, implementing the semantic decisions made before. We do not want to go into detail about a possible representation, since this is specific for each model checker.

Besides the transformation of the process model, the rules need to be transformed accordingly into the textual representation for the checking tool. The process elements in rules can be seen as patterns, which can be matched to sub-states of the process model. Hence, one or more sub-states may fulfill a pattern in a rule. While the G-CTL operators are transformed straight-forward, the process patterns are replaced by a test, whether the according sub-state is currently active. Of course this test must be congruent to the sub-state representation in the process transformation.

After the checking tool is called and results are available, the transformation layer is also responsible for parsing the checking results. The textual results are mapped back to the graphical elements of the process model.

2.3 Extended Verification

The verification has to be tool supported. One option is to use a “standard” checking tool like SMV (Symbolic Model Verifier) [10] or one of its derivate systems. A drawback of these well-known checkers is that they support only one type of state. In such a direct transformation of the EPC models into an automata model we may transform the elements *event* and *function* directly into states (only one type of states) which are connected according to the control flow.

However, business process models like EPCs provide two types of states in the control flow: functions and events. It is desirable to distinguish between different types of states, in our example the functions and events. An example of a checking concept distinguishing different types of states is the COV (Component Verifier) [11].

In general, CTL model checkers need automata models as input for the checking. These automata models are represented in a specific structure, e.g. the Kripke structure. However, as a simplification we use ordinary automata representations (which may easily be transformed into Kripke structures [12]).

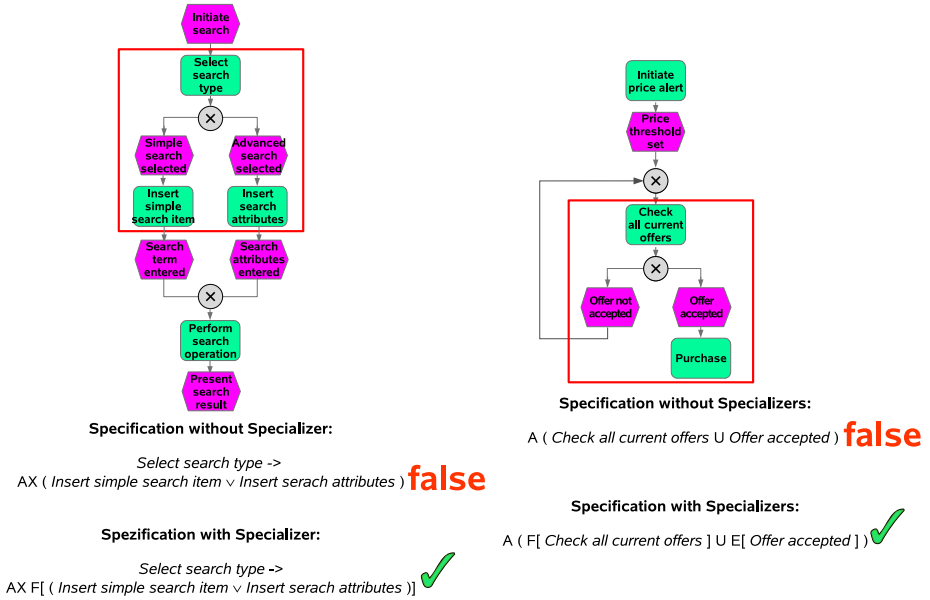


Fig. 3. Specializers in Temporal Logic

The two examples in figure 3 will illustrate the usage and advantage of differentiating between the states. The upper specifications do not distinguish between different elements. The lower specification distinguish by using specializers: F for functions and E for events. In the example on the left side the specification without specializers expects that directly after the function *Select search type* the functions *Insert simple search item* or *Insert complex search item* have to follow. The model on which this rule is applied is the already known search example. Such a specification or rule may be defined in the situation when we would like to keep the denomination of the events after *Select search type* open (e.g. for customizing at design time) and are only interested in that they are followed by standard functions (such as *Insert simple search item* or *Insert complex search item*).

The lower specification contains the specializer F (for function) directly after the *Always neXt* (AX). This indicates that only function elements have to be considered in the checking. An event (or an element of another type) is ignored. This specification is true (as we would expect it in the domain semantics).

The example on the right side of figure 3 contains a loop. The process is a price alert process. If a price falls below (or rises upon) a certain threshold there is a price alert and the system purchases.

It may be of interest if the process *Check all current offers* is performed until the threshold is met and the *Offer (is) accepted*. The first specification without specializers turns out to be false although the model meets the requirement. When we use the specializers then the specification is true in our model as expected. In our example *Always* the function *Check(s) all current offers* (due to

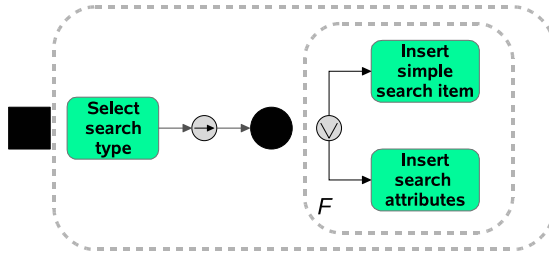


Fig. 4. *Extended Graphical-CTL (EG-CTL)*

the specializer only functions are considered) *Until* the event *Offer accepted* becomes true. The event *Offer not accepted* is in the loop back to the function *Check all current offers*. Since *Offer not accepted* is an event and not a function the checker does not care about it during the checking process due to the appropriate usage of the specializer.

With this specializer concept in temporal logic specifications we are able to select specific element types and focus on these. This is an extension of the temporal logic CTL we call ECTL (Extended CTL). In order to handle the specializers the algorithm of the model checker has to be modified. A more detailed description of the modified checking algorithm may be found in [11]. Due to the specializers the specification may be more precise. The expressiveness of the temporal logic is extended and captures different types of elements.

The graphical version of the extended CTL is the *Extended Graphical-CTL* (EG-CTL). Figure 4 presents the graphical version of the ECTL rule of the example in the left of figure 3.

3 Related Work

The first approaches to verify software models with systems like model checkers appeared in the 1980's. Examples for early approaches based on model checking are [13] or [10]. However, despite the presence of model checking there is still a gap concerning its applicability to business processes. The base of all the business process checking is the formalization of business process models like [14] (graph grammar based approach) and [15] which enable to apply formal methods for business processes. The transformation of EPC models to Petri nets also formalize them [16]. In this case the semantics are restricted. The formalization proposed in [8] uses a fix-point-semantics-based definition of the semantics of EPCs which is also used for model checking.

Many approaches (also the majority of the here referenced approaches) consider BPMN. However, executable models like BPEL are object of formalization as well [17]. Further formalization approaches are based on the pi-calculus [18]. An issue of research to be addressed by formalization approaches are the joins after branching in general and the problem of OR-joins in specific [15]. In the EPC model [19] the semantics as base of the formalization have been analyzed e.g. by [20]. An example for the BPMN analysis may be found in [21].

Examples for approaches employing model checking on business processes are [22], [23] or [24]. [22] evaluates different checking technologies for being applied on business processes. [23] and [24] focus on the aspect of transactions in e-commerce systems. In [25] a large number of business processes have been investigated and different checking concepts are applied. One important conclusion is that several concepts could be combined in order to improve their effect. An example of an approach for the verification of business process systems based on Petri nets is [26] using BPMN. With Petri nets the business processes are mapped to the Petri net elements similar to our Kripke structure mapping. Petri net based verification is, for instance, based on bi-simulation and algebraic solutions as in [27].

The approach presented in this paper relies directly on the push-button model checking technology and temporal logic requirement specifications. Most approaches applying formal methods for checking business process models use straight-forward model transformations. These transformations result in a loss of information and, therefore, verification precision. The reason is the incompatible semantics of the business process models and the verification models which causes several problems resulting in different alternative approaches to tackle them [28]. Moreover, additional information (such as organizational units in EPC models) is lost during the transformation due to a surjective mapping of notational elements between business process model and verification model. Two approaches transforming business process models to verification models are [29] (SMV Kripke structures) or [30] (Petri nets).

An approach which proposes a graphical representation of models and specifications is [31]. In this approach the business process notation are UML activity diagrams and the result of the LTL-based checking is presented in a textual manner. The VERBUS approach (Verification for Business Processes) provides a architecture with three layers which is similar to our approach [32]. This approach uses BPEL4WS as model and works with the well-known model checkers Spin and SMV (or NuSMV).

In the domain of formal methods approaches may be found which concentrate on an increase of semantic expressiveness of the specification languages (e.g. the μ -calculus [33] and [34] or in the multi-valued logic research as in [35]). Extensions to the temporal logic for LTL have been proposed in [36] or [37], for instance. In these approaches a link to software models or business process models is missing and the general idea of a specialization on different model elements is not considered. In contrast to [36], [37], [38] and [39] we are able to explicitly distinguish and mix specializers for different model elements.

4 Conclusions and Future Work

The business process verification framework allows to extend existing modeling tools and specification notations. Furthermore it integrates verification systems supporting new and more specific checking operators.

The specification uses a graphical notation based on the business process notation enriched with temporal operators of the temporal logic

CTL (Computational Tree Logic). This graphical specification language is called G-CTL (Graphical-Computational Tree Logic). Moreover an enhanced model checker is integrated supporting further checking operators such as the specializers. These allow to differentiate between different types of business process model elements in the checking algorithm. The new CTL-based logic is called ECTL (Extended CTL). The graphical version of the logic is EG-CTL. The integration concept is realized by the Eclipse-based **B**usiness **A**pplication **M**odeler (BAM).

Currently our modeling concept is applied to other notations. The latest version of the editor supports the modeling language i^* (pronounced “i star”). A modeling concept used in the early phase of the development in order to understand the problem domain. However, other common notations such as BPMN may also be supported by BAM. Further specification operators may enhance the checking language. At the moment we are working with wild card operators which allow to abstract from a certain element in the business process.

References

1. Sommerville, I.: Software Engineering, 9th edn. Addison-Wesley, Reading (2010)
2. Wynn, M.T., Verbeek, H.M.W., van der Aalst, W.M.P., ter Hofstede, A.H.M., Edmond, D.: Business Process Verification - Finally a Reality! *Business Process Management Journal* 15(1), 74–92 (2009)
3. Breitling, M.: Business Consulting, Service Packages & Benefits. Technical report, Intershop Customer Services, Jena (2002)
4. Feja, S., Speck, A., Witt, S., Schulz, M.: Checkable graphical business process representation. In: Catania, B., Ivanović, M., Thalheim, B. (eds.) ADBIS 2010. LNCS, vol. 6295, pp. 176–189. Springer, Heidelberg (2010)
5. Anders, E.: Modellierung und Validierung von Prozessmodellen auf Basis variabler Modellierungsnotationen und Validierungsmethoden als Erweiterung für Eclipse, Diploma Thesis (2010)
6. Feja, S., Fötsch, D.: Model Checking with Graphical Validation Rules. In: 15th IEEE International Conference on the Engineering of Computer-Based Systems (ECBS 2008), Belfast, NI, GB, pp. 117–125. IEEE Computer Society, Los Alamitos (2008)
7. Pulvermüller, E., Feja, S., Speck, A.: Developer-friendly Verification of Process-based Systems. *Knowledge Based Systems* 23(7), 667–676 (2010)
8. Kindler, E.: On the semantics of ePCs: A framework for resolving the vicious circle. In: Desel, J., Pernici, B., Weske, M. (eds.) BPM 2004. LNCS, vol. 3080, pp. 82–97. Springer, Heidelberg (2004)
9. Mendling, J.: Metrics for Process Models: Empirical Foundations of Verification, Error Prediction, and Guidelines for Correctness, 1st edn. Springer, Berlin (2008)
10. McMillan, K.L.: Symbolic Model Checking. Kluwer Academic Publishers, Dordrecht (1993)
11. Pulvermüller, E.: Reducing the Gap between Verification Models and Software Development Models. In: The 8th International Conference on Software Methodologies, Tools and Techniques (SoMeT 2009), pp. 297–313. IOS Press, Amsterdam (2009)

12. Bérard, B., Bidoit, M., Finkel, A., Laroussinie, F., Petit, A., Petrucci, L., Schnoebelen, P.: *Systems and Software Verification – Model-Checking Techniques and Tools*. Springer, Berlin (2001)
13. Emerson, E.A., Clarke, E.M.: *Characterizing Correctness Properties of Parallel Programs Using Fixpoints*. In: de Bakker, J.W., van Leeuwen, J. (eds.) *ICALP 1980*. LNCS, vol. 85, pp. 169–181. Springer, Heidelberg (1980)
14. Klauck, C., Müller, H.J.: *Formal business process engineering based on graph grammars*. *International Journal on Production Economics* 50, 129–140 (1999)
15. Börger, E., Sörensen, O., Thalheim, B.: *On Defining the Behavior of OR-joins in Business Process Models*. *The Journal of Universal Computer Science (J. UCS)* 15(1), 3–32 (2009)
16. Langner, P., Schneider, C., Wehler, J.: *Petri net based certification of event-driven process chains*. In: Desel, J., Silva, M. (eds.) *ICATPN 1998*. LNCS, vol. 1420, pp. 286–305. Springer, Heidelberg (1998)
17. Cámara, J., Canal, C., Cubo, J., Vallecillo, A.: *Formalizing WSBPEL Business Processes Using Process Algebra*. *Electronic Notes in Theoretical Computer Science* 154(1), 159–173 (2006)
18. Ma, S., Zhang, L., He, J.: *Towards formalization and verification of unified business process model based on pi calculus*. In: *Proceedings of the 6th ACIS International Conference on Software Engineering Research, Management and Applications (SERA)*, pp. 93–101. IEEE Computer Society, Los Alamitos (2008)
19. Scheer, A.W.: *ARIS - Modellierungsmethoden, Metamodelle, Awendungen*. Springer, Berlin (1998)
20. Mendling, J., Neumann, G., van der Aalst, W.M.P.: *Understanding the Occurrence of Errors in Process Models Based on Metrics*. In: Chung, S. (ed.) *OTM 2007, Part I*. LNCS, vol. 4803, pp. 113–130. Springer, Heidelberg (2007)
21. Grosskopf, A.: *xBPMN. Formal control flow specification of a BPMN based process execution language*, Master’s thesis (2007)
22. Köhler, J., Tirenni, G., Kumaran, S.: *From Business Process Model to Consistent Implementation: A Case for Formal Verification Methods*. In: *6th International Enterprise Distributed Object Computing Conference (EDOC 2002)*, pp. 96–106 (2002)
23. Anderson, B.B., Hansen, J.V., Lowry, P.B., Summers, S.L.: *Model checking for design and assurance of e-Business processes*. *Decision Support Systems* 39(3), 333–344 (2005)
24. Anderson, B.B., Hansen, J.V., Lowry, P.B., Summers, S.L.: *The application of model checking for securing e-commerce transactions*. *Communications of the ACM* 49(6), 97–101 (2006)
25. 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)
26. De Backer, M., Snoeck, M.: *Business Process Verification: a Petri Net Approach*. Technical report, Catholic University of Leuven, Belgium (2008)
27. Morimoto, S.: *A survey of formal verification for business process modeling*. In: Bubak, M., van Albada, G.D., Dongarra, J., Sloot, P.M.A. (eds.) *ICCS 2008, Part II*. LNCS, vol. 5102, pp. 514–522. Springer, Heidelberg (2008)
28. van Dongen, B.F., Jansen-Vullers, M., Verbeekm, H.H.M.W., van der Aalst, W.M.P.: *Verification of the SAP reference models using EPC reduction, state-space analysis, and invariants*. *Computers in Industry* 58(6), 578–601 (2007)

29. Pulvermüller, E.: Composition and correctness. *Electronic Notes in Theoretical Computer Science (ENTCS)* 65(4) (2002)
30. van der Aalst, W.M.P.: Formalization and Verification of Event-driven Process Chains. *Information and Software Technology* 41(10), 639–650 (1999)
31. Förster, A., Engels, G., Schattkowsky, T., Van Der Straeten, R.: Verification of Business Process Quality Constraints Based on Visual Process Patterns. In: *Proceedings of the First Joint IEEE/IFIP Symposium on Theoretical Aspects of Software Engineering (TASE 2007)*, pp. 197–208 (2007)
32. Fisteus, J.A., Fernández, L.S., Kloos, C.D.: Applying model checking to BPEL4WS business collaborations. In: *Proceedings of the 2005 ACM Symposium on Applied Computing (SAC)*, pp. 826–830. ACM, New York (2005)
33. Bradfield, J., Stirling, C.: Modal logics and mu-calculi: an introduction. In: *Handbook of Process Algebra*, pp. 293–330. Elsevier Science Publishers, Amsterdam (2001)
34. Kozen, D.: Results on the propositional mu-calculus. *Theoretical Computer Science* 3(27), 333–354 (1983)
35. Chechik, M., Devereux, B., Easterbrook, S., Gurfinkel, A.: Multi-Valued Symbolic Model-Checking. *ACM Transactions on Software Engineering Methodology* 12(4), 371–408 (2003)
36. Chaki, S., Clarke, E.M., Ouaknine, J., Sharygina, N., Sinha, N.: State/Event-based software model checking. In: Boiten, E.A., Derrick, J., Smith, G.P. (eds.) *IFM 2004*. LNCS, vol. 2999, pp. 128–147. Springer, Heidelberg (2004)
37. Jonsson, B., Khan, A.H., Parrow, J.: Implementing a Model Checking Algorithm by Adapting Existing Automated Tools. In: Sifakis, J. (ed.) *CAV 1989*. LNCS, vol. 407, pp. 179–188. Springer, Heidelberg (1990)
38. Giannakopoulou, D., Magee, J.: Fluent Model Checking for Event-based Systems. In: *Proceedings of the 9th European Software Engineering Conference (ESEC) held jointly with 10th ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE)*, pp. 257–266. ACM Press, New York (2003)
39. Kindler, E., Vesper, T.: ESTL: A temporal logic for events and states. In: Desel, J., Silva, M. (eds.) *ICATPN 1998*. LNCS, vol. 1420, pp. 365–384. Springer, Heidelberg (1998)

A Query-Driven Approach for Checking the Semantic Correctness of Ontology-Based Process Representations

Michael Fellmann, Oliver Thomas, and Bastian Busch

University of Osnabrueck,
Institute of Information Management and Corporate Governance,
Katharinenstr. 3, 49069 Osnabrueck, Germany
{Michael.Fellmann, Oliver.Thomas,
Bastian.Busch}@uni-osnabrueck.de

Abstract. The paper presents an approach to check the semantic correctness of business process models using queries in conjunction with an ontology-based process representation. The approach is based on the formalization of the semantics of individual model elements by annotating them with concepts of a formal ontology. In order to ensure semantic correctness, constraints are formalized as queries which are executed against the ontology-based process representation. The effectiveness of this approach is demonstrated by a user experiment. The experiment shows that searching for constraint violations using the query language produces more accurate results and is less time consuming in comparison to manual search when large models have to be checked.

Keywords: Process Modeling, Correctness, Ontology, Query, OWL, SPARQL.

1 Introduction

Models are important to manage complexity. They provide a means for understanding the business process, and understanding already is a benefit. This is indicated by a study from Gartner revealing an increase in efficiency of 12 percent gained solely by documenting actions and organizational responsibilities using process models [16, p. 4]. Moreover, process models serve for optimization, reengineering, and implementation of supporting IT systems. Due to the importance of process models, model quality is important. According to ISO 8402, quality is “the totality of characteristics of an entity that bear on its ability to satisfy stated and implied needs”. One fundamental aspect of model quality is the correctness of a model.

Approaches to check the correctness are often described as either “verification” or “validation” approaches. According to the IEEE 1012-1998 definition, “verification” means to check whether an artifact and/or its creation comply with a set of given requirements, thus targeting the internal constitution of a model. In contrast to that, validation means the eligibility of a model in respect to its intended use [3, p. 24] – in other words: if the criterion is something outside the model [2; 17, p. 2]. Hence, validation implies artifact-external aspects, human judgment and experience. Intuitively, verification of process models is more amenable to machine processing

than validation. However, human experience may also be externalized and formally specified in an ontology which can be connected to model elements via semantic annotations. In this way, annotated models with machine processable semantics in conjunction with formalized constraints also enable the automation of validation tasks. This contributes to the blurring line between validation and verification. Therefore, we prefer using the term “correctness checking” instead of either verification or validation.

While there are numerous approaches available to ensure (a) the syntactical correctness, (b) correctness in regard to the formal semantics and (c) correctness in regard to linguistic aspects, only a few approaches focus on the correctness of the model contents, i.e. its semantic correctness. A major problem regarding semantic correctness checks is how to automate them. This problem is rooted in natural language being used for labeling model elements, thus introducing terminological problems such as ambiguity (homonyms, synonyms) and other linguistic phenomena. Model creators and readers do not necessarily share the same understanding as the concepts they use are usually not documented and mix both discipline-specific terminology and informal, ordinary language. Therefore, it is hard for humans to judge if a model is semantically correct and almost impossible for machines (apart from using heuristics) because the model element labels are not backed with machine processable semantics. The result is that the machine cannot interpret the contents of model elements. Therefore, we have developed an approach to encode the model element semantics in a precise, machine readable form using OWL ontologies [8; 9; 10]. This approach is also introduced in section 3.1.

We extend our previous work in demonstrating that the query language SPARQL [18] is useful and usable to check the semantic correctness of an ontology-based process representation. We use this query language as (a) it is a language which has been standardized by the World Wide Web Consortium (W3C) and gained broad acceptance in the Semantic Web and AI community, (b) it provides a simple structure for queries consisting of triples which intuitively reflect the structure of process graphs and (c) the query language is not specialized to any specific process modeling language such as EPC, BPMN or UML Activity Diagrams but can be used with any modeling language. Moreover, SPARQL supports disjunction and negation and is thus more expressive than SWRL which we have used in our previous work.

In order to demonstrate the effectiveness of our approach, we present a user experiment in which we have compared manual search and browsing with the usage of SPARQL for checking process models. We report on the insights we have gained by exposing novice users to a query environment and comment on the overall feasibility and usefulness of our approach. To the best of our knowledge, this is the first empirical experiment testing the usage of SPARQL to query business process models in a real-world setting.

The paper is organized as follows. In section 2, we present related work. In section 3, we give a short introduction to our approach of semantic correctness checking which we evaluate and comment on in section 4. In section 5, we summarize and conclude the paper.

2 Related Work

In the area of *formal semantics of process models*, criteria such as „soundness“, „relaxed soundness“ or „well-structuredness“ have been developed. Originating from workflow management, these criteria are used to detect shortcomings such as deadlocks, livelocks, missing synchronisations and other defects regarding the formal semantics [4; 17, p. 7; 22]. Correctness in this sense abstracts from the individual semantics of model elements which is given by natural language and concentrates on formal procedures. It ensures that a model is built correctly which does not require that its content is correct.

Correctness beyond formal semantics is discussed e.g. in the context of compliance. Compliance can be understood as the conformity of something such as a process model to the entirety of relevant legal liabilities, directives and rules as well as to the internal guidelines and best practices of an enterprise [25]. This clearly goes beyond syntax and formal semantics and requires also checking the individual model elements and their semantics. Most approaches in this area aim at detecting compliance violations caused by the model structure or execution semantics [1; 11; 19] or the modeling style [12]. Some approaches also consider running processes [14] or the analysis of finished processes [21]. Although these approaches address aspects of semantic correctness and partly make use of machine reasoning, in contrast to our work they usually do not propose the use of a standard ontology language such as the Web Ontology Language (OWL).

First approaches to *ontology-based correctness checking* can be found in the context of Semantic Web Services. Semantically annotated process models are checked with an emphasis on logical preconditions and effects which are specified relative to an ontology [5; 23; 24]. These approaches usually require both the annotation of preconditions and effects and ensure that the model is consistent. They do not build upon a formal representation of the (intentional) semantics of individual model elements. Following this argument, a function “receive guest” and “welcome guest” in a hotel service process may have the same preconditions and effects, but differ considerably. Our approach enables capturing such differences by using a single annotation of a model element in order to associate it with its intended meaning explicitly specified in a formal ontology. Semantic constraints then allow checking if a model complies with a set of requirements using this explicitly specified meaning along with the deductions that are possible due to its formal representation. Our approach is therefore orthogonal to approaches considering preconditions and effects. So far, there are only a few works using constraints together with semantic process descriptions [6; 7; 15; 20]. To the best of our knowledge, none of these approaches makes use of the full expressivity of a description logic such as SHOIN(D) which is permitted by our approach.

3 Approach for Semantic Correctness Checking

3.1 Ontology-Based Process Representation

A first step towards checking the semantic correctness of semiformal process models is the representation of the process models using a formal ontology language. The use

of the ontology-based representation is twofold. On the one hand, it allows the connection of process models with domain knowledge in order to improve the interpretation and derive new facts not explicitly specified by the modeler but relevant for correctness checking. On the other hand, it provides for a machine processable representation enabling the automation of such derivations and therefore using logic and reasoning to automate correctness checking tasks.

For ontology creation, top-level or upper ontologies may be used as a basic backbone structure that helps bootstrap ontology development and reaching ontological commitment in the sense of a shared “contract” between the different involved stakeholders. We use the Suggested Upper Merged Ontology (SUMO) (see <http://ontologyportal.org>) as a backbone to structure the domain representation. This ontology provides basic distinctions such as between abstract and physical entities forming the basis of a subsumption hierarchy. The subsumption hierarchy of such ontologies does not only serve for disambiguation purposes (e.g. *Service* as subclass of *ComputerProcess* vs. subclass of *Product*). It also provides for the specification of semantic constraints on varying levels of generality. The ontology can easily be enriched with additional domain knowledge, e.g. encoded in the OWL version of the MIT process handbook containing the description of more than 8,000 business functions (see www.ifi.uzh.ch/ddis/ph-owl.html).

We use the Web Ontology Language (OWL) standardized by the W3C as it has gained a broad acceptance both inside and outside the AI and Semantic Web community. Specifically, we use OWL-DL 1.0 which corresponds to the description logic SHOIN(D) which is a decidable fragment of first order logic. Automatic classification can leverage existential restrictions of properties on classes as well as restrictions on their domain and ranges. Moreover, OWL-DL also provides specific characteristics of properties such as symmetry, transitivity, reflexivity etc. leading to additional inferences relevant for semantic constraint checking.

In the following, we give a short introduction to our ontology-based process representation (cf. Fig. 1). In general, we use an instance-centric representation approach since it fits the nature of SPARQL queries intended to operate on graph data and which we use for semantic constraint checking. The suggested ontology-based process representation consists of a model representation (cf. ❶ in Fig. 1) generated by instantiating generic concepts of process modeling languages (cf. ❷ in Fig. 1). This representation is annotated with domain knowledge (cf. ❸ in Fig. 1). The namespace-prefix `p:` is used for indicating the process space in general and `ex:` for indicating example processes. The creation of a process model representation ❶ in the ontology is done by considering its graph structure. For each node, an instance is created and for each arc, a property is created respectively. These model representation instances are annotated by using the property `p:equivalentTo` to link to instances representing elements of the process domain ❸ such as e.g. `ex:heck_admission`.

The properties having their domain and range on the `p:ProcessGraphNodeClass` are used to represent relations between nodes of the process graph. To allow for intuitive querying of the process graph (which is important to specify constraints using a query language later on), we use several properties with specific characteristics. Direct following relations are represented by properties in the form of `p:hasAfterX` where “X” denotes the type of the following node. As these properties

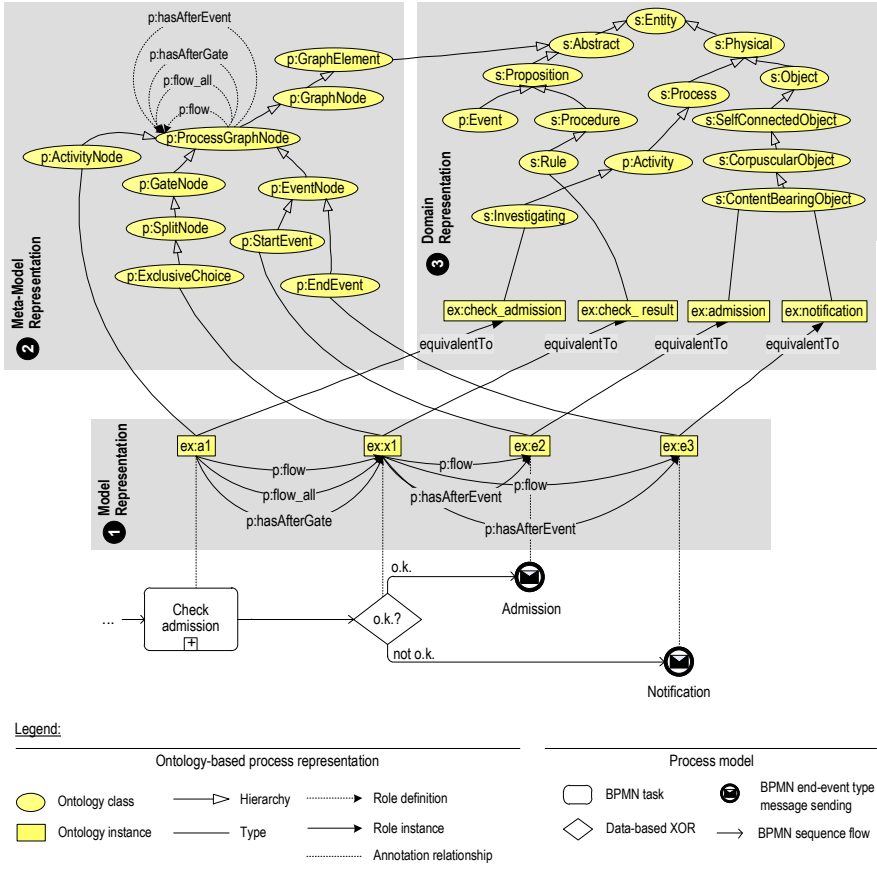


Fig. 1. Ontology-based Representation of Process Models

have domain and range restrictions, explicit typing of the nodes in a query is not necessary. The transitive relation $p:flow$ connects arbitrary nodes with the set of following nodes in the process graph. In a similar way, the transitive relation $p:flow_all$ connects nodes except for the case when an exclusive decision node such as an XOR-gate is reached (hence “all” means that literally all imaginary tokens in the control flow have to pass the path).

3.2 Using SPARQL Queries for Constraint Checking

OWL does not support constraints. Nevertheless, a limited form of constraints can be added by using the Semantic Web Rule Language (SWRL) [13]. SWRL adds to OWL the capability of specifying simple rules in the form of a logical implication. However, since OWL and SWRL are designed to adhere to the Open World Assumption, rules cannot easily be formulated that detect missing information. In an open world, this is in fact not reasonable as missing information is merely unknown,

and hence not absent or false. In contrast, for checking business process models, a sort of (temporarily) closed world can be assumed where the failure to derive a fact is considered as a form of negation (negation as failure, NAF). Also, when checking business process models, constructs for expressing logical disjunctions are required (e.g. “a process should contain either x or y ”) which SWRL does not provide. Therefore, we use the SPARQL query language to formulate semantic constraints.

The basic SPARQL syntax is relatively simple. After a SELECT keyword, the variables are listed which contain the return values (comparable to columns of a table). In the following WHERE clause, multiple graph patterns can be specified for describing the desired result by using so called *triple patterns* with subject s predicate p and object o separated by blanks and followed by a dot. On each position in the triple pattern, variables may be used which are prefixed with a question mark. Below we show the basic structure of such queries.

```
SELECT ?var1 ?var2 ?varN
WHERE{
    s1 p1 o1 . s2 p2 o2 . sN pN oN
}
```

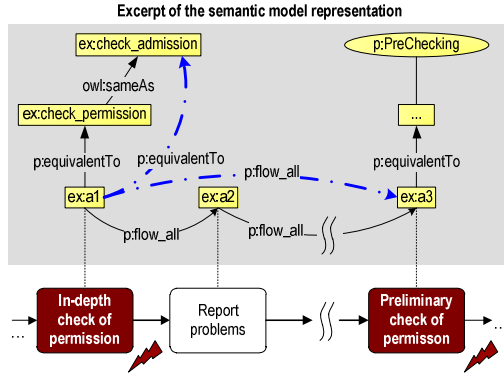
The subject and object of a triple pattern usually specify known individuals, classes or blank nodes, the predicate specifies a property between them that must be present in the ontology. Triple patterns are interpreted as a conjunction. However, they can be augmented with the constructs UNION for expressing logical disjunction and OPTIONAL for triple patterns that are not required to be satisfied. The upcoming 1.1 version of SPARQL will additionally support NOT EXISTS for detecting missing triples and specifying a cardinality for predicates, e.g. `node1 flow{1,5} node2` will match if 1 to 5 flow predicates are in between the subject and object of the triple pattern. For more information, see the SPARQL specification [18]. For query processing, we use the ARQ query processor (<http://jena.sourceforge.net/ARQ/>).

A simple example of checking the semantic correctness would be a constraint that after an in-depth check no preliminary check has to follow. This constraint originates from the examination of real-world processes in a public administration of a capital city in the north-west of Germany. Models violating this constraint in fact have been found in practice and were corrected by manually searching and browsing in the process models (we report on that in [8]). A corresponding SPARQL query for detecting the violation of this constraint is shown in Fig. 2 along with a visualization of an excerpt of the ontology-based model representation created as described in section 3.1.

The sample query does not only demonstrate the usage of SPARQL for constraint checking, but also the usage of terminological background knowledge provided by the ontology asserting that `ex:check_permission` is the same as `ex:check_admission`. Hence, the `p:equivalentTo` property between `ex:a1` and `ex:check_admission` can be inferred. Moreover, as `p:flow_all` is a transitive property, the triple `ex:a1 p:flow_all ex:a3` can be inferred. As `ex:a3` is annotated with an ontology instance that belongs to the class of `p:PreChecking` activities, there is a solution for the query hence indicating a constraint violation.

```

SELECT ?node1 ?node2
WHERE {
  ?node1 p:equivalentTo ex:check_admission .
  ?node1 p:flow_all ?node2 .
  ?node2 p:equivalentTo ?x .
  ?x rdf:type p:PreChecking .
}
    
```



Legend – addition to Fig. 1

- Elements causing the constraint violation
- Shaded model elements are part of the SPARQL query result set
- $\text{--}\{owl:oneOf\}\text{--}$ Class membership by enumeration
- $\text{--}\{owl:oneOf\}\text{--}$ Inferred property
- $\text{--}\{owl:oneOf\}\text{--}$ Omitted parts of the model/ontology

Fig. 2. Sample Element Flow Constraint

		Constraint Matter	
		Process	Resource
Constraint Focus	Structure	1 Element flow constraint	2 Resource usage constraint
	Occurrence	3 Element occurrence constraint	4 Resource occurrence constraint

Fig. 3. Characterization of Constraints

Basic types of constraints that can be specified with SPARQL can be characterized according to the matter of the constraint and its focus (cf. Fig. 3). The constraint matter specifies the subject of a constraint which is either the process, i.e. the set of nodes and arcs which constitute the core process graph, or the resources which are involved in the process and which appear as additional nodes and edges e.g. in the form of organizational units assigned to tasks. The constraint focus is either the structure of a process graph involving several nodes connected by edges or the occurrence of specific nodes anywhere in the process graph.

The constraint types portrayed are to be understood as basic types. In practical applications, any combination of the four types may be combined in a single

constraint. For example, if an organizational unit *government representative* is present anywhere in the process (resource occurrence constraint), then an additional sequence of activities such as *report results to head of administration* has to be performed (element flow constraint) involving at least one information system for archiving the results (resource usage constraint).

4 Evaluation

4.1 Evaluation Scenario

We have evaluated our approach by conducting an experiment with 21 participants. 14 of the participants were students studying either Information Systems (8), Business Studies (2) or Computer Science (4). The rest of the participants recently has graduated in Information Systems (4), Business Studies (1) or is doing a PhD-Thesis (2). We used models in three different sizes with 11, 39 and 217 model elements. The participants had to answer 10 questions per model (a) by browsing and searching in the model with the tool *Microsoft Visio 2003* (including CTRL+F as allowed search “tool”), or (b) by using *SBPMQuery*, a graphical interface for querying OWL-DL-ontologies with SPARQL which we have adapted for the experiment (see also [20]). In case (a), the model was organized in a Visio document containing one page for the smallest model and multiple linked pages for the medium and the big model with an additional overview page. In case (b), the models were represented in one OWL-DL file according to the ontology-based representation introduced in section 3.1.

To simplify querying for the participants, we have shifted the various flow relations between the nodes of the represented process graph to the level of the annotated domain instances. Therefore, a graph pattern to find the following activity after a start event such as `?x a StartEvent . ?x equivalentTo order_received . ?y a Activity . ?y equivalentTo ?z . ?x flow_all ?y` can be simplified to just one triple `order_received flow_all ?z`. This is not a serious limitation as the simplified representation can be generated automatically. If the ontology contains multiple models, the queries have to be executed against each simplified model which can be extracted automatically. Afterwards, the results can be aggregated to a single result. The only limitation of the described simplification is that it does not provide for constraints spanning different models, e.g. “find a function that is used in more than 5 models”. However, the user is free to run such model-spanning queries against the complete knowledge base by not using the described simplification.

4.2 Results of the Evaluation

In order to measure the effectiveness of SPARQL queries for the task of semantic correctness checking, we have selected effort and quality as basic criteria. We determined the effort by measuring the time required for answering the questions. We determined the quality of the answers by counting the correct, incorrect and missing answers. Moreover, we have compared the results obtained by using the query language with the results obtained by using manual browsing and search.

Fig. 4 shows the average amount of errors of all participants in checking the big model with 217 model elements for the constraint types described in section 3.2. As

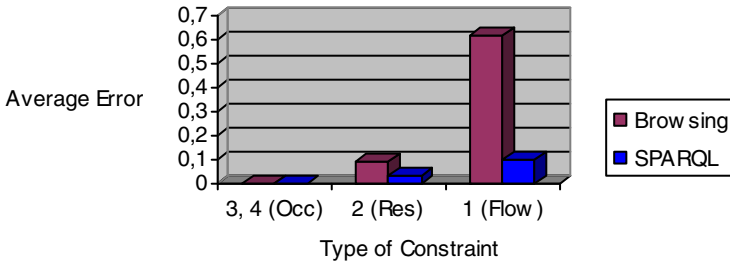


Fig. 4. Comparison of Error Frequency for Manual Browsing and Querying

participants made zero errors in checking element or resource occurrence constraints, we hence combined them to a single column *Occ*. It is obvious, that in general manual browsing generates more errors than working with a query language. Interestingly, checking element flow constraints (column *Flow*) produced the highest number of errors when using manual search. The amount of errors is significantly higher than the errors made in checking resource usage constraints (column *Res*). These results indicate that checking large models manually is an error-prone task and that SPARQL outperforms manual search regarding the accurateness of the results. The biggest advantage for the query language has been detected in regard to the analysis of the flow structure of a model.

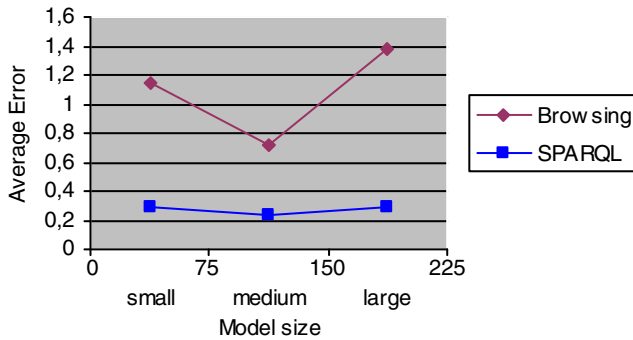


Fig. 5. Error Frequency for Arbitrary Constraints Dependent on Model Size

We also analyzed the amount of errors for checking arbitrary constraints composed of those described in section 3.2. As can be seen in Fig. 5, the error frequency for manual search is first decreasing. Above a model size of approx. 110 elements, it increases dramatically. In comparison to that, the error frequency for the query language remains relatively stable and is thus not dependent on model size. We

interpret the initial decrease of the error frequency for manual search as an artifact of the experiment as the participants became accustomed to the task of analyzing a model visually. The subsequent increase is due to the complexity and the cognitive effort required for analyzing large models manually. Regarding SPARQL, a slight initial decrease of the amount of errors might also be the consequence of getting familiar with querying, whereas an increased error frequency is likely to be the result of transferring the query results from the screen to paper.

Next, we analyzed the time required for answering the 10 questions for each of the three models in relation to the model size. Fig. 6 shows the result for the small, medium and big model. As can be seen easily, at first there is a learning curve when using SPARQL. When the model size exceeds approx. 190 model elements, the query language outperforms manual search.

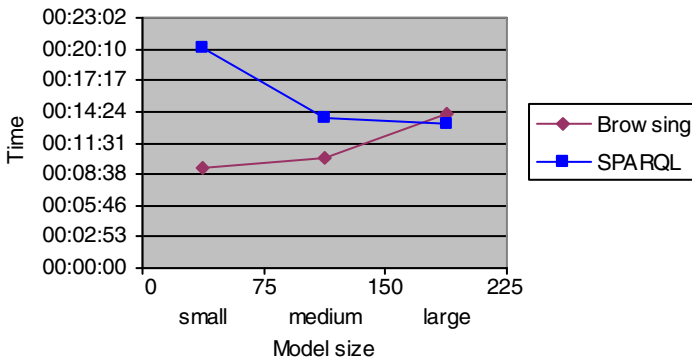


Fig. 6. Time Needed to Answer the Questions Dependent on Model Size

To complement our evaluation, we also conducted a short survey at the end of the experiment. The main results of this survey are that the participants feel confident with the accurateness and completeness of the results generated by the query language in contrast to the results found by manual search. 95% of the participants even stated that they would use SPARQL again if the tool at hand supported that language. To our surprise, participants with expertise in SQL did not produce better results than those without that knowledge, so we conclude that the usage of SPARQL can be learned in a short time (our introduction took approx. 30 minutes) and does not require any prior knowledge.

5 Conclusion

We have developed a semantic business process model representation which allows semantic correctness checking of business process models in regard to four basic constraint types: element flow, element occurrence, resource usage and resource occurrence. To check the correctness, background knowledge formalized in an ontology is used. This provides for more abstract constraints as inferred knowledge

from the ontology (e.g. subclass relations, transitive properties) is used to check whether the contents of a model are semantically correct or not. The required constraints can be specified as SPARQL queries. We have demonstrated the effectiveness of checking the ontology-based process representation with such queries by conducting a user experiment. The main result of the experiment is, that a query language such as SPARQL outperforms manual search in terms of the time needed and the accurateness of the results when the number of model elements exceeds 190.

We currently work on a prototype for user-friendly model annotation which will offer multiple ways of selecting ontology instances. We thereby also explore the issue of keeping the original model and its ontology-based representation synchronized.

References

1. Awad, A., Decker, G., Weske, M.: Efficient Compliance Checking Using BPMN-Q and Temporal Logic. In: Dumas, M., Reichert, M., Shan, M.-C. (eds.) BPM 2008. LNCS, vol. 5240, pp. 326–341. Springer, Heidelberg (2008)
2. Chapurlat, V., Braesch, C.: Verification, validation, qualification and certification of enterprise models: Statements and opportunities. *J. Computers in Industry* 59(7), 711–721 (2008)
3. Desel, J.: Model Validation - A Theoretical Issue? In: Esparza, J., Lakos, C.A. (eds.) ICATPN 2002. LNCS, vol. 2360, pp. 23–43. Springer, Heidelberg (2002)
4. Dijkman, R.M., Dumas, M., Ouyang, C.: Formal semantics and automated analysis of BPMN process models. Preprint Technical Report 5969, Queensland University (2007)
5. Drumm, C., Filipowska, A., Hoffmann, J., Kaczmarek, M., Kaczmarek, T., Kowalkiewicz, M., Markovic, I., Scicluna, J., Vanhatalo, J., Völzer, H., Weber, I., Wieloch, K., Zyskowski, D.: Dynamic Composition Reasoning Framework and Prototype. Project IST 026850 SUPER, Deliverable 3.2, SAP (2007)
6. El Kharbili, M., Stein, S.: Policy-Based Semantic Compliance Checking for Business Process Management. In: Loos, P., et al. (eds.) Proc. of MobIS 2008. CEUR Proceedings, vol. 420, pp. 165–177. RWTH Aachen (2008)
7. El Kharbili, M., Stein, S., Markovic, I., Pulvermüller, E.: Towards a Framework for Semantic Business Process Compliance Management. In: Sadiq, S., et al. (eds.) Proc. of GRCIS 2008, Montpellier, France. CEUR Proceedings, vol. 339, paper 1. RWTH Aachen (2008)
8. Fellmann, M., Hogebe, F., Nüttgens, M., Thomas, O.: An ontology-driven approach to support semantic verification in business process modeling. In: Esswein, W., Turowski, K., Jührisch, M. (eds.) Proc. of MoBIS 2010, Dresden, pp. 99–110. GI, Bonn (2010)
9. Fellmann, M., Hogebe, F., Nüttgens, M., Thomas, O.: How to ensure correct process models? A semantic approach to deal with resource problems. In: Abramowicz, W., et al. (eds.) Proc. of Informatik 2010, Leipzig, pp. 280–285. GI, Bonn (2010)
10. Fellmann, M., Hogebe, F., Thomas, O., Nüttgens, M.: What's inside the Box? Prospects and Limitations of Semantic Verification in Process Modelings. In: Klink, S., et al. (eds.) Proc. of EMISA, Karlsruhe, Germany. LNI, vol. 172, pp. 85–100. Bonn, Köllen (2010)
11. Goedertier, S., Vanthienen, J.: Designing compliant business processes with obligations and permissions. In: Eder, J., Dustdar, S. (eds.) BPM Workshops 2006. LNCS, vol. 4103, pp. 5–14. Springer, Heidelberg (2006)

12. Gruhn, V., Laue, R.: Checking Properties of Business Process Models with Logic Programming. In: Proc. of MSVVEIS 2007, held in conjunction with ICEIS 2007, Funchal, Madeira, Portugal, pp. 84–93. INSTICC Press (2007)
13. Horrocks, I., Patel-Schneider, P.F., Boley, H., Tabet, S., Grosz, B., Dean, M.: SWRL: A Semantic Web Rule Language: Combining OWL and RuleML. W3C Member Submission 21 May 2004. W3C (2004)
14. Ly, L.T., Rinderle-Ma, S., Dadam, P.: Integration and verification of semantic constraints in adaptive process management systems. *J. Data & Knowledge Engineering* 64(1), 3–23 (2008)
15. Ly, L.T., Rinderle-Ma, S., Göser, K., Dadam, P.: On enabling integrated process compliance with semantic constraints in process management systems. *J. Information Systems Frontiers*, 1–25 (2009)
16. Melenovsky, M.J.: Business Process Management’s Success Hinges on Business-Led Initiatives, Stamford, Gartner Research (2005)
17. Mendling, J.: Empirical Studies in Process Model Verification. In: Jensen, K., van der Aalst, W.M.P. (eds.) Transactions on Petri Nets and Other Models of Concurrency II. LNCS, vol. 5460, pp. 208–224. Springer, Heidelberg (2009)
18. Prud’hommeaux, E., Seaborne, A. (eds.): SPARQL Query Language for RDF: W3C Recommendation 15 January 2008. W3C (2008)
19. Speck, A., Pulvermüller, E., Heuzeroth, D.: Validation of business process models. In: Buschmann, F., Buchmann, A.P., Cilia, M.A. (eds.) ECOOP 2003. LNCS, vol. 3013 (2004)
20. Thomas, O., Fellmann, M.: Semantic Process Modeling – Design and Implementation of an Ontology-Based Representation of Business Processes. *J. Business & Information Systems Engineering* 1(6), 438–451 (2009)
21. van der Aalst, W., de Beer, H.T., van Dongen, B.F.: Process Mining and Verification of Properties: An Approach Based on Temporal Logic. In: Tari, Z. (ed.) OTM 2005. LNCS, vol. 3760, pp. 130–147. Springer, Heidelberg (2005)
22. van der Aalst, W.M.P.: Formalization and verification of event-driven process chains. *J. Inform. and Software Technology* 41(10), 639–650 (1999)
23. Weber, I., Hoffmann, J., Mendling, J.: Beyond soundness: on the verification of semantic business process models. *J. Distributed and Parallel Databases* 27, 271–343 (2010)
24. Weber, I.M.: Verification of Annotated Process Models. In: Weber, I.M. (ed.) Semantic Methods for Execution-level Business Process Modeling. Lecture Notes in Business Information Processing, vol. 40, pp. 97–148. Springer, Heidelberg (2009)
25. Wecker, G., van Laak, H. (eds.): Compliance in der Unternehmenspraxis: Grundlagen, Organisation und Umsetzung. Gabler, Wiesbaden (2008)

Guaranteeing Soundness of Adaptive Business Processes Using ABIS

Falko Koetter¹, Monika Weidmann², and Daniel Schleicher³

¹ University of Stuttgart IAT, Germany
falko.koetter@iao.fraunhofer.de

² Fraunhofer IAO, Germany
monika.weidmann@iao.fraunhofer.de

³ University of Stuttgart IAAS, Germany
schleicher@iaas.uni-stuttgart.de

Abstract. The Internet of Services necessitates ad-hoc collaboration of companies in business processes. Each collaboration requires specific adjustments of the underlying process. While adapting these variable processes in collaboration with multiple parties, a need for guaranteeing the soundness of business process variants arises. In this paper we extend the ABIS approach of adaptive business process modeling with soundness concepts, apply them in an interactive variant creation algorithm and implement this algorithm in a prototype.

Keywords: business process management, adaptive business processes, process variants, process configuration process modeling.

1 Introduction

The *Internet of Services* is used for buying and selling services [14], so companies can work together in ad-hoc collaborations. Therefore, a business process has to span each participant, thus providing a cross-company process flow [1]. As we observed in our work with insurance companies, these interorganizational business processes are variants of a single base process, which is adapted for each individual business relationship. Therefore, a need for standardization arises, while still providing some means of individualization.

To achieve this, we developed ABIS [20], an extension of the Business Process Modeling Notation BPMN 2.0 [1] for *mass customizing* of business processes. It allows creating standardized *process templates*, from which *process variants* may be derived incorporating *process fragments* from multiple parties. Further on we call the process of creating a *variant* from such a template *binding*. Single occurrences of variability are called *variability points*.

As these processes involve multiple parties [15], every change to the process may change the communication pattern between participants. Therefore, binding has to be coordinated between all involved parties. As this defines the specifics

¹ <http://bpmn.org/>

of the business interaction, a need to ensure soundness of the resulting variants arises. *Soundness* is achieved if any series of choices made during binding leads to a syntactically correct process variant, obeying the constraints set in the process template. In this paper we describe how ABIS enables to create process templates from which only sound process variants may be created. This is achieved by defining *dependencies* between variability points and alternatives, thus preventing inconsistent choices from being made. The main contribution of this paper is adapting an existing dependency concept [13] and using it to define an *interactive* binding algorithm. This algorithm is then implemented in a prototype using the web-based Oryx editor [2].

This paper is structured as follows. We first give a motivational example in Section 2. In Section 3 we examine related work. In Section 4 we explain how soundness in ABIS is achieved, including dependency concepts and the binding algorithm. Section 5 describes the ABIS prototype. Section 6 contains future work and a conclusion.

2 Motivational Example

To model variability in business processes ABIS introduces two new diagram types [20]: *Process templates*, from which process variants may be created and *process fragments*, which may be inserted into the process template at predetermined spots. During the binding process an *alternative* is selected for each variability point. If this alternative is applied, the variability point is said to be *bound*. Note that the finished variant is plain BPMN 2.0 and may be edited and executed like any other BPMN process.

There are two kinds of variability points in ABIS, which are marked with a *puzzle piece*. On the one hand any attribute of a BPMN element may be a *variable attribute* consisting of a number of alternatives, which may be defined using either an *explicit* value or a constraint for a user-defined value. On the other hand a *variable region* is a placeholder for a *process fragment*, which will be chosen as an alternative during binding and takes the variable region’s place in the sequence flow. A process fragment may in turn contain variability points itself, thus allowing for *recursion*.

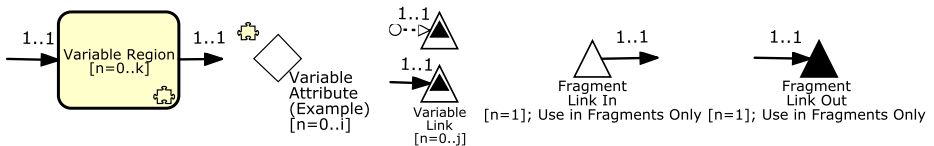


Fig. 1. ABIS modeling elements including the number of times they may be used (n) and cardinalities of connecting flows

² <http://oryx-editor.org>

To model these capabilities we introduced new modeling elements as shown in Figure 1. The *variable region* has to have exactly one incoming and outgoing sequence flow. These flows correspond to *fragment link in and out*, which serve to mark the beginning and the end of a process fragment. If a fragment replaces a variable region, the incoming and outgoing sequence flows of the region are connected to the elements neighboring fragment link in and out in the fragment. If an additional *sequence* or *message* flow from a fragment to the main diagram is needed, a *variable link* may be used, which is connected to a specified target during fragment insertion.

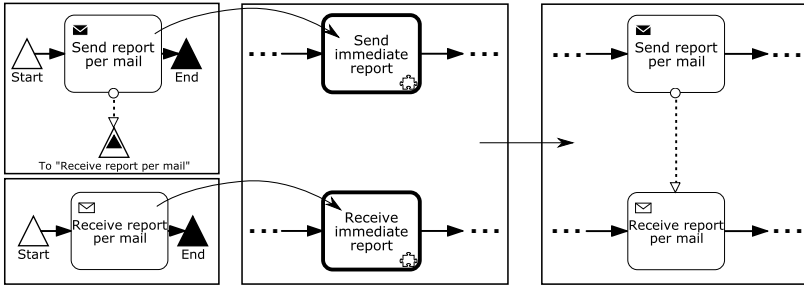


Fig. 2. Two fragments (left) are inserted into a process template (middle) to create a process variant (right)

Figure 2 illustrates the use of the new elements in an example. Consider the pictured part of a process template containing two *variable regions* handling reporting. If no immediate report is to be given, both may be bound with an empty fragment (not pictured here) and essentially left out of the variant. In our example reports are to be given per mail. Thus, participants may separately design *process fragments* handling their part of the reporting. To the left of Figure 2 two simplified sample fragments are pictured. A *variable link* is used to extend the message flow beyond the boundaries of the upper fragment. To the right the result of binding is displayed.

Consider now binding one variable region to the *empty fragment* and the other to the custom fragment. This results in an *invalid* process variant, as either the variable link cannot be resolved or an immediate report is expected which never arrives. Therefore, immediate reports have to be handled by both participants or not at all. They cannot be sent without being received and they cannot be received without being sent beforehand. We need to guarantee this in order to create a sound process variant.

As we see in this example, there is a need for *dependencies* between variability points. Either both or none of the two fragments have to be inserted. Also, the order of insertion is important. The lower fragment has to be inserted first to provide the *target* of the variable link in the upper fragment.

3 Related Work

In this section we present related work in the area of variable business processes. Generally, there are two ways to resolve variability in business processes: At *runtime* [11] or at *design time*. We only consider the latter, as in comparison to ad-hoc processes, which allow changes to a running process instance [4], ABIS only introduces variability at design time and treats the *process variant* as part of the agreement between the involved parties. In previous work [20] we compared concepts for variability modeling. For the purposes of this paper, we will focus on their implementation, dependency concepts, and binding.

The first concept *Provop* allows variability modeling by using so-called *options* [7]. These options represent single operations like insertion, deletion and modification. Applying options to create a variant is not commutative and may lead to contradictions [6]. This may be alleviated by specifying *dependencies* like implication or mutual exclusion, which are modeled in a separate diagram. Soundness may be guaranteed by checking each bindable variant [8]. The second concept *PESOA* uses UML-like stereotypes to add variability capabilities to BPMN [19]. Variability points are grouped in features, which may be added or left out as a whole. Contrary to ABIS all alternatives are modeled within the main diagram, thus not allowing for separate fragment creation. The third concept *process configurator* uses tree-based logic to allow selecting a set of features, which is then broken down to modifications of a model [21]. The model itself is plain BPMN, augmented by annotations. While this allows to use existing modeling tools, it also makes variable parts harder to distinguish from regular process flow which is one reason why ABIS extends BPMN. To enable modeling *inter-organizational* process templates, we combined qualities of these existing approaches, i.e. ease of use, separate alternative modeling and dependency modeling. Prototypes exist for all three concepts, allowing binding in a single transformation step [5, 21, 17]. In comparison, our approach focuses on guiding the user through the binding one choice at a time, immediately showing the consequences each choice has.

Previous work on dependencies has also been done in the field of *software product line engineering* concerning mass customization of software. In [9] a taxonomy of dependencies is given, distinguishing dependencies between variability points, between an alternative and a variability point and between alternatives. [16] further introduces two kinds of dependencies, *requires* and *excludes*. *Configurable Event Process Chains* (C-EPCs) introduce dependencies as *requirements*, a set of logical expressions constraining choices and included elements in the finished EPC [18]. These are displayed alongside the diagram, indicating which parts of the process are affected by them. In [12] a *questionnaire-based* approach to binding C-EPCs is presented, allowing binding by answering a questionnaire. For this, additional *dependencies* are introduced, defining an order between decisions. In [2] another taxonomy for model dependencies is introduced. Here, there are *property* and *existence* dependencies. The former concerns the value of a property, while the latter concerns the existence of a construct. Dependencies are further classified as being *symmetric* or *asymmetric*. While symmetric

dependencies are mutual, i.e. each element affects the other, asymmetric dependencies represent a one-directional dependency, i.e. a logical implication. As we will see, our concepts support all of these dependency types.

4 Soundness in ABIS

To enable sound process variants, three steps are to be taken. First, ABIS must be extended to allow defining the *order of binding*. Second, it must be possible to *restrict alternatives* based on the choices made before and the current state of the diagram. Finally, the *binding algorithm* needs to enforce these two concepts, thus creating only sound process variants. The remainder of the section will describe these three steps in detail. In order to take the first two steps, we base the dependency concept of ABIS on the use of dependencies in *variability descriptors* [13], as they are used for exporting a process template. Similar to C-EPs [18, 12], in [13] two kinds of dependencies are distinguished: *Dependencies* are defined between *variability points* and determine *when* choices can be made. *Enabling conditions* are defined for *alternatives* and determine *what* choices can be made.

Dependencies. A dependency defines a source-target relation between two variability points. The target *depends* on the source. The target may only be bound when the source has already been bound, e.g. the condition for an audit may only be defined *after* the kind of audit has been chosen.

Enabling Conditions. An enabling condition defines a condition which has to be true in order for a particular alternative to be chosen. Each alternative may have an enabling condition. As enabling conditions are essentially Boolean expressions, a formal language is needed. Variability descriptors [13] use an extension of XPATH for enabling conditions. To enable *existence* and *property* dependencies [2], we further extend XPATH with the following ABIS specific functions: `selectedValue(variabilityPoint)` returns the selected value of the specified variability point. This is either the value of a *variable attribute* or the name of a *process fragment*. `exists(elementName)` returns true if a BPMN element named *elementName* exists. `elementAttribute(elementName, attributeName)` returns the attribute named *attributeName* of the element named *elementName*.

In our example (see Section 2), enabling conditions are used to guarantee immediate reports are handled correctly. As previously described the choice in *Send immediate report* can be deduced from the choice in *Receive immediate report*. To model this, each alternative of *Send immediate report* has an enabling condition. The *upper fragment* may only be chosen if the lower fragment has been chosen at *Receive immediate report* (`selectedValue("Receive immediate report") = "Lower fragment"`). The *empty fragment* may only be chosen if the empty fragment has been chosen at *Receive immediate report* as well (`selectedValue("Receive immediate report") = "Empty fragment"`). Considering the two alternatives at *Receive immediate report*, these two enabling conditions are mutually exclusive. Only one of the alternatives will be enabled for any given

choice at *Variable Region 1*. In fact, no choice by the user is needed at *Send immediate report* if the previous choice has been made.

Binding. ABIS supports the creation of a process variant by successively binding each variability point and applying the choices to the process template, guiding the user through the binding process, making one choice at a time and immediately seeing the results.

During binding dependencies are needed to determine which variability points may be bound. Due to the binary relation defined by the dependencies, variability points and dependencies form a *directed graph*. This *dependency graph* G is a tuple of variability points V and dependencies E , defined as follows:

$$G = (V, E)$$

$$V = \{vp | vp \text{ is a variability point}\}$$

$$E = \{(s, t) | \exists \text{ Dependency with source=s and target=t}\}$$

A variability point can only be bound if all its dependencies have been satisfied. Considering the dependency graph this condition is met if the variability point has no incoming edges from other points which have not been bound. If the graph is acyclical, it imposes a partial order on all variability points. Using topological ordering, such a partial order may be extended to a total order. Therefore, we will adapt an algorithm for topological ordering [10] to create our binding algorithm.

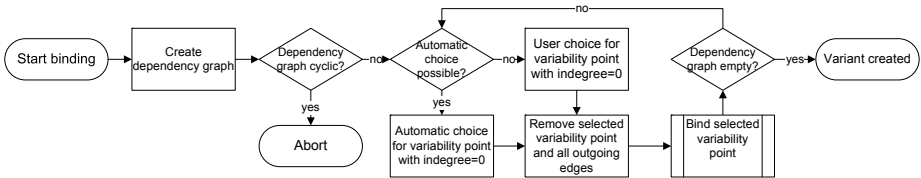


Fig. 3. Basic binding algorithm

Figure 3 shows the **basic binding algorithm**. Whenever a variability point is bound, it and all its outgoing nodes are removed from the dependency graph. Therefore, all variability points which depend on it lose this dependency. If the graph is acyclical, all variability points will eventually lose their dependencies and may be bound. This results in a *variant*. We will examine the single steps more closely in the next paragraphs.

The first step is to *create the dependency graph*. This is achieved by creating a node for each variability point and an edge for each dependency. Next we check if the *dependency graph is acyclic*. If there is a cycle in the graph, there is no topological ordering for it so there is no valid binding order. Therefore, binding cannot be performed and the algorithm is *aborted*. If the graph is acyclical, the main binding loop starts. First, the next variability point for binding has to be determined.

To do this, ABIS checks if an *automatic choice is possible*, by checking all available variability points for the number of enabled alternatives. If this number is

exactly one, the only available alternative is *chosen automatically*. This may be used by the modeler to propagate choices throughout the diagram, as seen in Figure 2. If no automatic choice can be made, an available variability point and an enabled alternative is *chosen by the user*. After a choice is made, the variability point in question is *removed from the dependency graph* including all of its outgoing edges. Given a variability point v_{bound} , removal is performed as follows:

$$V := V \setminus \{v_{bound}\}$$

$$\forall (s, t) \in E : s = v_{bound} \rightarrow E := E \setminus \{(s, t)\}$$

Then, the variability point in question is bound. We will look at the *binding subprocess* in more detail in the next paragraph. If the *dependency graph is now empty*, a *variant has been created* and binding is finished. If there is still at least one variability point left, the main binding loop starts again checking if now an *automatic choice is possible*.

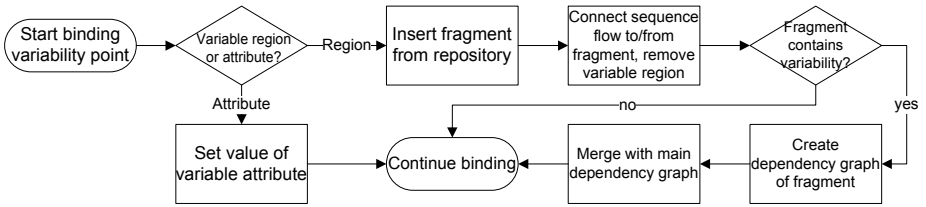


Fig. 4. Subprocess for binding a variability point

Figure 4 shows the **subprocess for binding a variability point**. It is called from the main algorithm (see *Bind selected variability point* in Figure 3). Two cases need to be distinguished: In case of a *variable attribute*, binding is straightforward. The *value of the variable attribute is set* to the specified alternative. In case of a variable region, multiple steps are necessary to bind it. First the selected fragment is *fetched from the repository and inserted* into the main template. Then, it is *connected to the sequence flow* as seen in Figure 2, thereby replacing the variable region.

Now, the possibility of recursive variability has to be taken into consideration. If a process fragment contains no variability points itself, the main binding algorithm can *continue*. If a process fragment itself is variable, its variability needs to be bound before binding of the main template can continue, because the dependencies rely on the *variable region* to be fully bound. Conceptually, a variant of the fragment has to be created in a separate binding process. Only after this *fragment variant* is created, it can be inserted into the template. To allow the user binding the fragment in its final context, binding a variable fragment is done *after* it has been inserted into the main process template. Therefore, we have to bind the variability points of the fragment before regular binding continues. We can emulate the separate binding of the fragment using additional dependencies without modifying the main algorithm.

To achieve this, first the *dependency graph of the fragment is created*. As we need to observe both the existing dependencies of the fragment and of the main template, we have to *merge the newly created dependency graph with the main dependency graph*, ensuring correct binding order. Given G_f , the graph of the fragment and G_m , the main dependency graph, merging is performed as follows:

$$\begin{aligned} G_m &= (V_m, E_m), G_f = (V_f, E_f) \\ E_{s,t} &= \{(s, t) | s \in V_f, t \in V_m\} \\ E_m &:= E_m \cup E_f \cup E_{s,t} \\ V_m &:= V_m \cup V_f \end{aligned}$$

A dependency is added from each variability point of the fragment to each variability point of the main template, thus ensuring each variability point of the main template can only be bound after each variability point of the fragment has been bound.

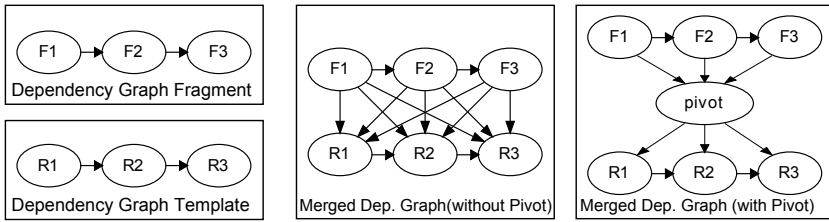


Fig. 5. Merging two dependency graphs with and without pivot element

Figure 5 shows an example of merging two dependency graphs. The variability points of the fragment have to be bound in the order $F1, F2, F3$, the points of the process template in the order $R1, R2, R3$. In the center of Figure 5, the merge is shown, enforcing the correct binding order $F1, F2, F3, R1, R2, R3$. For this merger 9 dependencies had to be added. In general, if the graphs G_m and G_f contain n and m variability points respectively, $n * m$ dependencies need to be added. Considering multiple levels of recursion, the amount of dependencies increases exponentially.

To alleviate this, we use the transitivity of dependencies. If b depends on a and c depends on b , a has to be bound before c . We utilize this by introducing an additional node to the dependency graph. This *pivot node* serves as an intermediary between the two merged graphs. As seen on the right of Figure 5, the pivot point depends on each point of the fragment. Each variability point of the main process template in turn depends on the pivot point. Using the *pivot point* v_{pivot} merging is performed as follows:

$$\begin{aligned} G_m &= (V_m, E_m), G_f = (V_f, E_f) \\ E_{s,pivot} &= \{(s, v_{pivot}) | s \in V_f\} \\ E_{pivot,t} &= \{(v_{pivot}, t) | t \in V_m\} \\ E_m &:= E_m \cup E_f \cup E_{s,pivot} \cup E_{pivot,t} \\ V_m &:= V_m \cup V_f \cup \{v_{pivot}\} \end{aligned}$$

As v_{pivot} depends on each given v_f of the fragment and each given v_m of the main template depends on v_{pivot} , each v_f has to be bound before v_m . In comparison to the previous approach, this only introduces $n + m$ new dependencies.

After both dependency graphs are merged, the *subprocess* (see Figure 4) is finished and the main binding algorithm (see Figure 3) can continue. Using the algorithm, all *dependencies* as well as *enabling conditions* are enforced, thus enabling *sound* process variants.

5 Prototype and Evaluation

To evaluate the ABIS concept, we created a **prototype** which supports the modeling of the new diagram types as well as binding a *process variant* using the algorithm described above. The prototype is based on Oryx, a web-based tool for collaborative modeling [3]. Its architecture is plugin-based, where extensions can be made on client and server side. Figure 6 shows the architecture of the ABIS prototype integrated within Oryx. The ABIS extensions are highlighted in grey color and will be explained in the following paragraphs.

Modeling languages are implemented using *stencil sets*, containing a number of shapes and rules for connecting them. An existing *stencil set* may be extended using a *stencil set extension*, adding additional shapes and rules. Two such extensions have been created for the new diagram types.

To ensure compatibility with the existing *BPMN 2.0* implementation, instead of explicitly defining attribute variability using element properties, any existing attribute can be made variable using a variability description language. Using this language, variability is serialized and stored in the existing attributes, thus eliminating the need for additional properties. Variable attributes can be defined using the *Variability Wizard* plugin. This plugin allows defining a variable attribute by creating alternatives and selecting dependencies from a checklist of all variability points.

The binding algorithm (see Section 4) is performed by the *Binding Plugin*. To achieve this, the editor is put in a read-only binding mode, during which all *variability points* are highlighted. The user may select one of the elements and

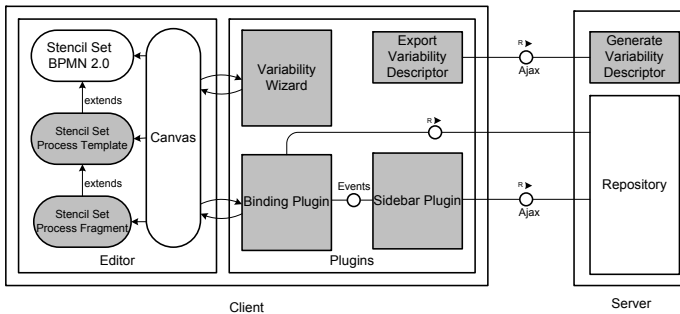


Fig. 6. Architecture of the ABIS prototype. (based on [3])

choose one of the available alternatives from a sidebar using the *Sidebar Plugin*. Process fragments are fetched from the Oryx *repository*. Figure 7 shows Oryx in binding mode. Available variability points are indicated by horizontal lines, unavailable points by vertical lines. Note the ABIS shapes in the toolbar to the left. To the right, the *sidebar* can be seen, offering a choice of fragments for the selected variable region.

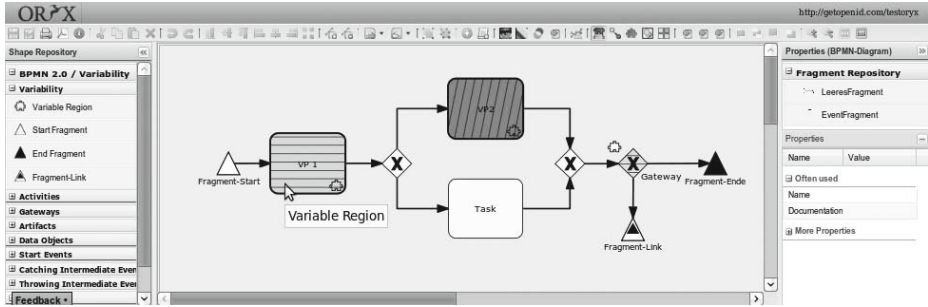


Fig. 7. ABIS implementation in Oryx during binding

For **evaluation** of the prototype we modeled two variable processes, one process constructed to cover as many different variability scenarios as possible (a customizable order process for an *online shop*) and one real life example as part of the openXchange project (a *property claims management* process). For this we identified 18 business processes, from which 9 templates and 31 fragments were generated. The maximal depth of recursion used was 3. On average, one fragment could be reused per process template. Considering dependencies we identified 102 valid process variants in total. After defining dependencies and enabling conditions, it was impossible to create an invalid process variant. Our evaluation showed our approach was not only sufficient to model all necessary variability, but also enforced soundness during binding. While binding was straightforward, defining enabling conditions and dependencies necessitated some debugging, as they had to be written in XPATH. The capability to define them visually would further enhance the understandability of our approach.

Additionally, we performed a stress test recursively inserting process fragments up to a recursion depth of 100. While editor response times deteriorated, this was mostly due to the larger amount of shapes to be drawn by the Oryx editor.

6 Future Work and Conclusion

We plan to further evaluate our prototype modeling more real world processes in cooperation with business users. Also, we want to extend the collaborative capabilities of the extension, allowing distributed binding by synchronizing multiple editor instances. In future work we may have to look into ways to support not only the *business user* during binding, but also the *process modeler* by evaluating soundness during template creation. During binding we observed

that not all process details need to be visible for each stakeholder. Therefore, we plan utilizing ABIS to allow modeling of business processes in *abstraction levels*.

In this paper we extended the ABIS approach for modeling adaptive business processes with dependency concepts, thus allowing the modeler to guarantee sound process variants. Further on, we defined a binding algorithm for the creation of these variants and implemented ABIS using the Oryx editor. The ABIS prototype provides the means to model ABIS diagram types and allows for interactive binding to create process variants. We validated our concepts using the prototype, showing our concept enables the user to model sound variable processes.

Acknowledgments. The work published in this article was partially funded by the openXchange project of the German Federal Ministry of Economy and Technology under the promotional reference 01MQ09011. The author D. Schleicher would like to thank Ericsson for financial support.

References

1. Barros, A.P., Dumas, M.: The Rise of Web Service Ecosystems. *IT Professional* 8(5), 31–37 (2006)
2. Bodenstaff, L., Wombacher, A., Reichert, M., Wieringa, R.: MaDe4IC: An Abstract Method for Managing Model Dependencies in Inter-Organizational Cooperations. *Service Oriented Computing and Applications* 4, 203–228 (2010)
3. Decker, G., Overdick, H., Weske, M.: Oryx – An Open Modeling Platform for the BPM Community. In: Dumas, M., Reichert, M., Shan, M.-C. (eds.) *BPM 2008*. LNCS, vol. 5240, pp. 382–385. Springer, Heidelberg (2008)
4. Dorn, C., Burkhart, T., Werth, D., Dustdar, S.: Self-adjusting recommendations for people-driven ad-hoc processes. In: Hull, R., Mendling, J., Tai, S. (eds.) *BPM 2010*. LNCS, vol. 6336, pp. 327–342. Springer, Heidelberg (2010)
5. Giese, C., Schnieders, A., Weiland, J.: A Practical Approach for Process Family Engineering of Embedded Control Software. In: *Proceedings of the 14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems*, pp. 229–240. IEEE Computer Society, Washington, DC, USA (2007)
6. Hallerbach, A., Bauer, T., Reichert, M.: Issues in Modeling Process Variants with Provop. In: Ardagna, D., Mecella, M., Yang, J. (eds.) *Business Process Management Workshops*. *Lecture Notes in Business Information Processing*, vol. 17, pp. 56–67. Springer, Heidelberg (2009)
7. Hallerbach, A., Bauer, T., Reichert, M.: Modellierung Und Darstellung von Prozessvarianten in Provop. In: *Modellierung 2008 Conference*. LNI, vol. 127, pp. 41–56. GI (2008)
8. Hallerbach, A., Bauer, T., Reichert, M.: Guaranteeing soundness of configurable process variants in provop. In: *Proceedings of the 2009 IEEE Conference on Commerce and Enterprise Computing*, pp. 98–105. IEEE Computer Society, Washington, DC, USA (2009)
9. Jaring, M., Bosch, J.: A Taxonomy and Hierarchy of Variability Dependencies in Software Product Family Engineering. In: *Proceedings of the 28th Annual International Computer Software and Applications Conference*, Washington, DC, USA, vol. 01, pp. 356–361. IEEE Computer Society, Washington, DC, USA (2004)

10. Kahn, A.B.: Topological Sorting of Large Networks. *Commun. ACM* 5, 558–562 (1962)
11. Koning, M., Sun, C.-a., Sinnema, M., Avgeriou, P.: VxBPEL: Supporting Variability for Web Services in BPEL. *Inf. Softw. Technol.* 51(2), 258–269 (2009)
12. La Rosa, M., Lux, J., Seidel, S., Dumas, M., Hofstede, A.H.M.T.: Questionnaire-Driven Configuration of Reference Process Models. In: *Proceedings of the 19th International Conference on Advanced Information Systems Engineering*, pp. 424–438. Springer, Heidelberg (2007)
13. Mietzner, R.: A Method and Implementation to Define and Provision Variable Composite Applications, and Its Usage in Cloud Computing. Dissertation, Universität Stuttgart, Fakultät Informatik, Elektrotechnik und Informationstechnik, Germany (August 2010)
14. Oberle, D., Bhatti, N., Brockmans, S., Niemann, M., Janiesch, C.: Countering Service Information Challenges in the Internet of Services. *Business & Information Systems Engineering* 1(5), 370–390 (2009)
15. Patig, S., Casanova-Brito, V., Vögeli, B.: IT requirements of business process management in practice – an empirical study. In: Hull, R., Mendling, J., Tai, S. (eds.) *BPM 2010. LNCS*, vol. 6336, pp. 13–28. Springer, Heidelberg (2010)
16. Pohl, K., Böckle, G., Linden, F.J.v.d.: *Software Product Line Engineering: Foundations, Principles and Techniques*. Springer-Verlag New York, Inc., Secaucus (2005)
17. Reichert, M., Rechtenbach, S., Hallerbach, A., Bauer, T.: Extending a Business Process Modeling Tool with Process Configuration Facilities: The Provop Demonstrator. In: *CEUR Proceedings of the BPM 2009 Demonstration Track, Business Process Management Conference 2009 (BPM 2009)*, Ulm, Germany, vol. 489 (2009)
18. Rosemann, M., van der Aalst, W.: A configurable reference modelling language. *Information Systems* 32(1), 1–23 (2007)
19. Schnieders, A., Puhlmann, F.: Variability Mechanisms in E-Business Process Families. In: *9th International Conference on Business Information Systems (BIS 2006)*, pp. 583–601. Springer, Heidelberg (2006)
20. Weidmann, M., Koetter, F., Kintz, M., Schleicher, D., Mietzner, R.: Adaptive Business Process Modeling in the Internet of Services (ABIS). To appear in *Internet and Web Applications and Services (ICIW)* (2011)
21. Werner, A., Müller, H.: Geschäftsprozesskonfigurator - Softwaregestützte Geschäftsprozessberatung. *ERP Management* 5(2), 25–28 (2009)

Merging Business Process Variants^{*}

Wassim Derguech and Sami Bhiri

National University of Ireland, Galway
Digital Enterprise Research Institute
{firstname.lastname}@deri.org
www.deri.org

Abstract. We propose in this paper a merging algorithm for integrating a set of process variants into a single configurable process model. This integrated process model should (i) subsume the behaviours of all original models, (ii) ensure a trace back of the origin of each element and (iii) derive any of the input models by means of configuration and individualization. Existing solutions either fail in respecting all these requirements or allow for merging only pairs of process models. However, our algorithm allows for merging a set of process models at once.

Keywords: business process modeling, reuse, merging, configuration.

1 Introduction

Reference process models describe proven practices for a specific industry. They are often aligned with emerging industry-specific and cross-industry standards. We refer the reader to [12] for an overview. One of the scenarios of reference process modelling is the reference process model customization [7]. It begins with a reference process model that provides configuration facilities. This model can be configured to specific needs of an enterprise e.g., by refining business rules or enabling/disabling some activities. In this scenario, there is an increasingly important need to assist business analysts in creating the reference model (i.e., the configurable model).

Configurable process models are constructed via the aggregation of several variants of a process model [12]. Such models are considered for example when companies become subject of acquisitions and mergers, in case of improvement of existing business processes or simply when different business analysts define their customized process models for achieving the same business goal. We call these business process models *business process variants*. Since they achieve in essence the same business goal, these variants slightly differ from each other in their structure [10]. Therefore, managing these variants can be made easier by handling the common parts just once and not for each variant separately.

Manual creation of configurable process models is tedious, time-consuming and error-prone task. Gottschalk et al. [9] mention that it took a team of five

^{*} This work is funded by the Lion II project supported by Science Foundation Ireland under grant number SFI/08/CE1/I1380 Lion-2.

analysts and 600 man-hour to merge 25% of an end-to-end process model. In this paper, we are proposing an automatic merging method that allows for creating a configurable business process model from a collection of process variants. We have defined a set of three requirements that our algorithm should respect:

1. The merged model should allow for the behaviour of all the original models. Traditionally, this operation is manually made by business analysts which comes with the risk that some aspects of the original models are accidentally neglected [5]. With the automation support for merging process variants, this risk can be minimized considerably.
2. Each element of the merged process model should be easily traced back to its original model [9]. A business analyst needs to understand what do the process variants share, what are their differences, etc. This can be made possible if he can trace back from which variant does an element originate.
3. Business analysts should be able to derive one of the input models from the merged process model [9]. Indeed, merging business process variants does not necessarily lead to the creation of a configurable process model. This requirement asks that the resulting merged model should provide configuration facilities in order to customize it and derive one of the input models.

The contribution of the paper is an algorithm that allows for merging a collection of business process variants given as input and delivers a configurable business process model. The resulting model should subsume the behaviours of all input models and these original models can be derived by means of configuration and individualization. For more information about the configuration and individualization algorithms, we refer to [4,8,12].

Several methods have been proposed to merge business process variants such as [5,6,9], their main weakness resides in the fact that they allow for merging only pairs of process variants. In order to merge a collection of process variants, they should merge a first pair then they add one variant at a time to the configurable process model. In contrast to existing proposals, our algorithm allows for merging all original process variants at once.

The remainder of the paper is organized as follows: Section 2 introduces our running example and the adopted notation which will be used in the rest of the paper. Section 3 presents the merging algorithm. Section 4 discusses the related work and Section 5 concludes the paper and states our future work.

2 Prerequisites

In this section, we introduce our running example which will be used to explain the steps of the proposed merging algorithm. Then, we present our notation which will be used for defining the merging algorithm.

2.1 Running Example

Several modelling languages have been proposed to represent business process models (e.g., Event-driven Process Chain (EPC), UML Activity Diagrams and

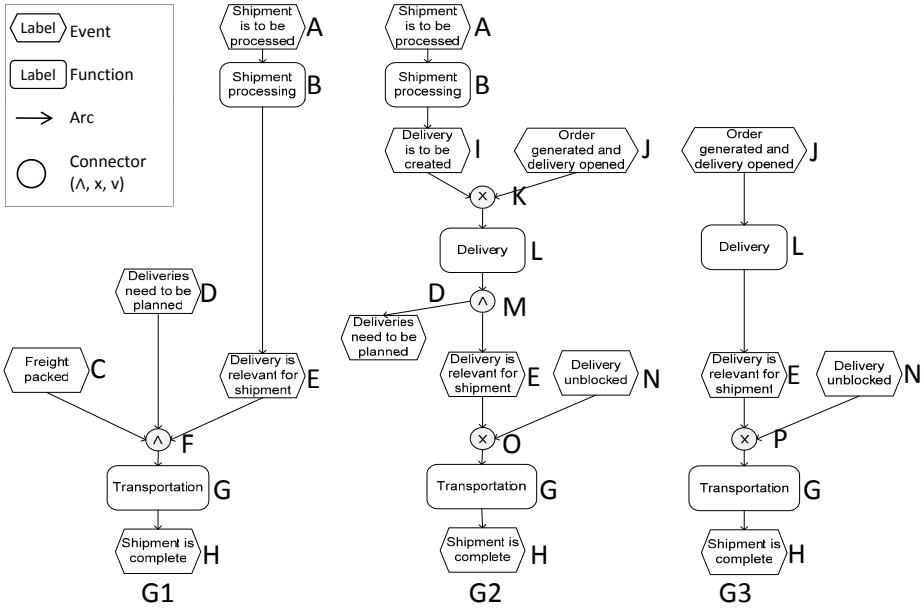


Fig. 1. Three business process variants (adapted from [9])

Business Process Model and Notation (BPMN)). In this paper we will adopt EPC [11] to illustrate our running example as well as for showing results of the steps of the merging algorithm.

Our running example depicted in Fig. 1 presents three process variants that follow the EPC notation (G1, G2 and G3). These process models describe a shipment process that involve three functions: “*Shipmentprocessing(B)*”, “*Delivery(L)*” and “*Transportation(G)*” (for presentation purposes, each node will be referred by a letter (i.e., A,B,...P) as marked in Fig. 1.

- Function “*B*” appears in G1 and G2 and in both variants it is triggered with the same event (i.e., “*A*”).
- Function “*L*” appears in G2 and G3. In G3 it is triggered with the event “*J*”, however, in G2 it is triggered either by the event “*J*” or “*I*”.
- Function “*G*” appears in all these variants. In G2 and G3 it is triggered either by the event “*E*” or “*N*”, however, in G1 it is triggered by all the events “*C*”, “*D*” and “*E*”.

The work presented in this paper consists of merging the three process variants of the Fig. 1 in order to generate a configurable process model. The outcome of our merging algorithm is depicted in Fig. 3b.

Next, we will present our notation for describing business process models. This notation will be used for presenting our merging algorithm.

2.2 Notation

Even though we will use EPC for illustrating the results of our merging algorithm, our proposed merger is not exclusively made for this modeling notation. It is still possible to adapt some steps of the algorithm to allow its applicability for other modeling notations. In our work, we represent a business process as a directed graph with annotated nodes and edges. A *business process graph* G is a triple $\langle WN, RN, E \rangle$ where:

- WN is the set of work nodes: in EPC, work nodes are either function or event nodes. Function nodes are used for representing tasks of the business process. Event nodes are used for presenting preconditions that need to be satisfied to perform a particular function. A work node n of the graph G is a triple $(id_G(n), \lambda_G(n), \tau_G(n))$ where:
 - $id_G(n)$ is a string that represents a unique identifier.
 - $\lambda_G(n)$ is a string that represents the label of the node.
 - $\tau_G(n)$ represents the type of the node (i.e., event or function in EPC).
- RN is the set of routing nodes: in EPC, connectors are the routing nodes. They are used for controlling the flow of tasks that need to be performed for achieving the goal of the business process. A routing node is a tuple $(id_G(n), \lambda_G(n), \tau_G(n), \eta_G(n))$ where:
 - $id_G(n)$ is a string that represents a unique identifier.
 - $\lambda_G(n)$ is a string that indicates if the connector is a join or a split connector.
 - $\tau_G(n)$ is a pair $(PID, Operator)$ where PID represents the process identifier and $Operator$ is its corresponding operator (i.e., \wedge , XOR , \vee). We deliberately mention the process identifier as part of $\tau_G(n)$ because configurable process models are also represented using this notation. Besides, a connector in a configurable process model can have multiple operators, having in mind that each operator can originate from a different process model. This notation will allow for keeping track where a connector originate from. In case of multiple variants of a configurable connector, $\tau_G(n)$ contains a set of pairs $(PID, Operator)$.
 - $\eta_G(n)$ is a flag that indicates whether this node is configurable or not, this applies when G represents a configurable process model.
- E is the set of directed edges which connect nodes from WN and RN . Entries of E are as follows: $(Source, PID, Destination)$ where:
 - $Source \in WN \cup RN$. It represents the source node of the edge.
 - $Destination \in WN \cup RN$. It represents the destination node of the edge.
 - PID represents the set of process identifiers where this edge originates from.

We use $\bullet n = \{m/(m, PID, n) \in G\}$ to denote the set of input nodes of n . We use $n\bullet = \{m/(n, PID, m) \in G\}$ to denote the set of output nodes of n .

3 Merging Business Process Models

For merging business process models, we start by representing them according to our notation. Before merging $G1$, $G2$ and $G3$ of the Fig. [11](#), we start by a

pre-processing step (see Section 3.1). Then we merge and post-process these business process graphs (see Section 3.2). Finally, a reduction step (see Section 3.3) is recommended to ensure that the final result is a reduced configurable process model.

3.1 Pre-processing Business Process Graphs: Resolving Conflicts of Start and End Nodes

The issue that we consider here concerns preserving start and end nodes of any process model. In other words, if a node is a start node in a particular model it should remain as a start node in the merged model. This property should be preserved for end nodes as well.

We pre-process all the business process graphs that we need to merge before proceeding to the merging step.

The solution that we propose consists of checking all the start/end nodes of all the models that we need to merge and verify if they are start nodes everywhere (i.e., in all the models). If a node appears as a start/end node in some models and not a start/end node in others, then it is considered as a conflict and has to be resolved. The resolution is as simple as creating a new identifier and assigning it to the conflicting nodes.

Let's go back to our running example. We can notice that the work node "*Deliveries need to be planned (D)*" is a start node in the first process graph (i.e., G1 of Fig. 1) and it is not a start node in the second variant (i.e., G2 of Fig. 1). To resolve this conflict, we operate as follows: we select all nodes which are start nodes in each of the input models (i.e., $\bigcup_{i=1}^k \{n_{G_i} / |\bullet n_{G_i}| = 0\}$), then select all nodes which are not start nodes in each of the input models (i.e., $\bigcup_{i=1}^k \{n_{G_i} / |\bullet n_{G_i}| \geq 1\}$) and finally determine their intersection. The result of this intersection is the set of nodes that need to be resolved. In our example the only node that need to be resolved is "*Deliveries need to be planned (D)*". The resolution of the conflict related to this node consists of generating a new identifier for the node "*D*" that is "*DStart*" and assign this identifier to nodes which appear as start nodes in all the models, in our case only in the first variant.

We proceed similarly to resolve the possible conflicts related to end nodes, to ensure that each end node in a model is an end node in the merged model.

Once the conflicts related to start and end nodes are resolved, we move to the next steps that consist of merging the process graphs and post-processing them which is the aim of the following sections.

3.2 Merging and Post-processing Business Process Graphs

The merging phase consists of a simple set union operation between all the process graphs. The merged process graph (MG) = $\langle \bigcup_{i=1}^k WN_i, \bigcup_{i=1}^k RN_i, \bigcup_{i=1}^k E_i \rangle$ where k represents the number of the initial business process graphs.

After merging the business process graphs, we expect obviously that some edges having the same source and destination will appear in the set of edges E .

For example, in our running use case, we have in G1 of Fig. 1 an edge between “*Shipment is to be processed (A)*” and “*Shipment processing (B)*” presented by $(A,(1),B)$ in the set of edges E_{MG} . The same edge appear in the second variant (i.e., G2 of Fig. 1) which is presented by $(A,(2),B)$ in the set of edges E_{MG} . For this, we need to merge these two edges into a single edge having as PID a set of both process identifiers where it originates from (i.e., $(A,(1,2),B)$).

Algorithm 1 ensures that all edges having the same source and destination are merged into a single edge. This algorithm starts by detecting edges that need to be merged (i.e., in line 2) then, it removes them from the set of edges E (i.e., in line 3) and finally, it merges edges with the same source and destination and add them back to the set of edges E (i.e., in line 4).

Algorithm 1. Merge edges having the same source and destination

Input: Graph G: A graph that represents a configurable business process graph.

```

1 begin
2    $EdgesToBeMerged \leftarrow \bigcup_{i=1}^n (Src, PID_1, Dest)_i \in E_G / \exists (Src, PID_2, Dest) \in E_G$ 
   and  $PID_1 \neq PID_2$ ;
3    $E_G \leftarrow E_G \setminus EdgesToBeMerged$ ;
4    $E_G \leftarrow E_G \cup \{(Src, PID, Dest) / PID = \bigcup_{i=1}^k PID_i / (Src, PID_i, Dest) \in EdgesToBeMerged\}$ ;
5 end
```

The resulting MG needs some post-processing in order to be well formed according to some requirements imposed by the modelling notation. Recall that in this paper we consider EPC as a modelling notation. A well formed EPC should satisfy several requirements. We assume that the input models are well formed, i.e., respect all the EPC requirements [11]. After the merging operation two requirements are violated. These requirements are:

1. for each $n \in WN$: $|\bullet n| \leq 1$. This requirement means that each work node must have at most one input.
2. for each $n \in WN$: $|n \bullet| \leq 1$. This requirement means that each work node must have at most one output.

Changes that need to be applied to have a well formed model need some additional features that are provided by C-EPC [12,13]. C-EPC stands for Configurable EPC. It is an extended version of EPC where some *connectors* can be marked as configurable. A configurable connector can be configured by reducing its incoming branches (in the case of a join) or its outgoing branches (in the case of a split) [9]. The result will be a regular connector with a reduced number of incoming or outgoing branches. In addition, the type of a configurable OR can be restricted to a regular XOR or AND.

Next, we will use configurable connectors in order to respect the requirements previously mentioned (i.e., $\forall n \in WN, |\bullet n| \leq 1$ and $\forall n \in WN, |n \bullet| \leq 1$).

Each work node has a single entry and a single exit. To ensure that each work node has a single entry, Algorithm 2 operates as follows: if a work node has more than one input, we create a configurable connector (XOR-Join) that becomes the new destination in stead of that work node (i.e., lines 3 to 12). Finally, it creates a new edge from the new configurable connector to the work node that previously had more than one entry (i.e., line 13).

Algorithm 2. Single Entry

Input: Graph G: A graph that represents a configurable process graph.

```

1 begin
2   foreach Node in  $WN_G$  and  $|\bullet Node| > 1$  do
3      $IDs \leftarrow \emptyset$ ;
4     CreateNewCXOR(CXOR);
5      $\lambda(CXOR) \leftarrow join$ ;
6      $\tau(CXOR) \leftarrow (id(G), XOR)$ ;
7      $\eta(CXOR) \leftarrow true$ ;
8     foreach  $\{(Source, PID, Destination) \in E_G / Destination = Node\}$  do
9        $E_G \leftarrow E_G \setminus \{(Source, PID, Destination)\}$ ;
10       $E_G \leftarrow E_G \cup (Source, PID, CXOR)$ ;
11       $IDs \leftarrow IDs \cup \{PID\}$ ;
12    end
13     $E_G \leftarrow E_G \cup \{(CXOR, IDs, Node)\}$ ;
14     $RN_G \leftarrow RN_G \cup \{CXOR\}$ ;
15  end
16 end

```

Fig. 2 illustrates an example of this transformation. The left hand side of this figure depicts three input edges for the event “Delivery is relevant for shipment” which has been changed in the right hand side of this figure by inserting a configurable XOR connector and an edge from this connector to “Delivery is relevant for shipment” that has the label of all previous edges (i.e., “1,2,3”).

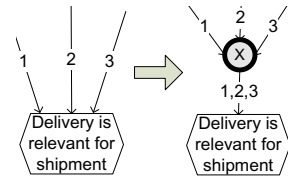


Fig. 2. A work node has a single entry

We proceed similarly to ensure that each work node has a single exit.

At this level, the merged process model is completely constructed and Fig. 3a depicts the resulting model. However, during this post-processing step, we have inserted several configurable connectors. This may lead to the appearance of several connector chains. Indeed, in Fig. 3a, the rounded regions represent

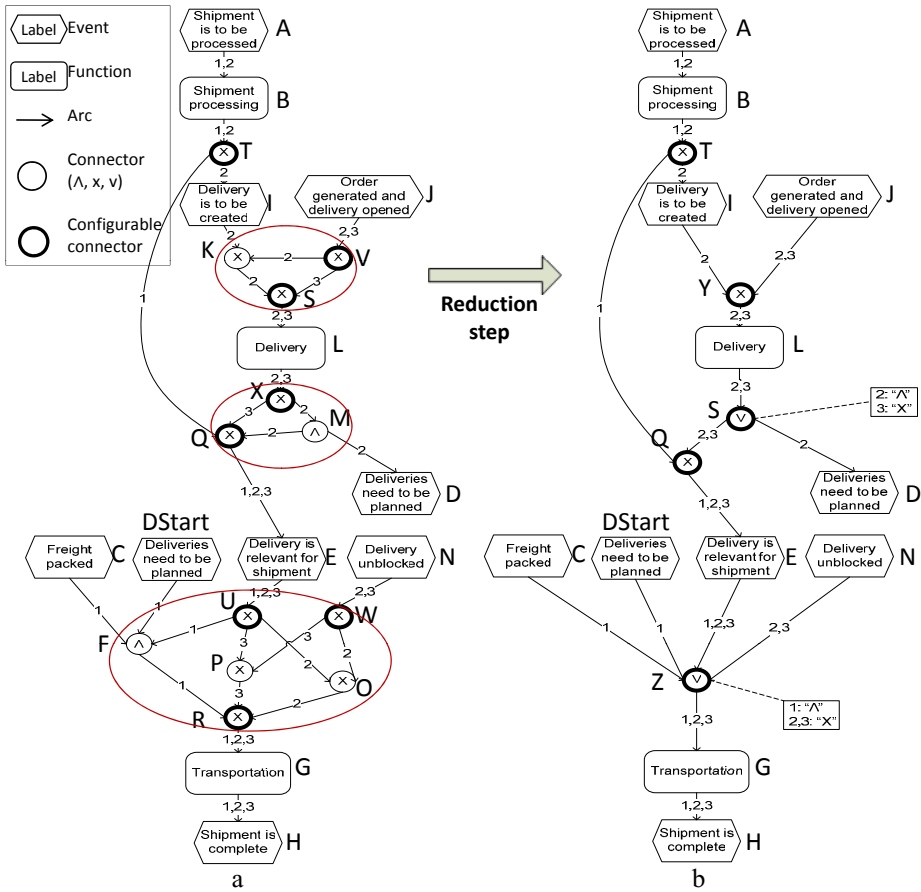


Fig. 3. Configurable business process model before and after the reduction step

various connector chains. In the following, we will show how we reduce these connector chains in order to obtain a reduced configurable process model such as depicted in Fig. 3b.

3.3 Reduction of the Configurable Business Process Graph

In this section, we present two simplification rules that help for reducing connector chains. These rules should reduce the business process graph while preserving its behaviour. These reduction rules are: (i) merging consecutive split/join connectors and (ii) removing trivial connectors.

Merge Consecutive Connectors. Algorithm 3 from line 3 to 27, merges any consecutive split/ join connectors into a single connector. It starts by parsing all the edges of G . If the source and the destination of an edge are two connectors of the same type (i.e., join or split) (i.e., line 3), then the algorithm fetches all

the adjacent connectors having that type in order to create a set of connectors that have to be reduced (i.e., lines 5 to 7). If one of the connectors is either an AND or an OR (i.e., line 11), then the new connector should be a configurable OR keeping trace back to the original operator with the identifier of the process where it originates from (i.e., line 14) otherwise it is a configurable XOR (i.e., line 9). Then, the algorithm continues to parse the remaining edges to detect input/output edges of the current connectors in order to link them to the new connector (i.e., lines 18 to 26). Line 27 of algorithm 3 is a call of the algorithm 11. It allows for merging edges having the same source and destination.

In Fig. 4.a, connectors F , P , O and R are four consecutive join connectors which are merged into Z in the Fig. 4.b. In Fig. 4.b there are three edges from U to Z with labels “1”, “2” and “3” which respectively originate from (Fig. 4.a) the edge from U to F , the edge from U to O and the edge from U to P . These three edges are merged into a single edge with the label “1,2,3” in the Fig. 4.c.

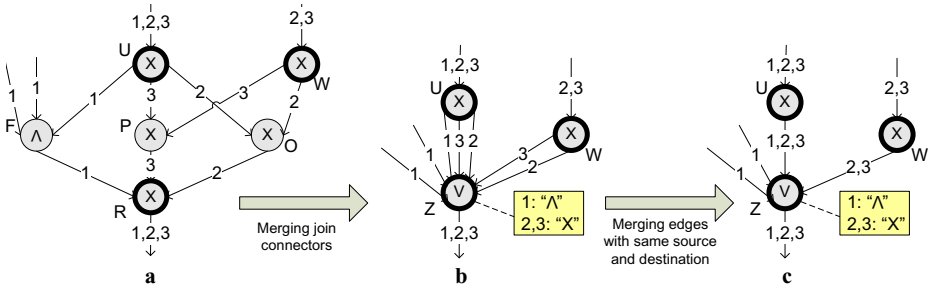


Fig. 4. Reducing a business process graph by merging consecutive connectors

Remove trivial connectors. A *trivial connector* is a connector that has only one input and one output edge. They do not provide any useful routing information. Thus, they can be removed without altering the process behaviour. To remove these connectors, Algorithm 3 from line 28 to 35, starts by fetching routing nodes that have one entry and one exit (i.e., line 28), then, it creates a new edge to connect the input and the output nodes of that routing node (i.e., line 31). After that, it removes the edges that were linking this routing node (i.e., line 32 and 33). Finally, it removes disconnected nodes (i.e., lines 35). *Disconnected nodes* are nodes without input or output edge. In our running example, Fig. 4.c depicts two trivial connectors (i.e., U and W) which are to be removed. Fig. 3.b depicts the resulting optimal configurable process model.

4 Related Work

For merging process variants, Gerth et al. [3] propose a formalism to detect equivalent business process models based on the detection of equivalent fragments contained in these models. This work is presented in the context of detection and resolution of version conflicts. It is implemented as a tool support for model

Algorithm 3. Reduce a business process graph**Input:** Graph G : A graph that represents a configurable process model.

```

1 begin
2   foreach ( $Src1, PID1, Dest1$ ) in  $E_G / \{Src1, Dest1\} \subset RN_G$  and
3     ( $\lambda(Src1) = \lambda(Dest1)$ ) do
4      $ToBeReduced \leftarrow \{Src1, Dest1\}$ ;
5     foreach ( $Src2, PID2, Dest2$ )  $\in E_G / Src2$  or  $Dest2 \in ToBeReduced$ 
6       and ( $\lambda(Src2) = \lambda(Dest2) = \lambda(Src1)$ ) do
7       |  $ToBeReduced \leftarrow \{Src2, Dest2\}$ ;
8     end
9     CreateNewConnector(ConfigConnector);
10    Operator = "XOR";
11    foreach  $Connector \in ToBeReduced$  do
12      | if  $\tau(Connector) = (PID, "AND")$  or  $\tau(Connector) = (PID, "OR")$ 
13        | then
14          | Op = "OR";
15        | end
16      |  $\tau(ConfigConnector) \leftarrow \tau(ConfigConnector) \cup \tau(Connector)$ ;
17    end
18     $\tau(ConfigConnector) \leftarrow \tau(ConfigConnector) \cup \{(id(G), Op)\}$ ;
19     $\eta(ConfigConnector) \leftarrow true$ ;
20    foreach ( $Src, PID, Dest$ )  $\in E_G / Src \in ToBeReduced$  do
21      |  $E_G \leftarrow E_G \setminus (Src, PID, Dest)$ ;
22      | if  $Src \in ToBeReduced$  and  $Dest \notin ToBeReduced$  then
23        |  $E_G \leftarrow E_G \cup (ConfigConnector, PID, Dest)$ ;
24      | else
25        |  $E_G \leftarrow E_G \cup (Src, PID, ConfigConnector)$ ;
26      | end
27    end
28    MergeEdgesWithSameSourceSameDestination( $G$ );
29    foreach  $Node \in RN_G$  and  $|\bullet Node| = |Node \bullet| = 1$  do
30      |  $Source \leftarrow m / (m, PID, Node) \in E_G$ ;
31      |  $Destination \leftarrow m / (Node, PID, m) \in E_G$ ;
32      |  $E_G \leftarrow E_G \cup \{(Source, PID, Destination)\}$ ;
33      |  $E_G \leftarrow E_G \setminus \{(Node, PID, Destination)\}$ ;
34      |  $E_G \leftarrow E_G \setminus \{(Source, PID, Node)\}$ ;
35    end
36    RemoveDisconnectedNodes( $G$ );
37 end

```

merging in IBM WebSphere Business Modeler [6]. The merge procedure defined here is not intended to be fully automated, it is rather developed for reducing the number of false-positive differences and conflicts in models management.

Gottschalk et al. [5] define an approach exclusively intended for merging EPC models. This approach consists first of transforming EPCs into a so called abstraction of EPCs, namely function graphs. The second step is the combination of these function graphs by means of set union operations. Finally, they transform back the combined function graph into an EPC. The object in their approach is not to create a configurable EPC, there are no configurable connectors introduced which would allow for extracting one of the original models.

Both approaches discussed previously [3,5] do not allow for the second nor the third requirements of Section 1. Indeed, the generated merged models do not allow to trace back where an element of the model originates from. As well as they do not provide any possibility to configure the obtained model in order to derive one of the input models.

As we share the same requirements of Section 1, La Rosa et al. [9] propose a technique that allows for the three of them. The proposed technique starts by computing a similarity measure between nodes of pairs of process models. Then, given a mapping between different elements of the original models, they propose a merge operator that computes maximum common regions (MCR) and then links elements of the second models, which are not in the MCR, to the MCR of the first model. Similar to our approach, they use arcs' annotations to allow for tracing back the origin of an element.

All the proposed approaches [3,5,9] are intended to merge pairs of process variants. This means that if a business analyst need to merge several process variants, he has to merge a first pair and then add the other variants one by one. However, our approach allows for merging process models at once.

5 Conclusion and Future Work

In this paper, we have presented a novel algorithm that allows for merging a collection of business process models in order to create a configurable process model. Even though we used EPC to illustrate our running example and results of the steps of our algorithm, our approach is not exclusively made for EPCs. Indeed, we represent a process variant as a business process graph according to the notation shown in Section 2.2 which has been used in this work.

Our algorithm ensures that the resulting configurable model includes the behaviours of the original business process variants by considering work nodes with identical labels and preserving the status of start and end nodes.

Current approaches, discussed in the related work section, provide algorithms for merging pairs of process models. Unlike these solutions, our algorithm allows for merging a collection of business process variants at once.

In the near future, we intend to develop a tool support and conduct an empirical evaluation of this algorithm.

In Section 3.1 we considered resolving only conflicts related to start and end nodes. This does not necessarily mean that these are the only possible conflicts.

Thus, while defining our testbed for future evaluation of our work, we plan for a deep analysis of process variants in order to look for other possible conflicts.

A strong assumption that we need to break in the future work concerns the fact that our approach allows for merging work nodes with identical labels whereas the approach of [9] supports approximate matching. As part of our future work, we plan to allow for such feature. In fact, the labels of functions and events have a specific meaning which needs to be interpreted in order to understand the process. Two different labels might mean the same and have as a result the same semantic during the process execution. As well as if two labels are the same, the meaning might be different depending on the context.

References

1. Braun, R., Esswein, W.: Classification of reference models. In: *Advances in Data Analysis*. Springer, Heidelberg (2007)
2. Fettke, P., Loos, P., Zwicker, J.: Business process reference models: Survey and classification. In: Bussler, C.J., Haller, A. (eds.) *BPM 2005*. LNCS, vol. 3812, pp. 469–483. Springer, Heidelberg (2006)
3. Gerth, C., Luckey, M., Küster, J.M., Engels, G.: Detection of semantically equivalent fragments for business process model change management. In: *SCC (2010)*
4. Gottschalk, F.: *Configurable Process Models*. Ph.D. thesis, Eindhoven University of Technology, Eindhoven, Netherlands (December 2009)
5. Gottschalk, F., van der Aalst, W.M.P., Jansen-Vullers, M.H.: Merging Event-Driven Process Chains. In: Chung, S. (ed.) *OTM 2008, Part I*. LNCS, vol. 5331. Springer, Heidelberg (2008)
6. Küster, J.M., Gerth, C., Förster, A., Engels, G.: A Tool for Process Merging in Business-Driven Development. In: *CAiSE Forum*. CEUR-WS.org, vol. 344 (2008)
7. Küster, J.M., Koehler, J., Ryndina, K.: Improving business process models with reference models in business-driven development. In: Eder, J., Dustdar, S. (eds.) *BPM Workshops 2006*. LNCS, vol. 4103, pp. 35–44. Springer, Heidelberg (2006)
8. La Rosa, M.: *Managing Variability in Process-Aware Information Systems*. Ph.D. thesis, Queensland University of Technology, Brisbane, Australia (April 2009)
9. La Rosa, M., Dumas, M., Uba, R., Dijkman, R.: Merging Business Process Models. In: Meersman, R., Dillon, T.S., Herrero, P. (eds.) *OTM 2010*. LNCS, vol. 6426, pp. 96–113. Springer, Heidelberg (2010)
10. Li, C., Reichert, M., Wombacher, A.: Discovering reference models by mining process variants using a heuristic approach. In: Dayal, U., Eder, J., Koehler, J., Reijers, H.A. (eds.) *BPM 2009*. LNCS, vol. 5701, pp. 344–362. Springer, Heidelberg (2009)
11. Mendling, J., Nüttgens, M.: EPC Syntax Validation with XML Schema Languages. In: *EPK (2003)*
12. Rosemann, M., van der Aalst, W.M.P.: A configurable reference modelling language. *Information Systems* 32(1) (2007)
13. Weber, B., Mendling, J., Reichert, M.: Flexibility in Process-Aware Information Systems (ProFlex) Workshop Report. In: *WETICE*. IEEE CS, Los Alamitos (2006)

Empirical Validation of MoDe4SLA; Approach for Managing Service Compositions

Lianne Bodestaff^{1,*}, Andreas Wombacher¹, and Manfred Reichert²

¹ University of Twente, The Netherlands
{l.bodestaff, a.wombacher}@utwente.nl
² University of Ulm, Germany
manfred.reichert@uni-ulm.de

Abstract. For companies managing complex Web service compositions, challenges arise which go far beyond simple bilateral contract monitoring. For example, it is not only important to determine whether or not a component (i.e., Web service) in a composition is performing properly, but also to understand what the impact of its performance is on the overall service composition. To tackle this challenge, in previous work we developed MoDe4SLA which allows managing and monitoring dependencies between services in a composition. This paper empirically validates MoDe4SLA through an extensive and interactive experiment among 34 participants.

Keywords: SLAs, service composition, empirical validation, SLA management.

1 Introduction

Regarding the monitoring of Web service (WS) compositions, it is necessary to take composition structure as well as characteristics of services into account. This information is needed to assess composition performance. This is particularly important when considering the growing complexity of WS compositions. Particularly, providers of composite services struggle to manage these complex constellations. Different services are provided with different quality levels. Further, services stem from different providers, and have different impact on the composition. To meet Service Level Agreements (SLA) with its customers any company faces the challenge of managing its underlying services. For each SLA violation the company determines its impact on the composition and decides on how to respond. Generally, complexity of this decision process grows with the number of services being involved in the composition.

The goal of MoDe4SLA [12] is to determine for each service in a composition its *impact* on the composition performance. The latter is measured by analyzing different metrics (e.g. costs and response time) as used in SLAs. Through analysis of both *dependency structure* and *impact* it becomes possible to monitor composition performance taking dependencies between services into account. The advantage of such analysis is the possibility to explain SLA violations of a service composition through identifying badly performing services the composition depends on.

* This research has been supported by the Dutch Organization for Scientific Research (NWO) under contract number 612.063.409.

This paper validates our scientific solution (MoDe4SLA) for the real-life problem of managing complex service compositions through a *controlled, quantitative experiment* [3,4]. Experimental validation in Computer Science is recognized as being very important [5,4]. However, still a minority of research papers actually provides some experimental results [6,7]. Without validating developed approaches like MoDe4SLA, however, a researcher might steer his research efforts into a fruitless direction [5]. In our evaluation we conduct an experiment with 34 participants. These participants are asked to manage Web service compositions using our approach and to do this without using MoDe4SLA. We gather data on their experiences and analyze them.

This paper evaluates *usefulness* of our MoDe4SLA approach for managers burdened with maintenance of service compositions [8]. We first provide some background information on MoDe4SLA in Section 2. Related work is discussed in Section 3. We evaluate usefulness by asking experts to manage simulated runs of service compositions using MoDe4SLA (Sections 4 and 5). We conclude with a summary in Section 6.

2 Background and Example Scenario

MoDe4SLA [11,2] intends to support companies in managing their composite services by identifying and monitoring their dependency on other services requested from external providers. Our approach pinpoints to services causing SLA violations of the composition. SLAs describe constraints (e.g., response time and availability) a service has. Our abstract example scenario (Fig. 1) depicts a composition where every composition invocation triggers WS 1 - WS 7, WS 13, WS 16, and WS 17. In addition WS 8, WS 12, or the *Loop*-construct is chosen (*XOR*-construct where each outgoing edge is annotated with the *chance to be chosen* compared to its siblings). If the *Loop* is chosen, WS 9 - WS 11 are invoked in a repeated *sequence* (on average 3 times). Further, either WS 14 or WS 15 is invoked (*ORDISC*-construct with ratio 0.89 : 0.11). Except for services in the *Loop* sequence, all services are invoked in *parallel*.

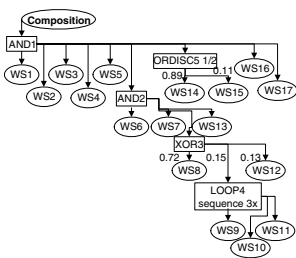


Fig. 1. Exemplary service composition

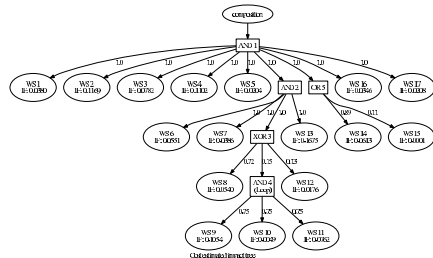


Fig. 2. Estimated impact tree for cost

Offering a composite service to customers implies that a company relies on other service providers. SLA constraints might depend on different services, or depend on them in different ways. For example, if a company offers information with fast response time by querying five providers and returning information of the fastest responding one, a cost constraint is influenced by all five services (invoking means paying), while a

impact of each service. Color *red* indicates worse performance than expected and *green* indicates proper performance. *Yellow* indicates the service is not performing perfectly, but still within boundaries set by the company, and *dark green* indicates a service runs better than anticipated on. Edges are colored in the same manner. For example, red indicates an edge contributes more often than expected.

Fig. 3 depicts the cost feedback tree of our example scenario. The yellow composition indicates the cost constraint is not met. This is caused by two factors: (1) *Performance of service WS 13* is bad since it violates its SLA (i.e., it is colored red) with an impact factor (IF) of 0.1763, i.e., almost 18% of the overall costs are contributed by this service. (2) Due to the *structure of the composition*, services WS 9 - WS 11 are contributing more often than expected (i.e., red incoming edges) which causes elevated costs. Although WS 17 is not functioning properly, its impact factor is too low (i.e., 2%) for causing major composition violations. Furthermore, several services are over-performing (i.e., dark green). If performance problems are solved, these services might positively influence overall costs.

3 Related Work

Menasce [9] presents response time analysis of composed services to identify impact of slowed down services. The result is a measure for the overall slow down depending on statistical likelihood of a service not delivering expected response time. As opposed to our approach, Menasce performs analysis at design-time rather than providing a runtime based analysis. A different approach with the same goal is the virtual resource manager proposed by Burchard et al. [10]. It targets a grid environment where a calculation task is distributed among different grid vertices for individual computation jobs. If a grid vertex fails to deliver the promised service level, a domain controller first reschedules the job onto a different vertex within the same domain. If this action fails, the domain controller attempts to query other domain controllers for passing over the computation job. Although the approach covers runtime, it follows a hierarchical autonomous recovery mechanism. MoDe4SLA focusses on identifying causes for correction on the level of business operations rather than on autonomous job scheduling. In the COSMA approach Ludwig et al. [11] describe a framework for life cycle management of SLAs in composite services. They recognize the problem of managing dependencies between different SLAs. Furthermore, their COSMA doc component describes composite specific dependencies but does not explicate what type of dependencies are considered. The SALMon approach by Oriol et al. [12] aims at monitoring and adapting SOA systems at runtime. Monitoring is done for SLA violations. Further, a decision component performs corrective actions so that SLAs are satisfied. Their approach does not focus on service compositions but on runtime adaptability. As a consequence, they are not concerned with dependencies between different SLAs. Moser et al. [13] describe an approach for automatically replacing services at runtime without causing any downtime for the overall system. The BPEL processes are monitored according to their QoS attributes and replacement of services and partners is offered on various strategies.

Although their approach has similarities to ours, their goals focus on runtime adaptability, and not on service compositions and their SLA dependencies. Sahai et al. [14] aim at automated SLA *monitoring* by specifying SLAs and not only considering provider side guarantees but focus also on distributed monitoring, taking the client side into account as well. Barbon et al. [15] enable run-time monitoring while separating business logic from monitoring functionality. For each process instance a monitor is created. Unique for this approach is its ability to also monitor *classes* of instances, enabling abstraction from an instance level. The smart monitoring approach of Baresi et al. [16] implements the monitor itself as a service. There are three types of monitors available for different aspects of the system. Their approach is developed to monitor specifically contracts with constraints. In [17] Baresi et al. present an approach to dynamically monitor BPEL processes by adding monitoring rules to the different processes. These rules are executed during runtime. Our approach does not require modifications to the process descriptions what might suit better to some application areas. An interesting approach in this direction is work by Mahbub et al. [18] who considers the whole state of the system in their monitoring approach. They aim at monitoring derivations of system behavior.

Most of the discussed approaches are evaluated by providing a proof-of-concept implementation (e.g., [19][20]). In addition, some approaches are validated by a performance study (e.g., [18][21]). However, to our knowledge, none of the approaches has been empirically validated. This complicates finding a suitable method to compare our approach with. Therefore, we choose to use straightforward bilateral monitoring as a baseline for evaluating MoDe4SLA.

4 Evaluating Usefulness: Setup

The design of our evaluation is described more extensively in Bodenstaff et al. [8]. To evaluate usefulness of the results from our MoDe4SLA analysis, we interview experts, asking them to make a statement on how useful they perceive the approach when managing compositions. We use the following criterion to evaluate usefulness:

MoDe4SLA is considered as being useful when experts testing it perceive the feedback given by MoDe4SLA as more useful for managing and maintaining the composition than when using bilateral monitoring results.

Common management approaches return bilateral monitoring results to users. They do not provide information on the relation between the different services, but merely return individual service performance. For evaluating our MoDe4SLA approach, we extend the implementation of an existing simulator - SENECA [22] - with generating impact models and with an analyzer module (cf. Fig. 4). This simulator randomly generates a *composition structure* for a given number of services. To each created service in the composition a randomly generated SLA is assigned. The *impact models* are derived based on the composition structure and SLAs. SENECA simulates invocation of the services according to the composition structure. Accordingly, services might violate their SLAs. The simulator gathers *runtime data* and generates *feedback models*.

For our evaluation we prepare three compositions of different complexity. The complete set of documents, including the questionnaire handed out to experts for evaluation can be found in Bodestaff et al. [23]. The first test case (TC1) consists of five services with three constructs, the second test case consists of ten services with one OR-split and one discriminative join. Finally, the third test case consists of seventeen services connected through five constructs. This case constitutes our example scenario (cf. Fig. 1).

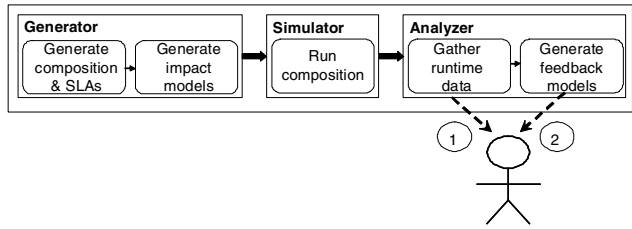


Fig. 4. Evaluating usefulness

For each composition two documents are prepared. The MoDe4SLA document contains feedback models for both response time and costs, while the control document contains performance data for each service resulting from bilateral monitoring, but does not provide information on how they are related [23]. Main goal of our evaluation is to test following hypothesis:

The MoDe4SLA document has a clear benefit over the control document when managing the composition.

We evaluate this hypothesis by conducting a survey considering the following research questions (RQ):

- RQ1** Accuracy of identifying malfunctioning services using MoDe4SLA in comparison to the use of bilateral monitoring results,
- RQ2** Efficiency in identifying malfunctioning services using MoDe4SLA compared to bilateral monitoring results,
- RQ3** Confidence experts have in their answers when using MoDe4SLA compared to bilateral monitoring results.
- RQ4** How complex is the MoDe4SLA approach for users?
- RQ5** Which possible improvements do experts suggest for the MoDe4SLA approach?

For this purpose, we prepare a questionnaire of 49 questions that experts answer before, during and after the experiment. Typically they five-level Likert item (i.e., Strongly disagree, Disagree, Neither agree nor disagree, Agree, Strongly agree) to rate responses. We start with a trial run on three colleagues to discover problems in examples, test cases, and questionnaire. Although no errors are found, a front sheet depicting graphically each composition is added. We conduct six more sessions with 34 participants from several universities and companies. Each session consists of:

1. A presentation explaining the goal of the approach.
2. Discussion of two examples where bilateral monitoring results and MoDe4SLA feedback model results are explained, including an interpretation discussion.
3. The evaluation: First some *introductory questions* are answered (Q1-Q7) after which the first test case with *bilateral monitoring* results is studied and the participant answers Q8-Q11. Then the MoDe4SLA feedback trees are studied and

Q12-Q18 are answered. These steps are repeated for the second and third case. We conclude with general questions (Q41-Q47).

5 Conclusions from Evaluating Usefulness

We introduce some demographics after which we discuss answers related to questions from Section 4. We conclude with an analysis of relations between different outliers in questions in Section 5.3. Statistics on all questions can be found in [23].

5.1 Demographics

The group of 34 participants consists of experts in developing and managing services. 11 experts are from industry, of which 9 are also active in academia, and additional 23 experts from academia. 15% of the experts have experience using tools for managing composite services. 15% of the experts have experience developing tools for managing composite services. 60% of the participants have not worked with composite services, while the remaining 40% have experience varying from less than one year to over three years. 9% of the participants consider themselves having a high level of expertise in managing composite services. They are in particular supportive for MoDe4SLA approach and therefore no explicit discussion of this participant group is done. The low number of experts in managing service compositions does not influence the overall result since also experts in services understand the complexity of service compositions and their management although they have not performed the task themselves yet.

Although our participants are familiar with service compositions, on average their expertise in managing them is not high. As advantage this inexperience helps us in determining how difficult it is to master MoDe4SLA. As disadvantage, we cannot expect much feedback on possible other approaches for managing service compositions.

5.2 Statistics

We start each test case with a question on how complex the participants feel the composition is (TC1 with 5 services: Q8, TC2 with 10 services: Q19, TC3 with 17 services: Q30) (cf. Fig. 5). We assume TC2 is perceived as less complex than TC1 since it contains only one construct: an OR-split with discriminative join while TC1 contains three constructs. We add this question since we assume the more complex the composition is, the more useful MoDe4SLA will be. Although participants consider the different test cases to be of different complexity, and although participants appreciate using MoDe4SLA even more when considering the complex test case (i.e., TC3), these differences were lower than expected as discussed in the following.

RQ1: Accuracy. We want to know whether participants feel that identifying problematic services can be done more accurately with than without MoDe4SLA. We have two questions giving us an insight on this.

The first question is asked for each test case, and for both response time and costs. We ask participants whether they perceive identifying the impact each service has on the composition, is easier with MoDe4SLA than it is without MoDe4SLA (for TC1: Q15

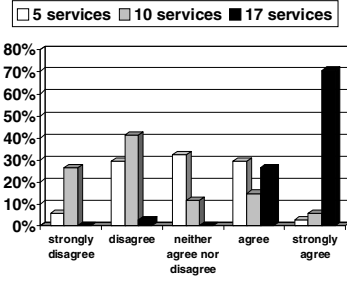


Fig. 5. The composition is complex

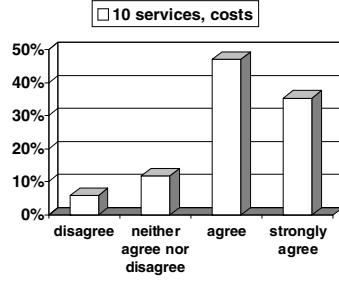


Fig. 6. It is easier to determine the impact of each service with the analysis than without

and Q16, for TC2: Q26 and Q27, for TC3: Q37 and Q38). MoDe4SLA is perceived as being more useful for response time than for costs, and as more useful for TC3 than for TC1 and TC2. The majority perceives the use of MoDe4SLA as very helpful for easier identification of the impact. Fig. 6 depicts the histogram with *least* positive responses for our approach. It still entails over 80% of the participants agreeing or strongly agreeing to the statement.

Second, for each test case we ask participants whether they consider MoDe4SLA being helpful when managing the composition with regard to accurately depicting malfunctioning services (for TC1, TC2, and TC3, and Q18, Q29, and Q40). Fig. 8 depicts results. 75-80% of the participants agree or strongly agree that MoDe4SLA is helpful to accurately depict these services even for the least complex composition.

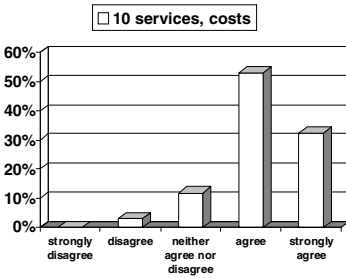


Fig. 7. It takes less time to see relations between different services and the composition

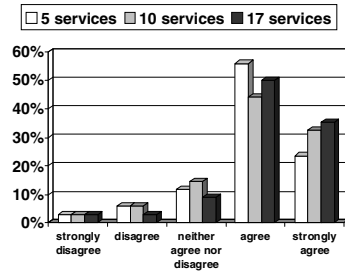


Fig. 8. MoDe4SLA is helpful when depicting malfunctioning services

RQ2: Efficiency. We investigate whether participants consider it as more *efficient* when using MoDe4SLA for managing service compositions than without. First, for each test case and for both response time and cost we ask participants to respond to the statement that it takes less time to see relations between the different services in a composition when using MoDe4SLA. Since MoDe4SLA relies on identifying relations and dependencies between the services, we assume that MoDe4SLA is helpful when trying to identify these relations. Depending on the test case, 85-100% of the participants (strongly) agree with this statement. Fig. 7 depicts the least positive responses.

Second, for each test case we ask participants whether they consider MoDe4SLA being helpful when managing the composition with regard to efficiently depicting malfunctioning services (for TC1, TC2, and TC3, and Q18, Q29, and Q40). Fig. 9 depicts results for these questions. Around 90% of the participants agree or strongly agree that MoDe4SLA is helpful to accurately identify these services.

RQ3: Confidence. To evaluate how confident participants are when making a choice on which services to adapt to get better performance, for each test case we ask three questions. First, we ask how confident they are making a choice *before* seeing MoDe4SLA models. Second, we ask how confident they are about their original choice when seeing the models. Third, we ask how confident they are making a choice when considering MoDe4SLA models. The aim of the second question is to find out whether participants feel MoDe4SLA giving additional support. If they feel more or less confident, MoDe4SLA apparently gives them additional insights. If they do not change their opinion, MoDe4SLA has not given additional insights. The change in confidence (i.e., the second question) is depicted in Fig. 10. For each test case at least 80% of the participants change their confidence level. The confidence level of participants before considering MoDe4SLA is depicted in Fig. 11. Experts have reasonable confidence levels in the first and second test case but no confidence in the third one. In Fig. 12 the confidence level goes up for all test cases after participants studied the MoDe4SLA files. On average participants feel more confident making choices on which services to adapt using MoDe4SLA than without it.

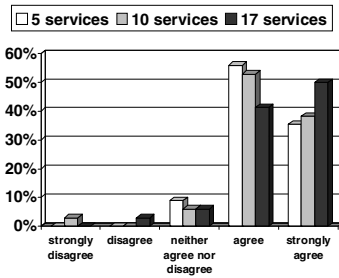


Fig. 9. MoDe4SLA is helpful to fast select services to renegotiate

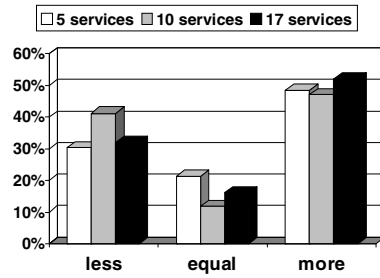


Fig. 10. How is your confidence in your selection for renegotiation?

We test usefulness of our approach by answering RQ1-RQ3. In addition, we ask participants at the end of the survey to respond to the statement that using MoDe4SLA is helpful when managing composite services (cf. Fig. 13). None of them disagrees or strongly disagrees with this statement. 94% of them agree or strongly agree with it.

RQ4: Complexity. Another important aspect is how difficult MoDe4SLA is to comprehend. We strive to develop an *intuitive* approach that is easy to understand for users. Of course, the positive evaluation results concerning MoDe4SLA usefulness after a short training period support our claim that its usability is good. However, we also want to

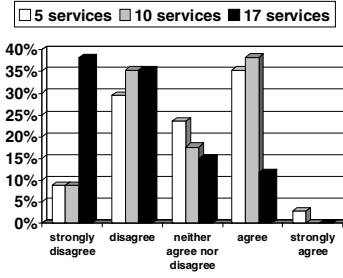


Fig. 11. I would feel confident in selecting services for renegotiation

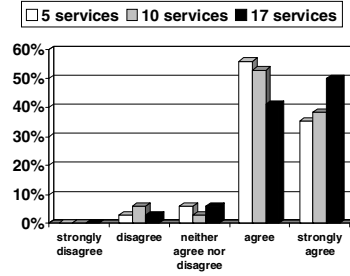


Fig. 12. I feel more confident selecting renegotiation services with MoDe4SLA than without

know whether participants feel the given explanation of MoDe4SLA was sufficient to use the models (cf. Q42). The presentation takes at most one and a half hour, including discussion and questions. Fig. 14 indicates that over 85% of the participants agree with this statement. A considerable group (around 12%) appreciate more explanation.

Furthermore, in Q45 we ask participants to name weak points of the approach. Here, we find indicators MoDe4SLA is less intuitive for some of the participants. 7 of them (i.e., around 20%) indicate they have problems understanding the values in the model. With these values, participants sometimes mean impact factors, but usually contribution factors (i.e., branch annotations) turn out to be hard to understand. The magnitude of numbers confused some of the participants. 2 of them (i.e., around 6%) state for them the feedback models are too complex to comprehend. In conclusion, about 75% of the participants have no difficulties understanding values in the feedback models.

RQ5: Possible improvements. We ask participants to state what they feel is most beneficial about the feedback models. Participants like the visualization part, especially the coloring. Furthermore, impact factors and analysis itself are beneficial. Also, we ask for possible improvements. The first is a reduction of the many numbers used in the models. As discussed in RQ4, for some participants there is too much information given. In addition, participants feel they are able to choose using colors and impact factors. Therefore, we consider filtering this information when models are presented to users. Second, participants appreciate some interpretation guidelines. For example, “a low impact factor indicates”, “a high ratio means”, and “from the combination of an impact factor and ratio you can derive”. Therefore, we consider extending the presentation with information on these statements. Third, related to the previous improvement, participants appreciate guidelines for decision making. It is beneficial if the models indicate which services to consider for change, and why. Currently, models only provide monitoring information without suggestions on how to improve performance. Developing such guidelines is part of our future work.

5.3 Evaluation Conclusions

Test cases. Although we introduce three test cases with different complexity levels, MoDe4SLA is already perceived as useful when managing a composition with only

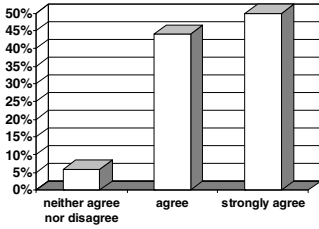


Fig. 13. MoDe4SLA is helpful for managing service compositions

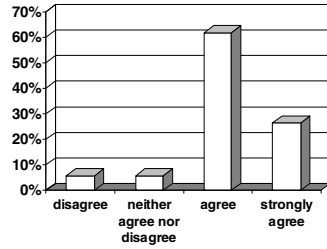


Fig. 14. The presentation is sufficient to understand MoDe4SLA

five services. Furthermore, our test case with seventeen services is considered as highly complex. However, real-life compositions are typically much larger. As a consequence, a proper management approach is definitely necessary in those cases. Further, TC2 is perceived as less complex than TC1 although it consists of twice as many services. The three constructs in TC1 make it more complex than TC2 that contains one construct.

Cost versus response time. When browsing through the different diagrams, it is clear that most participants struggle more with response time dependencies than with cost dependencies. As a result, MoDe4SLA is especially appreciated in response time models, which is also supported by answers of Q48 where beneficial parts are named. Response time of a branch depends on the interaction between different services: Whether service A contributes, does not only depend on whether it is invoked, and how fast it runs, but also on how fast its neighbor runs. Also for cost the contribution depends on the cost of other services, but this dependency is less strong.

6 Summary and Outlook

This paper presents our evaluation of usefulness of our MoDe4SLA approach. In several interactive sessions we conduct an extensive evaluation where 34 participants answer 49 questions. To support our hypothesis that MoDe4SLA models have a clear benefit over an approach that only supports bilateral monitoring (cf. Section 4), we investigate usefulness of our approach as perceived by experts. All three sub-questions concerning accuracy (RQ1), efficiency (RQ2), and confidence (RQ3) clearly indicate that participants benefit from the MoDe4SLA models. Though there are improvements to be considered, as discussed in RQ5, participants are able to properly understand MoDe4SLA within one and a half hour. These results give us support to continue developing our monitoring approach. So far, we only ask participants for their *opinion*. There are no good or bad answers. Of course, we want to know whether the answers our participants give are *effective* as well. In other words, we want to know whether their decisions are better when making them with MoDe4SLA than without. This is considered in our effectiveness evaluation in future research.

References

1. Bodenstaff, L., Wombacher, A., Reichert, M., Jaeger, M.C.: Monitoring dependencies for SLAs: The MoDe4SLA approach. In: Proc. SCC 2009, pp. 21–29 (2008)
2. Bodenstaff, L., Wombacher, A., Reichert, M.U., Jaeger, M.C.: Analyzing impact factors on composite services. In: Proc. SCC 2009, pp. 218–226 (2009)
3. Kitchenham, B.: Evaluating Software Engineering Methods and Tool Part 1: The Evaluation Context and Evaluation Methods. ACM SIGSOFT Software Engineering Notes 21(1) (1996)
4. Zelkowitz, M., Wallace, D.: Experimental Models for Validating Computer Technology. IEEE Computer 31(5), 23–31 (1998)
5. Tichy, W.: Should Computer Scientists Experiment More? IEEE Computer 31(5) (1998)
6. Sjøberg, D., Hannay, J., Hansen, O., Kampenes, V., Karahasanovic, A., Liborg, N., Rekdal, A.: A Survey of Controlled Experiments in Software Engineering. IEEE Transactions on Software Engineering 31(9), 733–753 (2005)
7. Tichy, W., Lukowicz, P., Prechelt, L., Heinz, E.: Experimental Evaluation in Computer Science: A Quantitative Study. Journal of Systems & Software 28(1), 9–18 (1995)
8. Bodenstaff, L., Wombacher, A., Jaeger, M.C., Reichert, M., Wieringa, R.: Monitoring service compositions in MoDe4SLA: Design of validation. In: Proc. ICEIS 2009 (2009)
9. Menascé, D.A.: Response-time analysis of composite Web services. IEEE Internet Computing 8(1), 90–92 (2004)
10. Burchard, L.O., Hovestadt, M., Kao, O., Keller, A., Linnert, B.: The Virtual Resource Manager: An architecture for SLA-aware resource management. In: Proc. on Cluster Comp. and the Grid, pp. 126–133 (2004)
11. Ludwig, A., Franczyk, B.: Managing dynamics of composite Service Level Agreements with COSMA. In: Proc. Int. Conf. on Fuzzy Systems and Knowledge Discovery (2008)
12. Oriol, M., Marco, J., Franch, X., Ameller, D.: Monitoring adaptable SOA-systems using SALMon. In: Proc. WS on Service Monitoring, Adaptation and Beyond (2008)
13. Moser, O., Rosenberg, F., Dustdar, S.: Non-intrusive monitoring and service adaptation for WS-BPEL. In: Proc. WWW 2008 (2008)
14. Sahai, A., Machiraju, V., Sayal, M., van Moorsel, A.P.A., Casati, F.: Automated SLA monitoring for Web services. In: Proc. Int. WS on Distributed Systems, pp. 28–41 (2002)
15. Barbon, F., Traverso, P., Pistore, M., Trainotti, M.: Run-time monitoring of instances and classes of Web service compositions. In: Proc. ICWS 2006, pp. 63–71 (2006)
16. Baresi, L., Ghezzi, C., Guinea, S.: Smart monitors for composed services. In: Proc. ICSOC 2004, pp. 193–202 (2004)
17. Baresi, L., Guinea, S.: Towards dynamic monitoring of WS-BPEL processes. In: Proc. ICSOC 2005, pp. 269–282 (2005)
18. Mahbub, K., Spanoudakis, G.: Run-time monitoring of requirements for systems composed of Web-services: Initial implementation and evaluation experience. In: Proc. ICWS 2005 (2005)
19. Baresi, L., Guinea, S.: Towards dynamic monitoring of WS-BPEL processes. In: Benatallah, B., Casati, F., Traverso, P. (eds.) ICSOC 2005. LNCS, vol. 3826, pp. 269–282. Springer, Heidelberg (2005)
20. Ludwig, A., Franczyk, B.: COSMA—An Approach for Managing SLAs in Composite Services. In: Bouguettaya, A., Krueger, I., Margaria, T. (eds.) ICSOC 2008. LNCS, vol. 5364, pp. 626–632. Springer, Heidelberg (2008)
21. Pistore, M., Traverso, P.: Assumption-Based Composition and Monitoring of Web Services. In: Test and Analysis of Web Services, pp. 307–335 (2007)

22. Jaeger, M.C., Rojec-Goldmann, G.: SENECA - simulation of algorithms for the selection of Web services for compositions. In: Bussler, C.J., Shan, M.-C. (eds.) TES 2005. LNCS, vol. 3811, pp. 84–97. Springer, Heidelberg (2006)
23. Bodenstaff, L., Wombacher, A., Reichert, M.: Evaluation data for MoDe4SLA. Technical report: TR-CTIT-10-05, University of Twente (2010)

Towards a Goal-Driven Approach for Business Process Improvement Using Process-Oriented Data Warehouse

Khurram Shahzad and Constantinos Giannoulis

Department of Software and Computer Systems, Royal Institute of Technology (KTH),
Stockholm, Sweden
mks@dsv.su.se

Department of Computer and Systems Science, Stockholm University (SU),
Stockholm, Sweden
constantinos@dsv.su.se

Abstract. An emerging approach for business process analysis is to use business intelligence practices by employing data warehouse and decision-making techniques. However, little work has been done on developing core methods and tools to guide process analysis and improvement. Our research addresses this issue by introducing a goal-driven approach for business process improvement using process warehouse. In this paper, we present our three step method and its evaluation through an empirical study. The results showed that the impact of applying our method for process improvement has been perceived positively.

Keywords: Process Improvement, Business Process Intelligence, Business-orientation of Process Warehouse, Goal-driven Process Improvement.

1 Introduction

Analysis and improvement of business processes, a core phase of business process management (BPM) lifecycle, has been receiving increasing attention among many enterprises. Facilitating sustainable process improvement requires an understanding of processes as well as approaches for continuously analyzing and improving them.

An emerging approach for business process analysis utilizes business intelligence practices, which attempts to boost the analytical capabilities of BPM systems by employing process-oriented data warehouse [1, 2], called process warehouse (PW) [3]. PW captures the data about executed business processes, such as actors, activities performed by actors, resources used, execution time and thereby can be used as an adequate basis for analysis and optimization of those processes [2, 3, 4].

Due to a typically large size of PW and lack of approaches for their efficient use, users commonly retrieve irrelevant data, miss vital information, and as a consequence, fail in the process analysis task [5]. Muehlen et al. [6] has argued that if the cognitive effort to extract and interpret the information from a process warehouse is high, the extracted information may be ignored and thus it doesn't bring any value for decision makers. It is because, users are accustomed to the business perspective in process analysis [7] and thereby far from capabilities to directly utilize the data

contained in warehouses for process analysis and improvement. Therefore, the question arises, how can process warehouse be used for analysis and improvement of business processes? An additional problem is that despite the PW concept exists for some years, only a limited number of implementations are reported (e.g. HP's business process cockpit [8]). A possible reason for this is the lack of methods that can guide business process improvement using process warehouse [5, 9].

In order to address the outlined problems, a starting point may be the consideration of the needs for business process analysts and decision makers. They are accustomed to the business perspective in the process analysis [10] and earlier studies [7, 11] contend they prefer to analyze business performance in accordance to goals. Therefore, our approach to address the discussed issues is based on a business orientation in the design and utilization of warehouse. In this study we introduce a three step method to identify goals, integrating them with PW and further using these goals to navigate the warehouse for process analysis and improvements. We further investigate the usability and impact of the method from users' perspective. For that, we have conducted an empirical study based on the evaluation model proposed by Hong et al. [12]. The key benefits that the method offers are: to support goal-based navigation of PW, to retrieve data relevant to the goals and to enhance the ways users analyze and improve business processes.

The rest of the paper is organized as follows: Section 2 briefly reports on work related to ours. In Section 3 the proposed method for process analysis and improvement is introduced. Section 4 presents the evaluation of the proposed approach. Section 7 summarizes the contribution of this study and elicits the directions of future work.

2 Related Work

Studies on business process analysis have focused on the utilization of process-oriented warehouses by proposing basic toolsets like Business Process Cockpit [8] and architectures like Business Process Intelligence System architecture [13], which allow users to monitor and analyze processes-related data. Other studies on process warehouses have focused on proposing adequate multidimensional data models for this kind of warehouse, such as generic [4], or domain specific (e.g. healthcare, chemical engineering). An analysis of such PW design proposals concluded incompleteness of the presented data models [14]. Also, a recent study [15] contend that these systems don't provide the means for process improvement, therefore improvement can be done solely based on human knowledge and experience.

Business orientation of warehouse: During the last decade a number of efforts have been reported on the business orientation in the data warehouse. At first, Sarda et al. [16] discussed the use of business metadata in data warehouse systems. Following that, Stefanov et. al. [7] extended Sarda's work and introduced an integrated view on data warehouse and enterprise models using "model weaving links". The model weaving approach was also used to make the relationship between data warehouse and enterprise goals visible and accessible [11] through an analysis tool to support and improve data interpretation. However, in these studies, the content scope is restricted to data-orientated information systems and methods that can guide process analysis and improvement are not discussed.

Methods for process improvement: Recent studies [9, 17, 18] have advocated the need for building methods and frameworks that can guide analysis and improvement of processes. These approaches use process execution logs or process warehouses as a data source. However, either these approaches have a limited scope (e.g. [18]'s scope is limited to agent assignment), or they do not facilitate extraction and interpretation of the information needed for process analysis and improvement.

This study combines the ideas of the aforementioned related work aiming at making use of process warehouse for goal-driven business process analysis. We propose a well-structured method that differs from existing ones in the way that it integrates goals with process warehouse, while providing the goal structure for utilizing the warehouse to retrieve data for analysis of process performance.

3 Goal-Driven Process Improvement: The Method

In this section we introduce our goal-driven process improvement method. The purpose of the method is to facilitate PW designer in integrating goals with PW and process manager to diagnose process related weakness and make changes to the process for possible improvements. The method consists of three steps: build a goal structure (section 3.1), integrate goals into a PW (section 3.2), analyze and improve business process (section 3.3).

3.1 Step 1: Build Goal Structure

Our method utilizes the process design framework [19] and the goal decomposition tree (GDT) proposed by [20] in agent programming. The process design framework is used as a reference to ensure that all essential process concepts are addressed and comprehensively represented, as exemplified in [21]. Whereas, GDT is a mechanism that captures both the declarative and procedural aspects of goals, which offers the ability to reason about goals. GDT provides traceability between goals (both top-down and bottom up) using logical inferences through a set of decomposition operators, allowing claims on a goal's achievement based on its sub-goals' achievement. Considering [20] we shift from goal-agent to goal-process coupling. Within the scope of this paper a *goal* is a state of a process in terms of the quality of a service property that is intended to be achieved (e.g. quality of service: timely, goal: treatment is timely). Figure 1 depicts the workflow of the tasks, followed by their description:

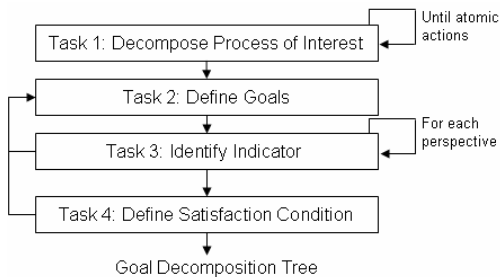


Fig. 1. Workflow of Step 1: Build goal structure

The *first task* is to recursively analyze a process from its functional perspective and break it down to atomic actions; where no further process sub-elements can be identified. A Process P is disaggregated from its functional perspective, $P = P_1 \wedge P_2 \wedge \dots \wedge P_n$, where P_1, P_2, \dots, P_n are sub-process elements of P . For the subsequent sub-process $P_1 = A_{11} \wedge A_{12} \wedge \dots \wedge A_{1n}$, where $A_{11} \dots A_{1n}$ are activities of P_1 . Apart from AND decompositions, more operators can be used (e.g. OR, SeqAND, SeqOR etc. [20]).

The *second task* is to define goals for the process by identifying the quality of service properties needed to be considered for process analysis. Based on the functional decomposition of a process, each goal is expanded into sub-goals, where achievement of the goal relies conventionally on the disaggregated achievement of sub-goals. Let G be the elicited goal of process P . The Goal (P, G) is expanded by disaggregation of the process $(P_1, G_1) \wedge (P_2, G_2) \wedge \dots \wedge (P_n, G_n)$.

The *third task* is to identify criteria for the achievement of goals. It is a bottom-up task in which indicators are set through a cognitive process for each goal by employing the process design framework with four perspectives (functional F , behavioral B , organizational O and informational I). Recall, process design framework covers basic building blocks and ensures a comprehensive representation of process concepts. Therefore, by employing the perspectives we aim for a comprehensive criteria for each goal. For the achievement of goal G defined on process P , indicators I_1, I_2, I_3, I_4 are set, relevant to each perspective; $(F, I_1), (B, I_2), (O, I_3), (I, I_4)$.

The *fourth task* is to specify satisfaction conditions (S) for each indicator, where satisfaction conditions of all indicators of a goal form the satisfaction condition of the goal. A satisfaction condition is the desired instance of an indicator that demonstrates goal achievement as defined by stakeholders. Satisfaction condition is a coupling of context and desired value, represented by $S(C, V)$. Desired value is the value aimed at being achieved, while context represents the conditions for which the value is valid. For indicators I_1, I_2, I_3, I_4 satisfaction conditions S_1, S_2, S_3, S_4 are defined respectively. When each satisfaction condition is satisfied, $(I_1 \models S_1) \wedge (I_2 \models S_2) \wedge (I_3 \models S_3) \wedge (I_4 \models S_4)$ then all satisfaction conditions are logically true, therefore, goal G is considered to be achieved.

3.2 Step 2: Integrating Goals with Process Warehouse

In this step, goals are integrated with a PW to later facilitate the acquisition of goal related information. We propose changes in a PW description at two levels, Schema and Data level.

Schema level: It defines the tables and segments of a table (in terms of attributes) relevant to a goal. This level captures information about the tables (dimension, fact) and attributes that are related with goals, through indicators. Our approach to capture schema level relationships is based on adding a stable structure of tables which captures information about goals, indicators, satisfaction conditions and their relationships with dimension, fact tables. Figure 2 shows the data model for stable structure of tables, which is hardcoded in our prototype (see details in Section 3.3).

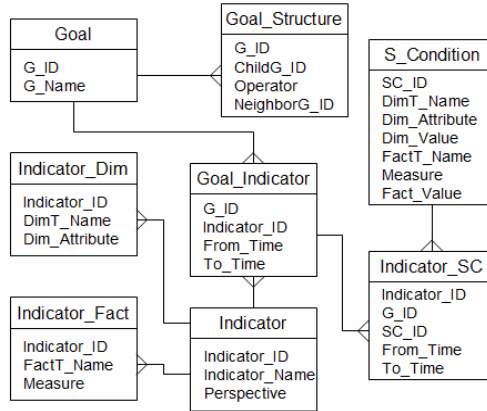


Fig. 2. Data model for stable part of PW

Data level: It defines the instances (in terms of records) of tables (dimension and fact) related to a goal. Our approach to relating records with goals is based on adding bitmaps to the dimension and fact tables that are defined relevant to the goal on the schema level i.e. we negate the addition of bitmaps to each table for each goal to avoid unnecessary increase in size of PW. The bitmap attribute can possess one value, either 1 or 0 representing relevance and non-relevance of a record, respectively. However, the relationship is defined on two levels, Schema and Data, so the bitmap value (1 or 0) does not represent the relevance or irrelevance of the complete record. Instead value 1 of a goal's bitmap represents the relevance of the record to the segment of the table defined at schema level.

Step 3: Analyze & Improve Business Process

The final step of our method is focused on utilizing PW for process analysis and improvement by identifying weaknesses that hinder goal satisfaction. This step consists of the following tasks: *condition identification*, *goal identification*, *information analysis*, *decision elicitation* and *process change solution*. Figure 3 depicts the workflow of the tasks followed by a brief description of each task, due to space limitations.

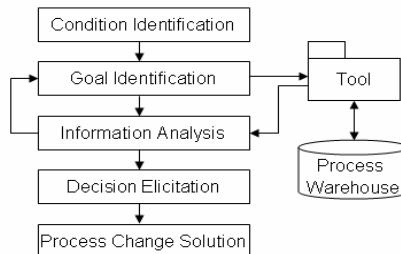


Fig. 3. Workflow for Step 3: Analyze and Improve Business Process

The *first task*, condition identification, triggers business process analysis. The reason could be either the occurrence of a process-related event or a predefined temporal state being reached. Consider a simple example, ‘registration resource purchases go over budget’ as a triggering event for process analysis.

The *second task* is to select a goal, or a set of goals from the goal structure built in Step 1 and examine their satisfaction against the real process data, stored in the warehouse. Based on the triggering condition, a goal structure is chosen. If the problem concerns only a certain set of sub-processes, or activities, i.e. not the whole process then goal analysis starts from a corresponding node in the goal tree. In case of only a single sub-goal in the tree being considered, all satisfaction conditions for all goal indicators must be satisfied for the (sub)goal to be considered satisfied. In case of more sub-goals being considered, conclusion on their satisfaction depends on the operators used (see Section 3.1. - AND, OR, SeqAND, etc.). In the example, wastage of resource is the main concern on registration sub-process. The chosen goal therefore becomes ‘efficient registration’, where ‘efficiency’ is the quality of service property related to wastage of resources.

The *third task*, information analysis, concerns the analysis of information retrieved from the warehouse. Analyzing the information to reach appropriate conclusions relies on comparing the satisfaction condition with the values (facts) acquired from the processes. If a goal is achieved, further analysis is not required. However, if a goal is not achieved due to values of facts not being equal to satisfaction condition, for each goal in the goal sub-structure the above tasks are repeated until leaf goals are reached. For the chosen goal in the example, the defined tasks need to be repeated for each activity, until the activity where resources are overused is identified.

The *fourth task* is to elicit the decisions whose realization will result in goal achievement. Input to this manual phase are both goals and the conclusions, while output is a decision taken based on one of the process perspectives. For the example, the decision becomes: reduce the usage of registration equipment in the registration activity, based on informational perspective. It is a cognitive task therefore there is a possibility that more than one decision are reached as a result of this task.

The *fifth task* is to facilitate the propagation of the obtained decision(s) to the analyzed process a pattern-based approach is used. As shown in the previous section, a decision concerns a change in the process, based on one or more of the process perspectives. As such, a decision gives a rise to a set of atomic process improvement patterns collected and outlined in Table 1.

Table 1. Process improvement patterns

Informational	Add resource type, Add resource, Remove resource type, Remove resource, Substitute resource type, Substitute resource, Substitute resource
Organizational	Add new responsibility type, Add new responsibility, Remove responsibility type, Remove responsibility, Substitute responsibility type, Substitute responsibility
Functional	Add activity/sub-process, Eliminate activity/sub-process, Replace activity/sub-process, Automate activity/ sub-process, Move activity/sub-process
Behavioral	Add condition, Remove condition, Substitute condition, Move condition, Add condition, Update condition, Sequence to parallel, Parallel to sequence add concurrency, Reorder sequence, Change Iteration.

If during process analysis several alternative goals are considered, or several alternative decisions are elicited, it is of benefit to have a mechanism to evaluate the consequences of different process change patterns. One possibility is to evaluate patterns along the four common process impact factors: *time*, *cost*, *quality*, and *flexibility* identified by Mansar et al. [22]. For an illustration, consider that the goal ‘register patient is efficient’ is not satisfied due to an overuse of nurses in the patient registration. Before choosing a process improvement pattern, from a set of alternative patterns, it is important to assess the patterns’ effect on the process. In Table 2 we have considered two pattern examples: the first, “remove responsibility”, and the second, “substitute responsibility type”. If a number of nurses are removed from the register patient activity, the time for the registration will increase and since the nurses can be used at some other place the cost of the process will decrease. However, the quality and flexibility will not be affected. To exemplify how different patterns may have different effects, we consider that another option is to substitute the nurses with registrars, who are trained for registration. Now, the activity time will decrease and the quality will increase, because registrars are more skilled than nurses for performing the activity and the activity cost will decrease due to a lower cost of the registrars compared to nurses.

Table 2. Effects of process improvement patterns

Process Improvement Patterns	Time	Cost	Quality	Flexibility
Remove responsibility	+/+	-/-	N/A	N/A
Substitute responsibility type (nurse by registrar)	-/-	-/-	+/+	N/A
+/+ Increase	+/- may increase	-/- Decrease	N/A	Not applicable

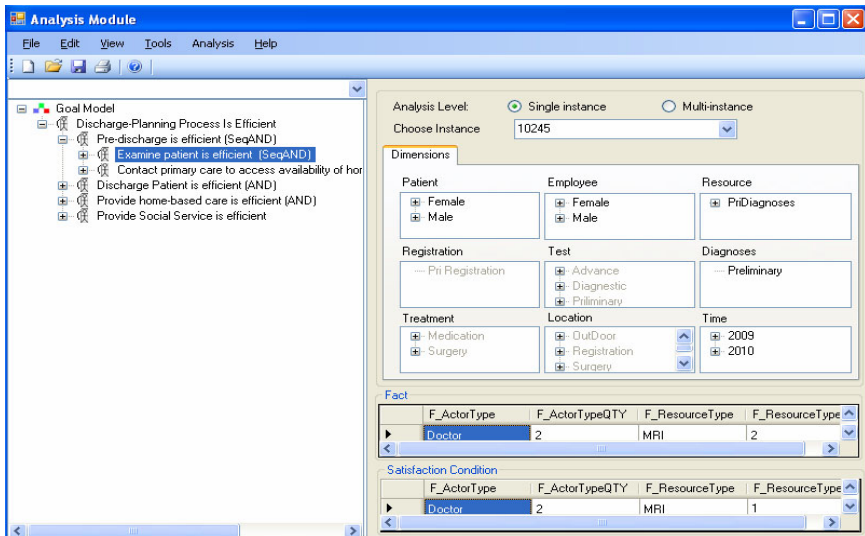


Fig. 4. Prototype of Goal Driven Process Analysis

To increase the usability of the proposed method, we have developed a prototype, as an instrument to facilitate the addition of goals, integration of goals with PW and goal analysis through a use interface for each task. Due to space limitation a screenshot of implemented prototype is shown in figure 4.

Following the choice of a goal for process analysis, both the corresponding satisfaction condition and the values are extracted from the warehouse. The major benefit of the tool is that it enables users to select goals from the tree (left pane in Figure 4), and set their desired satisfaction conditions for automatic retrieval of data from executed process instances. A user can choose to analyze a specific instance of a process or a set of instances for analysis by using the option given in the left pane.

4 Method Evaluation

In this section we briefly describe the empirical study we have conducted to investigate the impact of our proposed method on process analysis. We have adapted the evaluation model developed by Hong, et al. [12], which is based on the Technology Acceptance Model [23], the IS Success Model [24] and factors affecting data warehousing success. For this study, we have considered six out of the eight constructs of Hong et al. model (data quality, accessibility, training, perceived usefulness, perceived ease of use and perceived impact)¹ as well as all items related to the aforementioned constructs.

4.1 Research Instrument and Data Collection

For the empirical study, a PW was implemented for a healthcare process using a proprietary database management system and sample data of 100 patients were used. Based on the goals defined by Swedish Healthcare Institute [25], five goal structures were built, which were then added and linked to the warehouse using the prototype.

Based on an earlier study [26] where one of the authors had participated together with healthcare practitioners in Stockholm, Sweden, we identified eight process-related problems that trigger the need for process analysis.

Twenty university employees volunteered to participate in the study. A study mentor explained our approach in detail and demonstrated a few examples, followed by a question and answer session to address any individual concern. Participants were then asked to analyze the process for the given problems by using the approach described in section 3.3. Upon completion, participants were asked to fill a post task survey. The survey questionnaire was also adopted from Hong et al. [12] and consisted of 23 construct items (or simply items) as well as an answering scale: strongly disagree (1), simple disagree (2), weak disagree (3), not sure (4), weak agree (5), simple Agree (6) and strongly agree (7).

¹ The seventh construct, response time, has not been considered because in our method we don't aim to reduce response time, rather we aim at relevant content retrieval.

4.2 Data Analysis and Discussion

The data collected from the empirical study are analyzed to evaluate the use and perceived impact of our method. For analyzing the responses, frequency distribution analysis has been used in order to identify the distribution of responses on a scale of agree, not sure and disagree; where ‘agree’ encapsulates all those answered strongly agree, simple agree or weak agree and ‘disagree’ encapsulates all those answered strongly disagree, simple disagree or weak disagree.

Figure 5, shows that our method has overall received positive responses by a large number of participants, which indicates that they perceived our method positively. However, a significant percentage (70% and 65%) of participants disagrees with two construct items Q2 and A4. The disagreement with first item indicates that our method does not retrieve irrelevant information i.e. it retrieves relevant information, whereas with second item indicates that the task of retrieving the information is not effortless.

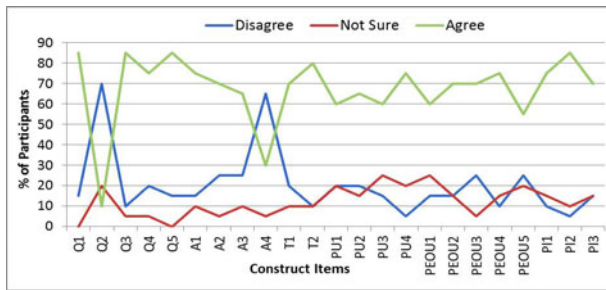


Fig. 5. Frequency distribution of construct items

Figure 6 summarizes the results of users’ perception of the method in terms of the six constructs. Values of each construct are calculated by taking a mean of the items belonging to the construct. More than 75% of the participants agree that the perceived impact is positive.

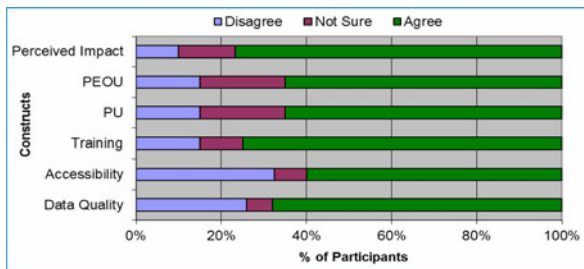


Fig. 6. Frequency distribution of constructs

To evaluate whether the participants’ background knowledge influences their perception of the method or not, we have classified participants into two groups; experienced and novice users. Users having passed a course on data warehousing and

having participated in a data warehousing project belong to the experienced group while users without any background knowledge of data warehousing belong to the novice group. Information about participants' background knowledge was collected during the study. Our sample includes 8 experienced and 12 novice users.

Figure 7 shows the difference between experienced and novice users who agree on the construct items. According to the figure, experienced users agreed with the construct items in a larger percentage than novice ones, indicating that these construct items were better perceived by experienced users than novice users.

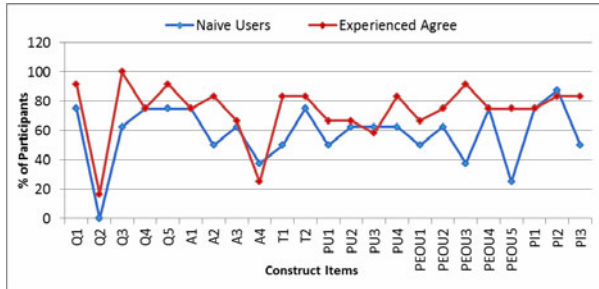


Fig. 7. Comparison of user groups on construct items

Figure 8 summarizes the results of users' perception of the method when considering the six constructs. The results show a greater difference in the perception of data quality, training and perceived ease of use than in accessibility, perceived usefulness and perceived impact.

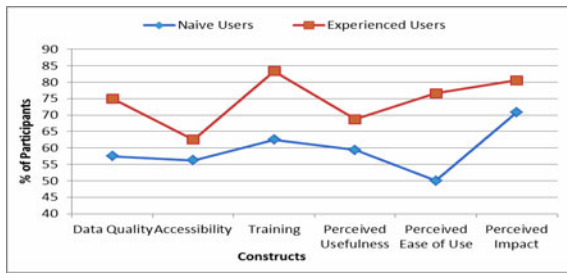


Fig. 8. Comparison of user groups on constructs

5 Conclusion

In this paper, we have proposed a three step method for goal-driven analysis of business process models aiming at eliciting possible improvement directions. In the first step a goal structure is defined for the process of interest. In the second step goals are integrated with process warehouse and in the third step goal structures are used for process analysis and improvement. To evaluate users' perceptions of our method's impact on process analysis, we have conducted an empirical study based upon an existing evaluation model.

Based on the results of the study we can conclude, a) the perceived impact of our method is positive; b) the construct items are better perceived by experienced users than novice users, c) experienced users perceived data quality, training and perceived ease of use more than accessibility, perceived usefulness and perceived impact.

Regarding future work, we are aiming for a refinement and further evaluation of the method. In addition, we intend to consider the augmentation of the automation for the proposed method steps.

References

- [1] Castellanos, M., Simitsis, A., Wilkinson, K., Dayal, U.: Automating the Loading of Business Process Data Warehouse. In: Proceedings of the 12th ACM International Conference on Extending Database Technology (EDBT 2009), Russia, pp. 612–623 (2009)
- [2] Mansmann, S., Neumuth, T., Scholl, M.H.: Multidimensional data modeling for business process analysis. In: Parent, C., Schewe, K.-D., Storey, V.C., Thalheim, B. (eds.) ER 2007. LNCS, vol. 4801, pp. 23–38. Springer, Heidelberg (2007)
- [3] List, B., Schiefer, J., Tjoa, A.M., Quirchmayr, G.: The Process Warehouse: A Data Warehouse Approach for BPM. In: Abramowicz, W., Zurada, J. (eds.) Selected Aspects of Knowledge Discovery for Business Information Systems. Kluwer Academic, USA (2000)
- [4] Casati, F., Castellanos, M., Dayal, U., Salazar, N.: A Generic Solution for Warehousing Business Process Data. In: Proceedings of the 33rd VLDB 2007, Austria, pp. 1128–1137 (2007)
- [5] Shahzad, K., Zdravkovic, J.: A goal-oriented approach for business process improvement using process warehouse data. In: Persson, A., Stirna, J. (eds.) PoEM 2009. Lecture Notes in Business Information Processing, vol. 39, pp. 84–98. Springer, Heidelberg (2009)
- [6] Muehlen, M., Shapiro, R.: Business Process Analytics. In: Brocke, J., Rosemann, M. (eds.) Handbook on Business Process Management 2: Strategic Alignment, Governance, People and Culture (International Handbooks on Information Systems). Springer, Heidelberg (2010)
- [7] Stefanov, V., List, B.: Explaining Data Warehouse Data to Business Users - A Model Based Approach to Business Metadata. In: Proceedings of the ECIS 2007, Switzerland, pp. 2062–2073 (2007)
- [8] Sayal, M., Casati, F., Dayal, U., Shan, M.C.: Business Process Cockpit. In: Proceedings of the 28th International Conf. on Very Large Databases (VLDB 2002), Hong Kong (2002)
- [9] Pourshahid, A., Amyot, D., Peyton, L., et al.: Business process management with the user requirements notation. *Electronic Commerce Research* 9(4), 269–316 (2009)
- [10] Shahzad, K., Zdravkovic, J.: Towards Goal-driven access to Process Warehouse: Integrating Goals with Process Warehouse for Business Process Analysis. To appear in the Proceedings of the Fifth IEEE International Conference on Research Challenges in Information Sciences (RCIS), Guadeloupe, France (2011)
- [11] Stefanov, V., Beate List, B.: Business Metadata for the Data Warehouse - Weaving Enterprise Goals and Multidimensional Models. In: Proceedings of the International Workshop on Models for Enterprise Computing (IWMEC) at the EDOC 2006, Hong Kong (2006)

- [12] Hong, S., Katerattanakul, P., Hong, S., Cao, Q.: Usage and Perceived Impact of Data Warehouses: A Study of Korean Financial Companies. *International Journal of Information Technology and Decision Making* 5(2), 297–315 (2006)
- [13] An, L., Yan, J., Tong, L.: Business Process Intelligence System: Architecture and Data Models. In: Proc. of the 6th Wuhan International Conference on e-business, China (2007)
- [14] Shahzad, K., Johannesson, P.: An Evaluation of Process Warehousing Approaches for Business Process Analysis. In: Proceedings of the 5th ACM-EOMAS in conjunction with CAiSE 2009. CEUR-WS Proceedings, vol. 458, pp. 1–14. ACM Press, New York
- [15] Pourshahid, A., Mussbacher, G., Amyot, D., Weiss, M.: Toward an aspect-oriented framework for business process improvement. *International Journal of Electronic Business* 8(3), 233–259 (2010)
- [16] Sarda, N.L.: Structuring Business Metadata in Data Warehouse Systems for Effective Business Support, <http://arxiv.org/abs/cs/0110020> (last access on December 19, 2010)
- [17] Shahzad, K., Zdravkovic, J.: A Decision-model Based Approach to Business Process Improvement. In: Proceedings of the IEEE-SMC 2010, Istanbul, Turkey, pp. 810–818 (2010)
- [18] Jablonski, S., Talib, R.: Agent assignment for process management: Pattern based agent performance evaluation. In: Cao, L., Gorodetsky, V., Liu, J., Weiss, G., Yu, P.S. (eds.) ADMI 2009. LNCS, vol. 5680, pp. 155–169. Springer, Heidelberg (2009)
- [19] Curtis, B., Kellner, M., Over, J.: Process Modeling. *Comm. of ACM* 35(9), 75–90 (1992)
- [20] Simon, G., Mermet, B., Fournier, D.: Goal decomposition tree: An agent model to generate a validated agent behaviour. In: Baldoni, M., Endriss, U., Omicini, A., Torroni, P. (eds.) DALT 2005. LNCS (LNAI), vol. 3904, pp. 124–140. Springer, Heidelberg (2006)
- [21] List, B., Korherr, B.: An Evaluation of Conceptual Business Process Modelling languages. In: Proceedings of the ACM Symposium on Applied Computing (SAC 2006), NY, USA, pp. 1532–1539 (2006)
- [22] Mansar, S.L., Reijers, H.A.: Best Practices in business process redesign: use and impact. *Business Process Management Journal* 13(2), 193–213 (2007)
- [23] Davis, D.F., Bagozzi, R.P., Warshaw, P.R.: User acceptance of Computer Technology: A Comparison of two Theoretical Models. *Management Science* 35(8), 982–1003 (1989)
- [24] De Lone, W.H., McLean, E.R.: Information Systems Success: The Quest for the Dependent variable. *Information Systems Research* 3(1), 60–95 (1992)
- [25] Institute of Medicine. Crossing the Quality Chasm: A New Health System for the 21st Century, <http://www.iom.edu/CMS/8089/5432.aspx> (last accessed on July 30, 2010)
- [26] Häggglund, M., Henkel, M., Zdravkovic, J., et al.: A New Approach for Goal-oriented Analysis of Healthcare Processes. *Stud Health Tech. Inform.* 160(2), 1225–1251 (2010)

Business Process Optimization Using Formalized Optimization Patterns

Florian Niedermann, Sylvia Radeschütz, and Bernhard Mitschang

Institute of Parallel and Distributed Systems, University of Stuttgart,
Universitätsstraße 38, 70569 Stuttgart, Germany
{florian.niedermann,sylvia.radeschuetz,
bernhard.mitschang}@ipvs.uni-stuttgart.de
<http://www.ipvs.uni-stuttgart.de>

Abstract. The success of most of today's businesses is tied to the efficiency and effectiveness of their core processes. Yet, two major challenges often prevent optimal processes: First, the analysis techniques applied during the optimization are inadequate and fail to include all relevant data sources. Second, the success depends on the abilities of the individual analysts to spot the right designs amongst a plethora of choices. Our deep Business Optimization Platform addresses these challenges through specialized data integration, analysis and optimization facilities. In this paper, we focus on how it uses formalized process optimization patterns for detecting and implementing process improvements.

Keywords: Business Process Optimization, Business Process Management, Process Optimization Methods, Adaptive Processes.

1 Introduction

In this section, we will first discuss the reasons for and our understanding of process optimization within *Business Process Management (BPM)* and the role of optimization techniques within this context. We will then move on to introduce our *deep Business Optimization Platform (dBOP)* and explain how it uses formalized patterns for assisting business analysts in detecting and implementing opportunities of improvement.

1.1 Business Process Management and Optimization

In the past decade, businesses have moved from tweaking individual business functions towards optimizing entire business processes. Originally, this trend - then geared towards fundamental process redesign and called Business Process Reengineering [3] - was triggered by the growing significance of Information Technology and the trend towards globalization [2]. The increasing volatility of the economic environment and competition amongst businesses has further increased its significance over the past years and also created the need for faster, often incremental process improvements.

To achieve this, nearly all companies have dedicated *Business Process Optimization (BPO)* staff tasked with both continuous optimization and running large-scale optimization projects. Technically speaking, the ultimate goal of *BPO* is the selection of the right process designs and the application of the most appropriate optimization techniques. There are a number of challenges associated with this goal: First, as business processes are cross-functional, the data pertaining to their analysis is spread over a multitude of different sources that need to be integrated. Second, finding interesting patterns in both the process data and the process itself requires specialized analysis methods that go beyond the capabilities of most process design tools. Finally, there is a plethora of different optimization techniques defined in literally hundreds of books, case studies and research papers on *BPO*. As they all are described on different levels of abstraction (with case studies on the one [3] and algorithmic approaches [13] on the other end of the spectrum) and apply to different process meta-models and application domains, selecting the right techniques is a very challenging task.

To address these challenges, an architecture is needed whose data integration and analysis capabilities go beyond the limited scope of most of today's process modeling tools and which assists its users in selecting and applying the most beneficial optimization techniques for any targeted process domain.

1.2 Deep Business Optimization Platform

In the previous section, we have listed the challenges that a platform for enhancing process performance has to address. As a possible solution we will briefly introduce in this section our *deep Business Optimization Platform (dBOP)*. The goal of the *dBOP* is to provide an integrated environment that supports semi-automated optimization during the process design, execution and analysis stages. For that purpose, the platform as shown in Fig. 1 consists of three main layers:

1. **Data Integration:** Data that is relevant to the process can be distributed across a number of relevant sources. While the most commonly used data source is audit/process execution data, other relevant data is typically contained in operational data sources (e.g., while the process data might contain the ID of the executing employee, it might not contain her trainings, work experience or similar attributes). Hence, the first layer provides an integrated view on the data relevant to the process and abstracts e.g. over different data formats used by different process engines. This is achieved by a semi-automated matching together with a custom developed reasoner that is explained in [10].
2. **Process Analytics:** In order to achieve meaningful optimization results, process specific "insights" need to be extracted from the integrated data layer. For this purpose, we leverage a range of specialized data mining techniques [6], such as (multidimensional) association rule mining or classification trees [4]. The results of the analysis are stored in the *Process Insight Repository*, which is used as the main data source for the optimization of the process. This layer also includes process matching capabilities for design-time optimization [8] and static process graph analysis methods.

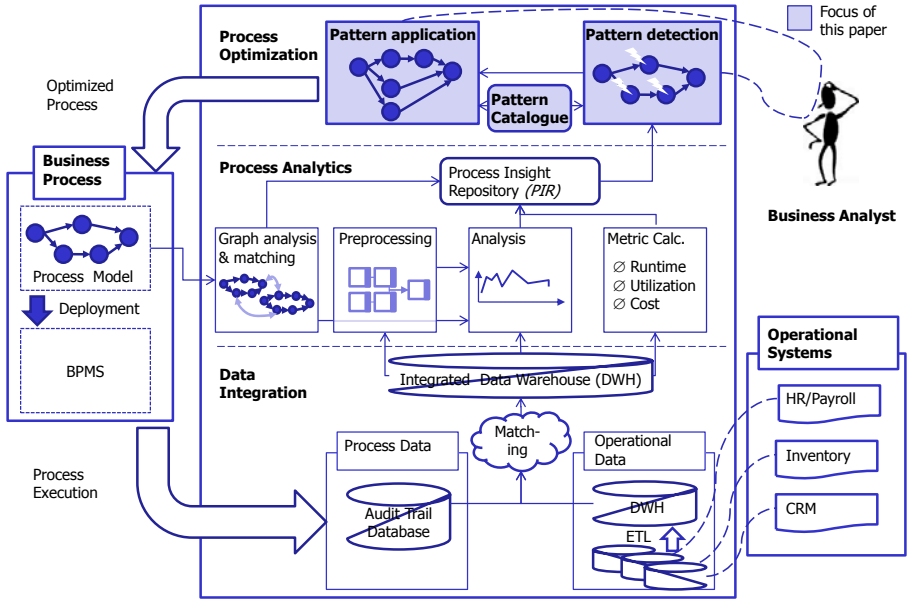


Fig. 1. *deep Business Optimization Platform Overview: Focus Optimization Layer*

3. **Process Optimization** Based on the analytics results, the actual process optimization is conducted using a catalogue of formalized optimization patterns. This step will be discussed in detail in Section 2.

As our previous work has discussed the integration [10] and analysis [9] [7] layers extensively, the remainder of this paper will focus on the optimization layer. Its specific contribution consists of explaining the basic methodology used to define and apply the optimization patterns that make up the pattern catalogue, as well as giving some simple pattern examples. In Section 2 we will explain the methods and concepts that are used to build the pattern catalogue as well as introduce the underlying process meta-model. Based on this, we will present a few sample patterns in Section 3 that are then applied in Section 4 to a small case study. In Section 5, we take a look at related work and assess how the *dBOP* approach compares to it before briefly discussing our current work and concluding the paper in Section 6.

2 Optimization Layer

While the data integration layer and analytics layer are crucial to obtain the required insights, the key layer of the *dBOP* is the optimization layer. To fulfill the *dBOP* platform goals, it has to meet a number of crucial requirements as shown in Fig. 2.

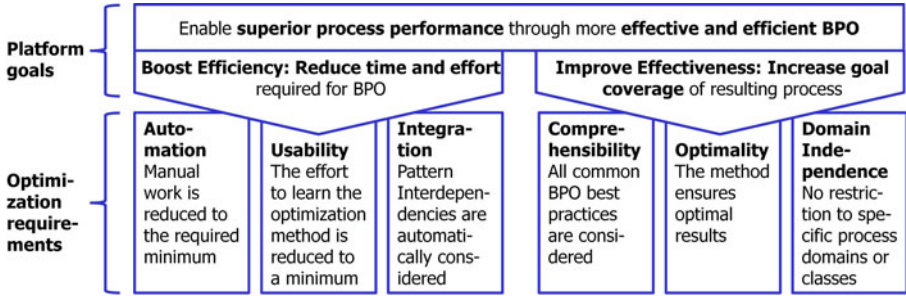


Fig. 2. Optimization Requirements

Three conceptual components are offered by the *dBOP* to meet these requirements:

1. **Optimization methodology:** A structured optimization methodology that makes sure that all requirements are considered while not neglecting usability.
2. **Process meta-model:** A graph-based process meta-model that allows for automated reasoning while being close to common modeling languages such as BPMN or UML.
3. **Pattern catalogue:** A catalog of process optimization patterns that contains formalized, extendable and domain-independent *BPO* best practices.

Each of these components will be discussed in this section in detail.

2.1 Optimization Methodology

First, we will discuss the optimization methodology. As shown in Fig 3, it consists of three steps: In the first step, the analyst has to pick the optimization goal. This is important, as there is not a single universal optimization goal, but rather a complex system of possibly conflicting goals (e.g., process cost vs. process time) together with associated constraints (e.g., utilization needs to be kept below x%).

Based on the selected goal function and possibly applicable constraints, the optimizer selects the appropriate patterns as well as the best execution order from the catalogue. Then, the optimizer "detects" instances that match the given patterns based on the graph structure and the results of the process analysis (see 6 for a detailed example). Once this has been done, the business analyst reviews the proposed changes as well as their estimated effects w.r.t. the selected goal function(s). For those instances he confirms, the process and/or organizational model is adapted accordingly through the application of the pattern.

2.2 Process Meta-Model

To meet the optimization requirements, a formal process model is required that is sufficiently powerful to allow for the required reasoning capabilities while

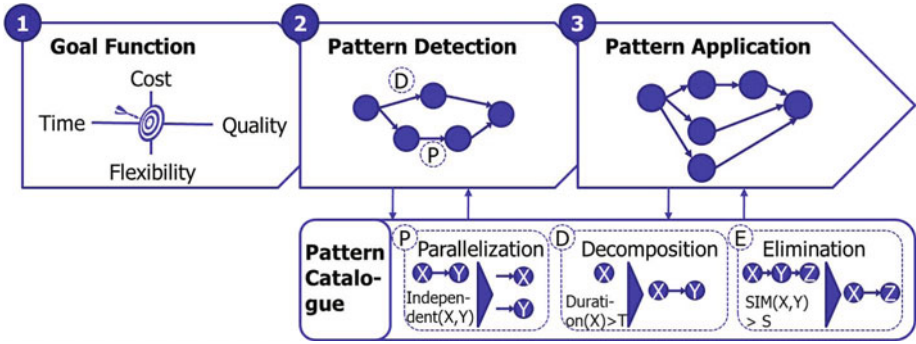


Fig. 3. *dBOP* optimization methodology

accessible enough to be employed by business analysts. We therefore use an adapted version of the graph model presented in [5], as it provides a good mix of usability (due to its proximity to, e.g., BPMN), formal reasoning capabilities and expressive power.

Definition 1. *Process Graph:* A Process Graph G is a tuple $(V, N, S, R, C, \iota, o, \rho, E_C, E_D, E_R)$, in which

1. V is the finite set of process data elements (also called variables).
2. N is the finite set of process nodes which includes the set of activities N_A , the start and the end node, the termination nodes N_T and the control nodes (XOR and AND Fork/Join) N_C
3. S is the set of node spheres. Spheres are process fragments that are bounded by Fork/Join nodes and that can be recursively nested. Spheres consist of one or many sequences. During optimization, spheres are often treated similar to activities.
4. R is the set of resources (e.g., staff members, machines) used by the process.
5. C is the finite set of conditions assigned to control connectors.
6. $\iota : N \cup C \cup \{G\} \rightarrow \wp(V)$ is the input data map, with $\wp(V)$ being the power set over V .
7. $o : N \cup \{G\} \rightarrow \wp(V)$ is the output data map.
8. $\rho : N \rightarrow \wp(R)$ is the resource usage map.
9. $E_C \subseteq N \cup N \cup C$ is the set of control connectors as explicitly modeled.
10. $E_D \subseteq N \cup N$ is the set of data connectors as denoted by dependencies between ι and o
11. $E_R \subseteq N_A \cup N_A$ is the set of resource connectors which indicates resource dependencies between activities.

Further, we define the functions $AVGDUR : N_A \cup G \rightarrow \mathbb{R}_{\geq 0}$ and $FREQ : N_A \rightarrow [0, 1]$, with the former being the average duration and the latter being the frequency of occurrence of an activity over all process instances. Additional operators for graph analysis and modification as well as functions for expressing the different analysis results will be introduced as needed in Section 3.

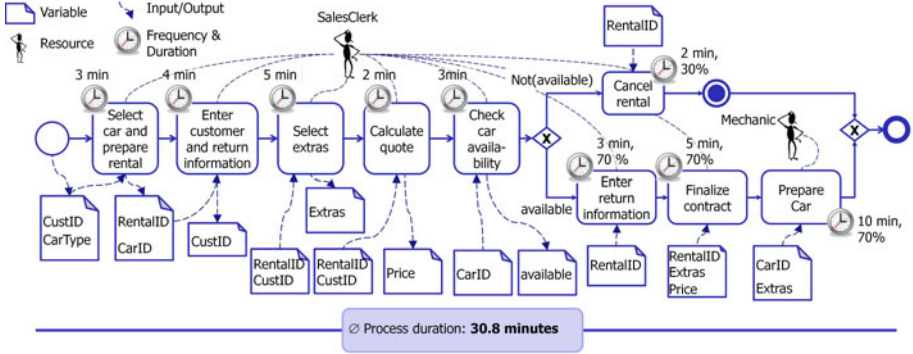


Fig. 4. Car Rental example process

An example for a simplistic car rental process modeled according to the process model is shown in Fig. 4. We will revisit this process again in our small case study in Section 4.

2.3 Pattern Catalogue

While the optimization methodology and the meta-model define the environment for the optimization, the pattern catalogue contains the optimization logic itself. It describes the relationship between patterns and the goal function, the relationships between the patterns and contains the detailed, formal detection and application logic for each pattern. The patterns are defined based on the meta-model introduced in the previous section.

The content of the catalogue was derived from two distinct sources: First, an in-depth study of *BPM* literature. Second, from interviews with several production engineers and *BPM* consultants as well as an analysis of successful *BPM* projects. As the excerpt in Fig. 5 shows, this has enabled the catalogue to cover a broad spectrum of optimization techniques, from the automation of decisions through data mining to the automated reengineering of knockout sequences.

To allow for the selection of the optimal patterns for a given scenario, the catalogue contains a wealth of meta-information about each pattern: First, each pattern is classified according to the changes achieved through its application, as shown in Fig. 5. Second, the catalogue lists whether a pattern contributes, prevents or is indifferent towards the fulfillment of each of the optimization goals (adapted from literature search and interviews - see for instance [11]). Third, it states the process stage (design, execution, analysis) during which the pattern can be applied. Fourth, it lists constraints, especially w.r.t. the execution order, that should be considered when combining this pattern with other patterns (e.g., always conduct "Activity Elimination" first). Finally, it indicates which data (just the model, process or integrated data) is required to run the pattern.

More details on the structure of the patterns can be found in the next section, where we provide the details for a few sample patterns.

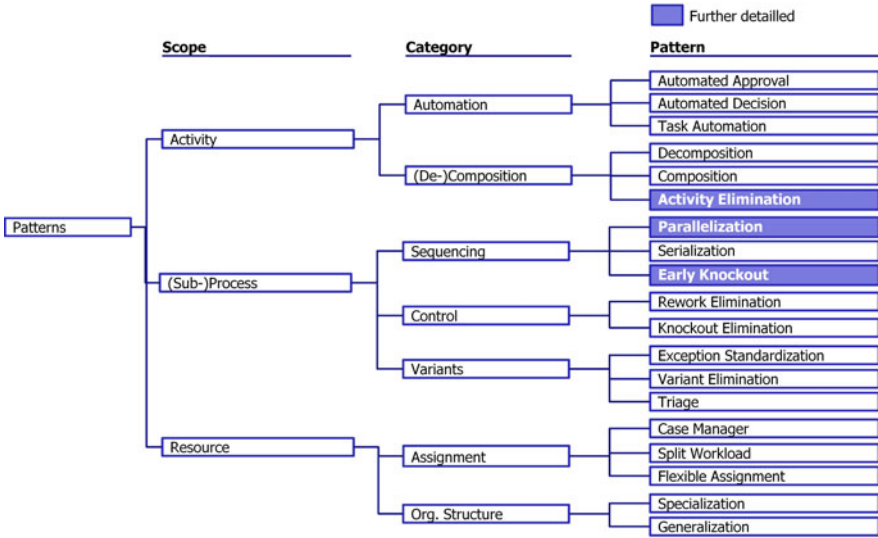


Fig. 5. Pattern Catalogue Excerpt

3 Pattern Examples

After having introduced the pattern catalogue in Section 2.3, we will now take a deeper look into two of the contained patterns. Specifically, we will look at the Activity Elimination (Section 3.1) and the strict variant of the Parallelization (Section 3.2) pattern. The template for the patterns is composed of three sections:

1. **Specification:** Lists the meta-information of the pattern that is used by the optimizer. The specification helps to ensure that the analyst employs the pattern properly, e.g., by ensuring that only appropriate patterns are considered and that they are executed in the right order (see above).
2. **Detection:** An algorithm that specifies how instances of the pattern are retrieved. It returns a set of pattern instances.
3. **Application:** An algorithm that describes the transformation logic to be applied when a pattern instance is confirmed.

Please note that due to the space constraints of this paper, we had to omit some of the details that are, however, not critical for the understanding of the respective patterns.

3.1 Activity Elimination

The core idea of the "Activity Elimination" pattern shown in Pattern 1 is that processes sometimes contain redundant activities. The elimination of these redundant activities is one of the few patterns that is beneficial (or at least does not hurt), no matter the goal function(s).

Pattern 1. Activity Elimination

Specification

Goals:	Time	↓	Cost	↓	Quality	→	Flexibility	→
Stage:	Design	✓	Execution	✗	Analysis	✓		
Data req.:	Model	✓	Process	(✓)	Operational	(✓)		

Detection**Require:** Similarity threshold $T \in [0, 1]$, Activity nodes N_A of process graph G **Ensure:** All found pattern *instances*

```

instances = {}
for all act ∈ NA do
  for all succ ∈ Successors(act, EC) do
    if SIM(act, succ) ≥ T then
      instances = instances ∪ newInstance(act, succ, SIM(act, succ))
    end if
  end for
end for
return instances

```

Application**Require:** Graph G , all *instances*, selected *instance*, *analyst* input**Ensure:** The optimized process graph G_{opt}

```

Gopt = G
if confirms(instance, analyst) then
  DeleteNode(inst.succ)
  RemoveDependents(instances, inst)
end if
return Gopt

```

The pattern is based on the notion, that activities that are highly similar are possibly redundant. Hence, this pattern uses an activity similarity function $sim_A : N_A \times N_A \rightarrow [0, 1]$ to measure the similarity of all process activities (e.g., by syntactic and semantic comparison of the activity labels and their in- and outputs, see [8]). All pairs that are above a certain similarity threshold are proposed as possible elimination candidates to the analyst. Once the analyst confirms the pattern, the redundant node is deleted and possibly dependent pattern instances (i.e., those that contain the redundant node) are removed from the instances to be reviewed.

This pattern also demonstrates why we call our approach *semi*-automated. The redundancy of an activity depends significantly on its semantics and its effects on the process context. As this information is usually not explicitly modeled, the platform can not decide based solely on the available information whether an activity actually is redundant. What it can do, however, is identify likely candidates and perform their proper elimination after confirmation from the analyst.

Pattern 2. Parallelization (Strict)**Specification**

Goals:	Time	↓	Cost	↗	Quality	↘	Flexibility	↘
Stage:	Design	✓	Execution	✗	Analysis	✓		
Data req.:	Model	✓	Process	✗	Operational	✗		

Detection

Require: Spheres S of process graph G

Ensure: All found pattern *instances*

$instances = \{\}$

for all $sphere \in S$ **do**

for all $seq \in Sequences(sphere)$ **do**

$parFragments = GetIndependentFragments(seq, E_D, E_R)$

if $|parFragments| \geq 2$ **then**

$instances = instances \cup$

$newInstance(parFragments, Predecessor(seq), Successor(seq))$

end if

end for

end for

return $instances$

Application

Require: Graph G , all *instances*, selected *instance*, *analyst* input

Ensure: The optimized process graph G_{opt}

$G_{opt} = G$

if $confirms(inst, analyst)$ **then**

$InsertNodeAfter(pred(inst), fork), InsertNodeBefore(succ(inst), join)$

for all $parFragment \in ParFragments(inst)$ **do**

$UpdateGraph(E_C, parFragment), Move(parFragment, fork, join)$

$RemoveDependents(instances, inst)$

end for

end if

return G_{opt}

3.2 Parallelization

Pattern 2 shows the strict variant of the "Parallelization" pattern. Parallelization is based on the notion that significant time savings can be realized if activities that do not depend on each other are executed in parallel [13]. However, parallelization can increase cost (if more resources are used) and reduce flexibility and process quality, as it increases complexity. When applying this pattern, different degrees of strictness can be applied, where the lenient version only considers data dependencies and the strict version also considers resource dependencies (in the pattern, this is implemented by different versions of the `GetIndependentFragments(seq, E_D, E_R)` function).

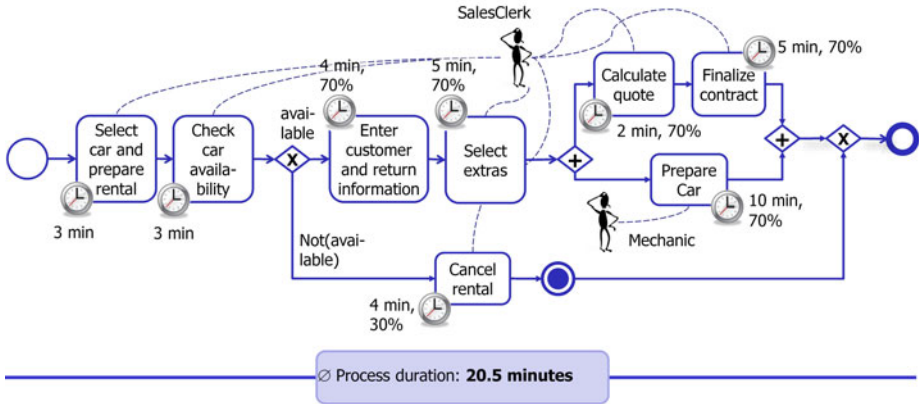


Fig. 6. Optimized Car Rental Process

To find possible applications for the pattern, the optimizer scans each of the spheres of the process for sequences that might be parallelizable - i.e., where resource and data flow allow for parallelization of the control flow. These sequences are then presented to the analyst. If an instance is confirmed, the sequences are split into parallelizable fragments (using the `ParFragments` function) according to their resource and data dependencies. These fragments are then executed in parallel by embedding them in an AND Fork/Join pair.

4 Case Example

In this brief case example, we will apply three patterns to the example process shown in Fig. 4 to reach our goal of reducing process time, namely the two patterns introduced in the previous section and the "Early Knockout" pattern. This pattern moves a knockout sequence $kSeq$ (a decision path including a termination node) as well as the relevant decision nodes to the earliest possible point. The earliest possible point is determined by the data dependency graph: It is right after the first node $target \in Predecessors(kSeq, E_D)$, i.e., the first node that has any relevant outputs for the knockout sequence. [13] discusses this.

As the optimized process in Fig. 6 shows, the application of these patterns was successful in reducing the average process duration by 10.3 minutes. This is achieved as follows:

1. **Activity Elimination:** The only match with a significant similarity according to the measures of [8] is composed of the nodes *Enter customer and return information* and *Enter return information*. The elimination of the latter yields an improvement of **2.1 minutes**.
2. **Early Knockout:** The knockout which is enacted if the car is not available can be executed right after the activity *Select car and prepare rental*, as it only needs the CarID and RentalID as inputs. Moving the knockout accordingly creates savings of **3.3 minutes**, as some activities only need to be executed in the success case.

	Algor- ithmic	Pattern- based	Best Practice	Comments
Automation				Patterns require analyst confirmation to make sure that their findings apply (e.g., due to context)
Usability				Algorithmic approaches often require a strong formal background, while best practices offer little concrete guidance
Integration				Both algorithmic and best practice approaches usually focus on single techniques, without considering interdependencies
Compre- hensibility				There are some, esp. context-heavy or strategic, best practices that can't be matched to patterns
Optimality				Patterns are heuristics which are likely to yield good results, however, not always optimality
Domain In- dependence				Algorithmic approaches have strict requirements for the process domain and the underlying formalism

Fig. 7. Assessment of the *dBOP* Optimization Approach

- 3. Parallelization:** The activities *Calculate quote* and *Finalize contract* on the one hand and the activity *Prepare Car* on the other hand are both resource- and data-independent and hence can be parallelized. This creates savings of **4.9 minutes**, as *Prepare Car* now no longer needs to wait for the other activities to complete.

5 Related Work and Evaluation

This paper is part of our work on the *dBOP* platform. Prior work of the authors [7] [8] gives both an overview of the platform and provides a broader review of related work. The integration and analysis layers are the subject of [10], [9] and [6]. Overall, the approach of a system that automatically adapts according to a set of rules and feedback from its execution can be conceptually seen as an application of cybernetics [14] to *BPM*. The workflow controlling framework discussed in [15] and the process analysis approach of [1] are somewhat similar to our platform in that they use custom analysis tools to gain process insights. However, their integration capabilities are limited and they lack an (automated) optimization layer, leaving the optimization completely to the analyst.

As mentioned in Section 2, especially the pattern catalogue relies heavily on existing *BPO* literature such as [2], [3] and [12]. A particular important source for our work are existing surveys on *BPO* techniques such as [11] and research into particular optimization techniques like [13]. We have used these and other papers as both a source of additional patterns and leveraged some of their findings, e.g., on the different optimization goals.

In order to position the *dBOP* amongst the existing literature, we have qualitatively evaluated the different approaches against the requirements we defined in Fig 2. As Fig 7 shows, the algorithmic approaches found in Computer Science are typically proven to be "optimal" (with regards to some criterion) and

run fully automated, however, they are often restricted to narrow process domains. Further, the algorithms typically can not be readily combined. The case study/best practice approach in Business literature on the other hand is typically broadly applicable, however, offers only very general advice and leaves most of the work to the analyst. Our own approach provides an integrated set of formalized best practices that can be applied to arbitrary business processes and, while not fully automated, provide a great deal of support to the business analyst.

6 Conclusion and Outlook

In this paper, we have presented a platform as well as a methodology for the use of formalized process optimization patterns embedded in an integrated analytics environment. Our pattern-based approach is positioned in the middle of most of the work on *BPO* in Computer Science on the one and Business literature on the other hand in that it offers both a considerable degree of automation while being largely domain-independent. This makes it well-suited for increasing the efficiency and effectiveness of *BPO* through assisting business analysts with detecting and implementing opportunities of improvement.

As the core parts of the *deep Business Optimization Platform*, i.e., the integration, analysis and optimization components as introduced in Section 4.2, have largely been successfully implemented, our current focus is on broadening its scope and showing its usefulness in "real world" application scenarios. For this purpose, we are currently cooperating with several manufacturing and service companies on the application of the *dBOP* to their processes. Further, we are currently conducting a user study for demonstrating the advantages of the different *dBOP* components over other approaches.

References

1. Castellanos, M., Casati, F., Dayal, U., Shan, M.C.: A comprehensive and automated approach to intelligent business processes execution analysis. *Distributed and Parallel Databases* 16(3), 239–273 (2004)
2. Champy, J.: *Reengineering Management*. HarperCollins, New York (1995)
3. Hammer, M., Champy, J.: *Reengineering the corporation: a manifesto for business revolution*. Brealey, London (1993)
4. Han, J., Kamber, M.: *Data mining: concepts and techniques*. Morgan Kaufmann, San Francisco (2006)
5. Leyman, F., Roller, D.: *Production Workflow*. Prentice-Hall, Englewood Cliffs (2000)
6. Niedermann, F., Maier, B., Radeschütz, S., Schwarz, H., Mitschang, B.: Automated process decision making based on integrated source data. In: *Proceedings BIS 2011* (2011)
7. Niedermann, F., Radeschütz, S., Mitschang, B.: Deep business optimization: A platform for automated process optimization. In: *Proceedings of the 3rd International Conference on Business Process and Services Computing* (2010)

8. Niedermann, F., Radeschütz, S., Mitschang, B.: Design-time process optimization through optimization patterns and process model matching. In: Proceedings of the 12th IEEE Conference on Commerce and Enterprise Computing (2010)
9. Radeschütz, S., Mitschang, B.: Extended analysis techniques for a comprehensive business process optimization. In: Proceedings KMIS (2009)
10. Radeschütz, S., Niedermann, F., Bischoff, W.: Biaeditor - matching process and operational data for a business impact analysis. In: Proceedings EDBT (2010)
11. 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)
12. Sharp, A., McDermott, P.: *Workflow Modeling: Tools for Process Improvement and Applications Development*. Artech House Publishers, Boston (2008)
13. Van der Aalst, W.M.P.: Re-engineering knock-out processes. *Decision Support Systems* 30(4), 451–468 (2001)
14. Wiener, N.: *Cybernetics: Control and communication in the animal and the machine*. MIT Press, Cambridge (1948)
15. zur Mühlen, M.: *Workflow-based process controlling: foundation, design, and application of workflow-driven process information systems*. Logos Verlag (2004)

Facilitating Business Process Improvement through Personalized Recommendation

Mukhammad Andri Setiawan, Shazia Sadiq, and Ryan Kirkman

School of ITEE, The University of Queensland, 78 Staff House Road,
The University of Queensland, Brisbane, QLD. 4072. Australia
{andri,shazia}@itee.uq.edu.au,
ryan.kirkman@uqconnect.uq.edu.au

Abstract. Process improvement continues to be a high priority for organizations and hence organizations continually invest in expert advice and reference models. However, a valuable and often overlooked source of successful practice is the experiences and knowledge of individuals who perform various activities within the business process. In this paper, we promote the use of this existing knowledge to assist process users and designers towards process improvement. There are two main challenges in this regard: The identification of the so-called best practice and its recommendation to an individual user in such a way that it fits the individual user's current level of experience. In order to address the above challenges we take a design science approach towards the development of methods for multi-criteria based process ranking and personalized recommendation. The methods are evaluated using a real scenario and simulated data. Results of the analysis experiments indicate that the developed methods can assist in facilitating effective learning mechanisms within the organization's user base that may subsequently lead to process improvements.

Keywords: Business Process Improvement, Personalized Recommendation, Organizational Learning, Business Process Analysis, Multi-Criteria Decision Making.

1 Introduction

Process improvement continues to be named number one priority for organizations [1]. Improvement is typically solicited through expert advice and successful practice reference models [2]. However, a valuable and often overlooked source of successful practice is the experiences and knowledge of individuals who perform various activities within the business process which can be considered domain experts in a particular aspect of the overall operations. This knowledge constitutes the corporate skill base and is considered a valuable information resource towards fundamental step in achieving competitiveness [3].

To remain competitive, business processes often face a dynamic environment which forces them to have the characteristic of ad-hocism in order to tailor

circumstances of individual process cases or instances. This creates business process variants [4], that is, the same process may have different approaches to achieve the same goals. The variants include the creativity and individualism of the knowledge worker, which is generally only tacitly available. Each variant has different time taken, different task set and/or sequence and different cost, and consequently a different level of perceived success in achieving the goal.

Although variance is both necessary and desirable, it does not always lead to the most efficient practices within an organization. In fact, the higher the allowable variance within a process, the more a (inexperienced) user may struggle to find the best approach to address a particular case. These users are required to have deep knowledge of the process they are working on if they are to be successful [5].

In this paper we will present an approach that is intended to assist such users and promote intrinsic process improvement to facilitate a change and an improvement by learning from already existing successful practices. The approach is intended to provide assistance to users that allow them to learn from the best process variants as done by previous (arguably experienced) users. Rather than forcing users to make design decisions to handle particular cases, we promote the use of existing knowledge in the Business Process Management (BPM) system to allow users to address instance specific requirements in best possible way. We believe that such an approach will guide the future user to gain the benefit of both user perspective as well as organisational perspective.

In our research, we use process knowledge as the source of best practice. The key challenge in this regard is the identification of the so called *best variants* from the potentially large record of variant models and executions which later will be used as the user performance classifier. This identification is fundamentally dependent on the criteria that define *best*. These criteria are generally many and relate to different aspects of the variant. These could include criteria such as cost (e.g. dollar value of a shipment process); time (e.g. time taken for an approval process); relevance (e.g. insurance claims higher than a certain value); popularity (e.g. the frequency of a particular set of field tests in a complaints response process) and so on.

A further challenge in the recommendation is what so called *best* means to individual users of the processes. Such a recommendation may not be entirely practical as an organization's employee base is bound to have a spectrum of experiences ranging from novice users to experts. A recommendation that does not fit an individual user's current level of experience may be counterproductive.

In order to address the above challenges we take a design science approach towards the development of methods for multi-criteria based process ranking and personalized recommendation. The development of the methods is underpinned by two main areas of study: multi-criteria decision making and personalized learning.

The rest of the paper is organized as follows. In section 2 we present some related works to support the user recommendation in business process analysis through MCDM methods. In section 3, we introduce the basic approaches relating to process ranking and user recommendation. In section 4 we present an experimental evaluation of the approach. Finally, in section 5 we draw conclusions, and discuss limitations and future works.

2 Related Works

Today, many organisations have implemented Business Process Management System (BPMS) in managing, monitoring, controlling, analysing and optimizing their business process [6]. BPMS allows organisations to design business process models, execute process instances in accordance with the models, enable users/applications to access task lists and execute task operations [7]. The system is working for the whole life-cycle of business processes and meant to implement business strategies by modelling, developing, deploying, and managing business process so that organisations can have the benefit of innovation and optimization. The implementation of BPMS is expected to increase the quality of the business process which is fundamental to the organisation's performance and competitiveness. An important step in the BPM cycle and BPMS functionality is the analysis of post-execution logs [8]. Business process analysis is a mature field [6] with a broad meaning [9] which has a range of different scenarios and strategies implemented such as simulation and diagnosis, validation, and performance analysis of business processes [10]. Several works regarding this activity have been conducted e.g. *Observational Analysis* which inspects the business process based on the diagrams provided [11] with some specific analysis such as *Business Process Redesign* [12, 13], *Process Validation* to check on how the system behaves in conforming to a particular context [10], *Process Verification* to check whether the business process model is free of logical errors [10], and *Performance Analysis* to evaluate the ability to meet the requirements regarding the resource utilization, throughput times, etc [10].

In implementing the performance analysis, providing a ranking of completed business process instances against a given set of criteria is a desirable function as it able to show how good the performance of a process instance was. The ranking given can then be used in many applications e.g. *Business Process Redesign* [13]. In order to perform such a ranking, often multiple conflicting criteria that characterize the process instances are involved e.g. throughput time, process variant, popularity, currency, etc. A well known approach to measure the performance of an instance in a given set of alternatives is Multi-criteria decision making (MCDM). MCDM is an approach which provides an effective framework for alternatives comparison based on the evaluation of multiple conflict criteria [14, 15] such as the one described for ranking process instances. By definition, MCDM is a process to make choice from a countable set of countable or uncountable alternatives using 2 or more criteria [15]. Traditionally this technique ranks and selects alternatives by their composite values or scores in a ratio scale. Most approaches in MCDM involve two basic stages; (1) scale the values of all criteria to make them comparable; (2) rank ordering of the decision alternatives accordingly.

One of the most widely used MCDM approaches is Simple Additive Weighting (SAW) [16], also called "vector-maximum" problem [17]. The concept of SAW, also known as weighted sum method is to acquire the weighted sum of performance ratings of each alternative (instance in our case) under all attributes (or criteria) [15, 18]. In most problems, SAW has shown an acceptable result and has a large following as it is easy to understand and implement [15]. Other than the SAW method, there is another method called Weighted Product (WP) which also widely used for MCDM problems [16]. The weighted product method uses multiplication for connecting

attribute ratings, each of which is raised to the power of the corresponding attribute weight [19]. This multiplication process has the same effect as the normalization process for handling different measurement units. The logic of WP is to penalize alternatives with poor attribute values more heavily [20]. We will utilize the above two methods in our approach to rank and subsequently recommend a scenario based on past business process instances against defined criteria. The best past process instance which is selected using the MCDM method as the best alternative is expected to contain experiences and knowledge on best to handle the process instance. Users are then able to learn the tacit knowledge from those best instances as explained later. Despite the goodness of the best instances (as the best practices) given by the ranking mechanism, it is often that users who perform the business activities have different backgrounds, and different level of experience/knowledge. Expecting all users to perform the defined best practice in business process activities may prove to be impractical and demoralizing for some users. Studies have shown that a recommendation based on the most relevant conditions and information for an individual user (learner) is preferred [21] and potentially more productive. This scenario is commonly found in many domains e.g. e-commerce applications [22]; e-learning environments [23]; and most commonly in sports environments where the level of the athlete's training and skill determines which competition to aspire for. As an individual gains experience in the given domain, expectations towards excellence can also be raised. Thus recommendations should be delivered to users not only based on their preferences or based on highest level of achievement, but also designed by the system to match the current performance of users, or so called personalized recommendations. Personalized recommendation has been well studied particularly in the area of e-learning. Some studies have suggested a learning process along with the measurement of learners' performance [24] and matched to a certain situations [22] could achieve a better learning result.

Process improvement is a mature field [25], with many methodologies [26]. One of the dominant approaches is through the use of process reference models define as best practices which are often embedded in software solutions [2]. Others have also attempted to use the knowledge owned by individual domain experts to achieve process improvement [2], [25]. However, to the best of our knowledge, none of the methodologies consider user's initial capability as the base information before recommending the *best practice*. The use of techniques from personalized learning to facilitate is the aim of this approach. This approach works by suggesting users to learn and use the best process at a level immediately above their current performance level. An evaluation level of Bloom's taxonomy in the cognitive aspect is used to differentiate the performance level [27]. This evaluation level of *user performance* is important in our approach as we argue that users will only learn the best scenario of completing a business process instance from the most suitable recommendation scenario, which is doable and reachable by the user.

3 Approach

Analysing and/or monitoring a given process against criteria such as time or cost are widely available through business process analysis tools. However there can be a

number of criteria that characterize the processes, and are in turn used for process improvement as the ultimate goal of the user recommendation in the business process analysis. In general, improving business process will usually have goals, such as reducing costs, improving productivity, improving competitiveness, and reducing service or production time [28].

In an application wherein there is some degree of flexibility in execution, a process user will endeavour to learn from previous precedents, but may struggle to identify the most suitable or efficient precedent. Identification of the relevant decision criteria is thus fundamental to promote knowledge sharing and transfer. In addition, the criteria need to be measurable or quantifiable. Accordingly, in this paper we identify a set of general criteria consisting of *weight* and *popularity*. The criteria are not exhaustive and may be extended, but are utilized in this paper to maintain a manageable scope and convey the workings of the proposed approach. The efficiency (with respect to throughput time) and cost (with respect to resources utilized) is collectively calculated as *weight*. The popularity of the process model shows how many times a particular process (*variant*) model has been selected by user/used previously. It may also be necessary to conduct process matching on structural similarity in the presence of variants. This aspect is not the focus of this paper and instead we rely on existing methods, such as that given by [29] to identify the various (groups of) variant models discovered. The input of criteria is collected and analysed from the execution log of process activities where it shows the throughput time of process instances, and popularity of variants as well as the cost to be borne.

With conflicting criteria, MCDM approach is a suitable solution to rank the alternatives. As described previously, a number of MCDM methods are available to rank set of alternatives. In fact choosing the appropriate MCDM method is somewhat of a challenge in itself [15]. Decision makers are faced with different methods which unfortunately may have different result in the ranking outcome, especially when alternatives given are at the similar level of performance, as different MCDM methods will weigh the criteria with different performance ratings.

To increase the level of confidence for the decision makers in choosing the suitable MCDM methods, we synthesise two well-known MCDM methods to create a consensus between the MCDM methods. The average rank as described by [15] is used to achieve the consensus. The average score computed from the two MCDM methods is then utilized to formulate a personalized recommendation of best process instances. In the context of business process analysis the alternatives are assumed to be process instances; and the criteria are characteristics which define process instances' behaviour. Ranked alternatives as described in previous work [30] are then used as the source of learning from within approach.

Based on the selected criteria and ranking method, the best precedent business processes are obtained. These are in turn used as the baseline scenario to determine user's experience level and support for users when they perform operations within business process activities. The key idea is supporting users in improving their performance by having a recommendation which has the most appropriate contents and learning paths adapted to the users' current performance and promoting learning among peers.

User performance level gives a portrayal of how workers do/complete tasks and activities. A user profile collected from the execution log is used to determine the user

performance level. There are many different ways to conduct user profiling but essentially it classifies users based on the Bloom's taxonomy into three different levels, *novice*, *advanced*, and *expert*. Profiling method development is beyond the scope of this paper, but some examples can be shown e.g. summary of throughput time of user in completing each activity.

A process instance is called an *expert level qualifier* if it has the highest rank e.g. the lowest average rank score on MCDM synthesis approach. Subsequently we also define an *advanced level qualifier* instance if it's on the median rank from the set of the alternatives, and lastly a *novice level* where it falls behind the *advanced* instance. A scale of performance is introduced as shown in Fig. 1 where it shows set of alternatives (the business process instances) that are ranked based on the MCDM synthesis approach. These set of alternatives will later be used as the recommended instance which will work as the guidance for users to perform their activities.

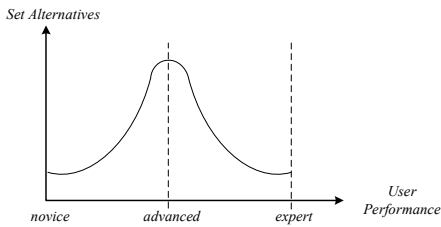


Fig. 1. Current Performance level

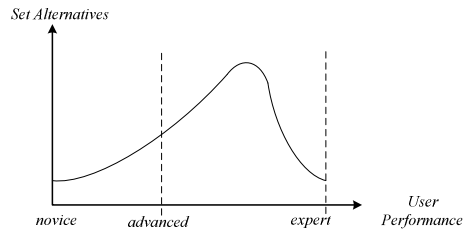


Fig. 2. Expected Future Users Performance level

The *expert* process instance is the ultimate goal on how a business process instance should be performed by user. While it is useful for an organisation if all users are following the guidance of an expert instance, but for novice users, an expert process instance could be beyond their current capability. Thus, a mid level called advanced level has the position as the bridge before user able to reach the expert level.

An example target in the future is shown in Fig. 2 which explains how in the future, after user recommendation is performed by users for a while, the overall users' performance levels can be improved, where more users are already shifting from novice level to advanced level (and eventually expert level). This is the main goal of the proposed approach, which is to eventually train all process users to perform at the highest level of efficiency by providing precedents of work practice that encourage achievable improvements, peer learning and healthy competition.

4 Experimental Evaluation

In this section we study the working of the proposed approach in detail and draw conclusions on its efficacy and feasibility. An example of a business process in use at a real business will be examined. The business process used as an example will be the bid tendering and completion process of a building services consultancy in Cairns, Australia. A building services consultancy usually deals with organisations looking to build new buildings or developments, e.g. a property developer, or organisations looking to retrofit existing buildings with new electrical, air-conditioning and

communication infrastructure. In the latter case, these projects are often local schools buildings that need to be upgraded to cope with additional demand caused by increased enrolment and increased computer usage.

The general overview of the process that will be used as an example follows the lifecycle of a “*bid*”. A bid represents a submission by the building services consultancy to an organisation looking for a contractor to take care of electrical and mechanical design work. This submission details what services will be rendered, and in what way.

First, the opportunity to submit a bid must be identified. This opportunity must be approved by management. If this is successful, the bid document must be drafted and submitted to the company that requested tenders. If the bid is successful, the work detailed in the bid must be completed. This work is then subjected to internal quality assurance mechanisms before it is released to the client. The final step in the process is to collect payment from the client.

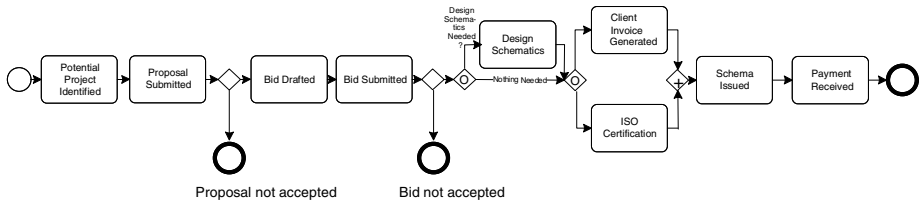


Fig. 3. Bid Tendering Process

On the bid tendering process above, process variants might exist e.g. whether design schema is needed or not (schema is needed if the similar works were never done before). The design schematics activities also can be broken down into few sub activities such as design general electrical schematics, design air-conditioning schematics, design fire protection relay circuit, and design acoustic schematics. The design schematics sub activities were recorded separately from the main execution log, and in the main execution log, all design schematics were recorded as one aggregated design schematics activity. There also exist parallel processes done while completing the process *Client Invoice Generated* and *ISO Certification*. Based on Fig. 3 above, an execution log of business process activities is collected. The result of the execution log shows some general data, overall process, and distributions of throughput time of each activity. Based on this initial data distribution on the real case study, a simulation tool was built to simulate the whole business process.

In this research, we have generated execution log of business processes through our simulation tool which generates 10,000 process instances. From those 10,000 process instances, we collected 6,471 completed processes including variants that may exist in completing the process. 5 performers were recorded as users who did all activities in completing each process instance (these are anonymously identified as performers A-E). From those past process instances, we then ranked all processes based using MCDM approaches, simple additive weight method and weighted product method, then synthesized as shown in Table 1.

Table 1. Average method to Synthesize MCDM Rank on Two Methods (SAW and WP)

Process ID	Rank on SAW	Rank on WP	Average
5272	1	1	1
4765	2	2	2
⋮	⋮	⋮	⋮
5826	24	24	24
8570	26	25	25.5
⋮	⋮	⋮	⋮
790	3253	3117	3185
5423	3254	3120	3187
⋮	⋮	⋮	⋮

Table 1 shows that some processes are on different rank according to the result of MCDM query with SAW and WP methods e.g. process 8570 ranked 26 using SAW method, but ranked 25 using WP method; process 5649 ranked 3255 on SAW method and ranked 3121 on WP method. Synthesizing both methods resulting process number 8570 is better than process number 6083. While on the top of the rank, both methods did not show a different result, on the subsequent records, we see that for the same processes' id, they show different rank, as seen on process number 8570, 6083, 3259, 790, etc.

To measure user's performance (and classify users), we summarize the average time spent by each user on each activity and aggregate them. The measurement is especially focused on three activities, Design Schematic, Get ISO Certification, and Schematic Issued. Based on the real case experience, these three activities were activities which influence a lot of users' performance. The throughput time criterion is selected as the user profile basis as this criterion directly linked to user performance, while other criteria e.g. process popularity, onetime cost, etc are independent criteria to users. Time aggregation of user's average time spent on activities is closest method to be compared to the best process instance performance.

Table 2. Time performance of best process and median best process to complete a process instance. (Note: In the parallel process, the time spent consumption is calculated based on the process with higher time spent).

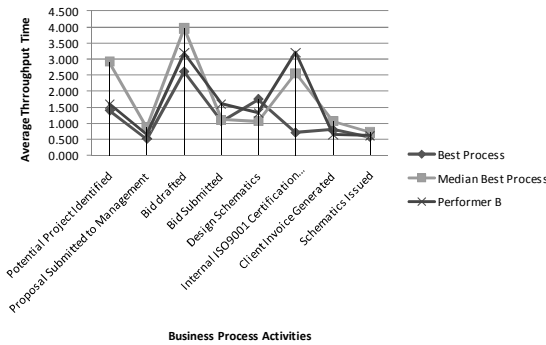
Activity	Best Process Instance Throughput Time (Process ID: 5722)	Median Best Process Instance Throughput Time (Process ID: 5649)
Schematic Design	1.748	1.427
Internal ISO9001 Certification Requested	0.704	2.560
Schematics Issued	0.578	0.614

In Table 2, we have selected two *best past process instances* as classifiers for users' performance that is *novice*, *advanced*, or *expert*. Performer who spent the aggregate average throughput time more than the median threshold is categorized as novice, for those who did better than median threshold is categorized as advanced,

and for users spent the least aggregate time and even better than the best process instance is categorized as expert. Please also note that in some business processes, some process instances shows no schematic design activity log. The rationale is that on some bid tendering process, the company has already had the same schematic from some similar projects, thus schematic design activity is no longer needed.

Fig. 4 shows an example of the work practice and performance of Performer B. Note that the overall performance (in terms of time to complete) of best process instance is better than the average time done by the performer *B*, except in the schematic design and client invoice generated activities (note that some activities has higher throughput time as it is caused by the contribution of some performers who did not perform very well on the activities). At the same time, the performer *B* surpassed the median best process. A conclusion derived from Fig. 4 is that the performer *B* is already on advanced level, thus this user only needs some little extra effort to become an expert user.

A best past precedence of business process instance is giving the idea on how to perform well in completing the process instance. For particular users who already surpassed the recommended process on each activity, the task is to maintain and/or become peers tutor to others who still underperformed. For underperformed users, the task is to improve their current performance based on the closest performance of *best process instance* for them e.g. performer with very low performance level (*novice level*) gets a recommendation which recommend the user to achieve *advance level*.



Activities	Best Instance	Performer B
Potential Project Identified	1.396	1.598
Proposal Submitted to Management	0.517	0.649
Bid drafted	2.602	3.199
Bid Submitted	1.070	1.602
Schematic Design	1.748	1.325
Internal ISO9001 Certification Requested	0.704	3.202
Client Invoice Generated	0.804	0.650
Schematics Issued	0.578	0.620
Payment Received	1.396	1.892

Fig. 4. Overall performance comparison of Best Process vs Performer B (the lesser the time needed, the better the performance is)

Based on the analysis done on the simulation’s result, we found that among 5 performers who work in the company, the performer *E* is the best performer and stated as *expert* user, the performer *B* is considered as *advanced* level, and the rests are on *novice* level. A deeper investigation of the data reveals that one of the most influencing activities within the process is the Design Schematic activity.

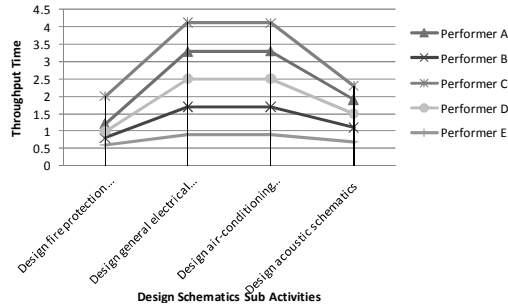


Fig. 5. Design Schematics sub activities comparison of all performers

From the chart shown in Fig. 5, in general, there are big gaps between performers on the design general electrical schematics, and the design air-conditioning schematics, but the gaps between the design fire protection relay circuit and the design acoustic schematics are quite small. This explains that some users are having some difficulties in completing process instance that involving some specific sub design schematic's activities, even though they still have the chance to perform well in other activities. Thus for these *novice level* users, high demanding and complex activities such as design general electric schematics and design air-conditioner schematics should be avoided for now. A recommendation is then given to users in novice level not to perform these difficult tasks/activities, as it will contribute in worsening the whole business process instance. The best past process instance is delivered to users to be the recommended process instance for learning with some key information communicated to users e.g. throughput time to be spent to complete a process instance and which process variants may be suitable for specific level of user.

The above is based on simulated data analysis, which was generated using the distributions from the real case. It is intended to provide a clear understanding on the working of the proposed approach as well as to present a preliminary evaluation. However, this work is not without limitation. The segregation between *novice*, *advanced*, and *expert* level is based on the performance of specific process instance that shows how a good process instance was performed. As clarified previously, a process instance consists of many activities that were performed by multiple users. While the best past process instance overall does not mean that each activity inside the corresponding instance is also the best among others. To overcome this limitation, we promote an aggregate throughput time spent on activities of users to measure the users' performance against the best past process instance to classify which user belongs to which level. Within this way, users can get the most suitable recommendation which is closest to their current level of performance.

Additionally, we anticipate that in addition to the system interaction, there is a variety of user-to-user interactions both professionally and socially which may influence the work practice and employee progress. This is indeed another interesting research question which if studied could provide further insights into how organizational learning may occur to facilitate process improvement.

5 Conclusion and Future Works

In summary, this paper has presented a method for creating recommendations from data in process execution logs regarding business process activities and instances. The recommendations came from the capitalization of previous practices and experiences within the organization and are intended to deliver the process knowledge to promote an intrinsic process improvement and change, leading to a *socialization of work practice* that is beneficial to both the individual user, as well as the organization.

We have utilized a multi criteria based ranking to provide recommendations on best practices from previous instances and provide a personalized recommendation based on the initial information of individuals' current level of experience. The approach promotes an effective learning mechanism within the organization by ensuring that personalized recommendations target each individual users performance improvement relative to their current level of experience.

Lastly, we acknowledge that the true value of the recommendations derived from the above approach can only be determined in a real-life setting. In our future work, we plan to extend the criteria in the decision-making framework. Further experiments are also needed in the future by using the extended criteria set, and larger populations of instances in order to identify and address any scalability issues in the proposed approach. We also plan to develop a prototype recommendation service and deploy it in an industrial setting to gain empirical feedback on the value of the recommendations and its impact on user progress and overall process improvement.

References

1. Gartner: Meeting the Challenge: The 2009 CIO Agenda. Egham, UK (2009)
2. Seethamraju, R., Marjanovic, O.: Role of process knowledge in business process improvement methodology: a case study. *Business Process Management Journal* 15, 920–936 (2009)
3. Kock, N.: *Business Process Improvement Through E-Collaboration: knowledge sharing through the use of virtual groups*. Idea Group Publishing, London (2005)
4. Lu, R., Sadiq, S.K.: Managing process variants as an information resource. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) *BPM 2006*. LNCS, vol. 4102, pp. 426–431. Springer, Heidelberg (2006)
5. Schonenberg, H., Weber, B., van Dongen, B.F., van der Aalst, W.M.P.: Supporting flexible processes through recommendations based on history. In: Dumas, M., Reichert, M., Shan, M.-C. (eds.) *BPM 2008*. LNCS, vol. 5240, pp. 51–66. Springer, Heidelberg (2008)
6. van der Aalst, W.M.P., ter Hofstede, A.H.M., Weske, M.: Business process management: A survey. In: van der Aalst, W.M.P., ter Hofstede, A.H.M., Weske, M. (eds.) *BPM 2003*. LNCS, vol. 2678, pp. 1–12. Springer, Heidelberg (2003)
7. Yujie, M., Shensheng, Z., Jian, C.: Supporting collaborative work with process management enhanced corporate portal. In: *IEEE International Conference on Systems, Man and Cybernetics*, vol. 5, pp. 4206–4213 (2004)
8. Biazzo, S.: Approaches to business process analysis: a review. *Business Process Management Journal* 6, 99–112 (2000)
9. Vergidis, K., Tiwari, A., Majeed, B.: Business Process Analysis and Optimization: Beyond Reengineering. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 38, 69–82 (2008)

10. van der Aalst, W.M.P.: The Application of Petri Nets to Workflow Management. *The Journal of Circuits, Systems and Computers* 8, 21–66 (1998)
11. Phalp, K., Shepperd, M.: Quantitative analysis of static models of processes. *Journal of Systems and Software* 52, 105–112 (2000)
12. Mansar, S., Reijers, H.: Best practices in business process redesign: validation of a redesign framework. *Computers in Industry* 56, 457–471 (2005)
13. Mansar, S., Reijers, H., Ounnar, F.: BPR Implementation: A Decision-Making Strategy. In: *Business Process Management Workshops*, pp. 421–431 (2006)
14. Shyur, H.-J., Shih, H.-S.: A hybrid MCDM model for strategic vendor selection. *Mathematical and Computer Modelling* 44, 749–761 (2006)
15. Hwang, C., Yoon, K.: *Multiple attribute decision making: methods and applications: a state-of-the-art survey*. Springer, Heidelberg (1981)
16. Yeh, C.H.: A Problem-based Selection of Multi-attribute Decision-making Methods. *International Transactions in Operational Research* 9, 169–181 (2002)
17. Zimmermann, H.J.: *Fuzzy set theory—and its applications*. Kluwer Academic Pub., Dordrecht (2001)
18. Fishburn, P.C.: Additive utilities with incomplete product sets: application to priorities and assignments. *Operations Research* 15, 537–542 (1967)
19. Yoon, K., Hwang, C.: *Multiple attribute decision making: an introduction*. Sage Publications, Inc., Thousand Oaks (1995)
20. Chen, S.-J.J., Hwang, C.L.: *Fuzzy multiple attribute decision making: methods and applications*. Springer-Verlag New York, Inc., Secaucus (1992)
21. Drachler, H., Hummel, H., Koper, R.: Personal recommender systems for learners in lifelong learning networks: the requirements, techniques and model. *International Journal of Learning Technology* 3, 404–423 (2008)
22. Santos, O.C., Boticario, J.G.: Modelling recommendations for lifelong learning. In: *First International Conference on the Applications of Digital Information and Web Technologies, ICADIWT 2008*, pp. 174–179 (2008)
23. Tang, T., McCalla, G.: Smart recommendation for an evolving e-learning system. *International Journal on E-learning* 4, 105–129 (2005)
24. Day, R., Payne, L.: Computer-managed instruction: an alternative teaching strategy. *J. Nurs. Educ.* 26, 30–36 (1987)
25. Harrington, H.J.: *Business Process Improvement: The Breakthrough Strategy for Total Quality, Productivity, and Competitiveness*. McGraw-Hill, Inc., New York (1991)
26. Kettinger, W.J., Teng, J.T.C., Guha, S.: Business Process Change: A Study of Methodologies, Techniques, and Tools. *MIS Quarterly* 21, 55–80 (1997)
27. Anderson, L., Krathwohl, D., Airasian, P., Cruikshank, K., Mayer, R., Pintrich, P., Raths, J., Wittrock, M.: *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives, Abridged Edition*. Allyn & Bacon, Boston (2000)
28. Mansar, S., Reijers, H., Ounnar, F.: Development of a decision-making strategy to improve the efficiency of BPR. *Expert Systems with Applications* 36, 3248–3262 (2009)
29. Lu, R., Sadiq, S.K.: On the discovery of preferred work practice through business process variants. In: Parent, C., Schewe, K.-D., Storey, V.C., Thalheim, B. (eds.) *ER 2007. LNCS*, vol. 4801, pp. 165–180. Springer, Heidelberg (2007)
30. Setiawan, M.A., Sadiq, S.: Socialization Of Work Practice Through Business Process Analysis. In: *ICEIS, Funchal, Madeira - Portugal* (2010)

System Architecture for Handling the Information Overload in Enterprise Information Aggregation Systems

Philipp Katz¹, Torsten Lunze², Marius Feldmann¹, Dirk Röhrborn²,
and Alexander Schill¹

¹ Technische Universität Dresden, Fakultät Informatik, Deutschland,
philipp.katz@tu-dresden.de

² Communardo Software GmbH, Dresden, Deutschland,
torsten.lunze@communardo.de

Abstract. Nowadays, many information workers are experiencing a serious problem: too many information sources have to be tracked and monitored for important information leading to information overload. State-of-the-art information aggregation systems have not solved this problem yet. In this paper a novel system architecture is proposed which can be applied in highly scalable information aggregation systems solving this problem. Such a system can be used to aggregate heterogeneous information sources and present the results to a user in a personalized manner. The architecture is derived from a specific information processing model described in this paper. The validity of the system architecture has been confirmed by a web-based information aggregation system realized based on this architecture. The implementation is supposed to be a framework for various novel information extraction and relation detection algorithms discussed in follow-up publications.

Keywords: Enterprise 2.0, System Architecture, Information Aggregation, Personalization.

1 Motivation

“We have directed all of our energies and intelligence to inventing machinery that does nothing but increase the supply of information. As a consequence, our defenses against information glut have broken down; our information immune system is inoperable. We don’t know how to filter it out; we don’t know how to reduce it; we don’t know to use it. We suffer from a kind of cultural AIDS.” [1]

These words from a speech given by Neil Postman about the characteristics of the information age point out a central problem, everybody involved in current information technology has already experienced: The overwhelming amount of information can – if at all – hardly be controlled, categorized and processed by human beings. Many information workers know this problem all too well. They have to check huge amounts of messages from various information sources such as Web feeds, e-mail accounts or instant messaging, they have to evaluate the

relevancy of the messages and have to organize them manually. It is evident that this task is time consuming and inefficient, making it desirable to employ system support for improving and automating this task.

Currently, the problem of a huge information load from various information sources is not covered by state-of-the-art enterprise information systems in an appropriate manner. As its central contribution, this paper proposes the architecture of a highly scalable enterprise information aggregation system which is intended to help information workers to gather relevant information efficiently. This architecture can be used in enterprise information systems intended to aggregate various heterogeneous information sources and present the retrieved information in a personalized manner to each user. The system architecture is derived from a message processing model forming a further central contribution of the following discussion. It is intended as a framework for investigating new information extraction and relation detection algorithms in future work.

The remainder of this paper is structured as follows: In Section 2, a concrete use case is provided to substantiate the identified problem from a practical perspective. Furthermore, important state-of-the-art systems are discussed and their central shortcomings are pointed out. After this preliminary discussion, the processing model building the foundations of the proposed system architecture will be introduced and discussed in detail in Section 3. Section 4 presents the announced architecture and discusses its main features. An overview of the technical realization of the proposed enterprise information system is presented in Section 5. A summary and outlook in Section 6 concludes this work.

2 Use Cases and State of the Art

In the preceding section we outlined the general motivation behind our work. We will now describe a use case illustrating a typical scenario embedded in an enterprise context. Based on the scenario we will derive the requirements for a system handling personalized information streams in an enterprise context. We will present an analysis of important existing approaches, arriving at the conclusion that no existing system is available to cover the formulated requirements.

2.1 Problem Scenario

ACME is a medium sized, fictitious software company. Information exchange takes place through various internal and external information streams, involving traditional e-mail communication, RSS¹ and Atom Web feeds, microblogging services such as Twitter and instant messaging applications such as Jabber or Skype. Typical employees of ACME can be characterized as knowledge workers, spending much of their time reading, processing, searching and organizing information and messages from various sources, identifying tasks, prioritizing and delegating those tasks.

¹ Really Simple Syndication or Rich Site Summary.

Alice works as a technical support employee for ACME and among many other messages she receives an e-mail with an urgent support request concerning one of ACME’s software products. After further investigation involving ACME’s internal Wiki system and e-mail archives, she suspects a bug in the respective product, hence she opens a new ticket in ACME’s bug tracking system containing the problem description and a quick note about the customer. Then she sends a quick notification to Bob, who is working in the development department of ACME, to underline the urgency of the problem. After Bob has read Alice’s e-mail, he connects to the bug tracker to get the detailed problem description. While working on the problem, Bob has further questions, so he exchanges several e-mails with Alice and the customer. After verifying and fixing the bug, he commits the modified source code to the Subversion repository and notifies Alice. Alice on her part contacts the customer, telling him the bug was fixed for the next product release of the software product. After a few days, manager Mary finds an open ticket with a high priority, which Bob forgot to close after solving the problem. In ACME’s internal microblogging service, Mary posts an entry to check for the open ticket. In the vast amount of microblog messages, Bob misses Mary’s entry, so she sends him another reminder via chat. After that, Bob connects to the tracking system, closes the forgotten ticket and replies to Mary’s post, stating the problem has already been solved.

2.2 Central Issues

The given scenario outlines the motivating problem of information overload which was introduced in Section 1.1 combined with the heterogeneity of different news streams in a typical enterprise context. Furthermore, it illustrates the emerging problem of so called “information silos” – closed systems, storing data, making it difficult or even impossible to interchange information between other systems combined with permanent information overflow. Given the above usage scenario we can identify the core problems which constitute the motivation behind our work: Enormous amounts of heterogeneous streams of news need to be aggregated, consumed and processed, relevant information needs to be identified and the user must be supported in fighting information overload. Given the focus on enterprise usage, aspects such as stability and scalability are core issues that have to be covered by the system.

2.3 State of the Art Approaches

There are several existing products for enterprise based information aggregation systems. Attensa offers StreamServer², a business solution for aggregating information from various sources. It allows consolidating and filtering this information and creating topic-specific information streams. Furthermore, users’ behaviors are tracked in order to create user models for delivering personalized and ranked information streams. Another product is NewsGator’s Social Sites³.

² <http://www.attensa.com/product/>

³ <http://www.newsgator.com/products/social-sites-for-sharepoint-2010.aspx>

It is an integration for Microsoft SharePoint, offering a news aggregation system with collaborative features aimed at enterprise usage.

Other efforts focus on efficiency and distribution aspects. Cobra [2] for example is an approach for aggregating and filtering vast amounts of news from RSS feeds for big user groups. Therefore it employs a highly distributed architecture consisting of different components for aggregation, filtering and delivery to users.

Many approaches which lay their emphasis on typical Internet Information Retrieval tasks such as filtering, ranking and personalizing news streams exist. Much recent work has been put into considering Web feeds. NectaRSS [3] and PersoNews [4] for example are ranking systems for RSS news which employ implicitly trained user profiles for filtering and ranking incoming news items. CoffeeReader [5] aims at creating a collaborative reading process, where social interactions such as recommendations, tagging, etc. are used to reduce the amount of information overflow, helping to point out relevant news.

2.4 PRISMA

Our analysis of existing approaches shows that in the state of the art no overall information aggregation approach for enterprise usage covering aggregation of heterogeneous sources, information extraction and personalization, combined with a distributed and scalable architecture designed for deployment in enterprise scenarios exists. Therefore, we introduce PRISMA⁴, a system addressing the central issues in enterprise news aggregation, as presented in Section 2.2.

3 Processing Model

Based on the outlined requirements, we derived a specific processing model which describes different stages of message representations throughout the proposed system. The goal of this processing model is to transform heterogeneous messages from various information sources to a homogeneous personalized visualization format. The message representations are connected to each other via core information processing steps thus resulting in a message processing workflow within the information system. The identified five stages are depicted in Figure 1. The messages in PRISMA run through the following stages:



Fig. 1. Stages of information in the PRISMA stream

⁴ PeRsonalization of Information Stream Aggregates.

Heterogeneous Message: Heterogeneous sources producing messages in varying categories and with different properties need to be covered by PRISMA. On one hand, the attribute “heterogeneous” describes different variants of accessing the information source technically. For example, the aggregated sources offer different subscription paradigms – whereas Web feeds need to be pulled in certain intervals, services such as XMPP⁵ rely on a push-based paradigm. On the other hand, the attribute “heterogeneous” implies the requirement for supporting different message protocols and formats.

Homogeneous Message: The cumulated raw messages derived from heterogeneous sources are transformed to a common representation. It serves as an internal interchange format for all further steps in the processing queue. Thus, the following processing steps do not have to differentiate between messages from different sources. This format needs to be highly generic, enabling the system to transform the heterogeneous message stream without losing vital information from the original source.

Enriched Message: Meta data expressing semantic properties is an important prerequisite for further processing steps. PRISMA aims at employing information extraction and text mining techniques to enrich the stream of messages with meta information. Considering typical message flows in an enterprise context, our information extraction approaches need to cover wide ranges of information varying in quality. As an obvious example, we need to consider variance in text lengths when employing our algorithms. Further challenges are different languages and styles between different messages. Our aim is to describe single messages with a set of keyphrases or “tags” and to identify semantic relations between different messages. Based on these semantic relations messages can be aggregated to semantic clusters named “activities”. These semantic clusters can be explored for personalization or message visualization purposes.

Personalized Message: Our goal is a filtered and ranked message stream, taking into consideration the individual interests of PRISMA’s users combined with contextual factors. Therefore, a relevance calculation has to be employed which depends on users’ profiles. Profiles are generated in implicit and explicit manners. Considering the targeted enterprise context, it is suitable to exploit relations between different users which can be modelled in hierarchial forms of existing corporate structures. For example, colleagues working in the same team are likely to have a bigger intersection of interests than users working for different departments.

Presentation: The user interface is the final destination for a message in the processing stream. Novel techniques for information extraction and personalization need to be accompanied by innovative metaphors regarding an appropriate presentation to the user tightly integrating with preexisting enterprise workflows.

The described workflow illustrates PRISMA’s transformation process, consuming messages from heterogeneous sources and performing a stepwise refinement,

⁵ eXtensible Message and Presence Protocol.

yielding in a filtered and personalized news stream. The processing model constitutes the basis for the design of PRISMA's system architecture which will be explained in the following chapter.

4 System Architecture

In the following, after introducing fundamental design goals and decisions, an overview of the necessary system components is provided and the interaction between the components is described. Furthermore, a specific focus is directed to the inter-component interaction realized by a message queue for scalability and reliability purposes.

4.1 Design Goals and Fundamental Decisions

Three fundamental design goals have been identified in order to create a solution applicable in the desired usage context:

Scalability: The architecture has to be able to scale on the number of messages, users and sources.

Reliability: The architecture has to be able to handle incoming messages in a fast and reliable way.

Flexibility: The architecture has to be able to dynamically add new processing nodes to handle message peaks and to ensure constant availability.

In order to fulfill these principles during the message processing, the components have to be able to work independently. The coupling between the components should be reduced to a minimum. A central problem arises from varying and potentially high processing times needed by the different components. This leads to a queue-based design approach, where individual components take their inputs from one queue, perform their work and write their output back to the same or a different queue. We identified the need for a central coordination instance which is responsible for monitoring the queues, controlling their message flows and starting or stopping new processing nodes on demand.

A fundamental concept used in the system is the principle of subscriptions: Users specify sources (e. g. an RSS feed) they are interested in via the user interface of the aggregation system. This specification is formalized as a subscription sent to the PRISMA system. A subscription contains information necessary for the information source access such as the address of an information source, information about the access protocol or credentials.

4.2 Component Overview

Based on the design principles and the processing model described in Section 3 the nine components which are depicted in Figure 2 have been identified:

Aggregator: The Aggregator accesses different information sources and transforms the received heterogeneous messages into a homogeneous representation (transition 1 as depicted in Figure 1). For every supported information

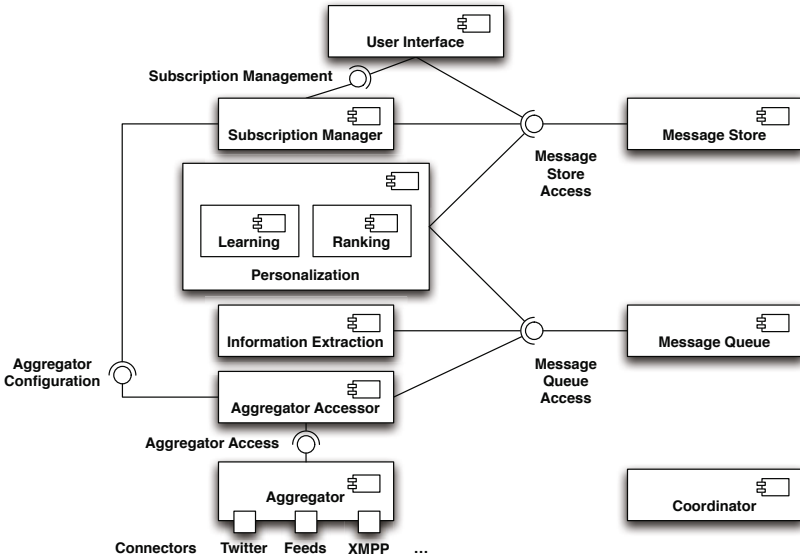


Fig. 2. Components and their interfaces of the PRISMA architecture

source a dedicated source adapter has to be offered by the Aggregator. These adapters realize the physical interaction with the information source. Each instance of the Aggregator manages a set of subscriptions. For every subscription it receives it selects the appropriate source adapter and accesses the associated information source.

Aggregator Accessor: This component is responsible for receiving messages from the Aggregator and forwarding them to a Message Queue.

Information Extraction: The Information Extraction component is responsible for enriching messages with meta information and to detect relations between messages fully automatically (transition 2 as depicted in Figure 4).

Personalization: This component determines the priority of a message for the set of users the message is intended for. Furthermore, it manages available user models. For both tasks the Ranking and Learning subcomponents are embedded into the Personalization component (transition 3 as depicted in Figure 4).

Message Queue: The Message Queue handles different message states and realizes the exchange of messages between separate components (Section 4.1).

Coordinator: The Coordinator is responsible for configuring and monitoring the other components. For example, the Coordinator will set up a Message Queue, as well as Information Extraction and Personalization components and create necessary connections for inter-component communication. Furthermore, it monitors instances of the Aggregator, Information Extraction and Personalization components and replicates them dynamically on demand.

Subscription Manager: This component accepts subscriptions to information sources specified by users and forwards them to an appropriate Aggregator via the Aggregator Accessor.

Message Store: After an incoming message has been processed and is personalized, it is stored in the Message Store, from where it can be requested by the User Interface component.

User Interface: The User Interface component is responsible for displaying retrieved messages to the end user (transition 4 as depicted in Figure 1). Furthermore, it offers means to manage the users' subscriptions and to modify the user profile data manually.

The components provide and use several interfaces. As their intentions could be easily derived from the associated components, we will not discuss them in detail here.

4.3 Distribution

As mentioned in Section 4.1, scalability is one of PRISMA's core design goals. This leads to the need of being able to replicate the introduced system components and to distribute them over various physical nodes. Thus, different processing stages of the PRISMA architecture can be parallelized. In order to facilitate the intended message processing flow, the following preconditions have to be met:

1. The Aggregator Accessor needs to know at least one Aggregator for receiving the messages.
2. The Aggregator Accessor needs to know exactly one Message Queue to store incoming messages.
3. The Message Queue must be accessed by at least one Information Extraction and at least one Personalization component.
4. All Personalization components will store their results in one shared dedicated database. For realizing a distributed database clustering or summarization concepts applied to user profiles may be exploited.

Depending on the components' distribution structure, a Message Queue has to be accessible by different nodes. Individual components can treat the queues as "black boxes" – they only need to comply with the queue interface, but do not need to know about further components connected to the queue. Technical details about the queues are managed by the Coordinator which is responsible for their initializations and for setting up connections between individual components and the queues.

A possible runtime configuration of the PRISMA architecture is depicted in Figure 3. In this example one Aggregator, one Aggregator Accessor, a Message Queue, an Information Extraction and two Personalization components are distributed to eight physical nodes.

4.4 Message Flow

The message flow of an incoming message is depicted in Figure 4. The Aggregator component checks the data sources for new messages. As soon as a message

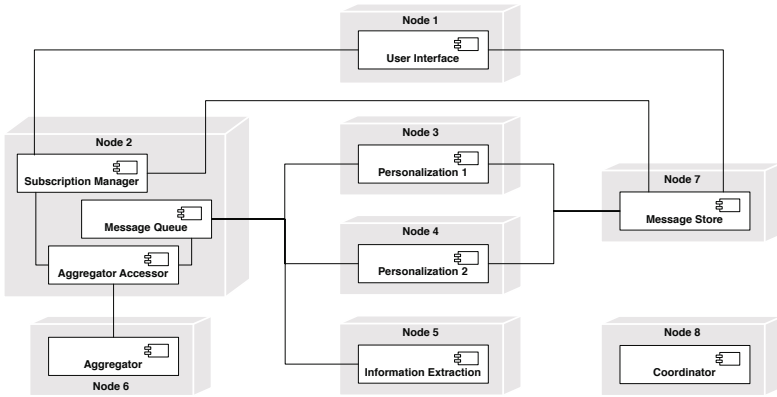


Fig. 3. Distributed components for parallel message processing

is received, it will be transformed to a homogeneous format as described in Section 3. After the message is transmitted to the Aggregator Accessor, it is stored in the Message Queue connected to the Aggregator Accessor. The Information Extraction component checks for new items in the queue, pulling a message on arrival to extract meta data and to detect semantic relations to former messages. Upon completion, the message is handed back to the queue. The Personalization component pulls the message from the queue which has been enriched by the Information Extraction. It performs a ranking algorithm considering users’ profiles. Finally, the processed message is removed from the queue and put into a message store, which is accessed by a frontend presenting the ranked result.

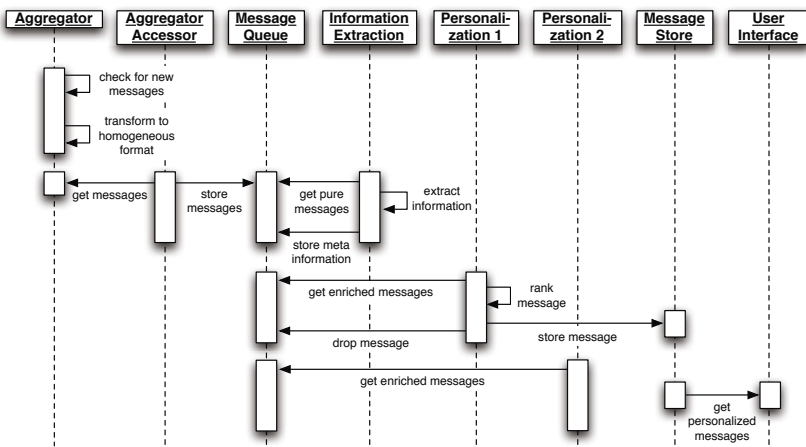


Fig. 4. Components and their interaction for handling incoming messages

For reliability purposes, it must be assured that the messages will be fully processed even if a component or a node processing the message fails. Therefore, the Message Queue provides a mechanism to lock a message once it is pulled from the queue by assigning an in-process flag. If the Coordinator detects a failing component, it can perform a rollback, recovering the messages which were in process by the failing component and hand them over to another component.

5 Implementation

For validation purposes, an enterprise information aggregation system has been implemented based on the described system architecture. The implementation constitutes the architectural framework for PRISMA based on the system architecture and the fundamental principles introduced in Section 4. The Java-based implementation has been deployed in various realistic test scenarios to validate its flexibility and scalability.

For communication purposes between remotely distributed components, the Jabber-RPC extension (XEP-0009 [6]) of XMPP is applied. Thus, an XMPP server forms a fundamental part of the different system components. In order to avoid an unnecessary overhead for deploying and managing such a server, the embedded and lightweight open source library Vysper[6] is used.

The implemented prototype allows subscriptions to heterogeneous information sources. For accessing these sources, several adapter modules for the Aggregation component have been implemented. These modules cover XMPP, RSS and Atom Web feeds as well as dedicated source adapters for external social media aggregators such as Collecta[7]. Basic implementations of the personalization and information extraction components have been realized based on state-of-the-art algorithms. For example, the core of the information extraction component is formed by a machine learning approach based on trained models. Due to the focus on the general system architecture of PRISMA, technical and algorithmic details of these components are not in the scope of this document.

For user interaction and result visualization purposes, a basic Web-based frontend has been conceived. The user interface offers various views. The main view is depicted in Figure 5. In this figure, a stream of messages aggregated from various information sources is displayed. This message stream can be filtered by several tools such as a tag cloud or by the priority which is calculated based on the current user's profile.

To detect potential bottlenecks within the system architecture, scalability tests have been applied. Besides aggregating messages from real world input sources as described above, we developed a message generator capable of generating message streams with text content of variable lengths and at arbitrary intervals. Monitoring functionalities have been developed to measure performance and throughput in PRISMA's Message Queue. We executed an initial

⁶ <http://mina.apache.org/vysper/>

⁷ <http://collecta.com/>



Fig. 5. Web-based frontend for PRISMA

performance test, using one Information Extraction and Personalization component at a time. The described test setup of the current implementation taking dummy messages with up to 50,000 characters allowed for a throughput of only 3.5 messages per second, confirming that replication and decentralization of single components are crucial aspects of PRISMA. Tests with our basic replication mechanism have proven the applicability of automatic creation of new component instances and of their deployment.

6 Summary and Outlook

In this paper a novel architecture for a flexible and scalable enterprise information system has been presented. It is intended to manage the huge information load users currently have to cope with. The architecture can be applied in systems which intend to aggregate information from various information sources and to represent the information to end users in a personalized manner. The core of the architecture is formed by a Coordinator component and a set of Message Queues the various replicable system components interact with. In order to confirm the feasibility of the architecture, it has been implemented and applied to a real enterprise information system.

Future work will focus to the systems components intended for information extraction and personalization. For both efficient and effective algorithms will be further evaluated and developed. In addition, enhanced strategies for component

replication will be investigated. Based on the experiences we have gained with a Web-based user interface, novel UI concepts are currently under exploration. The Web-based UI has central disadvantages in regards to the effectiveness a user can work with and process messages. Due to this, we are working on a frontend based on the Apple iPad which enables enhanced UI interaction possibilities. The results of our further investigations will be presented in follow-up publications. We will especially focus on details of the applied algorithms in the information extraction and personalization components and about the UI concepts.

Acknowledgments. This paper is the first in a series of publications presenting the results of the research project PRISMA which are developed in a cooperation between the Technische Universität Dresden and the Communardo Software GmbH. The PRISMA project is funded by the Free State of Saxony and the EU (European Social Fund). We would like to thank Eduardo Jacobo Miranda Ackerman and Livia Czernohorsky for their valuable feedback to this paper.

References

1. Postman, N.: Informing Ourselves to Death. Speech given at a meeting of the German Informatics Society (Gesellschaft für Informatik), Stuttgart, Germany (October 1990), http://w2.eff.org/Net_culture/Criticisms/informing_ourselves_to_death.paper (retrieved January 18, 2011)
2. Rose, I., Murty, R., Pietzuch, P., Ledlie, J., Roussopoulos, M., Welsh, M.: Cobra: Content-based Filtering and Aggregation of Blogs and RSS Feeds. In: NSDI 2007: 4th USENIX Symposium on Networked Systems Design & Implementation (2007)
3. Samper, J.J., Castillo, P.A., Araujo, L., Merelo, J.J.: NectaRSS, an intelligent RSS feed reader. *Journal of Network and Computer Applications* 31(4) (November 2008)
4. Banos, E., Katakis, I., Bassiliades, N., Tsoumakas, G., Vlahavas, I.: PersoNews: A Personalized News Reader Enhanced by Machine Learning and Semantic Filtering. In: *On the Move to Meaningful Internet Systems 2006: CoopIS, DOA, GADA, and ODBASE. LNCS* (2006)
5. Aizenbud-Reshef, N., Guy, I., Jacovi, M.: Collaborative feed reading in a community. In: *Proceedings of the ACM 2009 International Conference on Supporting Group Work, GROUP 2009* (2009)
6. Adams, D.J.: XEP-0009: Jabber-RPC (2006), <http://xmpp.org/extensions/xep-0009.html> (retrieved January 21, 2011)
7. Lunze, T., Feldmann, M., Eixner, T., Canbolat, S., Schill, A.: Aggregation, Filterung und Visualisierung von Nachrichten aus heterogenen Quellen – Ein System für den unternehmensinternen Einsatz. In: *Proceedings of the GeNeMe 2009 Workshop, Dresden* (2009)

Automated Process Decision Making Based on Integrated Source Data

Florian Niedermann, Bernhard Maier, Sylvia Radeschütz,
Holger Schwarz, and Bernhard Mitschang

Institute of Parallel and Distributed Systems, University of Stuttgart,
Universitätsstraße 38, 70569 Stuttgart, Germany
{florian.niedermann,bernhard.maier,sylvia.radeschuetz,holger.schwarz,
bernhard.mitschang}@ipvs.uni-stuttgart.de
<http://www.ipvs.uni-stuttgart.de>

Abstract. Decision activities are frequently responsible for a major part of a process's duration and resource consumption. The automation of these activities hence holds the promise of significant cost and time savings, however, only if the decision quality does not suffer. To achieve this, it is required to consider data from diverse sources that go beyond the process audit log, which is why approaches relying solely on it are likely to yield sub-optimal results. We therefore present in this paper an approach to process decision automation that incorporates data integration techniques, enabling significant improvements in decision quality.

Keywords: Data Mining, Decision Automation, Data Integration, Business Process Management, Data-driven Processes.

1 Introduction

In this section, we first discuss the role and complexities of decisions in business processes. Then, we demonstrate the importance of decision automation by considering the various effects that automation can have on a business. Finally, we briefly introduce our *deep Business Automation Platform (dBOP)* that provides an integrated environment for *Business Process Optimization (BPO)*, including the automation of decisions.

1.1 Decisions and Influence Factors in Business Processes

A decision fundamentally consists of a set of one or several nodes in a business process that determines, in the presence of several alternatives, which process path to take. A typical (manual or non-automated) is made by a human actor who weighs several factors against each other to arrive at the decision results. Some of these factors include:

- **Process data:** As most processes are today executed using some kind of *Business Process Management System (BPMS)* that supports handing data over between different activities, one decision factor is the process data. This is also the data that is typically written into the audit log of the *BPMS*.

- **Application systems:** In many processes, the decision maker utilizes one or several application systems - either to gain additional information or for decision support.
- **External services:** In some instances, data provided by an external source can be instrumental to the decision - a popular example being the role of credit rating agencies in many retail processes.
- **Decision maker attributes:** Whenever the decision maker does not follow strict formal rules, her or his characteristics, experiences and other attributes influence the decision. These attributes can be either explicit (e.g., formal education, years of experience) or implicit (e.g., cultural values, biases etc.).
- **Other implicit and explicit knowledge:** A wide range of other factors can influence decisions. This can include non-codified knowledge about the work item, the business process or external influences. For instance, major events or a casual conversation with a supplier can have substantial impact on a decision.

1.2 Motivation for Decision Automation

Depending on the nature of the process, decisions and their preparation can easily account for a significant proportion of a processes' duration, cost and resource requirements. Hence, there are strong reasons for business to automate decisions as much as feasible:

- **Faster processes:** While a human actor often can take minutes to decide upon a complex matter, a decision model can arrive at a conclusion within the fraction of a second.
- **Dealing with resource bottlenecks:** If a decision involves human actors, an organization invariably needs to have a sufficient number of people with the according qualifications. This is especially challenging if the demand is fluctuating, as this either leads to idle workers or to long waiting times.
- **Reducing process cost:** As a manual decision involves a (costly) human actor, its cost is typically reduced through automation.
- **Enabling new business models:** Through the combination of the factors mentioned above, automated decisions enable new business models that would not be economical or even possible with a manual decision maker.

Despite this importance, the multitude of influence factors discussed in the previous section suggests that it is nearly impossible to create a decision model that exactly replicates the decision of a human actor in any given situation. This is frequently, however, neither necessary nor desired (e.g., human bias can negatively influence decision quality). Further, research in machine learning has shown that the behavior of complex systems (such as decisions, see Section 3.2) can in various scenarios be reasonably well predicted using a sufficiently large subset of the system's attributes [4], such as the one that can be provided by integrating process and operational data.

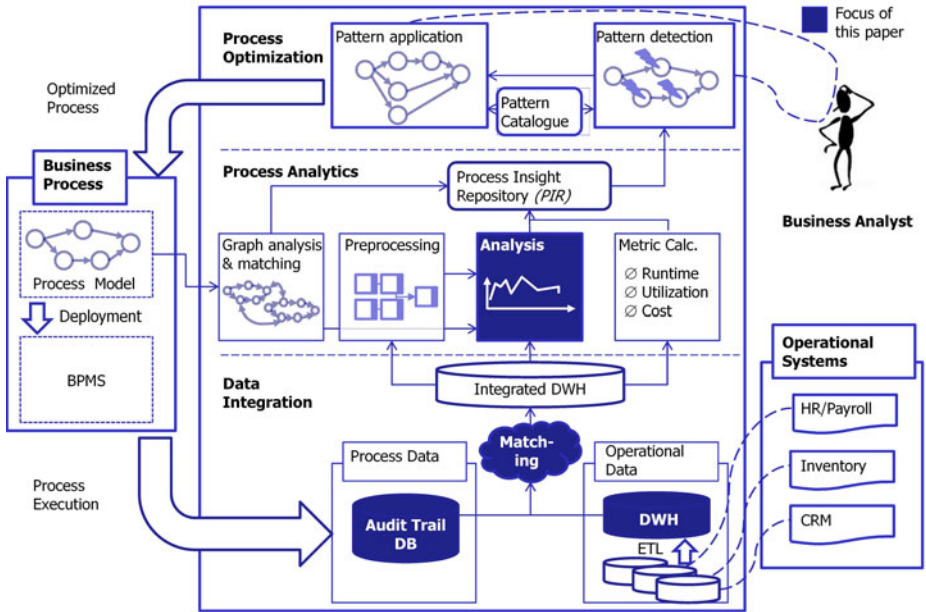


Fig. 1. Deep Business Optimization Platform overview

1.3 The Deep Business Optimization Platform

In order to make decision automation viable, we need a platform that provides the required data, the appropriate analysis techniques as well as a means to use the decision model in process execution. Our *deep Business Automation Platform (dBOP)* shown in Fig. 1 provides these and other facilities for (semi-)automated process optimization spread over three architectural layers:

1. **Data Integration:** As data relevant to the process and its decisions can be spread over a variety of heterogeneous data sources, the first platform layer provides the means to match and integrate process data with other data sources. We discuss some of the aspects of this layer in Section 2.
2. **Process Analytics:** In order to achieve meaningful optimization results, process specific "insights" need to be extracted from the integrated data layer. One of these analysis techniques is the learning of a decision classifier that is the subject of Section 3. This layer also includes process matching capabilities for design-time optimization [10] and static process graph analysis methods.
3. **Process Optimization:** Next to customized analysis techniques, the platform also contains a broad set of formalized best practice process optimization patterns. These patterns (such as parallelization, task elimination or decision automation) utilize the analysis results to determine which modification of the process are most beneficial under a certain goal function given

by the process analyst. We only briefly visit this layer in Section 3.3, more details can be found in 11.

As our previous work (see for instance 13, 12 or 9) has discussed general aspects of the platform extensively, this paper focuses on the specific components required for decision automation. In Section 2, we show how our integration approach enables us to cover a large subset of the decision influence factors listed in Section 1.1. Next, Section 3 shows how the integrated data is used to build and provide an automated decision classifier. In Section 4, we evaluate our approach using the decision classifier implemented in the *dBOP* and demonstrate how integrated data can improve the classifier quality. Finally, we take a look at related work in Section 5 before providing a brief discussion of our future research plans on the subject and the conclusion of the paper in Section 6.

2 Data Integration

In this section, we illustrate the Data Integration capabilities of the *dBOP*. First, we discuss the properties of process data vs. other operational data sources. Then, we explain how our matcher helps to integrate these heterogeneous data sources and how they are consolidated to provide the input for the decision classifier.

2.1 Process and Operational Data

To achieve integration, process activities pass data between each other. As most processes are today executed on some kind of *BPMS*, this process data is recorded in their audit log. The nature of process data is *flow-oriented*, i.e., while it contains information about the process flow, activity durations etc., the amount of information about the process subjects (e.g., work items, customers, resources) is often reduced to a minimum.

Operational data, stored in some application system or a *Data Warehouse (DWH)*, on the other hand is *subject-oriented*. It contains comprehensive information about the process subjects, but little information about the process flow (e.g., it might not contain information about process paths or cancelled instances).

The relationship between process and operational data and the decision influence factors is conceptually illustrated in Fig 2. As we can see, both process and operational data cover different areas. Further, three observations can be made: First, even by combining process and operational data, we are unlikely to capture all relevant information. This is, however, quite often not necessary, as we have discussed in Section 1.2. Second, process and operational data have some overlapping attributes (such as ID values). These attributes can be used to match and integrate the data, as we show in the next section. Finally, the information gain of operational data varies with the richness of the subject oriented information contained in the process data (e.g., because it is newly captured during the process). The last observation is revisited and quantitatively evaluated in Section 4.

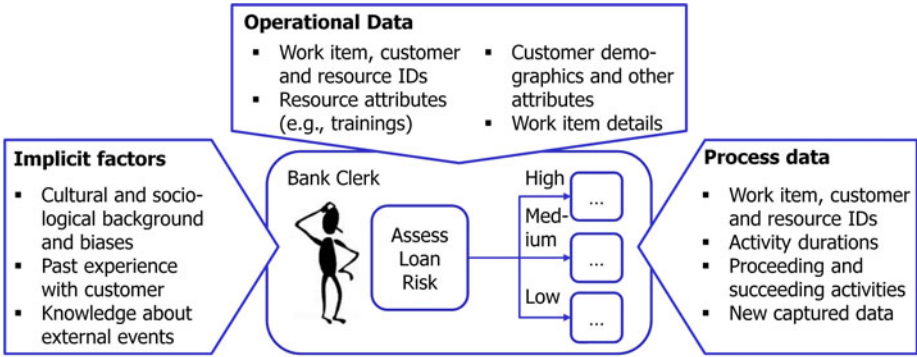


Fig. 2. DecisionAttributes

2.2 Integration Approach

As prior sections have shown, decision automation requires the integration of heterogeneous data sources. Due to the different paradigms of process and operational data, classical schema matching approaches like [1] struggle with their integration (e.g., because they have no specific methods to propagate matchings). This is why we have developed a specific approach for integrating operational and process data. As Fig. 3 illustrates, it consists of three steps:

- 1. Annotation and matching:** First, the matches between the process and operational data models need to be determined. This can be done using a variety of techniques, including direct matches pointed out by an analyst, matches found out using a semantic reasoner or through natural language processing (e.g., by matching synonyms).
- 2. Matching propagation:** After the initial matches have been determined, the matches are propagated based on a set of matching rules that utilize the specific properties of process data. One of these rules, for instance,

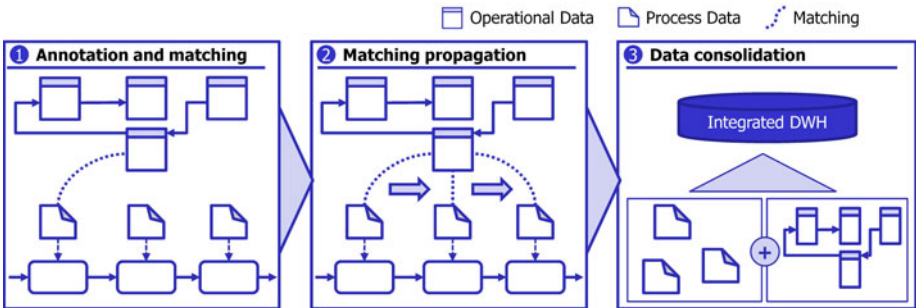


Fig. 3. dBOP Data Integration Approach

propagates the matchings along data flow mappings (such as the one realized by <assign> statements in BPEL).

3. **Data consolidation:** Finally, the data is consolidated into an integrated DWH, similar to the one discussed in [2]. This DWH is then used as the data source, e.g., for the construction of the decision classifier.

More details on the integration approach can be found in [13] and [12].

3 Decision Automation

After the previous section has discussed how we can get a large share of the decision influence factors through the use of data integration, this section will show how this data can be used to build a concrete decision classifier. First, we will show how decisions can be identified in a graph-based process model. Then, we will discuss how these decisions can be transformed into a classification problem. Finally, we show how the classifier is used by the *dBOP* to implement decision automation during process modeling and execution.

3.1 Identifying Decisions

To explain how decisions are identified, we first need some meta-model to represent processes. For the scope of this paper, we use a greatly simplified version of the process model graph presented in [7], please see there for further details. We use this meta-model, as it presents a good mix of usability (due to its proximity to prominent modeling languages, such as BPMN), implementability and formal reasoning powers. The concepts presented can, of course, also be applied with minor modification to other meta-models like Petri nets.

Definition. *Simplified PM Graph:* A simplified PM Graph G is a tuple $(V, O, N, C, E, \iota, o, \mu$ and χ are functions, in which

1. V is the finite set of process data elements (also called variables).
2. O is the finite set of operational data relevant to the process.
3. N is the finite set of process nodes.
4. C is the finite set of conditions.
5. $E \subseteq N \cup N \cup C$ is the set of (control) connectors.
6. $\iota : N \cup C \cup \{G\} \rightarrow \wp(V)$ is the input data map, with $\wp(V)$ being the power set over V .
7. $o : N \cup \{G\} \rightarrow \wp(V)$ is the output data map.
8. $\mu : \wp(V) \rightarrow \wp(O)$ performs the matching, i.e., integrates operational data with variables.
9. $\chi : E \rightarrow C \cup \{\emptyset\}$ determines, which condition has been assigned to a control connector.

Let further be $\vec{N}: N \rightarrow \wp(N)$ denote the immediate successor nodes and $\vec{E}: N \rightarrow \wp(E)$ the immediate successor control connectors of a process node

and let the predicates $SUCC(n_1, n_2)$ and $PRED(n_1, n_2)$ denote that n_2 is the successor/predecessor of n_1 respectively.

Based on this definition, we now define a decision as a node with several conditional outgoing connectors and exactly one outgoing connector without a condition (the *default* connector). Further, we define the set of associated nodes as the nodes that provide the input for the outgoing control connectors. More precisely, we define a decision as follows:

Definition. *Decision:* A Decision D in a simplified PM Graph G is a tuple (n_D, V_E, A, R) in which

1. n_D is the decision node for which holds true: $|\vec{E}(n_D)| \geq 2$ and $\exists e_{Default} \in \vec{E}(n_D) : \chi(e_{Default}) = \{\emptyset\} \Rightarrow \forall e \in \vec{E} \setminus \{e_{Default}\} : \chi(e) \neq \{\emptyset\}$.
2. V_E is the set of data attributes used to select a path with $V_E = \bigcup_{\forall e \in \vec{E}(n_D)} \iota(\chi(e))$.
3. A is the minimal set of associated nodes, with $\bigcup_{\forall a \in A} o(a) \supseteq V_E$ and $\forall a \in A : \exists v \in V_E : v \in o(a) \wedge \neg \exists n \in N : v \in o(n) \wedge SUCC(a, n) \wedge PRED(n, n_D)$
4. R is the set of possible decision results, with $R = \vec{N}(n_D)$.

For the sake of simplicity, we will focus on decisions, where $A = \{n_D\}$, i.e., the only node that is relevant for the decision is the decision node itself. While the principle method for automating decisions with multiple associated nodes is the same, there are some added complexities (such as resequencing of activities or effects of the automation on the process context) whose discussion goes beyond the scope of this paper.

3.2 Decision Automation as a Classification Problem

In machine learning, a classification problem typically has three components [4]: A set of class labels that should be "learned", a set of input data to use for the learning and a classification algorithm that processes the data to learn the classification rules. Using the definition of a decision introduced in the previous section, we can define a decision classifier as follows:

Definition. *Decision classifier:* A decision classifier D_{CL} for a decision D is a tuple (D, I, L, Alg) in which

1. D is the decision the classifier seeks to learn.
2. I is the set of classification input data. The input data is made up by the matched input data of all associated activities, i.e., $I = \bigcup_{\forall a \in A} \mu(a)$.
3. L is the set of class labels to be learned from the input data, which is made up by the different decision results, i.e., $L = R$.
4. Alg is some classification algorithm that is used to learn the decision (e.g., a decision tree or a multilayer perceptron).

Decision automation can therefore be successfully transformed into a "standard" classification problem.

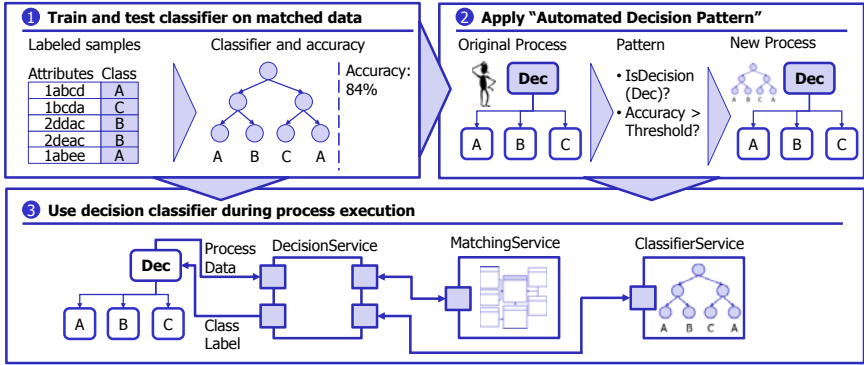


Fig. 4. *dBOP* Decision Automation Implementation

3.3 Implementation

As we have shown that decision automation can be treated like a classification problem, we can now use one of the many established classification algorithms to solve this challenge. The *dBOP* offers for this purpose a broad spectrum of classifiers which are taken from the WEKA library [5], with the default recommendation being the use of classification trees (see Section 4 for details on this choice).

As Fig 4 shows, the implementation of decision automation within the *dBOP* consists of three steps. First, the classifier for each decision in the process is trained and tested. The classifier along with the testing results are then forwarded to the *dBOP* optimizer. For these classifiers that the optimizer deems (based on user preferences) to be sufficiently good, it employs the "Automated Decision" process optimization pattern (see [11] for details on the pattern mechanism) to rewrite the process to include the automated decision. Finally, during process execution, the decision is handled by an automated decision service. It takes the original decision's input, enriches it per the defined mappings with operational data and feeds it to the decision classifier who in turn determines the class label.

4 Evaluation

After the decision automation approach has been explained thoroughly in the previous sections, this section will now quantitatively evaluate how well the approach performs in a sample application scenario. First, we will explain the evaluation scenario as well as the evaluation design. Then, we will discuss the evaluation results and their implications on the approach presented in this paper.

4.1 Evaluation Design

The evaluation is based on a simplified loan approval scenario shown in Fig 5. In it, a bank clerk classifies loan requests into high, middle and low risk loans. As

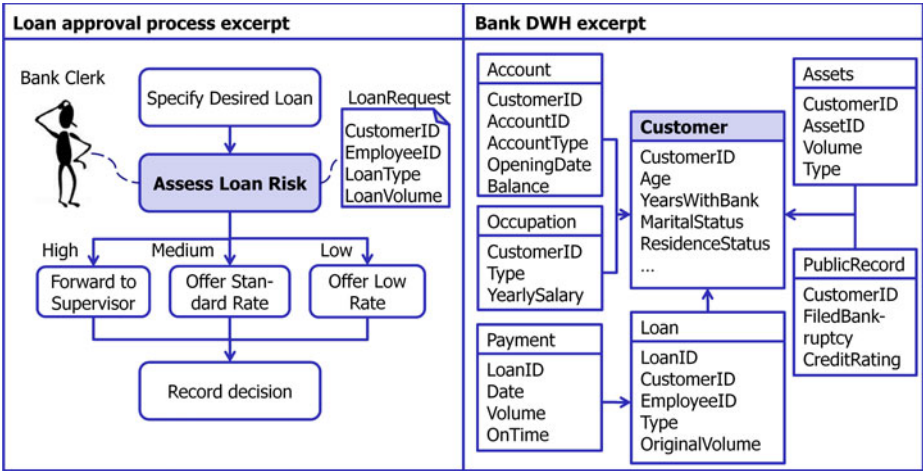


Fig. 5. Loan Approval Scenario

only existing customers can get a new loan, most of the information important to the decision is stored in operational data sources as shown in Fig. 5. We chose this scenario despite its typically already high degree of automation, as the factors involved are fairly complex and both the implicit and explicit decision factors are well publicized [17]. This allows us both to generate realistic test data and makes the scenario representative for other, less complex, decisions.

To assess the feasibility of our approach for automating the "Assess Loan Risk" activity using a decision classifier D_{CL} , the evaluation is set up as follows:

- **Input data:** To assess the impact of data integration on the quality of D_{CL} , we conduct the measurements using different input data sets: A minimal process data set containing only the loan type and volume, a rich process data set containing additionally the customer’s account balance and credit rating and the integrated data set, containing the complete set of attributes shown in Fig. 5.
- **Classification algorithms:** To assess the suitability of different types of algorithms, we employ three different ones: The WEKA implementations (in brackets: the WEKA names) of the C4.5 decision tree (J48), the naive bayesian classifier (NaiveBayes) and a multilayer perceptron (MultilayerPerceptron).
- **Sample size:** In total, we use a set of 27.000 sample processes. The set is split into 18.000 training and 9.000 testing samples (without cross validation).
- **Quality measurement:** The quality of the classifier is measured by the classification accuracy, defined as the share of correctly classified samples compared to the total number of classified samples.

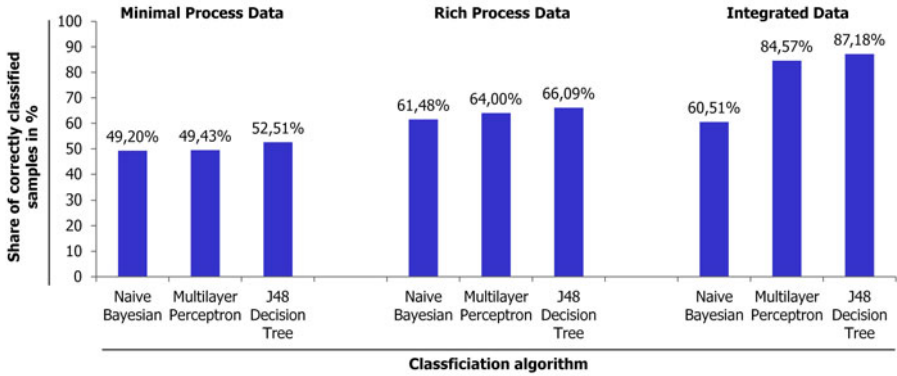


Fig. 6. Evaluation Results

4.2 Results

The results of the evaluation are shown in Fig. 6. The following observations can be made: First, non-surprisingly, using integrated data (87,18% accuracy with the best classifier) creates clearly superior results over both the minimal (52,51%) and the rich process data (66,09%). Second, while the benefit of using integrated data is somewhat reduced by the richer process data, it still remains significant. Third, the choice of classifier makes a great difference: The naive bayesian classifier performs notably worse than the other two classifiers. This is largely because the classifier assumes that attributes are independent [6], which is not the case in this scenario. The best performing algorithm is the decision tree, with the multilayer perceptron trailing behind. In this scenario, this could be attributed to the greater flexibility of decision trees with regards to the partitioning of training data. An alternative explanation can be found in [8].

Overall, the evaluation has demonstrated the feasibility of our approach for the given application scenario. As we on purpose selected a scenario with a multitude of influence factors and a complex decision logic, it is reasonable to assume that the approach also works under other circumstances. Further, it has shown that the inclusion of operational data significantly improves the quality of the decision classifier - to an extent that largely depends on the initial information richness of the process data.

5 Related Work

This paper is part of our work on the *dBOP* platform [9]. The data integration layer is discussed in [13]. Our integrated warehouse is similar to the process warehouse presented in [2], however, it offers better support for connecting process and operational data. The methods employed in the analysis layer are adapted from standard data mining and machine learning literature [6] [4]. Examples for their application can be found in [12] and [10]. The optimization layer builds heavily on existing research into business process optimization techniques, such

as [14]. Its role within the *dBOP* is the subject of [11]. Overall, the approach of a system that automatically adapts according to a set of rules and feedback from its execution can be conceptually seen as an application of cybernetics [18] to *BPO*. The workflow controlling framework discussed in [19] and the process analysis approach of [3] are somewhat similar to our platform in that they use custom analysis tools to gain process insights. However, their integration and analysis capabilities are limited and they lack an optimization layer.

Various other papers deal with the application of machine learning and data mining techniques to process data under the umbrella term of *Process Mining*. Closest related to this paper is the decision mining approach presented in [15] and [16]. The focus of the presented approach seems to be, however, more on process model validation (i.e., verification of whether a certain process execution instance conforms to a given process model) and less on actual decision automation. It hence only considers process data and does not provide a classifier for process execution, which restricts its application to decisions with limited complexity.

6 Conclusion and Outlook

In this paper, we have presented an approach for the automation of decisions in business processes. We have shown how decision automation can be transformed into a standard classification problem and demonstrated both qualitatively and quantitatively, that data integration can greatly increase the quality of a decision classifier. Further, we have presented our *dBOP* platform, which allows for a direct application of the analysis results through process rewriting and an integrated decision classifier service.

Future research plans on the topic include the application of machine learning techniques to a broader set of process optimization scenarios, such as the selection of process resources or the retrieval of process variants. Further, we are planning to do an empirical study based on business expert interviews to determine which level of accuracy would constitute, in a practical environment, a "good" classifier and what the ramifications of its use would be. Additionally, we are exploring the application of our data-driven approach to different areas of Business Process Management, such as the construction of simulation models and the definition and verification of business rules.

References

1. Aumüller, D., Do, H.H., Massmann, S., Rahm, E.: Schema and ontology matching with COMA++. In: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data (2005)
2. Casati, F., Castellanos, M., Dayal, U., Salazar, N.: A generic solution for warehousing business process data. In: Proceedings of the 33rd International Conference on Very Large Data Bases, pp. 1128–1137 (2007)
3. Castellanos, M., Casati, F., Dayal, U., Shan, M.C.: A comprehensive and automated approach to intelligent business processes execution analysis. *Distributed and Parallel Databases* 16(3), 239–273 (2004)

4. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern classification. Wiley Interscience, Hoboken (2000)
5. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: An update. *ACM SIGKDD Explorations Newsletter* 11(1), 10–18 (2009)
6. Han, J., Kamber, M.: Data mining: concepts and techniques. Morgan Kaufmann, San Francisco (2006)
7. Leyman, F., Roller, D.: Production Workflow. Prentice-Hall, Englewood Cliffs (2000)
8. Liu, X., Bowyer, K.W., Hall, L.O.: Decision trees work better than feed-forward back-prop neural nets for a specific class of problems. In: 2004 IEEE International Conference on Systems, Man and Cybernetics, vol. 6, pp. 5969–5974. IEEE, Los Alamitos (2005)
9. Niedermann, F., Radeschütz, S., Mitschang, B.: Deep business optimization: A platform for automated process optimization. In: Proceedings of the 3rd International Conference on Business Process and Services Computing (2010)
10. Niedermann, F., Radeschütz, S., Mitschang, B.: Design-time process optimization through optimization patterns and process model matching. In: Proceedings of the 12th IEEE Conference on Commerce and Enterprise Computing (2010)
11. Niedermann, F., Radeschütz, S., Mitschang, B.: Business process optimization using formalized patterns. In: Proceedings BIS 2011 (2011)
12. Radeschütz, S., Mitschang, B.: Extended analysis techniques for a comprehensive business process optimization. In: Proceedings KMIS (2009)
13. Radeschütz, S., Niedermann, F., Bischoff, W.: Biaeditor - matching process and operational data for a business impact analysis. In: Proceedings EDBT (2010)
14. 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)
15. 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)
16. Rozinat, A., van der Aalst, W.M.P.: Decision mining in business processes (2006)
17. Thomas, L.C.: A survey of credit and behavioural scoring: forecasting financial risk of lending to consumers. *International Journal of Forecasting* 16(2), 149–172 (2000)
18. Wiener, N.: Cybernetics: Control and communication in the animal and the machine. MIT Press, Cambridge (1948)
19. zur Mühlen, M.: Workflow-based process controlling: foundation, design, and application of workflow-driven process information systems. Logos Verlag (2004)

Making Decision Process Knowledge Explicit Using the Decision Data Model

Razvan Petrusel¹, Irene Vanderfeesten², Cristina Claudia Dolean¹,
and Daniel Mican¹

¹ Faculty of Economical Sciences and Business Administration, Babes-Bolyai University,
Teodor Mihali str. 58-60, 400591 Cluj-Napoca, Romania

{razvan.petrusel, cristina.dolean, daniel.mican}@econ.ubbcluj.ro

² School of Industrial Engineering, Eindhoven University of Technology, P.O. Box 513, 5600
MB Eindhoven, The Netherlands

i.t.p.vanderfeesten@tue.nl

Abstract. In this paper we present an approach for mining decisions. We show that through the use of a Decision Data Model (DDM) we can make explicit the knowledge employed in decision making. We use the DDM to provide insights into the data view of a business decision process. To support our claim we introduce our complete, functional decision mining approach. First, a ‘decision-aware system’ introduces the decision maker to a simulated environment containing all data needed for the decision. We log the user’s interaction with the system (focusing on data manipulation and aggregation). The log is mined and a DDM is created. The advantage of our approach is that, when needed to investigate a large number of subjects, it is much faster, less expensive and produces more objective results than classical knowledge acquisition methods such as interviews and questionnaires. The feasibility and usability of our approach is shown by a case study and experiments.

Keywords: Decision Mining, Product Data Model, Decision-aware System, Decision Workflow.

1 Introduction

In the area of enterprise financial decisions, there are a lot of fuzzy, not formally sound decision making processes. If we focus only on the data elements used in the decision process, our experience shows that managers tend to disregard some data items. This happens not just because they consider these data items unimportant but because it just slipped their mind or they just don’t know about it. For example, when a manager intends to contract a loan for their business, some decision makers consider important the amount paid to suppliers in the previous months while others might not. People may also perform decision making in unstructured situations by using feelings, intuition, etc.

Decision processes have been first researched in the early 60’s. The root of current well known decision processes is Simon’s model [1]. The focus of decision theory is on producing several decision alternatives and on how to perform the choice between

those alternatives. Decision theory assumes that the user knows which data items are needed to make informed decisions even if the actual values are uncertain [2]. Less attention was given to identifying relevant information or how to manipulate all the data items that need to be considered. But this is at the core of a business decision process. We argue that a regular person making a business decision does not always know which information is needed or relevant and does not have a clear overview of how available data should be aggregated.

We are aiming to provide a better insight into the decision process by making the implicit knowledge used in the decision process explicit. We are looking at different persons performing the same decision and we try to evaluate the process that they perform when making a decision. This involves a lot of mental activities which we need to capture and to make explicit in a model. We propose to use a Decision Data Model (DDM) as a graphical representation that can depict the data used in the decision process; and is easily understood by persons with less domain knowledge.

The aim of this paper is to: i) show how a model explicitly depicting the knowledge behind the data used in a decision making process is created and ii) to introduce an initial evaluation of the usefulness of such a model. Our approach includes all the necessary steps to automatically mine such a model based on the interaction of the decision maker with software. The framework includes a 'decision-aware system', a mining algorithm and the DDM format for representing the mined knowledge.

First, we introduce the reader to an overview of the decision mining approach. Then, Section 3 explains the concept of DDM and discusses the related research areas. In the mining approach (Section 4) we define the concepts we use and explain the steps we follow in order to create a DDM out of user activity logs then show the mining algorithm and a running example. The next section introduces a case study and a brief discussion over a DDM mined from an expert user. In the last sections we provide an evaluation of the approach and the conclusions.

2 General Approach

The general approach for making explicit the relevant knowledge for a decision process is illustrated in Fig. 1. The highlighted area is the main focus of this paper.

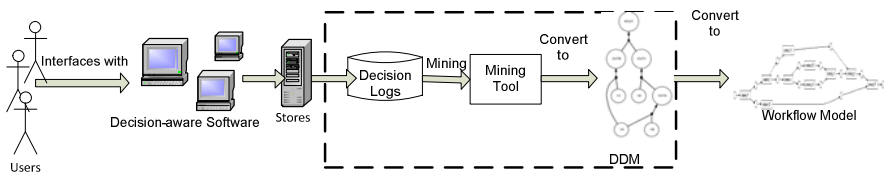


Fig. 1. General approach for making decision process knowledge explicit

First, we ask the decision makers to use our ‘decision-aware software’. This software provides the decision makers with decision scenario data, ranging from trivial to critical (e.g. all the data and information outputted by the information system of a company for a business decision). The software contains no reference that could guide or influence the decision making process. The term decision-aware designates the fact that the software is built so it will [3]: a) enable the user to perform all the mental steps towards making a decision within the boundaries of the system (mental steps are, for example, viewing a data element, comparing data elements, calculating new data elements, etc.); b) ‘force’ the user to decompose a mental pattern into basic thinking items; and c) allow the user to express each basic thinking item as an interaction with the system so it can be logged.

The software stores all details regarding the user interaction with the system as decision logs [3]. For the purpose of this paper we focus only on the data elements that are used by a decision maker while performing a specific decision process. This is referred to as a ‘trace’ of the process. Each trace should consist of: a) basic data items available in the simulation scenario and used by the decision maker; b) data items inputted by the user in addition to the basic data items available in the system (the user may type in new data and use it in deriving new items; we log for each such data item: the mandatory description and the value inputted by the user); c) derived data items calculated by the user; d) the type of interaction with the system; and e) timestamp of each interaction.

The log storing a trace is converted to a Decision Data Model by our decision mining tool. After that, the DDM may be converted into a workflow process model [4]. This last step will not be elaborated upon in this paper.

We basically hypothesize that our approach has some advantages over classical knowledge acquisition methods (such as interviews, questionnaires): a) the DDM depicts clearly the mental actions of the user and their order, b) the DDM can be easily understood (therefore knowledge is easier disseminated), and c) our decision mining approach will take less time compared to classic knowledge acquisition methods when applied to a large number of users. Since the decision makers in our approach interact with web-based software, the potential number of subjects is unlimited. All those users can perform the decision process at any time, from anywhere, at no cost and don’t require assistance from a human (the users guide of the software is enough to find out how to use it). A questionnaire or an interview requires human involvement, therefore is slower and more expensive (since resources are scarce). The best suited classical tool for acquiring knowledge about a process is direct observation (reporting while doing for mental processes). The results of observations or reports can be influenced by the observer. Our results are unbiased because they come straight from the user.

3 Background and Related Work

This research draws on several major fields of research: workflow management (especially process mining), decision making theory and analysis, decision support systems and software simulations. In this section we briefly discuss related work and introduce the notion of the PDM from the workflow domain.

The process mining methodology is present for over two decades. The result of process mining is a model that reflects a real life process in an enterprise [7]. The decision mining approach we present in this paper resembles process mining in that it aims to automatically extract and create a model, but this is done of a mental decision making process rather than of some physical process in the enterprise. Current algorithms for process mining focus on the retrieval of a process model from an event log. In our mining approach we try to mine the processing of information. Our approach is based on the fact that the actions of a person will provide an external observer with a better understanding of a workflow than what the user says about that workflow. Therefore, we can produce a more objective model which shows what actually happened instead of what the user says has been done. This assumption is also used by various researchers in process mining that rely on the historic operational data available from event logs (or audit trails or transaction logs, etc) produced by the software tools used in an enterprise (ERP, CRM, SCM, etc) rather than on the prescribed workflows modeled by experts [8].

The term “decision mining” was used before in [9]. Even though the same term is used, it is very different from our research in terms of objectives, research focus, etc. Rozinat [9] looks just at a specific kind of activity in a workflow model (i.e. splits). The goal is to identify the points in which a choice was made and to determine the properties that influenced the choice of one or another of the branches.

The class of systems which aims to provide the user with all the necessary data and information in order to help him to make better decisions is the class of decision support systems (DSS) [2]. In order to create a successful decision-aware system, we need to implement defining features of DSS in a virtual environment in order to provide the user with the best decision experience. In many ways a ‘decision-aware’ system is similar to a DSS, because it is intended to help the user make a decision by providing necessary data and some tools to manipulate it. However, we focus on logging the user interaction with the software instead of providing guidance during decision making.

The Decision Data Model

As explained in the previous section, the DDM is used to depict the mined decision process. Therefore, the reader should be familiar with the concept of the DDM. Below, the notion of a DDM is explained in more detail. The DDM is highly similar to the well-known concept of a Product Data Model (PDM) from the area of business process (re)design, which is the starting point for the Product-Based Workflow Design (PBWD) methodology [4], [5]. We have adapted the general PDM definition from [4] to our purposes and find the term DDM more appropriate and adequate.

A DDM describes the structure of the process of information processing. It is similar to a Bill-of-Materials [6] but instead of a physical product it describes how to produce an informational product (e.g. a decision on an insurance claim, the allocation of a subsidy, or the approval of a loan). In a DDM the data elements that play a role in a decision and their relationships are made explicit in a graphical way.

Consider, for instance, the example given in Fig. 2. This example describes the calculation of the maximum amount of mortgage a client is able to borrow from a bank. The figure shows that the maximum mortgage (element A) is dependent either on a previous mortgage offer (E), or on the registration in the central credit register

(H), or on the combination of the percentage of interest (B), the annual budget to be spent on the mortgage (C), and the term of the mortgage (D).

Data elements are depicted by circles in the DDM. For each specific case instance of the decision process a data element may have a different value (e.g. the gross income of each client will be different). The actions that are taken on the data element values are called operations and are represented by hyperarcs. In general, an operation can be of different forms, e.g. an automatic calculation, a judgment by a human or a rule-based decision. However, in this paper we focus on operations that can be represented by an arithmetic formula using simple operators such as +, -, /, and *.

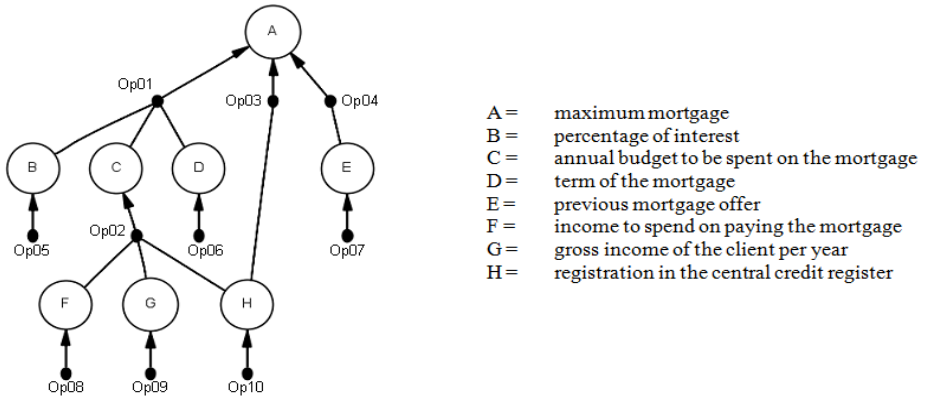


Fig. 2. The Decision Data Model for the mortgage example

Each operation has zero or more input data elements and produces exactly one output data element. The arcs are ‘knotted’ together when a value for all data elements is needed to execute the particular operation. Compare for instance the arcs from B, C, and D leading to A on the one hand, and the arc leading from E to A on the other hand in Fig. 2. In the latter case only one data element value is needed to determine the outcome of the process, while in the case of B, C, and D all three data element values are needed to produce A. An operation is executable when a value for all of its input elements is available.

Several operations can have the same output element while having a different set of input elements. Such a situation represents alternative ways to produce a value for that output element. For example, a value for the end product A in Fig. 2, can be determined in three alternative ways: (i) based on a value for E, (ii) based on a value for H, and (iii) based on values for B, C, and D. Also, a data element may be used as an input element to several operations. For instance, data element H is used in two operations: Op02 and Op03.

The top element of the DDM, i.e. the end product, is called the root of the DDM. The leaf elements are the elements that are provided as inputs to the process elements (e.g. the elements B, D, E, F, G, H). They are produced by operations with no input.

After the above informal introduction, the DDM can be formally defined as follows (this definition is adjusted from [4]):

Definition 1. A DDM is a tuple $(D, O; T)$ with:

- D : the set of data elements, $D = BD \cup DD \cup ID$, with
 - BD the set of leaf data elements
 - DD the set of derived data elements
 - ID the set of data elements inputted by the user
- $O \subseteq D \times P(D)$: the set of operations on the data elements.
Each operation, $o = (d, ds, ao)$:
 - has one output element $d \in DD$ and
 - has a set of zero or more input elements $ds \subseteq D$
 - has an arithmetic operation ao , specifying how to produce the output element d based on the input elements ds .
- D and O form a hypergraph $H = (D, O)$ such that its structure graph is connected and acyclic.

The DDM of Figure 2, contains six leaf elements: $B, D, E, F, G, H \in BD$. The leaf element B is for instance produced by an operation $op_{05} = (B, \emptyset, \emptyset)$. There are also two derived data elements: $A, C \in DD$. Data element A is produced by operation $op_{01} = (A, \{B, C\}, C/B * D)$. Note that, in general, the structure of the DDM is a network structure (it is not a simple tree), but does not contain cycles.

4 The Mining Approach and Algorithm

This section introduces the reader to the fundamentals of the proposed knowledge extraction method. Afterwards, a running example provides a better understanding of how our approach works.

Once the decision maker finished performing the decision process, we need to extract relevant data from the log and present it as a DDM. This is done in three major steps:

- A. parse the logs and output an XML file,
 - A1) export the logs from the decision-aware tool;
 - A2) filter the logs for just one trace. This is based on the Process Instance ID;
 - A3) run the mining algorithm on the individual trace so that relevant information is extracted from the logs and output the sets in Definition 1;
 - A4) input the Definition 1 sets into the specific structure of the DDM-XML file;
- B. import the DDM-XML file into ProM Framework,
- C. build the DDM (and the workflow model).

Activity A1 is performed by a web-service included in the decision aware system. It allows the mining application to retrieve the necessary data (as an XML file) (see also [10]). Depending on the context, Activity A2 can be performed by the decision-aware system (if a user wants to build the model right after he finished performing the decision process) or by the mining application (if a researcher using our approach wants to build one process model out of a log containing multiple traces). Activity A3 is performed by the stand-alone mining application (see also [10]). For a better understanding we will introduce the algorithm implemented in the application as

pseudo-code in Fig. 3. The input data for the algorithm is one trace in the activity logs outputted by the decision-aware system formatted as one XML file (Activity A1). The mining application also performs Activity A4 and outputs a DDM specific XML file (see also [10]), containing the elements in Definition 1. So far, this file needs to be manually uploaded into ProM (Activity B). The ProM plug-in creates the DDM graphical representation and the various workflow models (Activity C) [4].

The main concern of the remainder of this sub-section is introducing the reader into how the XML file containing the structure of the DDM is produced (activity A). The mining algorithm implemented in the mining application performs mining based on the logic steps in Fig. 3. It is used on the running example data in Table 1 to produce the model in Fig. 4. Further explanations on the decision aware system, the algorithm and the functions used are available in [10].

```

Create: Leaf_Nodes set; Derived_Data_elements set;
Root_Node set; Operations set; Operation_Data_Elements
set
Do case for each record
  Case Find_click_textbox_in WFMElt_Field () = True
    Add new item to Leaf_Nodes set
  Case Find_"=" _char_in Name_Field () = True
    Add new item to Derived_Data_Elements set
    Do Recognize_Data_Elements_Used_in_Operation
    Add new item to Operations set
    Add to Operation_Data_Elements set (name of
current operation as output, all data items (leaf,
derived and input data elements) as input)
  Case Find_edit_textbox_in_WFMElt_Field () = True
    Add new item to Root_Mode set
Endcase

```

Fig. 3. Mining algorithm logic in pseudo code

How the data items and the operations, that are performed by the user while calculating a derived data element, can be explicitly shown, is explained further in a running example. This is important because we need to show how a particular derived value fits into the overall decision process. For this short example, we suppose the user needs to calculate and input in a separate textbox the result of Formula 1. As a naming convention, we use *X* in front of any leaf data item (*BD* in Definition 1) and we assign plain letters for any calculated item (*DD*):

$$(XA + XB) / XC = XD. \quad (1)$$

Where: $XA = 1000$, $XB = 500$ and $XC = 5$.

When calculating such a result the mental actions performed by the user are:

- check for the value of *XA*, then remember it for the calculation,
- check for the value of *XB*, then remember it for the calculation,
- calculate the result of the addition of *XA* to *XB*,
- check for the value of *XC*, then remember it for the calculation,
- calculate the final result by dividing the result of the previous addition (c) by the value of *XC*.

Those mental activities need to be explicitly performed within the decision-aware system. The sequence of interactions allows us to generate a log of all the mental steps taken by the user while performing the calculation. This is how we make explicit a part of the knowledge employed by the user. The log sample, generated by the interaction of the user with the decision-aware system, for calculating the formula introduced above is shown in Table 1.

Table 1. Log explicitly depicting mental calculation steps expressed as interaction with the software and the DDM elements derived from the log

Time stamp	Workflow Model Element (WFMElt)	Name	Data attributes	Derived DDM sets
Time 1	click textbox	XA	1000	<i>Leaf element set (BD):</i> {XA, XB, XC} <i>Derived data element set (DD):</i> {A, B} <i>Root element:</i> {XD} <i>Operation set (O):</i> {op1, op2, op3, op4, op5, op6} Where: op1=(A, {XA, XB}); op2=(B, {A, XC}); op3=(XD, B); op4=(XA, Ø); op5=(XB, Ø); op6=(XC, Ø)
Time 2	click button Add Data Item	Add_XA	XA	
Time 3	click button Plus	plus	+	
Time 4	click textbox	XB	500	
Time 5	click button Add Data Item	Add_XB	XB	
Time 6	click button Equal	=XA+XB	1500	
Time 7	click button Add Data Item	Add_(XA+XB=)	{XA+XB=}	
Time 8	click button Divide	divide	/	
Time 9	click textbox	XC	5	
Time 10	click button Add Data Item	Add_XC	XC	
Time 11	click button Equal	=(XA+XB=)/XC	300	
Time 12	edit textbox	XD	300	

The log actually shows all the calculation steps that are now explicitly performed by the user as a sequence of interactions with the decision aware system. Assuming this to be a complete trace in which the user’s goal is only to produce the result of the formula, the final step that needs to be performed by the user within the system is to write down the value of the calculation (edit textbox XD in record 12). The model explicitly depicting how the goal value was produced is shown in Fig 4. Note that the order of the elements in the DDM input data is not important.

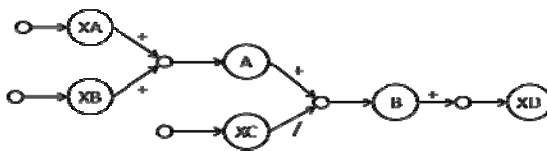


Fig. 4. DDM of the running example

5 Case Study and the Evaluation of the Approach

In order to demonstrate and evaluate our approach, we built a decision-aware system that implements the following simulation: “The user of the system is the decision maker in an enterprise that already decided to make an investment. Since the company doesn’t have enough cash available for the investment, the manager is faced with the decision of contracting a loan. The user needs to make a decision by choosing one alternative regarding a combination of: the loan value, the loan period, the loan type, and the installment type. For saving his decision, the user is required to

write down the values (for loan value and loan period) or select one of the available choices (e.g. for loan type there are 6 choices and for installment type, 2). The user needs to make all the decisions based only on the scenario data presented in the software, and cannot update any data item” [10]. He is allowed to input additional data elements, but once an element is added it cannot be updated. In order to perform the decision and to create a decision log, the user needs to interact with the decision-aware software. The goal of those actions is to derive new data (starting from basic data items provided in the simulation scenario) and build on it until the final decision values are calculated. An example of interactions showing the necessary steps for calculating a derived data item out of two basic data items is shown in Fig. 5.

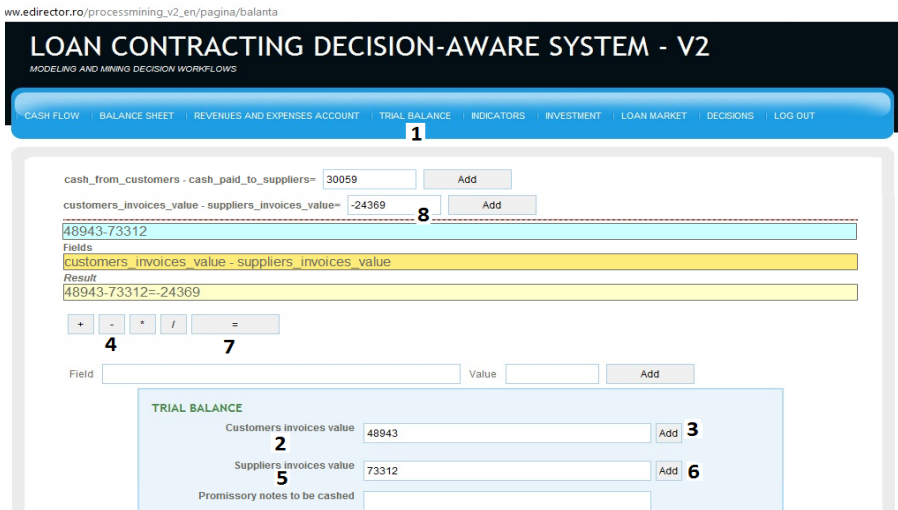


Fig. 5. Example of interaction sequence for calculating a derived data item

First, the user clicks one possible source of data about the managed enterprise (the Trial Balance in this instance – click no. 1), and is presented with empty textboxes of all the basic data items (see Promissory Notes for instance). The user needs to click a specific data item before the associated value is revealed (2), and then can add it to the calculation string (3). Then, the mathematical operation is selected (4) and another data item is added (5, 6). After all the calculation elements are in place, the equal sign is clicked (7) and the result is shown. The derived data element is added to the history list (8) from where it can be retrieved to be later used as an element in the next calculations. More on how the decision-aware system works can be found in [10].

After the user saves the final decision and logs out of the system, the decision-aware system will output an XML file containing the activity log for that particular trace [10]. The last step is the automatic conversion of the activity log XML into a DDM-XML file. By loading the DDM-XML file in ProM Framework the DDM and several workflow models can be produced.

In Fig. 6 we show the decision process model of an expert user aimed at choosing a value for the loan. From the case study we conclude that the mined DDM reveals some useful information:

- a) inconsistencies in the decision process of the user. For example, in this trace, one can notice a redundancy in the fact that the loaned amount depends on the cash at the beginning (XG) or at the end of the year (XH) when in the simulation data there is also such item as ‘net cash’, which was disregarded even if it is more meaningful,
- b) missing important data items that show up at a comparison with other models. For example, the current model is focused on past performance of the simulated company and disregards what future holds. When the expert was presented with a model produced by another expert, he admitted that he should have also considered the difference between accounts receivable and payables because it shows how much money the company still has to cash/pay from issued and received invoices in the near future. However, we found that expert models are quite similar (above 40% of basic data items are common to all expert traces for loan decision). More on the issue of model comparison can be found in [10] and will be subject to a stand-alone paper.

From this case study we conclude that our approach is feasible. The decision aware system correctly logs the user actions and it is possible to build a DDM from the logged information.

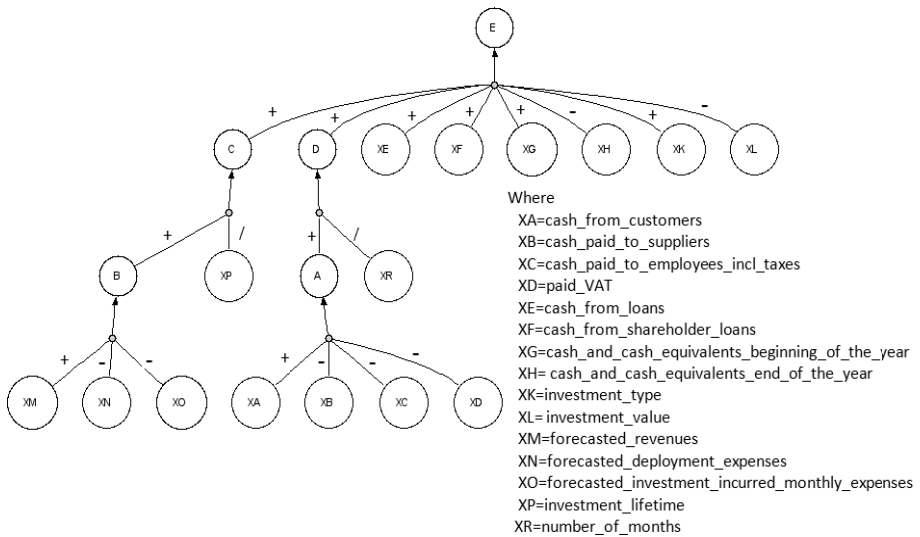


Fig. 6. Expert DDM model for the sub-decision of setting an amount for the loan

In addition to the case study, we tested our approach a experts from companies and students (bachelor and master level) from business faculties. There are two goals that we try to reach by this experiment. The first one is to gather a large number of traces (performed by users at various levels of knowledge), that can be mined and compared. The second goal is to collect the users’ reactions to our approach and their

understanding of this knowledge acquisition method and of the model produced. This evaluation tries to answer several questions: “is the DDM easily understandable?”, “does the DDM depict knowledge that otherwise would be hard to get?”, “can the user learn from such a model?”

Table 2 shows the results of this qualitative evaluation of the decision mining approach involving 33 intermediate decision makers. The decision makers were first asked to perform the decision in the decision aware system themselves. Afterwards, they were presented with a mined expert DDM and were asked to answer a questionnaire (see also [10]) with the following questions:

- Q1) “How much of the DDM model introduced after the software test can you understand?”
- Q2) “How much of the DDM model introduced after the software usage resembles your process?”,
- Q3) “To what extent do you feel that a DDM makes your knowledge explicit?”,
- Q4) “How much of your knowledge, about the loan contracting decision, would you be able to represent by yourself, using various representations (without the use of decision mining approach)?”,
- Q5) “Did the expert trace introduced as a DDM advance your knowledge on the loan contracting decision?”,
- Q6) “Did the expert trace introduced as a DDM reveal aspects of the decision you did not consider while performing the decision by yourself?”.

Table 2. Questionnaire results of experiment with 33 intermediate decision makers

Question no.	Q1	Q2	Q3	Q4	Q5	Q6
answer 1 (nothing, no)	0	0	2	0	0	1
answer 2 (a small part)	5	22	9	23	15	13
answer 3 (a large part)	26	10	22	10	14	18
answer 4 (completely, yes)	2	0	0	0	4	0
no answer	0	1	0	0	0	1
Average score	2,91	2,31	2,61	2,30	2,67	2,53

As can be seen from the table, the majority of the users can understand a large part of the mined expert DDM which they see for the first time (Q1) and believe that the approach makes most of the knowledge explicit (Q3). The majority of the users also identified a gap between their own decision process and the one from the expert (Q2). The user’s opinions are split about how much they learned about loan contracting by looking at the expert DDM (Q5) and about how many things they did not consider in the first place might be worth considering after all (Q6). Finally, the result for Q4 reveals that the largest part of the users find it difficult to formalize and represent this kind of knowledge on the decision process by themselves.

From this qualitative evaluation we conclude that the users are able to understand and use the DDM model, and that our decision mining approach is able to support in making decision process knowledge explicit. However, more research is needed to show that this approach is able to support learning and is a better and faster knowledge acquisition method.

6 Conclusions and Future Research

This paper introduces a complete framework aimed at making explicit the knowledge used in a business decision process. To do so we log the interaction of a decision maker with a decision aware system, from these logs the decision process is mined and represented by a DDM. The DDM model depicts: a) which data items were considered important and relevant by the user; and b) the new data items derived based on other data. We validated our approach (software, mining tool and the models we produce) by a case study and an experiment involving expert users and second year master and bachelor students. The conclusion we draw based on this qualitative assessment, is that our approach is feasible, that the DDM is easy to read and understand by decision makers, and therefore is a good tool to make decision process knowledge explicit. However, due to a limited qualitative evaluation, we were not able to prove yet that our method is a better knowledge acquisition method. By studying an expert's DDM one may learn from it and improve his/her knowledge on the decision process. Our approach may e.g. be used by professors for evaluating the progress in decision making training (by comparing the 'before' and 'after' models) or by professionals interested in an alternate knowledge extraction tool. More research is needed in this field. One of our major concerns, that will be investigated in the near future, is the comparison and integration of DDM models from different experts (i.e. building a model that aggregates individual traces) so that patterns present in different traces can be identified and pointed out to external observers. Due to the extremely different approaches of different users to the same decision process, standard process mining algorithms (Alpha++, Heuristic, Genetic and Fuzzy mining algorithms) output unusable spaghetti-like models. A new approach, tailored to the particularities of mental actions and processes, is required.

Acknowledgments. This work was supported by CNCSIS-UEFISCSU, project number PN II-RU TE code 292, number 52/2010.

References

1. Simon, H.A.: *The New Science of Management Decision*. Harper and Row, New York (1960)
2. Turban, E., Sharda, R., Delen, D.: *Decision Support and Business Intelligence Systems*, 9th edn. Prentice Hall, New Jersey (2010)
3. Petrusel, R., Mican, D.: Mining Decision Activity Logs. In: Abramowicz, W., Tolksdorf, R., Węcel, K. (eds.) *BIS 2010 Workshops*. LNBIP, vol. 57, pp. 67–79. Springer, Heidelberg (2010)
4. Vanderfeesten, I.: *Product-Based Design and Support of Workflow Processes*. PhD dissertation, Eindhoven University of Technology, Eindhoven (2009)
5. Reijers, H.A., Limam Mansar, S., van der Aalst, W.M.P.: Product-Based Workflow Design. *Journal of Management Information Systems* 20, 229–262 (2003)
6. Orlicky, J.A.: Structuring the Bill of Materials for MRP. *J. Production and Inventory Management*, 19–42 (1972)
7. van der Aalst, W.M.P., van Hee, K.: *Workflow Management: Models, Methods and Systems*. MIT Press, Cambridge (2002)

8. van der Aalst, W.M.P., van Dongen, B.F., Herbst, J., Maruster, L., Schimm, G., Weijters, A.J.M.M.: Workflow Mining: A Survey of Issues and Approaches. *J. Data and Knowledge Engineering*. 47, 237–267 (2003)
9. 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)
10. Petrușel, R., Vanderfeesten, I., Dolean, C.C., Mican, D.: Making Decision Process Knowledge Explicit Using the Product Data Model. Beta working paper series, WP 340, Technische Universiteit, Eindhoven (2011)

Sentiment Lexicon Creation from Lexical Resources

Bas Heerschop, Alexander Hogenboom, and Flavius Frasinca

Erasmus University Rotterdam

PO Box 1738, NL-3000 DR, Rotterdam, The Netherlands

basheerschop@gmail.com, {hogenboom,frasinca}@ese.eur.nl

Abstract. Today's business information systems face the challenge of analyzing sentiment in massive data sets for supporting, e.g., reputation management. Many approaches rely on lexical resources containing words and their associated sentiment. We perform a corpus-based evaluation of several automated methods for creating such lexicons, exploiting vast lexical resources. We consider propagating the sentiment of a seed set of words through semantic relations or through PageRank-based similarities. We also consider a machine learning approach using an ensemble of classifiers. The latter approach turns out to outperform the others. However, PageRank-based propagation appears to yield a more robust sentiment classifier.

Keywords: sentiment analysis, sentiment lexicon creation, sentiment propagation, page rank, machine learning.

1 Introduction

Sentiment analysis, also referred to as opinion mining, encompasses a broad area of natural language processing, computational linguistics, and text mining. In general, the aim is to determine the attitude of the author with respect to the subject of the text, which is typically quantified in a polarity. Recent developments on the Web – enabling users to produce an ever-growing amount of virtual utterances of opinions or sentiment through, e.g., messages on Twitter, blogs, or on-line reviews – advocate an array of possibilities for business information systems. Mining sentiment in the vast amount of data on the Web has many interesting applications, such as in the analysis of on-line customer reviews, reputation management, or marketing. Proper tools for sentiment mining can enable businesses to monitor the public sentiment with respect to particular products or brands, which can yield invaluable input for their marketing strategies.

In recent work, we assessed the state-of-the-art in sentiment analysis [1]. We showed that many approaches essentially rely on a lexicon containing words or phrases and their associated sentiment scores. Such lexicons often need to be created first. Automated methods include supervised learning on a set of manually rated documents and learning through related word expansion – expanding a small, manually created set of words by exploiting word relationships such

as synonyms, antonyms, and hypernyms. Several lexicon creation methods have been proposed, yet their performance has typically been evaluated by means of comparing generated lexicons with manually created golden lexicons. However, we argue that assessing lexicon creation methods in terms of their performance in the actual sentiment analysis process would be very insightful as well, as sentiment lexicons are typically developed for this process and should hence be evaluated as such.

Therefore, we propose to perform a corpus-based evaluation of sentiment lexicon creation methods. In this paper, we compare the performance of two commonly used variants of a sentiment propagating word expansion algorithm and a commonly used machine learning approach based on classifiers. Our focus here is on algorithms exploiting vast, readily available lexical resources like WordNet [2].

The remainder of this paper is organized as follows. First, Sect. 2 expands on WordNet and how this lexical resource can be exploited for sentiment lexicon creation. We then provide the specifics of our sentiment analysis framework as well as the considered sentiment lexicon creation approaches in Sect. 3. Subsequently, the evaluation of these methods is described in Sect. 4. Finally, we conclude in Sect. 5.

2 Related Work

When automatically creating a sentiment lexicon, a good starting point may be an existing lexical resource, the contents of which can subsequently be associated with a sentiment score. A widely used on-line (semantic) lexical resource is WordNet, the design of which is inspired by psycholinguistic theories of human lexical memory. WordNet is designed to be used under program control and enables the distinction between different word forms and word meanings. WordNet is organized into sets of synonyms – synsets – which can be differentiated based on their Part-of-Speech (POS) type. Each synset expresses a distinct concept and is linked to other synsets through different kinds of relations (e.g., synonymy, antonymy, hyponymy, or meronymy).

WordNet contains four main POS types: verbs, nouns, adjectives, and adverbs. The semantic network of English verbs in WordNet is typically considerably more complex than the network of nouns, which suggests that verb meanings are more flexible in usage than noun meanings [3]. Nouns in WordNet are grouped in a hierarchical way based on distinguishing features (i.e., modification, hyponymy, meronymy, and predication). This hierarchy seldomly exceeds more than a dozen levels. Adjectives in WordNet are divided into two classes: descriptive (e.g., “big” or “interesting”) and relational (e.g., “presidential” or “nuclear”) and may have several types of relations. Adverbs have the least complex structure of all POS types. Adverbs not derived from an adjective only have occasional antonym relations and derived adverbs are semantically related to their base adjectives.

The semantic relations expressed in WordNet can be exploited to generate a sentiment lexicon. A typical approach is to start with a seed set of words and their associated sentiment and to subsequently traverse the WordNet relations, while propagating the sentiment [4,5,6]. Rather than by traversing WordNet

relations, sentiment can also be propagated to synsets that are similar to positive or negative synsets. One way of accomplishing this is by using an algorithm inspired by Google’s PageRank algorithm [7], which uses the link structure of the Web to calculate a quality ranking for each Web page. Esuli and Sebastiani argue that this algorithm can also be employed to create a ranking of how closely synsets relate to positive or negative synsets by using the semantic structure of WordNet [8]. Their application uses eXtended WordNet¹, a publicly available version of WordNet in which each word occurring in a gloss of a synset is mapped to the synset to which it belongs.

Another way of creating a sentiment lexicon based on WordNet synsets and their associated sentiment is to iterate over WordNet synsets and assign sentiment scores to these synsets by means of a classifier which analyzes the glosses associated with the individual synsets. An example of a sentiment lexicon thus generated is SentiWordNet [9], where eight classifiers (trained using a semi-supervised method on different seed sets of glosses annotated with their associated synsets’ sentiment) have been used to analyze the gloss of each WordNet synset σ in order to assign scores quantifying how objective $Obj(\sigma)$, positive $Pos(\sigma)$, and negative $Neg(\sigma)$ each synset is. Each score is determined by the (normalized) proportion of the eight classifiers that have assigned the corresponding label to it and the sum of the three scores is constrained to equal 1 for each synset.

3 Framework

In order to assess the performance of different sentiment lexicon creation approaches, we propose to test the performance of a simple sentiment analysis framework on a corpus, while using lexicons created with our considered approaches. The sentiment classification is further detailed in Sect. 3.1. Our considered sentiment lexicon creation methods are discussed in Sects. 3.2, 3.3, and 3.4.

3.1 Sentiment Classification

We propose a simple lexicon-based sentiment classifier for investigating the performance of sentiment lexicons created by means of our considered methods. This classifier focuses on adjectives, adverbs, verbs, and nouns. The sentiment score of a document d is computed by aggregating the scores for each sentence s , which in turn are computed by aggregating sentiment scores for each non-stopword w in the sentences. The score $\text{eval}(d)$ of a document d is thus computed as

$$\text{eval}(d) = \sum_{s \in d} \sum_{w \in s} \text{score}(w), \quad (1)$$

after which the classification class $\text{class}(d)$ of a document d can be determined as

$$\text{class}(d) = \begin{cases} 1 & \text{if } \text{eval}(d) \geq 0, \\ -1 & \text{if } \text{eval}(d) < 0. \end{cases} \quad (2)$$

¹ <http://xwn.hlt.utdallas.edu>

Algorithm 1. Document scoring

```

input : A document  $d$ 
output: The sentiment score of document  $d$ 
1  $docScore = 0$ ;
2  $docScoreSentenceCount = 0$ ;
3 foreach  $sentence$  in  $d$  do
4    $sentenceScore = 0$ ;
5   foreach  $word$  in  $sentence$  do
6      $pos = \text{getPOS}(word, sentence)$ ;
7      $lemma = \text{getLemma}(word, pos)$ ;
8      $sense = \text{getWordSense}(word, sentence, pos)$ ;
9      $score = \text{getWordScore}(lemma, sense, pos)$ ;
10     $sentenceScore = sentenceScore + score$ ;
11  end
12   $docScore = docScore + sentenceScore$ ;
13 end
14 return  $docScore$ ;

```

In this process, documents are first split into sentences, after which the words in each sentence are tagged by a POS tagger. In order to subsequently assign sentiment scores to the individual words, we need to first retrieve the lemma and then disambiguate the word sense before we can extract the associated sentiment from our lexicon, as detailed in Algorithm 1.

For the word sense disambiguation process, we propose to use a Lesk algorithm [10], as it has a freely available implementation for WordNet [11], which has proven to yield satisfactory results (50–70% accuracy). The algorithm, described in Algorithm 2, selects the word sense that is semantically most similar to the words in the context (i.e., the other words in the sentence). This similarity is measured in terms of the overlap of an ambiguous word’s gloss and glosses of its context.

3.2 Traversing WordNet Relations

The sentiment classification process described in Sect. 3.1 requires a lexicon in order to find the sentiment scores associated with words. A typical approach to create such a lexicon is to start with a manually created seed set of words and their associated sentiment [4,5,6]. Such a seed set may for example contain the positive words “beautiful”, “proud”, “security”, “good”, and “success” and the negative words “unfortunate”, “distressed”, “sad”, “hate”, and “bad”. The scores of the seed words equal 1 for positive words and -1 for negative words.

For each word in the seed set, WordNet relations (hyponym, hypernym, and antonym relations) can then be traversed and each encountered word w can be stored with a computed word score based on the score of the seed word, a diminishing factor, and an iteration step (i.e., the number of relations between the word and the seed word). The word score must be multiplied by -1 when

Algorithm 2. Word Sense Disambiguation

```

input : The to be disambiguated word  $w$  and the sentence  $s$  that contains the
         word
output: The sense  $sense$  of  $w$  with the highest semantic similarity to the words
         in the context
1  $targetSenses = \emptyset$ ; // Senses of the target word  $w$ 
2  $targetGlosses = \emptyset$ ; // Glosses of the word senses for  $w$ 
3  $senseScores = \emptyset$ ; // Scores of the word senses for  $w$ 
4  $bestSense = \emptyset$ ; // Best sense for  $w$ 
5  $bestScore = -1$ ; // Score for best sense for  $w$ 
6  $k = 8$ ; // Considered context around  $w$ 
7 // Retrieve the sequence of words starting  $k/2$  words to the left of
8 //  $w$  and ending  $k/2$  words to the right of  $w$ , excluding  $w$ 
9  $context = getContext(s, w, k)$ ;
10 // Look up and add all senses of POS noun and verb for  $w$ 
11  $targetSenses = getSenses(w)$ ;
12 foreach  $sense$  in  $targetSenses$  do
13 | // Retrieve the gloss of the sense and the glosses connected to
14 | // it through hypernym, hyponym, meronym, and troponym relations
15 |  $targetGlosses = \{targetGlosses, getRelGlosses(sense)\}$ ;
16 end
17 foreach  $word$  in  $context$  do
18 | // Look up and add all senses of POS noun and verb for  $word$ 
19 |  $senses = getSenses(word)$ ;
20 | foreach  $sense$  in  $senses$  do
21 | | // Retrieve the gloss of the sense and the glosses connected
22 | | // to it through hypernymy, hyponymy, meronymy, and troponymy
23 | |  $glosses = getRelGlosses(sense)$ ;
24 | | foreach  $gloss$  in  $glosses$  do
25 | | | foreach  $targetGloss$  in  $targetGlosses$  do
26 | | | | // Each overlap which contains  $N$  consecutive words
27 | | | | // contributes  $N^2$  to the gloss sense combination score
28 | | | |  $senseScores[targetGloss] += overlap(gloss, targetGloss)$ ;
29 | | | end
30 | | end
31 | end
32 end
33 foreach  $sense$  in  $targetSenses$  do
34 | | if  $senseScores[getGloss(sense)] > bestScore$  then
35 | | |  $bestScore = senseScores[getGloss(sense)]$ ;
36 | | |  $bestSense = sense$ ;
37 | | end
38 end
39 return  $bestSense$ ;

```

Algorithm 3. Propagating WordNet from a seed set

input : The WordNet files, a list *seedWords* of the words to propagate, their associated scores ξ , an integer K denoting the maximum number of iterations, and a double *limit* which defines the score given to words in the last iteration

output: A *sentLexicon* containing all propagated words with their computed sentiment scores

```

1 sentLexicon =  $\emptyset$ ;
2 synsets = retrieveSynsets(); // Retrieve all synsets in WordNet
3  $\delta = \text{limit}^{\frac{1}{K}}$ ;
4 foreach word in seedWords do
5   |  $\xi = \text{score}(\text{word})$ ;
6   | propWord(synsets, sentLexicon, word,  $\xi$ ,  $\delta$ , 1,  $K$ ); // See Algorithm 4
7 end
8 return sentLexicon;
```

traversing an antonym relation. We thus define the word scoring function as

$$\text{score}(w, \xi, \tau, \delta, k) = \xi\tau\delta^k, \quad \tau \in \{-1, 1\}, \quad k \in \{1, \dots, K\}, \\ -1 \leq \xi \leq 1, \quad 0 < \delta < 1, \quad (3)$$

with ξ the score of the seed word, τ indicating whether to inverse (-1) the score or not (1), δ the diminishing factor, and k the iteration step with a constraint of a maximum number of iterations denoted as K . The word score function weights each word encountered with a confidence measure that represents how likely it is that the given word has the designated positive or negative sentiment of the seed word. We define an iteration as traversing a relation between two synsets. On every iteration of the algorithm, words in the graph that are closely related to a seed word will get a higher score than those that are less related to a seed word. Thus, a word that is not a seed word, but is a neighbor to at least one seed word, will obtain a sentiment score similar to that of its adjacent seed words. At each iteration this will then propagate to other words. If a word is reached through a different path, then the word is assigned the score obtained from the shortest path between the considered paths between a word and any of the seeds (i.e., the score of the lowest iteration step). This process, yielding a lexicon containing word, word sense, POS tag, and a computed sentiment score, is detailed in Algorithm 3. This algorithm in turn utilizes a recursive word propagation function detailed in Algorithm 4.

3.3 PageRank-Based Propagation

Rather than by traversing semantic relations in WordNet, sentiment can also be propagated to synsets that are similar to predefined positive and negative synsets. Google's PageRank algorithm [7] can be used to exploit the semantic structure of WordNet to create a ranking of how closely synsets relate to positive or negative synsets [8].

Algorithm 4. Propagating a single word in WordNet (propWord)

input: A set containing the parsed WordNet *synsets*, a *sentLexicon* for storing the propagated words with their computed scores, a *word* to propagate, the score ξ of the *word*, a diminishing factor δ , an integer k denoting the current iteration step, and an integer K denoting the maximum number of iterations

```

1 if  $k \leq K$  then
2   if word.ReachedInIteration >  $k$  then
3     // If this word has not been reached through another, shorter
4     // path (default path length equals  $\infty$ ), proceed propagation
5     synsetsWithWord = getSynsets(synsets, word);
6     foreach synset in synsetsWithWord do
7       pos = getPOS(synset);
8       foreach syn in synset.Synonyms do
9         | addToLexicon(syn,pos, $\xi$ );
10      end
11      foreach relation in synset.Relations do
12        |  $\tau = 1$ ;
13        | if relation.typeOf(antonym) then  $\tau = -1$ ;
14        | foreach syn in relation.Synonyms do
15          | | propWord(synsets, sentLexicon, syn,  $\xi\tau\delta^k$ ,  $\delta$ ,  $k + 1$ ,  $K$ );
16          | end
17        | end
18      end
19      word.ReachedInIteration =  $k$ ;
20    end
21  end

```

The input to PageRank is the parsed set of synsets, and its output is a vector $\mathbf{a} = (a_1, \dots, a_N)$ with sentiment scores for all N (117,659) synsets in WordNet, where a_i represents the score for synset σ_i . PageRank iteratively computes vector \mathbf{a} using

$$a_i^k = \alpha \sum_{j \in B(i)} \frac{a_j^{k-1}}{|F(j)|} + (1 - \alpha)e_i, \quad (4)$$

where a_i^k denotes the sentiment score of the i -th entry of \mathbf{a} at the k -th iteration, B represents backward links, F represents forward links, e_i is an element of the constants vector $\mathbf{e} = (e_1, \dots, e_N)$ with a constraint such that $\sum_{i=1}^{|N|} e_i = 1$, and α is a control parameter with a range of $[0, 1]$.

When creating a sentiment sentiment lexicon with PageRank, we first need to retrieve all N synsets (glosses) from eXtended WordNet and store their forward links. We subsequently loop over each synset and set its forward links as backward links of the synsets to which the synset points. After parsing eXtended WordNet, \mathbf{e} is initialized, such that all elements other than the seed synsets are

Algorithm 5. PageRank-based propagation in WordNet

```

input : The eXtendedWordNet files, a list posSeedSynsets and negSeedSets
        of the positive and negative seed synsets and their sentiment scores
        (respectively), a double  $\chi$  denoting the termination condition, and a
        double  $\alpha$  which defines the control parameter
output: A sentLexicon containing all ranked words with their computed scores
1 sentLexicon =  $\emptyset$ ;
2 synsets = retrieveSynsets(); // Retrieve all eXtendedWordNet synsets
3 foreach synset in synsets do setBackwardLinks(synset);
4 foreach seedSet in {posSeedSynsets, negSeedSynsets} do
5   | e = initializeE(seedSet);
6   | ak = initializeA();
7   | ak-1 = initializeA();
8   |  $\theta$  = 0;
9   | while  $\theta < \chi$  do
10  | | foreach aik in ak do aik = calculateAi(ak-1);
11  | |  $\theta$  = calculateCosAngle(ak, ak-1);
12  | | ak-1 = ak;
13  | end
14  | synsets = assignScores(synsets);
15 end
16 sentLexicon = buildLexicon(synsets);

```

assigned a value of 0, while seed synsets are assigned proportional values such that the sum of elements in \mathbf{e} equals 1. Alternatively, elements in \mathbf{e} can be assigned values based on their scores in SentiWordNet, i.e., by dividing synsets' positivity (negativity) scores greater than 0 by the sum of positivity (negativity) scores and by assigning other synsets a score of 0. Sentiment scores in \mathbf{a} are initialized at $\frac{1}{N}$. The PageRank algorithm iteratively updates the sentiment scores \mathbf{a} and stops when the cosine of the angle between \mathbf{a}^k and \mathbf{a}^{k-1} exceeds χ . Following Esuli and Sebastiani [8], we use $\chi = 1 - 10^{-9}$ and $\alpha = 0.85$.

To create both a ranking for positivity and negativity, the PageRank algorithm must be run twice; one time where the elements of \mathbf{e} are set for a positive seed of synsets and second time for negative ones. When the algorithm – described in Algorithm 5 – is completed, each extracted synset contains both a positive and a negative score. The sentiment associated with a synset is the sum of the positive and negative scores, which results in a real number in the interval $[-1, 1]$.

3.4 SentiWordNet

Besides creating sentiment lexicons by exploiting the WordNet relations or similarities, one could also create sentiment lexicons by iterating over WordNet synsets and their associated glosses and assign sentiment scores to these synsets by means of a classifier. SentiWordNet [9] has been created in such a way and would thus be a convenient resource to use in order to assess the performance of such a sentiment lexicon creation method.

In SentiWordNet, each WordNet synset σ has been assigned scores on objectivity $Obj(\sigma)$, positivity $Pos(\sigma)$, and negativity $Neg(\sigma)$. The sum of these scores always equals 1 for each WordNet synset. A score $Obj(\sigma) > 0$ means a less subjective word and thus weaker sentiment scores in terms of $Pos(\sigma)$ and $Neg(\sigma)$.

The objectivity, positivity, and negativity scores for all 117,659 WordNet synsets have been computed by an ensemble of eight ternary classifiers. Each classifier has classified a synset as either objective, positive, or negative, based on a vectorial representation of the associated gloss. The overall scores for a synset have then been determined by the (normalized) proportion of classifiers that have assigned the corresponding labels to the synset. The scores thus obtained have been evaluated on a set of 1,105 WordNet synsets which have been scored in a similar fashion by five human annotators.

The classifiers used by SentiWordNet differ from one another in their training data as well as in their implemented machine learning approaches. Training sets have been generated from a seed set of positive and negative synsets, which have been expanded by traversing WordNet relations such as see-also and antonymy. The considered number of expansion steps varies amongst the classifiers between 0, 2, 4, and 6. Neutral synsets in the training data have been determined as synsets which are neither positive nor negative in both the expanded seed sets and the General Inquirer lexicon [12]. The considered machine learning approaches are Support Vector Machines (SVMs) and Rocchio classifiers.

The sentiment scores generated by an ensemble of these classifiers can, combined with their associated synsets σ , easily be utilized as a sentiment lexicon. However, in our approach, we ignore the objectivity scores, as they implicitly influence the positive and negative scores. Instead, we define our own sentiment score for σ as a single real number computed by subtracting $Neg(\sigma)$ from $Pos(\sigma)$, which results in a real number in the interval $[-1, 1]$.

4 Evaluation

We have implemented the framework presented in Sect. 3 in order to be able to assess the performance of our considered sentiment lexicon approaches on a corpus. The implementation was done in C#.Net in combination with a Microsoft SQL Server database. For lemmatization and word sense disambiguation, we used functionalities provided by the open-source C# WordNet.Net WordNet API². Our POS tagger – with an accuracy of 98.7% [13] – is based on SharpNLP³ and is provided to us by Teezir⁴.

The performance of our considered sentiment lexicon approaches was evaluated on a collection of 1,000 positive and 1,000 negative English movie reviews⁵, which have been extracted from movie review web sites by Pang and Lee [14].

² <http://opensource.ebswift.com/WordNet.Net/>

³ <http://sharpnlp.codeplex.com/>

⁴ <http://www.teezir.com/>

⁵ <http://www.cs.cornell.edu/People/pabo/movie-review-data/>

Table 1. Experimental Results

Method	Positive			Negative			Overall		
	Precision	Recall	F_1	Precision	Recall	F_1	Accuracy	Macro	F_1
WordNetRel	51.0%	94.3%	66.2%	62.3%	9.4%	16.3%	51.9%		41.3%
PageRankSeed	49.8%	86.8%	63.3%	48.6%	12.5%	19.9%	49.7%		41.6%
PageRankSWN	49.6%	43.0%	46.1%	49.7%	56.3%	52.8%	49.7%		49.4%
SentiWordNet	56.3%	84.3%	67.5%	68.8%	34.6%	46.0%	59.5%		56.8%

The review classifications have been derived from the accompanying numerical review scores. On this corpus, we have evaluated the performance of our simple sentiment analysis framework when using sentiment lexicons created by utilizing our discussed algorithm for traversing WordNet relations (WordNetRel), two PageRank-based propagation methods, and SentiWordNet. The considered PageRank-based methods differ in their values for ϵ ; in our first variant, we distribute the weights equally amongst the synsets that are part of our seed sets (PageRankSeed), whereas our second variant is bootstrapped based on the SentiWordNet scores of all synsets (PageRankSWN).

In our evaluation, several performance measures have been taken into account. For both the positive documents and the negative documents, we report precision, recall, and the F_1 measure. Precision is the percentage of the positively (negatively) classified documents which have an actual classification of positive (negative). Recall is the percentage of the actual positive (negative) documents which is also classified as such. The F_1 measure is a weighted average of precision and recall. We also report some statistics on our full corpus. We report the macro-level F_1 measure, which is the average of the F_1 scores of the two classifications, and the accuracy, which is the total percentage of correctly classified documents. Our results are reported in Table 1.

Creating a sentiment lexicon when propagating sentiment by exploiting WordNet relations (WordNetRel) yields an overall F_1 measure of 41.3%. This approach also classifies relatively more documents as positive than as negative and provides a correct classification in over 50% of the time. Conversely, both PageRank-based algorithms appear to misclassify more than half of the documents in our test corpus, albeit in different ways. PageRankSeed exhibits a high recall on positive documents, whereas PageRankSWN's recall on positive documents is relatively low. Conversely, PageRankSeed has a low recall on negative documents, whereas PageRankSWN exhibits a high recall on negative documents. This renders the performance of PageRankSWN more stable over all documents. Like most other considered approaches, sentiment lexicon creation based on machine learning techniques (SentiWordNet) turns out to exhibit a slightly biased performance on our corpus; relatively more documents appear to be classified as positive than as negative. Yet, the SentiWordNet approach yields a macro F_1 measure of 56.8% and moreover correctly classifies a similar percentage of all documents in our corpus. The observation of many of the approaches typically being biased towards positive documents may be explained by people tending to avoid negative words

when expressing negative opinions, thus rendering purely lexicon-based sentiment classification more difficult [15,16,17].

Our results show that in terms of accuracy, the SentiWordNet approach outperforms all other considered approaches. Conversely, the PageRank-based sentiment propagation method bootstrapped using SentiWordNet scores appears to be the most robust approach in that the difference between its F_1 measures for positive and negative documents is smaller than is the case for the other considered approaches.

5 Conclusions and Future Work

In order for today's businesses to, e.g., keep a close eye on how their brands or products are perceived by the market, recent developments in the field of sentiment analysis may prove to be crucial for business information systems facing the challenge of extracting relevant information from the massive amount of data available through the Web. Many existing sentiment analysis approaches rely on lexical resources containing words and their associated sentiment. Creating such resources may be a cumbersome task, yet several methods for automated sentiment lexicon creation have already been proposed.

In this paper, we have performed a corpus-based evaluation of a number of distinct automated sentiment lexicon creation methods exploiting vast, readily available lexical resources. We have considered an algorithm exploiting semantic relations in a lexical resource in order to propagate the sentiment of a seed set of words, as well as a PageRank-based algorithm propagating the sentiment of a seed set of words to related words. We have also considered a machine learning approach based on Support Vector Machines. The latter approach turns out to outperform the others in terms of accuracy and macro F_1 measure. However, creating a sentiment lexicon with a PageRank-based propagation algorithm appears to result in the most robust sentiment classifier.

In future work, we would like to consider in our comparisons different languages (e.g., Dutch, Romanian, etcetera). Other possible directions for future work include the development and analysis of novel sentiment lexicon creation methods, focusing not only on existing lexical resources, but on, e.g., texts annotated for sentiment as well.

References

1. Heerschop, B., van Iterson, P., Hogenboom, A., Frasinca, F., Kaymak, U.: Analyzing Sentiment in a Large Set of Web Data while Accounting for Negation. In: 7th Atlantic Web Intelligence Conference (AWIC 2011), pp. 195–205. Springer, Heidelberg (2011)
2. Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press, Cambridge (1998)
3. Fellbaum, C.: English Verbs as a Semantic Net. *International Journal of Lexicography* 3(1), 259–280 (1993)

4. Kim, S., Hovy, E.: Determining the Sentiment of Opinions. In: 20th International Conference on Computational Linguistics (COLING 2004), p. 1367. ACL (2004)
5. Hu, M., Liu, B.: Mining and Summarizing Customer Reviews. In: 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2004), pp. 168–177. ACM, New York (2004)
6. Lerman, K., Blair-Goldensohn, S., McDonald, R.: Sentiment summarization: Evaluating and Learning User Preferences. In: 12th Conference of the European Chapter of the ACL (EACL 2009), pp. 514–522. ACL (2009)
7. Brin, S., Page, L.: The Anatomy of a Large-Scale Hypertextual Web Search Engine. In: 7th International World-Wide Web Conference (WWW 1998), pp. 107–117. Elsevier, Amsterdam (1998)
8. Esuli, A., Sebastiani, F.: PageRanking WordNet Synsets: An Application to Opinion Mining. In: 45th Annual Meeting of the Association of Computational Linguistics (ACL 2007), pp. 424–431. ACL (2007)
9. Esuli, A., Sebastiani, F.: SENTIWORDNET: A Publicly Available Lexical Resource for Opinion Mining. In: 5th Conference on Language Resources and Evaluation (LREC 2006), European Language Resources Association (ELRA), pp. 417–422 (2006)
10. Lesk, M.: Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. In: 5th Annual International Conference on Systems Documentation (SIGDOC 1986), pp. 24–26. ACM, New York (1986)
11. Dao, T., Simpson, T.: Measuring Similarity between Sentences. Technical report, WordNet.Net (2005),
http://wordnetdotnet.googlecode.com/svn/trunk/Projects/Thanh/Paper/WordNetDotNet_Semantic_Similarity.pdf
12. Stone, P., Dunphy, D., Smith, M., Ogilvie, D.: The General Inquirer: A Computer Approach to Content Analysis. MIT Press, Cambridge (1966)
13. Buyko, E., Wermter, J., Poprat, M., Hahn, U.: Automatically Adapting an NLP Core Engine to the Biology Domain. In: 9th Bio-Ontologies Meeting and the Joint Linking Literature Information and Knowledge for Biology (ISMB 2006), pp. 65–68. Oxford University Press, Oxford (2006)
14. Pang, B., Lee, L.: A Sentimental Education: Sentiment Analysis using Subjectivity Summarization based on Minimum Cuts. In: 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004), pp. 271–280. ACL (2004)
15. Dave, K., Lawrence, S., Pennock, D.: Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews. In: 12th International World Wide Web Conference (WWW 2003), pp. 519–528. ACM, New York (2003)
16. Taboada, M., Voll, K.: Extracting Sentiment as a Function of Discourse Structure and Topicality. Technical Report 20, Simon Fraser University (2008)
17. Turney, P.: Thumbs up or Thumbs down? Semantic Orientation Applied to Unsupervised Classification of Reviews. In: 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002), pp. 417–424. ACL (2002)

Matching Organizational Structure and Social Network Extracted from Email Communication

Radosław Michalski^{1,2}, Sebastian Palus¹, and Przemysław Kazienko¹

¹ Institute of Informatics at Faculty of Computer Science and Management

Wrocław University of Technology

Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland

² Research & Engineering Center

Strzegomska 46 B, 53-611 Wrocław, Poland

{radoslaw.michalski, sebastian.palus, kazienko}@pwr.wroc.pl

Abstract. The following paper presents the concept of matching social network and corporate hierarchy in organizations with stable corporate structure. The idea allows to confirm whether social position of an employee calculated on the basis of the social network differs significantly from the formal employee role in the company. The results of such analysis may lead to possible company management improvement enabling to gain a competitive edge. In order to perform this task the authors have made experiments with the use of two real-life datasets: Enron and mid-sized manufacturing companies showing which social network metrics may be suitable to match organizational structure and social network with good results.

Keywords: social network analysis, organizational design, enterprise management, corporate social networks, employee position evaluation.

1 Introduction to Social Networks and Social Network Analysis

1.1 Social Networks

Social network consists of *a finite set or sets of actors and the relation or relations defined on them* [15]. To help understanding this definition of a social network, some other concepts that are fundamental in this case should be explained. An actor is a *discrete individual, corporate or collective social unit* [15]. This can be a person in a group of people, a department within a company or a nation in the world system. Actors are linked to each other by social ties and these relations are the core of the social network approach.

Social networks are presented using graph structures, where nodes are actors and edges are connections between them. Hence, all graph theory methods and measured can be applied. A graph may be undirected, which means that there is no distinction between the two vertices associated with each edge, or its edges may be directed from one vertex to another. A graph is an ordered pair $G:=(V,E)$, where V are vertices or nodes and E are edges.

Social networks are often used to examine how organization employees interact with each other, characterizing the many informal connections that link executives together. For example, power within organizations often comes more from the degree to which an individual within a network is at the center of many relationships than actual job title. Social networks also play a key role in hiring, in business success, and in job performance. Networks provide ways for companies to gather information, deter competition, and collude in setting prices or policies [15].

1.2 Corporate Social Network Analysis

The power of social network analysis lays upon its difference from traditional social scientific studies, which assume that it is the attributes of individual actors - whether they are friendly or unfriendly, motivated or unmotivated, etc. - that matter. Social network analysis produces an alternate view, where the attributes of individuals are less important than their relationships and ties with other actors within the network. This approach has found to be useful for explaining many real-world phenomena, but leaves less room for individual agency, the ability for individuals to influence their success, because so much of it rests within the structure of their network.

The idea of corporate social network analysis consist of application of social network analysis methods (SNA) to social networks built on various company communication and event data. Such analysis can provide an additional information on groups, relations and information flow in a company which may be further used for improving company management in various ways [16]. Also regularities and anomalies in processes, key persons extracted from the social network and many other factors may be found and all these can be considered crucial in finding competitive edge for organizations.

Typically, social network actors are persons, i.e. customers, social networking sites users, employees, etc. What may be interesting is that corporate SNA can consider other type of actors – functional actors, i.e. warehouse workers threat as one entity or even non-human actors, like IT system, depending on data source availability. Thus, there may be different networks built based on the same data source and analyses may be performed using various combinations of entities. That makes corporate SNA task interesting, but, in some aspects, also complex and challenging.

As stated, corporate SNA uses mostly same tools as typical SNA – centrality metrics, clustering, group analysis, etc. However, due to its nature, corporate SNA takes into account also some other company information, such as process definitions and HR information. Conjunction of them can facilitate improvement of company organization and management, however, it also often requires development of new combined data analysis methods.

The following paper presents another point of view in the corporate SNA focusing on matching organizational structure and social network extracted from email communication. The goal is to prove that SNA can be valuable source of information about companies and in this case it can be used to deepen the knowledge about relation between formal position of employees in corporate hierarchy and real but informal role in social network. That kind of knowledge properly used may become another decision factor in company management allowing organizations to gain competitive edge.

Section 2 of this paper presents problem description, Section 3 characterizes related work in the field of corporate SNA. Section 4 introduces the concept of matching organizational structure and social network while in Section 5 authors present performed experiments and results achieved using real-life business cases to evaluate the introduced concept, what is further discussed in Section 6 of the paper. Conclusions and proposed future work directions are presented in Section 7.

2 Problem Description

Trying to gain a competitive advantage, companies are constantly searching for the solutions that would enable to beat their market opponents. It may be crucial to discover, among many ways to increase company effectiveness, own potential, hidden in corporate social network. The knowledge derived from this capacity, when properly extracted and interpreted, may lead to various positive effects in terms of company management [9], [13].

Company managers may often ask the question about the proper alignment of its employees in organization structure. The problem may be particularly important in fast-growing organizations, where medium-level management team may be chosen without prior adequate preparation and without the use of clear HR tools. Companies, in which some of employees are awaiting retirement, are the other example where such knowledge may be vital. If the company decides to search internally for the successor, social network analysis may become helpful in that task. It may be also helpful in extracting some prospective problems in the company, like managers avoiding communication with other employees within the organization.

The social network, that is built on the basis of employee communication logs, may be found useful in above tasks, because it can provide information about social network leaders, communication gaps and anomalies. However, the problem is what factors in social network analysis results should be considered as important ones and used in further company management decisions. The other problem is how to perform such analysis in order to regard it as meaningful and representative.

The idea proposed in this paper describes the effectiveness of matching organizational structure and social network, extracted from email communication and further possibilities to use those results as a base to redesigning company structure. The opportunity to compare key persons in social network and organization structure allows the management to answer the question about proper employees alignment in the organization they are responsible for.

3 Related Work

The task of building corporate social network using various data sources is well described, because this is a preliminary step in performing any further analysis. As a source for social network may be used: e-mail communication logs [1], ERP system logs [14], instant messaging systems [11], phone call records and many others. However - what is worth pointing out - social network may be even built without digital communication traces – employees sharing the same office or working on the

same floor are also related. The tasks for such information extraction may be harder and long-lasting than automated logs analysis, but it is valuable to consider such data sources either (i.e. as another social network layer).

The idea presented in this paper combines evaluation of employee position in social network and comparison with his formal position in organization. Commonly available and widely analyzed by other researchers Enron email logs are one of the datasets used in experiments [2]. The general concepts of node position evaluation in social networks were presented in [5] and the organization approach in social network analysis were shown in [4], [9], [12], all of them were based on Enron data.

Wide survey of other methods of researching organizational systems using SNA is presented in [16] and an interesting set of techniques of to discover and optimize organizational models is presented in [13].

The basic concept of the idea presented by this paper authors have been described by them in [10].

4 Concept of Matching Organizational Structure and Social Network

As a possible solution for solving problems described in Section 2 there is a concept of matching organization structure and social network presented. It consists of corporate social network building, using the available data and its further analysis and comparison with formal organization structure (see Figure 1).

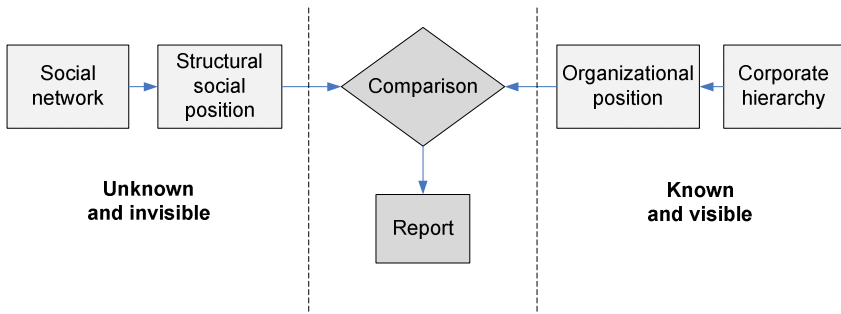


Fig. 1. The idea of matching organizational structure and social network

As a comparison result it is expected to know how many and which employees match to their level of corporate hierarchy. The authors have focused on answering the question whether there were any metrics or metric combinations that could be used in SNA to achieve best match between social network position and organizational position of an employee. If they are found and proved reliable, they may be used in other companies to support corporate management in human resources area. Presented idea focuses on matching employee level in corporate hierarchy rather than exact position. The approach follows from the fact that many employees in corporate hierarchy may have the same position in rank, what rarely takes place in social network. That is why it is hard to compare both ranks using well known comparison methods, i.e. using Kendall rank correlation [6].

The whole process consists of following steps:

- source data pre-processing
- social network building
- social network metrics calculating
- social network and corporate hierarchy comparison

The social network may be build using one or more layers, depending on the available source data. As the second input in comparison process there is the need of corporate hierarchy. It is worth mentioning that the analyzed period should not be too short and also well chosen, because in other case the built social network would not be reliable. For example, if the analyzed period refers to half a month - June, many employees would not even exist in social network due to their holidays. It is also important to have as much as possible information about corporate structure changes – dismissals, promotions, long sick leave data etc. All of those may be beneficial in explaining any found anomalies.

Node reduction and identification is the preliminary step in data pre-processing. If the source data are composed of external connections (i.e. emails sent outside analyzed company), they should be deleted. However, in order to provide additional information in SNA process, number of external connection can be used as the label of a node (this idea is not covered further in this paper). Later on all multiple instances of a node in the social network source data should be transformed to one instance (i.e. merging e-mail aliases) and all of those nodes must be later matched to company employees. It may certainly happen that an employee would not exist in social network because they have not used the analyzed medium.

The process of building social network consists of choosing the graph type (directed or undirected) and weight calculation method between nodes. The authors decided to build directed graph with the weight of an edge between node i and j is as follows:

$$w_{ij} = \frac{\sum e_{ij}}{\sum e_i} \quad (1)$$

where $\sum e_{ij}$ is the number of e-mails sent by node i to node j and $\sum e_i$ is a total number of e-mails sent by member i . It means that weight w_{ij} focuses on local neighborhood of an employee rather than on global network characteristic.

The next step is to calculate social network metrics used in comparison. It has been shown in Section 5 that some metrics are more suitable in matching social network to corporate hierarchy.

Final step of the process consists of comparison of social network and corporate hierarchy. It is accomplished by answering the question how many employees in social network rank are a good fit for certain level in corporate hierarchy. The basic result is made up of percentage coverage of each management level by correspondent social network rank employees. If a significant differences are found, more detailed analysis may be performed, even focusing on each employee when needed.

So as to test that the idea of matching corporate structure and social network, two networks have been build and analyzed. One is based on mid-size manufacturing company located in Poland source data and the other one uses data gathered from Enron corporation (see Section 5).

5 Experiments

5.1 Manufacturing Company

The first analyzed company is a manufacturing company located in Poland. The company employs 300 persons, whereas 1/3 are clerical workers, the rest - laborers. The period analyzed was half a year. The type of organizational structure is functional [3]. However, due to organization operating model and its consequences to organizational structure clarity as well as logs interpretation possibility, only a subset of organization have been chosen for current analysis: 49 clerical employees not directly related to manufacturing process. Three-level management structure exists in the selected company part: management board (2 persons), managers (11 persons) and regular employees (36 persons) and they work in twelve different departments. There were no organizational changes during the analyzed period.

Email logs were source data used to build social network . Because of email logs structure, there was no distinction between *To*, *CC* and *BCC* recipients. The resulting set of data contained 11,816 emails in total. Figure 2 shows visualization of the built social network in the analyzed company by using the ORA tool [8].

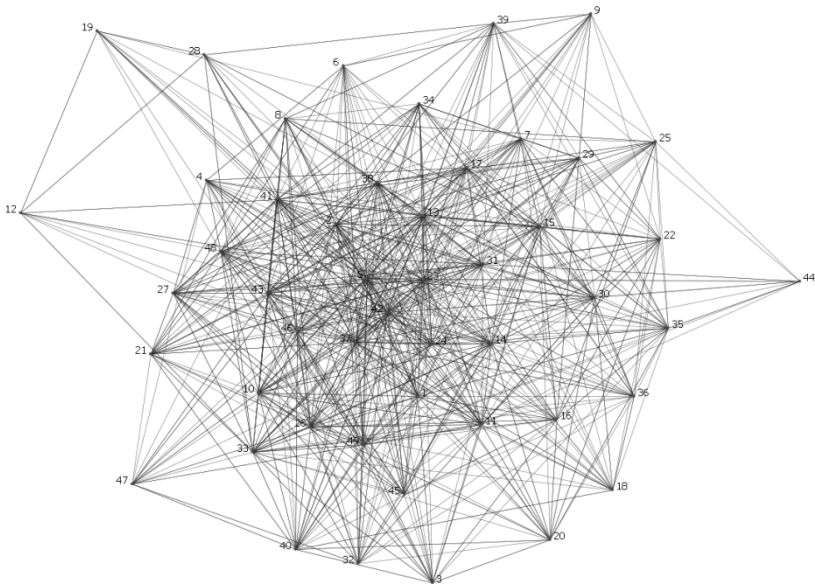


Fig. 2. Visualization of the social network in the manufacturing company

5.2 Enron

Enron, the another analyzed company, was one of the largest energy corporation around the world. It become especially famous worldwide in 2001 due to financial manipulation scandal. The Enron email dataset was made public by the Federal Energy Regulatory Commission during its investigation. The email dataset had a number of integrity problems which were corrected by researchers at MIT and SRI International [2]. The Enron hierarchy structure is still not publicly available. However, there are sources which can provide information concerning plenty of job positions of given employees and their department or division [12]. Because only some of employees existed in email corpus, authors have decided to analyze social network building only within limited set of managers and employees which positions - with high probability – were known.. The analyzed period comprised two years and due to limited information the authors assumed that there had been three level management (CEO, directors/managers, employees). Figure 3 shows organization structure of selected Enron part.

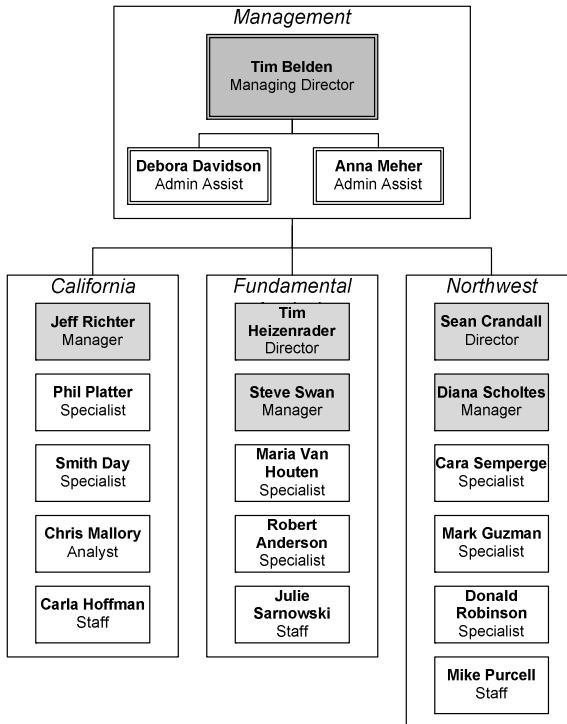


Fig. 3. Part of the Enron hierarchy used for analysis

5.3 Results

As stated in Section 4, the graphs built were directed with weight defined in Equation 1. The authors have decided to calculate most popular metrics used in social

network analysis within each network their built: in-degree centrality, out-degree centrality, centrality betweenness, centrality closeness, clustering coefficient Watts-Strogatz and centrality eigenvector [7]. Ranks of social network position using above metrics have been compared to organization structure as described in Section 4 to gain information how well the organization levels had been matched. Because it has been found that none of the used metric and tested metric combinations were able to make the clear distinction between the first (management board) and the second management level (managers and directors), further analysis has focused on distinction management as a whole from regular employees. The results are presented in Table 1 – all the metrics mentioned in this table are defined in [7].

Table 1. Accuracy of management level matching while using various social network metrics

	Percentage of the first and the second management level matched		Percentage of regular employees matched	
	Manuf. comp.	Enron	Manuf. comp.	Enron
In-degree centrality	85	67	94	85
Out-degree centrality	62	50	86	77
Centrality betweenness	38	33	78	69
Centrality closeness	46	33	81	69
Clustering coefficient	15	17	69	62
Centrality eigenvector	77	67	92	85

The results show that only some metrics are capable to make good distinction between management and employees. The best of them are: in-degree centrality and centrality eigenvector. It proves that the basic distinction between managers and others is based not on outgoing relations rather than on incoming relations (in-degree centrality) and the importance of employees contacting with us (eigenvector centrality). What is also interesting, none of tested metrics could not placed CEO or board members in the first place of the rank. It may be explained in few ways – top level managers did not contact with the others directly by email, rather by his assistants. The other explanation is that top managers were not using evaluated medium so often – in that case there is a need to gather other sources for having more social network layers.

6 Discussion

The idea of matching organizational structure and social network is regarded by the authors as another possible way to improve overall company management. The idea focuses on the comparison of calculated node position ranks using chosen metrics within organization structure. While choosing the metrics that gave good results in tested datasets and applying those in analyzed company, it may be found that similar

level of management team and employees were properly assigned to their management levels. If the results differ significantly, some more sophisticated analysis might be needed to answer the question why real-life communication and hierarchy do not necessarily cover organization chart. The reasons may differ: not the most important social network source was analyzed, the relations were changing too fast to give stable point of view or the company chose inappropriate persons to hold management positions.

It must be clearly stated that the analysis will apply in more effective way to companies with stable (probably functional) organization design [3], because other designs, such as matrix or horizontal design would not allow to create a hierarchy chart easy comparable with social network ranks.

There might be also another usage of proposed method. As mentioned in Section 2, the choice of new leaders in organization can be supported by described set of methods through recognition of those employees who belong to the upper level of management team (having compared to organization structure) as prospective candidates.

There is one more, maybe more controversial, application field of considered technique. If someone wishes to uncover corporate hierarchy or at least wants to know possible managers of this company using stolen (or somehow possessed) email logs and chooses proposed metrics, they may discover corporate managers in easier, faster and safe (passive) way. Later on those potential managers may be the target of industrial espionage or other actions.

Despite all the techniques regarding core data analysis, that may be very ambitious for SNA experts, the real challenge for companies is to properly interpret and make valuable use of the achieved corporate SNA results.

7 Conclusions and Future Work

Two real-world analyzed cases have proved that corporate social network analysis may be the way to get another point of view on the company. The different channels of communication between employees may be used as the data source to extract corporate social network and the results of such a network analysis, compared to organization design, can be used as a valuable decision support tool leading to company improvement. Overall, social network approach to the problem of corporate management appears to be very helpful, however, the analysis needs to be well interpreted in order to improve performance and social health of the company. This is only a tool. Still, human resources have to be managed by humans.

Future research in this area will focus on development of new reliable metrics for quantitative comparison and matching social network structures with corporate hierarchies. The basic goal is to develop such metric that will provide even better results than two regard as the best in this paper. As stated in Section 6, the proposed approach evaluated stable organizational structures and while thinking about more complex structures like matrix or horizontal ones [3] the analysis will be definitely harder to perform. However it is worth to try to develop a method suitable for such kind of structures. There is also a strong need to compare achieved results with other datasets (social network source and corporate hierarchy chart), which, sadly, are hardly available.

Acknowledgments. The work was supported by The Polish Ministry of Science and Higher Education, the development project 2009-11, the research project 2010-13 and the training within the "Green Transfer" project co-financed by the European Union from the European Social Fund.

References

1. Bird, C., Gourley, A., Devanbu, P., Gertz, M., Swaminathan, A.: Mining email social networks. In: Proceedings of the 2006 International Workshop on Mining Software Repositories (MSR 2006), pp. 137–143. ACM, New York (2006)
2. Cohen, W.: Enron Email Dataset, <http://www.cs.cmu.edu/~enron/>
3. Daft, R.L.: Organization Theory and Design, 10th edn. Cengage Learning, Cincinnati (2009)
4. Hossain, L.: Effect of organisational position and network centrality on project coordination. *International Journal of Project Management* 27(7), 680–689 (2009)
5. Kazienko, P., Musiał, K., Zgrzywa, A.: Evaluation of Node Position Based on Email Communication. In: Control and Cybernetics, vol. 38(1), pp. 67–86. Polish Academy of Sciences, Warsaw (2009)
6. Kendall, M.G.: Rank Correlation Methods. Charles Griffin and Co., London (1948)
7. Musiał, K., Kazienko, P., Bródka, P.: User Position Measures in Social Networks. In: Third SNA-KDD Workshop on Social Network Mining and Analysis held in conjunction with The 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2009, Paris, France, June 28, vol. (6), ACM Press, New York (2009), Article no. 6
8. ORA tool, CASOS at Carnegie Melon University, <http://www.casos.cs.cmu.edu/projects/ora/>
9. Palus, S., Bródka, P., Kazienko, P.: How to Analyze Company Using Social Network? In: WSKS 2010. Communications in Computer and Information Science, vol. 111, pp. 159–164. Springer, Heidelberg (2010)
10. Palus, S., Kazienko, P., Michalski, R.: Evaluation of Corporate Structure Based on Social Network Analysis. In: Lytras, M.D., De Pablos, P.O., Damiani, E. (eds.) Semantic Web Personalization and Context Awareness: Management of Personal Identities and Social Networking. IGI-Global, Hershey (in press, 2011)
11. Resig, J., Santosh Dawara, C.M.H., Teredesai, A.: Extracting social networks from instant messaging populations. In: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD) Workshop on Link Analysis and Group Detection (2004)
12. Rowe, R., Creamer, G., Hershkop, S., Stolfo, S.J.: Automated Social Hierarchy Detection through Email Network Analysis. In: Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 Workshop on Web Mining and Social Network Analysis, pp. 109–117 (2007)
13. Song, M., van der Aalst, W.M.P.: Towards comprehensive support for organizational mining. In: Decision Support Systems, vol. 46(1), pp. 300–317. Elsevier Science Publishers B. V., Amsterdam (2008)
14. Wagle, D.: The case for ERP systems. *The McKinsey Quarterly* 2, 131–138 (1998)
15. Wasserman, S., Faust, K.: Social network analysis: Methods and applications. Cambridge University Press, New York (1994)
16. Zack, M.H.: Researching organizational systems using social network analysis. In: Proceedings of the 33rd Annual Hawaii International Conference on System Science (2000)

Application Specific Communication Stack for Computationally Intensive Market Research Internet Information System

Peter Kurz and Andrzej Sikorski

Models and Methods, TNS Infratest Forschung GmbH, Landsberger Str. 284
D-80687 München, Germany

`peter.kurz@tns-infratest.com`

Technical University Poznań, Faculty of Electrical Engineering, Piotrowo 3A,
60-965 Poznań, Poland

`andrzej@et.put.poznan.pl`

Abstract. A robust and reliable transactional processing is a key quality factor in computationally intensive, multilayer information systems. We give three design patterns that model reliable session and transaction management in transactional web applications. These are: session timeout, server default action and split client-server state representation. Only the first design pattern can be successfully implemented with the standard WCF [1] facilities, thus we also include our optimized communication stack. The presented method significantly improves the operational capabilities of the state-of-the-art WCF [1] in respect of robustness (reliable finalization), flow control flexibility (synchronous, asynchronous) and efficiency (a significant performance boost).

Keywords: web services, transactions, session management, design patterns.

1 Introduction

Choice-based conjoint (CBC) analysis is considered an essential tool in market research industry [2]. It is a simple yet powerful survey method, such that respondents are presented with multiple product profiles and asked to choose the preferred one (maximizing the respondent utility). The main focus among the workers in the field has been to improve the design of product profiles with the aim of maximizing the information gained from the survey. Important academic research in the field led to efficiency improvements, yielding more information from fewer respondents. However, that gains come at a cost of huge computational complexity, that requires adoption of sophisticated algorithms and computational power. In this paper we give a report on the web service integrated respondent survey system based on the Sonnevend [3] polyhedral convex optimization referencing a global, hierarchical state maintained in a distributed database (on hierarchical aspects of utility estimation c.f. [2],[4],[5]).

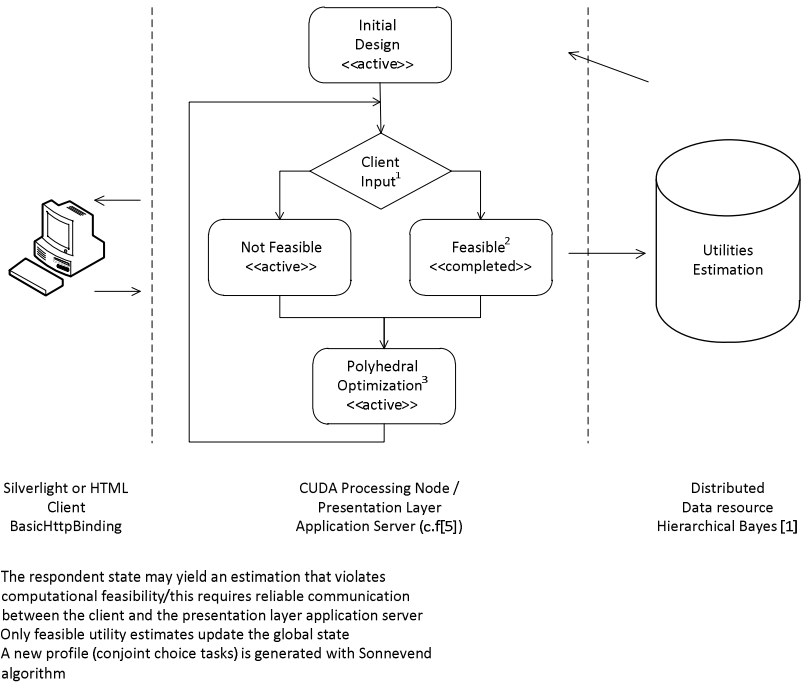


Fig. 1. The architecture of an adaptive market research multilayer survey system. The application server includes CUDA [6] computational and Silverlight presentation layer components. The survey session can be either in completed (feasible) or active state.

From the software system perspective the adaptive survey process and the underlying convex optimization can be viewed as a regular database transaction that includes multiple reads and updates (Fig.1.). The transaction status is a combination of user interaction, client application state variables and the current state of the optimization process. To handle the state transitions successfully reliable communication channels are required. There are two problems with WCF, however. Firstly, if the application terminates abruptly, which is the case when the browser shuts down, the standard WCF channels are no longer functional and the client is unable to send the finalization message to the server. There seems to be no workaround for this issue within the pure WCF [1]. Secondly, all communication WCF channels are asynchronous. These might seem to be superior to the synchronous counterparts, and in most cases they are. However, the synchronization and sequential control cannot be easily implemented with objects available in the *System.Threading* namespace (e.g. *AutoResetEvent* c.f. MSDN), as the initialization and completion routines of an asynchronous call run in the single UI thread [7]. Our solution takes advantage of the client host characteristics which offers a more robust environment that allows persistent state maintenance and reliable communication.

2 Completion Design Patterns

Let us now consider 3 simple but effective design patterns that allow reliable completion of database transactions initiated by a Web/Silverlight client application. These are: timeout based session termination, default server-side action and, the most featured one, state representation split between client and server. The first one, session timeout is in fact the only option presently available within the standard Silverlight communication framework. The two other require the web service call method described in this work.

The state model, assumed in this work, of a transactional Silverlight client is given in Fig. 2. The client sends SOAP messages to the application server, that can either initiate a transaction (*BeginTransaction*) or complete it. There are 2 possible states: *Active* or *Completed*. This can be also understood in a more general way, with *Active* state representing any pool of allocated resources, not just an active database transaction, and the locks acquired thereby. In particular an active transaction models a optimization process in an infeasible state.

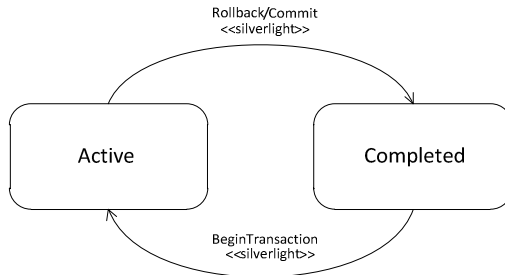


Fig. 2. The generic model of the application server state (c.f. [8]). It is assumed that the client session can be either in a active or completed transactional state. Our objective is to ensure that when the client terminates proper finalization is performed, in particular when browser is abruptly closed.

Operation other than *BeginTransaction*, *Rollback/Commit* are irrelevant to our model, as we are only interested in actions that require some finalization. Our objective is to ensure that after a transaction is initiated and the application server goes into the Active state, it will be eventually finalized in a proper manner, according to the observed business rules.

2.1 Timeout Based Session Termination

The first option considered herein is to maintain a timer for each active session. This timer is started at the server for each newly created session and restarted each time a new message from a client arrives. If the time after the last message exceeds the designated interval the Timeout event occurs. Subsequently the proper actions on the client side are taken and the session terminates. This solution does not require any cooperation from the client.

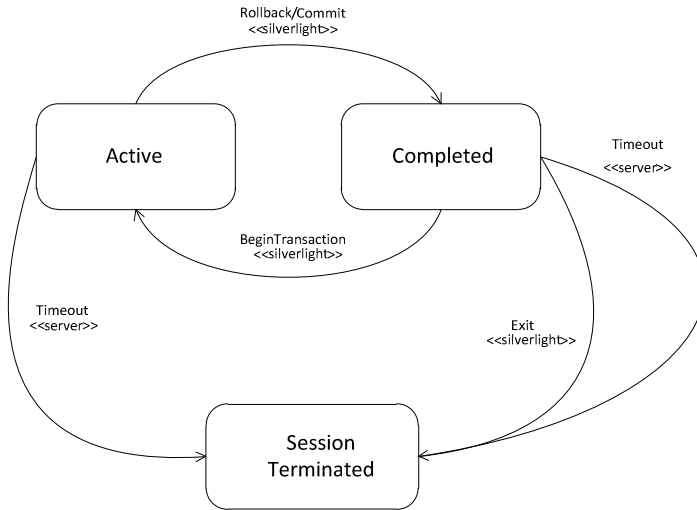


Fig. 3. The state diagram for the timeout design pattern. For each client session a timer is maintained, which fires an event when the designated time elapses.

If the session does not include an active transaction, timeout can be considered irrelevant. Nevertheless it can still occur for a session without any active transaction (i.e. *Completed* state) . In such a case timeout offers an opportunity to remove obsolete sessions and offers an opportunity to perform some additional housekeeping (e.g. finalization of allocated resources). The *Timeout* transition between *Completed* and *Session Terminated* states represents this. In both cases this transition is tagged with «server» stereotype, which reflects the fact that it is server side responsibility to handle it.

If the client transaction in a session is not active, the session can terminate leaving the database, or other resource, in a consistent state. The *Exit* transition in Fig. 3. represents a regular WCF asynchronous call that informs the application server that the client is terminating.

2.2 Default Server-Side Action

Another option is to designate a default action that should be performed at server side when a client shuts down. This method requires a reliable communication channel, which must be still available after the WCF facilities has been already closed. This channel is necessary to pass the *Finalize* message to the application server. In response the server initiates the default action, that can possibly be a transaction rollback.

Evidently the preferred way to shut down an application in the *Active* state is to go through the *Rollback/Commit* and *Exit* transitions. In this case the finalization is triggered solely by the standard WCF communication as both transition arcs are tagged with «silverlight». Unfortunately such a scenario is not what Silverlight and the web browser can guarantee.

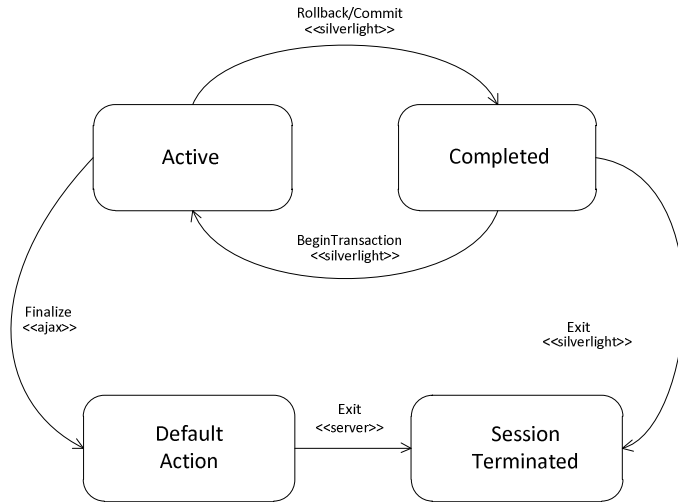


Fig. 4. Default action design pattern. When the Finalize message arrives at the server, default action for the session is taken and the session subsequently terminates.

The client application can be shut in an inadvertent way firing the Silverlight *Exit* event (which WCF is unable to handle, thus it is missing in the Fig. 4.) when the session is in the *Active* state. Although WCF is now not available the client can send its *Finalize* message through the channel available within the web browser – which is still active even after WCF shuts down. Because *Finalize* is a regular SOAP message, it can be send either through «ajax» channel (web browser) or standard WCF (not present in Fig.4.). We are going to discuss this opportunity in Sec. 4.

After Finalize message is received, the server performs appropriate handling (*Default Action* state) then the internal event *Exit* is issued and the session terminates. Evidently default action for each session can be designated independently. It can be viewed as just another business rule implemented by the application server being a part of a regular data processing. It is also worth noting, that it can be modified by regular SOAP calls as whenever new message arrives a correct completion action may vary according to the current session state.

Seemingly, this pattern is only a minor modification to the session timeout as these two state charts look similar. Yet, there are two important performance gains. Firstly, the server session is terminated immediately after client shuts down, releasing all allocated resources, in particular database locks. Secondly, the session management gets much simpler. The timeout design pattern requires that the session manager must either perform a continuous polling of all active entries or order the entries after the pending expiration time.

2.3 Split Client-Server State Handling

The 3rd pattern is the most featured and enables the most flexible processing. We named it “a split state” as now the session state is maintained independently at both ends of the channel. This is reflected in Fig. 5. with labels *Client* - and *Application*

Server processing. The idea is that each time the desired completion action changes, client part of the session is updated. This is correct, provided a reliable state transfer is guaranteed. See now, that the *Finalize* method includes one parameter (which can be a composite one) – representing, at the moment when the client is closing, its state.

The 2 arcs corresponding to switching between *Rollback* and *Commit* states represent events occurring at the client side – thus involves no WS calls.

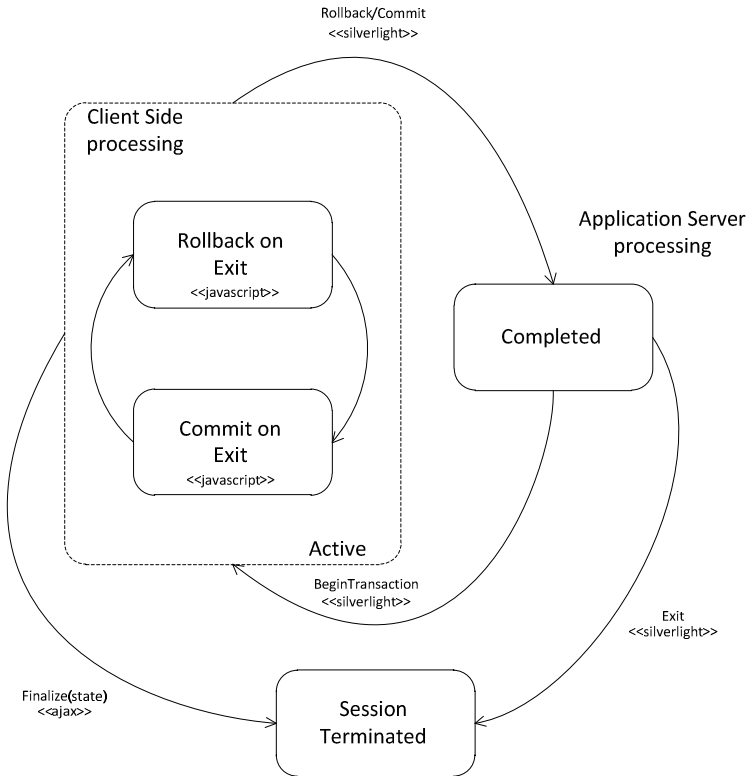


Fig. 5. Split-state management. The client updates its state which is guaranteed to be eventually submitted to the application server.

The client side state can be maintained in 2 ways. The first method is to keep the state data in javascript variables, updated each time the session state changes. Silverlight code can easily access the members of its host html page and manipulate them in any way. The evident advantage of this approach is that the sending of *Finalize* message will be now solely the web browser responsibility. When browser window is closed the unload event is fired, therefore a javascript code can handle it, sending *Finalize* message which now includes the representation of the client state. The other option is to keep the current state in the application variable and make it available to java javascript code when needed.

Silverlight supports communication between its application and hosting html page and the state representation can be easily passed. One option is to implement an

application class exposing accessors tagged with the *ScriptableMember* attribute. However in our production system we have chosen another way, that is, the current state is passed directly to the javascript routine when *Exit* event occurs.

3 Synchronous Web Service Calls

In this section we are going to describe the main technical component of our solution, namely the utilization of the AJAX /HttpXML object as a communication facility for the reliable message passing. Due to the limitation of Silverlight subset of the .NET framework (COM/DCOM including IDispatch automation is excluded from Silverlight [9]), the AJAX object is not available from within the client application. This is not an issue, as javascript has not such limitation and, as we have already pointed in the previous section, the application and html page can freely communicate.

Our objective is to maintain the maximum compatibility between WCF and AJAX communication. That is, the client side of the communication channel has to remain transparent to the application server. Consequently, the Silverlight SOAP compliant web service are unaware of the call method used at the client side. Both methods (WCF and AJAX) can consume services published via *basicHttpBinding* [9] Fig.6.

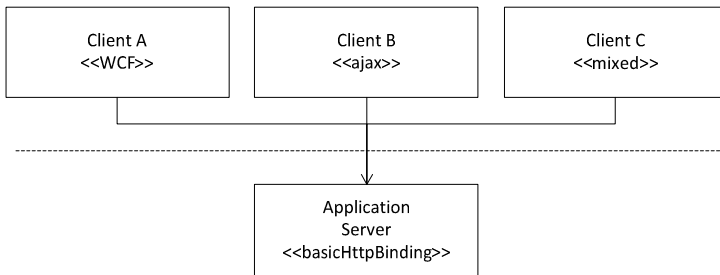


Fig. 6. Different types of client can access a service published via uniform binding (*basicHttpBinding* corresponds to SOAP in Microsoft terminology). The Client C uses both methods, thus stereotyped «mixed».

Client A is a standard WCF application, Client B uses our method while Client C (stereotyped as mixed) uses both of them (Fig. 6.). The third type of application probably makes little sense as we will see in the next section AJAX call performs much better than WCF. Thus, with AJAX facility already there, it is reasonable to give up WCF at all. As far as, asynchronous calls are concerned, these are still possible as AJAX is, by nature, asynchronous (Asynchronous Javascript and XML) and callbacks can be easily implemented with *ScriptableObject* classes.

Fig. 7. (c.f. [11]) gives the platform/browser independent initialization code of an AJAX object. For our purposes we need only one such object per client, which runs a single threaded and synchronous process, however nothing prevents the application to create as many as needed such objects. The *xmlHttp* variable is global and valid until the application terminates. The dispatch routine references this variable in the process of forwarding SOAP messages to the application server (c.f. Fig. 8.).

```
function initCall() {
    try { xmlHttp = new XMLHttpRequest(); } catch (e) {
        try {
            xmlHttp = new ActiveXObject("Msxml2.XMLHTTP");
        } catch (e) {
            try { xmlHttp = new ActiveXObject("Microsoft.XMLHTTP"); }
            catch (e) {return false;}
        }
    }
    return xmlHttp;
}
```

Fig. 7. The platform independent initialization of the AJAX communication channel [11]

The decisive advantage of AJAX object is that it is available and fully functional even when WCF is closed and Exit event fired. Moreover, it offers also a significant performance boost, contrary to the expectations that javascript could perform worse than C#/Silverlight. Another advantage of AJAX/HttpXML is the flexibility, as the calls can be configured as either asynchronous or synchronous (c.f. [13]).

Fig. 8. outlines the control flow in AJAX based service call. From the application perspective nothing changes with respect to the conventional WCF communication, except that there is no completion routine. The processing is now synchronous. After the call is issued the caller blocks until the response message arrives. Afterwards the response is deserialized and passed back to the caller and the application resumes its execution.

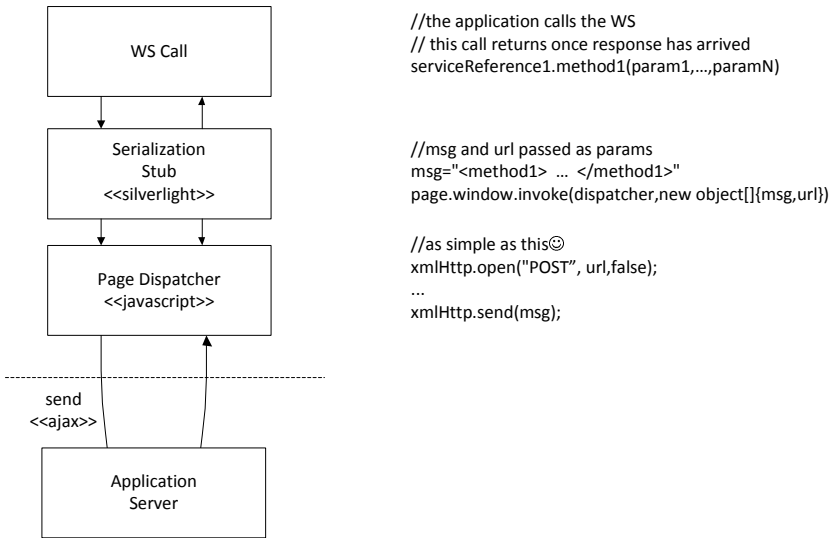


Fig. 8. The communication stack for AJAX based synchronous web service calls. To optimize the performance, the javascript layer of the stack is made extremely compact.

The entity named in Fig. 8. *Serialization Stub* which is a direct equivalent of WCF service reference can be generated automatically in a Visual Studio manner. This is possible with our tool which takes either a WSDL file or an Web Service endpoint as an input and creates the appropriate class definition. This AJAX oriented stub is much smaller and less complex than the standard one, opening up further opportunities for software developers, as generated routines can be subsequently modified or enhanced with application specific code.

4 Performance Evaluation

It is interesting to see how the AJAX based solution performance compares to the standard WCF asynchronous facility. Quite surprisingly, we have observed a significant performance boost when messages are dispatched via `html/javascript`. Our measurements were performed in 2 variants: local and network call. The predictable differences between them were confirmed by the experiments. (See Tab. 1.)

Table 1. The performance analysis. The network round trip simply adds to the total processing time. The server time is negligible, when the call returns immediately.

	AJAX (synchronous)	WCF	performance ratio
local call	4 ms	120 ms	30.00
network call (round trip 120 ms)	124 ms	240 ms	1.94
network call (round trip 300 ms)	304 ms	420 ms	1.38

To measure the performance of the WCF calls, we have implemented a completion routine that reissues the asynchronous call each time it completes. Each time the response arrives, the client increases the counter and finally, when designated number of call has been performed, it reports the total execution time. The code is given in Fig.9.

```

client.DoWorkCompleted += (s, ev) =>
{
    if (++i < designatedIterations) client.DoWorkAsync();
    else
    {
        double d = (DateTime.Now - dt).TotalMilliseconds;
        status.Text = d.ToString();
    }
};
dt = DateTime.Now;
designatedIterations = 1000;
client.DoWorkAsync();

```

Fig. 9. The performance measurement loop is simulated with a completion routine which repeatedly initiates an asynchronous call until the desired number of iterations is performed

The code in Fig. 9. is simple but one thing is noteworthy. The completion is now executed in the main thread of the application. This is an evident departure from the

former Silverlight asynchronous call architecture, where the completion code had been always scheduled in either a newly spawned thread or in a designated member of a thread pool [9]. Our guess is that this departure is a result of a confusion among Silverlight developers, annoyed by the requirement to access UI via the dispatcher. The time measurement of synchronous calls is trivial thus omitted.

5 Conclusions and Future Work

The presented solution was initially planned as a mere workaround, intended to remedy the Silverlight/WCF shortcomings with respect to finalization and synchronization. However, being both more efficient and flexible it became a viable candidate to replace the standard WCF in our Silverlight and WCF applications. To make this solution complete, in the future work we are going to enhance our utility tool (Sec. 2) with asynchronous interface and make it source level code compatible with WCF, facilitating greatly migration from WCF to AJAX.

The presented design patterns and communication stack has been already used in multiple market research applications, both internet and intranet. The market research applications relying on sophisticated computational infrastructure and databases are liable to pose considerable challenges for communication subsystem. The challenges present in our implementations included a proper integration of a computationally intensive multi-stage convex optimization (underlying an adaptive conjoint survey c.f. [4],[5]) and the reliable processing of hierarchical database updates. Both components (optimization and database) were published via an uniform, transactional *BeginTransaction, Commit, Rollback, Update* interface, what motivated our research in the area of reliable, transaction oriented communication channels. In our future work we are going to give a detailed report on the transactional aspects of numeric optimization.

References

1. Beres, J., Bill Evjen, B., Devin Rader, D.: Professional Silverlight 4. Wrox, Birmingham (2010)
2. Rossi, P.E., Allenby, G.M., McCulloch, R.: Bayesian Statistics and Marketing. Wiley Series in Probability and Statistics. Wiley, Hoboken (2006)
3. Sonnevend, G.: An Analytic Center for Polyhedrons and new Classes of global algorithms for linear (smooth, convex) Programming. In: Proc. of 12th IFIP Conference on System Modeling and Optimization, Budapest (1985)
4. Toubia, O., Simester, D.I., Hauser, J.R., Dahan, E.: Fast Polyhedral Adaptive Conjoint Estimation. *Marketing Science* 22(3), 273–303 (2003)
5. Nesterov, Y., Nemirovskii, A.: Interior point polynomial algorithms in convex programming. SIAM, Philadelphia (1997)
6. NVIDIA CUDA Compute Unified DeviceArchitecture, NVIDIA Corp.
7. Vaughan, D.: Synchronous Web Service Calls with Silverlight: Dispelling the async-only myth (2008),
<http://www.codeproject.com/KB/silverlight/SynchronousSilverlight.aspx>

8. Hill, D., Webster, B., Jezierski, E.A., Vasireddy, S., Al-Sabt, M., Wastell, B., Rasmusson, J., Gale, P., Slater, P.: Smart Client Architecture and Design Guide, Patterns and Practices. Microsoft, Redmond (2004)
9. John Papa, J.: Data-Driven Services with Silverlight 2: O'Reilly Media. O'Reilly Media, Sebastopol (2008)
10. Cleeren, G., Dockx, K.: Microsoft Silverlight 4 Data and Services Cookbook. PACKT Publishing, Mumbai (2010)
11. Darie, C., Brinzarea, B., Cherecheș-Toșa, F., Bucica, M.: AJAX and PHP Building Responsive Web Applications. PACKT Publishing, Mumbai (2006)
12. Zakas, N.C., McPeak, J., Fawcett, J.: Professional Ajax, 2nd edn. Wrox, Birmingham (2007)
13. Wenz, C.: Programming ASP.NET AJAX: Build rich, Web 2.0-style UI with ASP.NET AJAX. O'Reilly Media, Sebastopol (2007)

An Approach to the Semantization of ERP Systems

Robert Andrei Buchmann¹, Radu Meza², and Delia Pulcher³

Babes Bolyai University

¹ Faculty of Economic Sciences and Business Administration,
Business Information Systems Dpt., str. Teodor Mihali 58-60,
400591 Cluj Napoca, Romania
robert.buchmann@econ.ubbcluj.ro

² Faculty of Political, Administrative and Communication Sciences,
Journalism Dpt., str. G-ral Traian Moşoiu 71,
400132 Cluj Napoca, Romania
mezaradu@yahoo.com

³ Faculty of Economic Sciences and Business Administration,
Gazepower Project., str. Teodor Mihali 58-60,
400591 Cluj Napoca, Romania
delialoman@yahoo.com

Abstract. The paper promotes a methodology and application model for extending traditional, data-centric Enterprise Resource Planning Systems with semantics-oriented data models, in order to enrich interaction and reporting capabilities. The proposal includes a method of automating data semantization using a mix of semantic modeling and formal concept analysis, and an extended ERP architecture which integrates legacy systems in a Semantic Web wrapper system, providing new dimensions to traditional interaction and reports in a business environment, with specific examples regarding human resources management.

Keywords: ERP systems, semantization, formal concept analysis, RDF.

1 Introduction

ERP systems are one of the most important segments on the business information systems market. They are well established, mature, robust, and distributed, serving well the management needs that are data-centric. On the other hand, there's a lack of solutions open towards the relatively recent trend of Semantic Web. A possible reason could be that the Semantic Web paradigm itself moves quite slowly towards the business field, although management theory has long insisted on the abstract notions of competence and knowledge management [1].

As economic agents tend to guide themselves upon the requirement of efficiency rather than the principle of effectiveness, Semantic Web is still considered of marginal relevance by many pragmatic managers who promote knowledge but avoid talking about knowledge representation.

By manifesting itself on the common ground of artificial intelligence and Web technologies, Semantic Web might be able to speed up AI adoption in pragmatic situations which already rely on distributed systems. However, there are still thresholds to overcome.

One of the most relevant is the maturity of data-centric ERP systems and the "cost" that comes with this maturity. Companies who rely on such systems are not willing to replace them with knowledge-centric systems at least as long as the legacy systems serve the current operational needs. Current trends in ERP development are mobility and distributiveness rather than inference support. The relational model is mature enough to provide high performance queries for the most frequent scenarios and reports.

However, we consider it to be a safe bet that, as the Semantic Web (and the "Computational Web") take over the Web under the umbrella notion of *Internet Science* [2], ERP systems will, at some point, incorporate knowledge management capabilities, both of a semantic nature, and of a computational nature. Business intelligence modules are already available, although they emphasize data mining and model discovery rather than knowledge representation.

The paper proposes a methodology and an application model that can be applied, to various extents, on traditional data-centric ERP systems, in order to augment them with semantic support, by converting relational data structures to graph structures using a mix of semantic modeling and computational methods. The central idea is to provide the ability to generate extended, browsable and searchable reports that consume semantic graphs (linked data) through an external mechanism, beyond the SQL-based reporting mechanisms which are fundamental to most ERP applications.

The next section states the problem and some of its background (including related works). Section 3 lists the low cost instruments needed for implementation. Section 4 provides an architectural description with some implementation details, followed by a SWOT evaluation (strengths/weaknesses/opportunities/threats) and final conclusions.

2 Problem Statement and Background

For the purposes of this presentation, we use the less formal, working term of *data semantization* to designate any process that turns relational data collections into data structures that are fit for semantic modeling and processing. In our case, the end result is mapped on the RDF data model, based on an informational unit called *a triple*, consisting in a subject, a predicate/property and an object. Each of these is, usually, a resource with a global identifier (an URI). Multiple triples can be connected into graphs which, in turn, can be merged in a knowledge repository [3]. This data model serves as a foundation for the Semantic Web paradigm – the triple structure has the quality of being abstract enough to adapt to most paradigms and to fit legacy models and syntaxes (Prolog predicates, XML constructs, object-oriented constructs and even linguistic constructs) [4][5].

By extending the term, we assign the notion of *ERP semantization* to the process that extends the functionality of a legacy ERP system with semantized data and features that are able to consume it. This means that semantization takes place on at

least two levels, which are reflected by our paper: the model level and the presentation level. A domain-specific semantic business logic layer can be added to turn a semantized ERP to a full knowledge-centric system. This last goal falls out of this paper's scope and is subject to our research prospects.

The problem at hand is stated in the contexts of the slow adoption of ERP semantic support and the convergence between the computational and the semantically modeled knowledge management approaches. Until ERP developers will assimilate the requirements of turning databases into fact bases flexible enough for semantic processing, the methodology and architecture presented here are a low cost solution that can be applied on legacy systems, with minimal intrusion and without affecting the regular, established operations and usage scenarios of the original system.

Essential works related to our goal are the recent working drafts from W3C on a specialized RDB-RDF mapping language (a Turtle-based vocabulary called R2RML) [6] and its mapping methodology [7]. Until the language becomes widely supported, projects such as D2R (Database-to-RDF mapping language and server) provide interfaces for rewriting Web requests as SQL queries [8]. Our approach is more simplistic regarding the database mapping (it does not propose a language), but has better granularity due to the detection of formal concepts within tables and is less invasive with respect to the ERP architecture.

In the recent literature, semantization is becoming an attractive topic, but most research is biased towards the Web and related technologies (agents and services, mainly) or towards business process semantics. For example, [9] proposes a methodology for Web semantization by implementing an annotating crawler for a search engine, while [10] proposes an agent-based framework for achieving a similar goal. Closer to our goal is [11], where the authors approach ERP products with respect to their relation with web services. In [12], a new process-oriented technology of semantization is proposed, also based on web services.

Compared to these solutions, our paper is data-driven rather than process-driven, and takes a more business-oriented approach, with an emphasis on the data collections captured or exported from a legacy enterprise resource planning system. This might be considered a rudimentary approach to automated knowledge acquisition which has a much longer tradition than semantization. Most approaches to automation are computational and use various types of data sources and methods: [13] presents a way of extracting the logical structure of document using entropy analysis, while [14] and [15] employ neural networks and self-organizing maps for extracting knowledge from a database. Our research promotes an alternative computational approach – the formal concept analysis (FCA), an innovative methodology for taxonomy derivation, which is abstract enough to be also considered as a clustering method or an association rule detection method. During the last decade, FCA has been formalized and implemented with strong mathematization [16] and various algorithms [17][18]. One of our previous papers [19], from the early stages of our current research, takes a more theoretical approach to data semantization through FCA by insisting on the method's versatility with respect to other paradigms: statistics, linguistics, object-oriented and relational models.

3 Instrumentation

The tools required for our proposed ERP semantization methodology can be obtained freely:

- OpenRDF Sesame – an RDF management system developed by Aduna Software, with weak inferential capabilities (RDF Schema and the in-house Direct Type vocabulary) [20];
- OWLIM – an OWL knowledge and rule-management system developed by Ontotext AD [21] as a repository template for Sesame, thus bringing OWL and custom rules inference capabilities to Sesame;
- SIMILE – a toolkit of JavaScript APIs developed at MIT, for generating and feeding Web GUIs with RDF graphs expressed in a JSON serialization format [22]; the RDF-JSON conversion can be run through the Babel web service (also available at the SIMILE site) or, as it is our case, through a dedicated module of our application model with increased performance;
- Python is the language of choice for prototyping, mainly because it will allow us to experiment with Python-specific semantic support through libraries such as RDFLib (for RDF management) [23] and Fuxi (for rules and inferences) [24].

Regarding algorithms and methodologies, we employ an intuitive method for deriving RDF facts from relational data points, and the formal concept analysis (a variation on Ganter's algorithm [17]) for deriving a rudimentary ontological model (taxonomy). An object introspection module, (inspired by [4], chapter 9), is not fundamental to the proposed model, but rather a convenience that allows the model's developers to use an object-oriented syntax (instead of SPARQL queries) when accessing classes, instances and property values from the knowledge models.

4 The Proposed Model

We split the proposed solution in several layers which will be detailed in the next sections:

- On the model layer: the ERP original database and the knowledge repository;
- On the control and logic layer: the data semantization engine; the object introspection engine; the presentation content preparation;
- On the presentation layer: the extended ERP GUI (web reports and forms extending the original ones); the platform interface for privileged users (which are actually the knowledge engineers who are able to interact directly with the Sesame knowledge base and the OWLIM rule sets; they are responsible with performing the non-automated, higher level, knowledge acquisition).

In Fig. 1, the bolded shapes represent the original ERP components (based on a simplified version of the Navision architecture), while the other shapes represent the semantization extension, together with the platform-specific elements (Sesame, OWLIM and the platform interface).

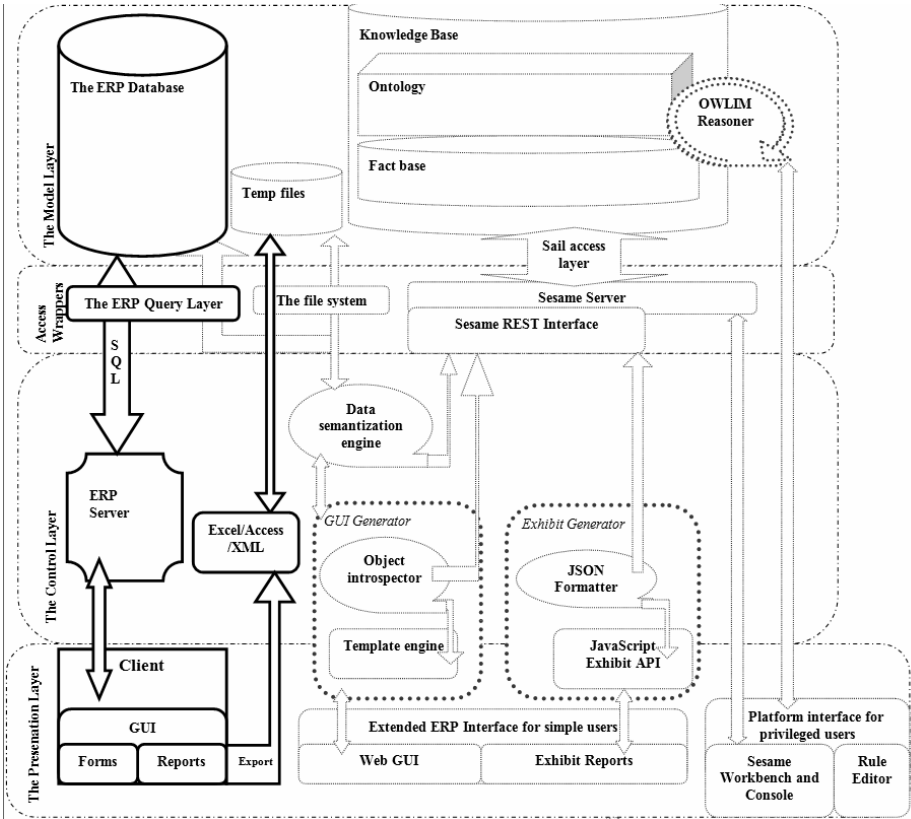


Fig. 1. The general architecture of the proposed model

4.1 The Automation of the Fact Base Acquisition

Knowledge acquisition in our proposal of a semantized ERP is performed on two levels: the facts and weak semantics (taxonomy) are automatically derived, while the strong semantics (rules) must be defined by the knowledge engineer through platform-specific tools (the OWLIM rule sets).

What our model proposes is a module responsible with converting data collections already existing in an ERP’s relational data store to a repository model which is more fit to semantic development. This involves two strategies for converting relational data to an RDF graph repository, resulting in a "knowledgeable" view upon the database: a relational-to-RDF mapping which can be applied to any data structure that fits the relational model, such as the exported result of a query; and the extraction of a class taxonomy from a table, by employing the formal concept analysis methodology.

a. The relational-to-graph mapping is accomplished with an intuitive conversion of the relational model underlying the legacy data store, through a wrapper layer for the queries supported by the relational data source. Depending on how the ERP exposes its data (queriable database or exported reports), it might be SQL-based, or CSV-based, or XML-based.

Let's consider the following table, with the name Employees:

Table 1. A sample table fragment from a ERP's human resources module

ID	Name	Birthdate	Gender	Social No.	Position	Dpt
A001	John Doe	17.12.1980	Male	1801217125792	Economist	D01
A002	Jane Doe	20.12.1980	Female	2801220125792	Network Admin	D02

The transformation is performed through the following steps:

1. The model generates a global resource ID for every record, using the primary key as a local identifier, concatenated with the company domain address in order to produce a global ID: <http://mycompany.com/resources#A001>
2. For every datapoint in the table, a RDF triple is generated, using the global record ID from the previous step as a subject, the field name as a property and the actual datapoint as a property value (object, with explicit type when available). Assuming that we have already defined a prefix in order to avoid repeating the company domain address, we would obtain, for every record, a triple set such as the following:

```
:A001      :hasID      "A001" .
:A001      :hasName    "John Doe" .
:A001      :hasBirthdate "12-17-1980"^^xsd:date .
```

```
.....
:A001      :hasDpt     :D01 .
```

The "has" word is attached to property IDs in order to avoid confusions with classes or data types with the same ID. For example, *Dpt* would become the class of all departments, while *hasDpt* would become the relationship between an Employee member and a Department member.

Assuming that we have an Employee-Department relationship between tables, it would be naturally captured by triples such as the last one, explicited for every related record.

3. For every record, an instanceOf relationship can be generated, using the table name as a class name:

```
:A001      rdf:type    :Employee .
```

4. The primary key is declared as a functional and inverse functional property, an OWL restriction that imposes a 1:1 mapping between resources and key values.

```
:hasID     rdf:type    owl:FunctionalProperty .
:hasID     rdf:type    owl:InverseFunctionalProperty .
```

5. On a vocabulary level, the current table becomes a property domain, and every field becomes a property range:

```
:hasDpt    rdfs:domain :Employee .
:hasDpt    rdfs:range  :Dpt .
```

(this reflects the relationship with other entities; a mapping might be necessary between table names and foreign key names)

```
:hasName   rdfs:domain :Employee .
:hasName   rdfs:range  :Name .
```

(this reflects the relationship with a custom datatype)

Special care must be taken if there's a possibility that the same field name is used in multiple tables, in order to avoid domain clashes. This might be avoided by adding an extra prefix to the property names (referring the table, a solution proposed by [7], which also prefixes primary key values) or by making the extra step of defining a superdomain for all properties with more than one domain.

Class, resource and property declarations will be inferred by the knowledge repository so they don't have to be explicitly created by the semantization engine. This process results in an RDF triple collection (graph) which is stored on the knowledge management platform (Sesame) through its REST interface.

In a Python-based environment, this is accomplished with the `urllib` library, capable of issuing HTTP requests with or without parameter data. On the server side, Sesame responds according to its REST protocol – the last part of the requested URL is treated as a request for a resource (a SPARQL query, usually). The following Python example issues a SPARQL selection query (through the HTTP GET method) for all the employee names and the departments they work for:

```
>>> server="http://localhost:8080/openrdf-sesame"
>>> request="/repositories/myrepository?query="
>>> namespaces="prefix
                :<http://mycompany.com/resources#>"
>>> myquery=" select ?name ?dpt where
                {?x :hasDpt ?dpt. ?x :hasName ?name}"
>>> formattedquery=urllib.quote_plus(myquery)
>>> temp=urllib.urlopen(server+request+namespaces
                +formattedquery)
>>> JSONresponse=temp.read()
>>> dataresponse=json.loads(JSONresponse)
>>> values=[(i['x']['value'],i['y']['value'])
                for i in
dictionary['results']['bindings']]
```

As the last three lines reflect, Sesame responds with a JSON serialization of the SPARQL result format, which must be parsed into Python dictionaries in order to extract the data.

b. The second strategy is to further decompose each table in a taxonomy of formal concepts, according to the FCA methodology. FCA algorithms are executed over a so-called *formal context* consisting in a bit matrix in order to compute a lattice of (partially ordered) *formal concepts*, which can be freely interpreted as statistical clusters, object-oriented or semantic classes, relational entities or linguistic concepts. The paper won't insist on the mathematization of FCA (available in the provided references). Instead we define *the formal concept* rather informally, as being *the set of objects sharing a certain set of attributes*.

An FCA algorithm would compute from Table 2 a lattice expressed by the Hasse diagram in Fig. 2. Every node is a class/cluster/concept defined by a set of shared attributes (*the intent*) and containing a set of objects sharing those attributes (*the extent*). Every arc represents a specialization/generalization, corresponding to the `rdfs:subClassOf` property. As attributes are added, the formal concepts become more precise, as attributes are removed, the formal concepts become more general (fewer features, more objects). The extremes can be mapped on standard classes such as `owl:Thing` (the class of all objects, regardless of shared attributes) and `owl:Nothing` (the class of objects that share everything – theoretically, it’s possible to have this class satisfied, but in our case there are attributes excluding each other).

Table 2. A sample of object-attribute bit matrix

	A1	A2	A3	A4
O1	X	X		
O2		X	X	X
O3	X		X	X
O4	X	X	X	

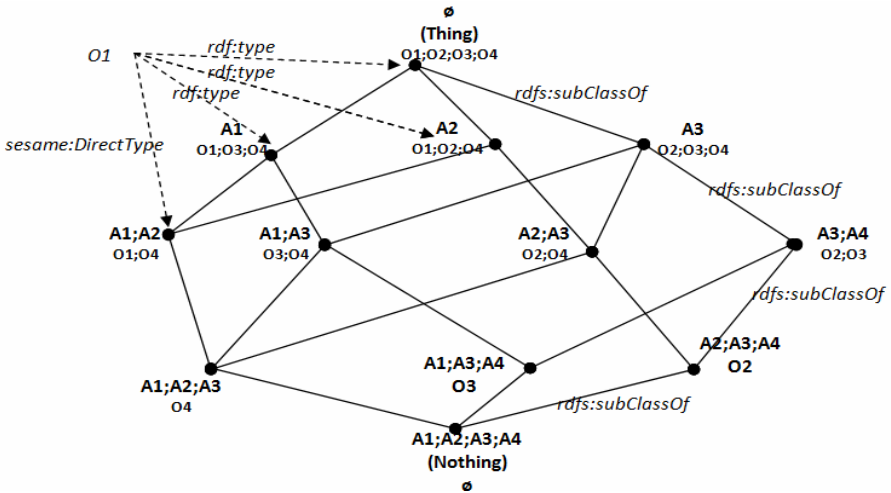


Fig. 2. A Hasse diagram extended with RDF properties

We extended the Hasse diagram in order to reflect the `instanceOf` (`rdf:type`) relationships between the members of an extent (an object) and the classes to which they belong (one or more formal concepts). For the smallest concepts we use the convenient `directType` relationship provided by the Sesame vocabulary in a closed world assumption (a resource has a certain *directType* if there are no intermediary classes between the resource and that type).

One of the challenges regarding the FCA taxonomy derivation is to extract relevant boolean *formal contexts* from the much more heterogeneous ERP database. This reduces the degree of automation since it must be accomplished through a query layer controlled with an HTML interface where the knowledge engineer must define the

relevant attributes. A fully automated derivation of a database is neither realistically usable, nor scalable.

The relevant attributes are created by assigning an attribute to each value in a field (ex: male, female), to subsets of field values (ex: related job positions) or to value ranges (ex: age ranges, salary ranges). Logical fields map trivially to FCA attributes. In an HTML interface, the user selects a relational entity and, for each of its fields, creates a set of mappings between FCA attributes and values/ranges/subsets whose presence will determine a value of 1 in the formal context. For example, the class of Employees might be split in the concepts determined by FCA: males, females, old, young, employee groups determined by position, department or salary package structure.

4.2 The Presentation Layer

The proposed model extends the ERP GUI with Web-based forms and reports generated from the RDF backend resulting from the previous conversion effort. In order to accomplish this, we employ AJAX libraries such as those provided by SIMILE. One of the essential components of SIMILE is EXHIBIT, an XML vocabulary powered by a JavaScript API and used for defining *RDF lenses* (a notion similar to *database views*). They can be mixed with HTML to generate web pages based on these lenses. The final HTML document is augmented with useful AJAX widgets such as a search engine, filtering facets, pop-ups describing resources related to the ones displayed (colleagues from the same department, in Fig. 3).

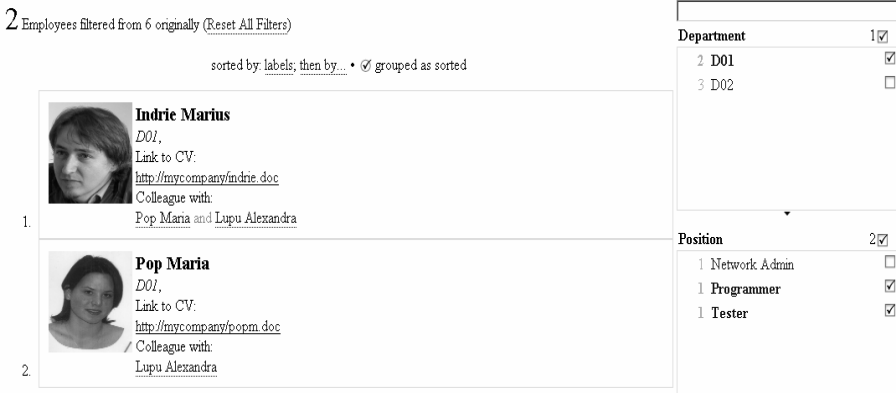


Fig. 3. A screenshot from the EXHIBIT-powered interactive report and (below) its RDF data source expressed as JavaScript Notation (for the Employee class)

The JSON representation of RDF can be obtained either by running RDF triples through the Babel web service or, in our case, through a customized conversion module. This is especially useful as the Sesame REST interface responds with a JSON representation of the standards SPARQL Results Format [25] and not with a traditional RDF serialization format, so the converter acts as a JSON reformatter that adapts the dataset to the requirements of EXHIBIT lenses. The lenses can be defined for each class of the taxonomy.

```
{types: {Employee : {pluralLabel: "Employees"}},
  properties: {DptColleagues : {valueType: "item"}},
  items : [
    {
      type : "Employee",
      label : "Lupu Alexandra",
      position : "Network Admin",
      department : "D01",
      DptColleagues : "Pop Maria",
      URLCV : "http://mycompany.com/lupu.doc",
      URLphoto : "poza2.jpg"},.....]]}
```

5 SWOT Evaluation and Conclusions

Strengths

The paper presents a low-cost methodology for turning a legacy ERP system into a fact base with a web interface that manages data models of a semantic nature. This can be easily extended into a full-fledged knowledge management system built upon the ERP data sources, by adding a custom rule base in the OWLIM inference engine.

The proposed solution adds value to legacy ERP systems, by opening them towards Semantic Web-based decision support capabilities for higher management levels.

Weaknesses

The proposed model does not provide integrity and synchronicity with the legacy data source. It acts as a semantic backup for the relevant information from the ERP system and new records added to the database do not reflect automatically in the knowledge repository; the knowledge engineer may add these records manually or perform the full semantization periodically, as with any backup operation.

OWLIM comes in two flavours: the free version (SwiftOWLIM) and the commercial version (BigOWLIM). In order to respect our policy of low cost semantization, the model employs the free version, which has an essential disadvantage – lack of support for inconsistency detection. However, this is compensated by the possibility of customizing the rule set using a text-based intuitive syntax for Horst rules. One can define rules for consistency checking (by producing special triples, with some key resources) like in the following example:

```
RuleID: r01
  :X owl:sameAs :Y
  :X owl:differentFrom :Y
  :X :hasInconsistentRelation :Y
  :X :r01 :Y
```

The proposed solution is weakly coupled with legacy ERP solutions. Requirements of robustness and reliability might bring forward the requirement for stronger coupling, where semantic queries are executed in a uniform manner, together with relational queries (via direct mapping languages such as those proposed by [6] and [8]). The Microsoft solutions that we experiment on are flexible enough from a usage standpoint, but not flexible enough from a reengineering viewpoint, thus the semantic extension is rather augmentative than integrated, with multiple points of conversion and interoperability, aspects that confer a „patchwork” heterogeneity to the model.

Opportunities

The Semantic Web emergence over data-centric systems, although initiated for a long time now, it’s still slow and lacks popularity expressed as software support in tools of high immediacy. However, there’s an inevitable migration towards Web 3.0 and the big actors on the Web software market spend a great deal of research in proposing standards for emergent semantic technology markets, a context to which our solution is correctly aligned on long term. In this context, it is fairly plausible that ERP systems will start migrating towards the new paradigm as well.

Threats

There are still skeptics stating that Semantic Web will have the same fate as artificial intelligence had during the 70s, that its development is much too slow with respect to the added value for business, and lacking spectacular results. There’s a lack of managers’ interest in implementing ERP semantization. However, there’s a much stronger support for Semantic Web due to its convergence with the discipline of Business Intelligence and due to the network effect manifesting in the Web environment. An important factor is the ability of RDF to unify data-centric systems with knowledge-centric systems in a consistent manner.

For practitioners, a semantization methodology for legacy systems would add significant value until semantics will be widely adopted as an ERP feature, rather than being seen as an experimental field that stands in the way of everyday operations.

Future efforts will be invested in an alternative implementation based on Python libraries instead of the standalone servers (with performance comparisons). For now, we consider the separation between the application server and the knowledge service to be preferable, with respect to Semantic Web openness principles (as opposed to the more protective database principles).

Acknowledgment. This paper represents results of the research project 2443/2009 funded by the UEFISCSU-CNCSIS Program IDEI, managed by Lect. Robert Buchmann Ph.D.

References

1. Ikujiro, N.: The Knowledge Creating Company. Harvard Business Review 69, 96–104 (1991)
2. Internet Science FP7 Call, http://cordis.europa.eu/fp7/ict/fire/internet-science_en.html
3. The Resource Description Framework, <http://www.w3.org/2001/sw/wiki/RDF>

4. Segaran, T., Evans, C., Taylor, J.: *Programming the Semantic Web*. O'Reilly, Sebastopol (2009)
5. Hebel, J., Fisher, M., Blace, R., Perez-Lopez, A.: *Semantic Web Programming*. Wiley, Chichester (2009)
6. The R2RML W3C Working Draft, <http://www.w3.org/TR/r2rml/>
7. The RDB Direct Mapping W3C Working Draft, <http://www.w3.org/TR/rdb-direct-mapping/>
8. The D2R Server Official Site, <http://www4.wiwiss.fu-berlin.de/bizer/d2r-server/>
9. Dedek, J., Eckhardt, A., Vojtas, P.: Web Semantization – Design and Principles. In: Snášel, V., Szczepaniak, P., Abraham, A., Kacprzyk, J. (eds.) *Advances in Intelligent Web Mastering - 2. Advances in Soft Computing*, vol. 67, pp. 3–18. Springer, Heidelberg (2010)
10. Beno, M., Misek, J., Zavoral, F.: AgentMat: Framework for Data Scraping and Semantization. In: *The Third Int. Conf. on Research Challenges in Information Science 2009 Fez*, pp. 225–236. IEEE Computer Society, Los Alamitos (2009)
11. Anjomshoaa, A., Karim, S., Shayeganfar, F., Tjoia, A.M.: Exploitation of Semantic Web Technology in ERP Systems, Research and Practical Issues of Enterprise Information Systems. In: *IFIP International Federation for Information Processing*, vol. 205, pp. 417–427. Springer, Heidelberg (2006)
12. Hepp, M., Leymann, F., Domingue, J., Wahler, A., Fensel, D.: Semantic Business Process Management: A Vision Towards Using Semantic Web Services for Business Process Management. In: Lau, F.C.M., Lei, H., Meng, X., Wang, M. (eds.) *ICEBE 2005*, pp. 535–540. IEEE Computer Society, Los Alamitos (2005)
13. Tang, Y.Y., Yan, C.D., Suen, C.Y.: Document Processing for Automatic Knowledge Acquisition. *IEEE Transactions on Knowledge and Data Engineering* 6(1), 3–21 (1994)
14. Shin, D.K., Lee, S.H., Lim, J.S.: Automated Knowledge Acquisition from Discrete Data Based on NEWFM. In: *Proceedings of the 2010 Third International Conference on Business Intelligence and Financial Engineering*, pp. 53–56. IEEE Computer Society, Los Alamitos (2010)
15. Elfadil, N., Isa, D.: Automated Knowledge Acquisition Based on Unsupervised Neural Network and Expert System Paradigms. In: Anderson, S., Felici, M., Littlewood, B. (eds.) *Computer Safety, Reliability, and Security. LNCS*, vol. 2773, pp. 134–140. Springer, Heidelberg (2003)
16. Ganter, B., Stumme, G., Wille, R. (eds.): *Formal Concept Analysis: Foundations and Applications. LNCS (LNAI)*, vol. 3626. Springer, Heidelberg (2005)
17. Ganter, B.: Two basic algorithms in concept analysis. In: Kwuida, L., Sertkaya, B. (eds.) *ICFCA 2010. LNCS*, vol. 5986, pp. 312–340. Springer, Heidelberg (2010)
18. Kuznetsov, S.O., Obiedkov, S.: Algorithms for the construction of concept lattices and their diagram graphs. In: Siebes, A., De Raedt, L. (eds.) *PKDD 2001. LNCS (LNAI)*, vol. 2168, pp. 289–300. Springer, Heidelberg (2001)
19. Buchmann, R.A., Meza, R., Hejja, A.: The Automated Derivation of Semantics from ERP Databases. *Journal of Applied Computer Science & Mathematics* 9, 27–31 (2010)
20. Open RDF official website, <http://www.openrdf.org/>
21. OWLIM official website, <http://www.ontotext.com/owlim/>
22. SIMILE project official website, <http://simile.mit.edu/>
23. RDFLib official website, <http://www.rdflib.net/>
24. FuXi official download page, <http://pypi.python.org/pypi/FuXi/>
25. The Serialization of SPARQL Query Results in JSON, <http://www.w3.org/TR/rdf-sparql-json-res/>

Open IT for Business: Transforming Information System Infrastructure with a Commercial BPM Suite

Sava Mintchev



Baring Asset Management, 155 Bishopsgate, London EC2M 3XY, UK
sava.mintchev@barings.com

Abstract. Corporate information systems must be continuously enhanced and adapted to ever changing business needs and priorities. At the same time, IT and Operations costs come under increasing pressure, so improved services must be delivered with increasing efficiency. In this paper we consider how such challenges can be met in the context of the financial services industry, and from the viewpoint of IT. We present our experience of utilising webMethods, a commercial BPM suite from Software AG, in successfully modernising the business system infrastructure at Barings, a global asset management firm, over a 3-year period. Our pragmatic approach is for IT developers to create and maintain executable processes in collaboration with business experts. We discuss the approach, resources, completed projects, difficulties encountered, future plans; and suggest techniques for building flexible and adaptable enterprise information systems.

Keywords: business process management, integration, reporting, service oriented computing, industrial experience.

1 Introduction

Most organisations build up over their existence a set of business information systems to support their activities. More often than not, these systems would be a mix of commercial off-the-shelf software from different vendors, in-house applications, and internal or external services; would use disparate technologies; and would have different ages and levels of maturity. Despite the diversity, there is a need for consistency of data across the enterprise, and for delivering comprehensive information from all systems.

In this paper we discuss our experience of meeting this need with a business process management  approach. The work has been carried out over the past three years at Baring Asset Management (BAM). We begin with an overview

¹ Baring Asset Management provides investment management services in developed and emerging markets to clients worldwide. The company operates from 10 countries, and has around 100 investment professionals, covering equity, bond and alternative asset classes. It is a subsidiary of MassMutual, a leading diversified financial services organisation.

of BAM business system architecture and its integration challenges (Section 2). The approach to change is outlined in Section 3, and in Section 4 we discuss the stages in transforming our information system infrastructure. In Section 5 we take a step back, and reflect on our experience of achieving flexibility in enterprise information systems.

2 Starting Point

2.1 Business Systems

The business activities of Barings are backed by a distributed enterprise IT infrastructure, with main business systems hosted in London, supporting round-the-clock market operations in different geographic locations. The key systems (shown in Figure 1) include:

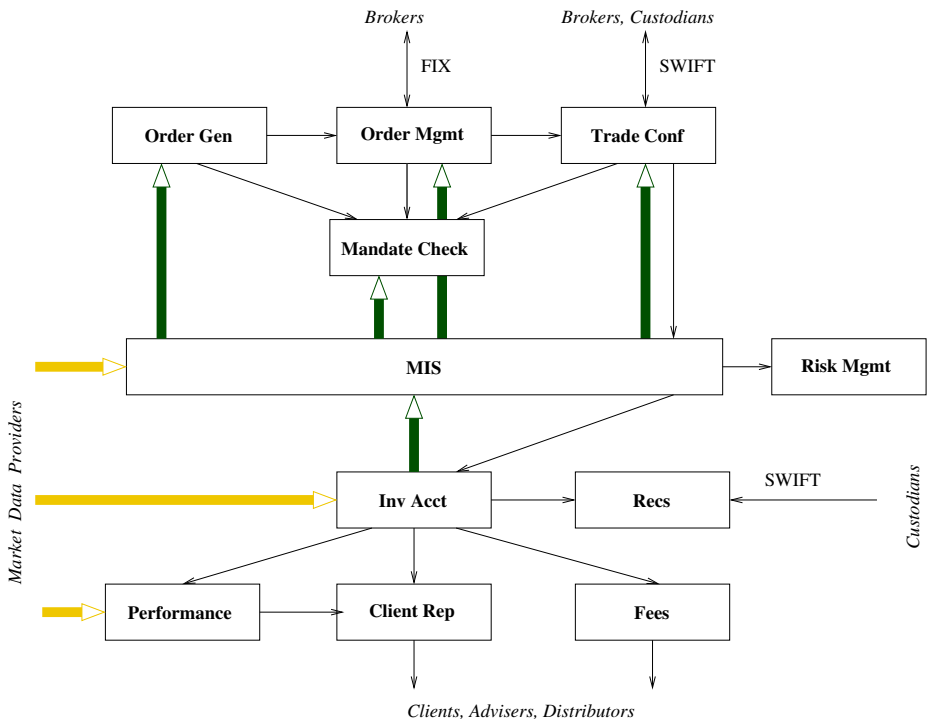


Fig. 1. Main business systems and data flows

Decision Support and Order Generation (Order Gen) - used by Investment Managers to model and analyse the impact of investment decisions on portfolios, and to generate trading orders;

Order Management - used by Dealers to execute trading orders in the market;

Trade Confirmation and Settlement - matches executed trade information from brokers with orders, and issues settlement instructions;

- Mandate Check** - ensures compliance of investments with client mandate restrictions (e.g. ethical);
- Management Information System (MIS)** - contains a consolidated view of funds, portfolios, and benchmarks from different investment accounting systems, and various market data providers;
- Risk Management** - for risk measure calculation, what-if analysis, and stress testing of portfolios;
- Investment Accounting (Inv Acct)** - book keeping of transactions and positions in all investment funds and client portfolios. We have two such systems used for different types of clients;
- Reconciliation (Recs)** - of portfolio cash and holdings with Custodian records;
- Performance Measurement** - for calculating portfolio performance against benchmarks, and performance attribution analysis;
- Fee Calculation** - for generating invoices and rebate statements, and financial analysis of investment portfolios
- Client Reporting** - production of client valuation packs, investment reviews, fund fact sheets and other reports.

MIS and the systems shown above it in Figure 1 belong to the front and middle office, while those below are considered part of the back office. In reality the distinction is blurred – some back-office systems also support front-office functions; for example, the Investment Accounting and Performance Measurement systems are used for client reporting.

In addition to these and other industry-specific systems, there are ERP applications and services to support the operations of a medium-sized company, including CRM, financial accounting, HR, project management, supply chain management, helpdesk *etc.*

Barings has been pursuing a buy-not-build, best-of-breed strategy for business application procurement. Consequently most systems come from different vendors, and are not directly compatible. Increasingly, software from external Application Service Providers (ASP) or software as a service (SaaS) is used, in preference to systems hosted internally.

The computing infrastructure is heterogeneous, but not diverse: we have Sybase and Microsoft SQL database servers running under Unix (HP-UX) and Windows. Older platforms, along with COBOL programs, have been decommissioned or replaced.

2.2 Integration Infrastructure

A large part of the software development and maintenance activity at BAM is dedicated to the integration of different business systems, applications, and services. Until a few years ago, the emphasis had been mostly on data integration, with less focus on process integration. B2B integration with external partners, counterparties, intermediaries, clients, data and application service providers has also been increasing in importance, volume and diversity.

In simple technical terms, four main integration styles can be identified [5]: file transfer, shared database, remote procedure calls, and messaging. All these

styles have been applied in BAM's integration infrastructure. Until 3 years ago, flat file-based ETL (extract / transform / load) feeds were most commonly used, controlled by the IBM Tivoli Workload Scheduler (TWS), which enforces timing and dependency constraints across the enterprise. Client-server applications developed in-house during the 1990-s utilised MIS as a shared database; many have since been replaced.

Synchronous remote calls (at the level of database servers, OS commands, component model, or web services) across system boundaries have been used infrequently for process integration, and for consolidated reporting from multiple sources. We have tended to avoid direct calls between systems because of the resulting tighter coupling.

Messaging has been applied for trade flows between systems (both internally and externally), as well as for some reference data. Several messaging solutions have been in use: Sungard MINT (for both internal and external communication via SWIFT); FIX engines (Sybase Financial Fusion; Cameron FIX); JBoss; TIBCO SmartSockets.

2.3 Analysis

In summary, the business system landscape at Barings has presented a number of integration challenges:

- Multiple disjointed systems with their own databases;
- Different data dictionaries and models;
- Data model gaps and overlap;
- Functionality gaps and overlap;
- Mixture of packages hosted internally, and ASP / SaaS services;
- Heterogeneous environments, multiple technologies, different interfaces.

On the positive side, our integration infrastructure has been based on a sound architecture, centred around the MIS database as a kind of “data hub”. It consolidates data from multiple sources (two Investment Accounting systems, numerous market data suppliers), and propagates it to front office systems. Thanks to MIS we have had no point-to-point connections between individual back-office and front-office systems.

3 Approach to Change

An internal BAM project analysed the state of the Enterprise Application Integration (EAI) market in 2000, but the market in proprietary EAI suites was then judged to be immature and unsettled. More recently, since 2004, our focus has been on standards-compliant modular solutions, and on building the infrastructure bottom-up (starting with the messaging and service layers) rather than top-down. In 2006 webMethods² was selected for use in application messaging and integration at BAM, in line with the strategy of our parent company - MassMutual.

² webMethods is now a product suite from Software AG.

3.1 The webMethods BPM Suite

The webMethods Business Process Management suite is built on the following main components:

Broker - robust implementation of persistent messaging, supporting different client interfaces;

Integration Server (IS) - a container for web services, also supporting various calling client interfaces. Together with Message Broker forms the basis of an Enterprise Service Bus (ESB) implementation

Process Runtime - a BPEL-compatible business process execution engine, hosted by Integration Servers

The suite also includes service development and process design tools, centralised administration and monitoring (of both the infrastructure and business activity), a semantic metadata repository and SOA governance module. There are numerous adapters for other commercial suites, and for industry-specific formats and protocols. The suite provides support for standard APIs (e.g. JMS), component models (EJB, COM, .NET), and Web Services, allowing application programmers to interface easily to systems without the burden of ensuring compatibility.

3.2 Design Considerations

The overall model we have chosen for data replication purposes is a *message bus*. Some systems provide data by publishing messages on the bus; other systems subscribe to the types of message which are relevant to them. In line with existing BAM practice, preference has been given to one-way synchronisation of data, with a clear source for each type of data.

Where necessary, routers have been implemented for specific message types; for example, trade execution messages from an Order Management system can be sent to different Trade Confirmation systems / services depending on the instrument type (bond, equity, foreign exchange, derivative).

Canonical message (document) formats – i.e. ones which are enterprise-wide rather than system specific – have the advantage of being applicable to all systems being integrated, thereby facilitating the creation of a publish-subscribe messaging bus. BAM canonical message (document) formats have been based on the MIS data model where applicable, because the data in MIS is a consolidation of data in other BAM systems. Other canonical documents can utilise standard formats; e.g. vertical standards (FixML, SWIFT) and general e-business standards (XBRL, ebXML).

System interfaces to the message bus are implemented as BPEL-compatible processes, performing the required data transformation. Distributed transaction processing has been avoided.

4 Projects

4.1 Integration Infrastructure Project

The first step towards modernising our business systems infrastructure was to put in place the webMethods BPM suite. This was done by a dedicated IT project, with the following scope:

- Training of developers and IT support analysts;
- Development of custom Utility and Exception Handling services for integrating webMethods into existing mechanisms for system monitoring;
- Adoption of guidelines and standards for development, deployment, and support processes;
- Delivery of core Development, System Test, User Acceptance Test (UAT), Production and Disaster Recovery (DR) webMethods BPM environments;
- Development and delivery to production of a business process automation “pilot” sub-project.

The project ran during the second half of 2007. It was completed with limited (10 days) consultancy from Software AG Professional Services, which is evidence of the maturity of the webMethods suite. The pilot sub-project was one for which there was a business requirement at the time. It went live in May 2008, and was implemented as a an executable process spanning 3 logical Integration Servers, 4 market data vendors, and 2 target systems (one of which externally hosted).

The main problem was to keep this project going despite demand for resources from other business-sponsored projects of higher priority. The project was completed under budget but later than originally planned.

4.2 Integration and Automation Projects

Subsequent projects have utilised the new infrastructure to integrate new and existing business systems, and to automate business processes. These run-of-the-mill projects have all been sponsored by business owners, not by IT, and managed following the established methodology based on PRINCE2. Starting in 2008, they included (see systems diagram in Figure 11):

- Replacement of Order Generation system
- New Performance Attribution system
- Replacement of Fee Calculation system
- New BI reporting (using SAP BusinessObjects) for financial management information
- New Investment Research application
- Replacement of Order Management system

Executable business processes have been designed in webMethods by developers. These processes are fully automated; so there was little or no involvement of business users in their creation. In relation to our “legacy” integration infrastructure controlled by the TWS scheduler, processes correspond to *schedules*, and process steps correspond to *jobs*. The people who control process executions are IT Support staff, and not business users - just as for TWS schedules.

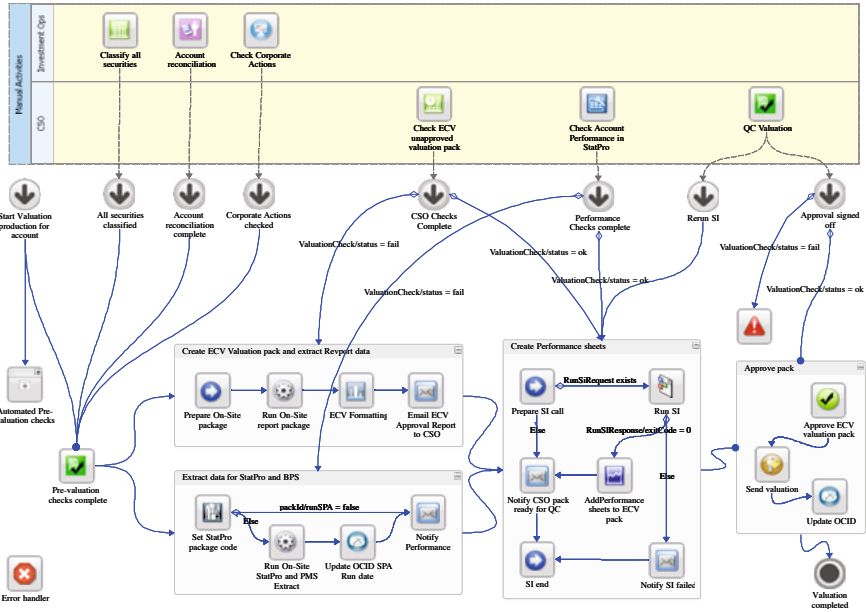


Fig. 2. Client Valuation business process

In order to implement the new processes, webMethods IS services have been created to interface to the new systems, and to existing systems (MIS and Investment Accounting). Projects also involved adding new external information sources (marked data providers, fund administrators) and consumers (administrators, wrapper providers).

The main problem we encountered in these and other webMethods-related projects has been in managing the relationship with the main users of the new technology – IT Support staff. Difficulties were partly due to the insufficient training and involvement of IT Support during the Integration Infrastructure project (Sect. 4.1); the new tools were also (seen as) more restrictive, and allowing rather less direct access “under the bonnet” than operating system commands. As a result, we made a couple of feature requests to the vendor (Software AG), one of which has been taken forward on the basis of user community voting.

4.3 BPM Workflow Projects

Having established the infrastructure, built a layer of services interfacing to business systems, and gained experience in designing automated business processes, we have since been able to move into “proper” BPM territory. One example is the project to improve client valuation production (part of Client Reporting in Figure 1).

The production of valuation statements is a process which involves a series of data quality checks, and running reports from two main sources. The associated workflow had been described in business operational procedures. The main

problems were identified by business users themselves as lack of systems support for workflow, and insufficient automation (manual co-ordination of activities; manual recording of the state of processes; manual checking of data and running of reports).

The project to improve valuation production started in 2010. Elements of agile methodologies have been applied: IT (developers and system testers in particular) working closely with the business; iterative development; adaptability to changing requirements; regular delivery to production. The first release went live in June 2010. The solution comprises an executable business process (Figure 2), services for interfacing to existing systems (Investment Accounting, Performance) and applications (Client Reporting). It utilises a centralised repository of client account valuation-related data, including frequency, deadlines, reports, benchmarks, distribution. The automated workflow facilitates co-ordination of activities between business teams, and allows the status of valuation production to be monitored at detailed and management summary level.

Some steps in the process remain manual; some of them can be automated, and will go live in subsequent releases.

The experience of this project has convinced us that an executable Business Process model definition is an excellent tool for capturing, refining, and discussing requirements with business people. Process design is an activity best shared between IT developers and the business. During the course of the project, the model from Figure 2 has evolved continuously; initially starting as a representation of the developer's understanding, it was later corrected and enriched in discussions with users to reflect existing procedures. Soon after go-live, when business managers had seen the effect of automated workflow and identified new bottlenecks, the process definition was again discussed and changed in a re-engineering exercise.

4.4 Resulting Environment

Table 1 gives an idea of the size of the webMethods-based environment created as a result of the projects from Sections 4.1 to 4.3. In explanation of the entries in Table 1: in webMethods, *documents* define data structures. Publishable documents are those which can be published as messages on Broker; they trigger the execution of processes which subscribe to a particular document type.

Table 1. Statistics about BAM processes in webMethods production environments

<i>Objects</i>	<i>Quantity</i>
Integration Server (IS) instances in Production	10
Deployment build projects	60
Executable Processes	35
Process instances run per day (average)	1,300
Documents - publishable	85
Documents - non-publishable	260
Services	2,360

The webMethods-related development has been carried out by the 8-strong Development and Integration team, with up to 3 developers engaged at any one time. The team has significant business knowledge and experience, with 5 members qualified to Investment Management Certificate (IMC) level.

The BPM suite has helped us to “open up” IT to business users, with new interfaces to systems, new composite applications, and graphical process models.

5 Towards Flexible Enterprise Information Systems

Technology (a BPM suite) does not on its own make enterprise information systems more flexible and maintainable. In this section we reflect on principles and techniques which, in our experience, can help achieve flexibility. What follows is a practitioner’s subjective view, rather than a systematic or comprehensive overview of this vast area of active research.

5.1 Executable Process Modelling

We have seen (in Section 4.3) the value of *executable* process definitions which can be understood, when represented in a graphical form, by both business and IT people. The closer process models get to executable processes, the narrower the gap between business procedures, requirements, and software gets; and the shorter the cycle of business process management can be.

Work on the translation of process modelling notations (BPMN, EPC) into executable definitions (BPEL) is well advanced in both research and industrial implementations (6, 7).

5.2 Process Modularity, Composition, Encapsulation

Executable process definitions are a type of software, and are subject to general software engineering considerations. Principles from structured and object-oriented programming should apply just as much to the design of processes and services. In order for a process to be understandable and maintainable, its graphical representation should fit on one page. To achieve that, processes can be composed of sub-processes; and design tools need to support and encourage composition.

Having a larger number of smaller, simpler processes (rather than a small number of large processes) would increase the importance of process repositories - process discovery, and query tools based on process semantics 4.

5.3 Unified Reporting and Integration

Traditionally, enterprise application integration (EAI) has been considered separately from production reporting and business intelligence (BI). A more holistic approach is called for, especially in the context of BPM and SOA. A service which extracts information from a system can equally be used as the source of data for integration (to load into another system), and as a source for BI / reporting.

Similarly, an existing production reporting program which outputs data in an XML format, can be used as an integration source, and wrapped as a service. A unified approach to reporting and integration, when we have applied it, has saved us effort throughout the software life cycle.

5.4 Combined Batch and Event-Driven Data Synchronisation

Traditionally, data synchronisation (including propagation and aggregation) between system databases had been done in batches, scheduled to run once or twice daily, after the close of markets, during out-of-office hours in the relevant time zones. Such batch synchronisation was entirely separate from event driven data flows (primarily orders / trades).

With a messaging-based BPM infrastructure in place, we have been able to combine scheduled batch with event-driven synchronisation. The underlying services which interface to business systems can be used in either mode, and are invoked in both scheduled batch and event-driven executable processes. Special consideration to performance must be given when designing such services.

5.5 Silo Avoidance

In the past, we have always bought stand-alone business applications to perform specific functions. This is unsurprising, as product selection is driven by business departments, and they focus on functionality and user interface. Data and system architecture considerations have been secondary. This has led to a silo effect, manifested in a number of integration challenges discussed in Section 2.3. Long-term we hope that vertical ontologies will be developed sufficiently, standards will be widely accepted, and software vendors will embrace them wholeheartedly; but in the meantime, we could alleviate the silo problem by buying component or service libraries, rather than complete packages. Some business functions can be automated using generic components (e.g a business rule engine) instead of industry-specific applications.

5.6 Support for Self-Service

To meet the needs of the increasingly sophisticated users of business information systems, IT needs to empower them, by giving them more freedom and flexibility in managing and accessing information. We have found that some simple techniques can go a long way.

User-Defined Attributes. It is important to enable the addition of new attributes³ to certain entities in an end-user GUI, without software changes (of database schema, code, message format definitions *etc.*). IT puts all software changes through a rigorous change management process, necessarily involving delays; while new attributes may be required at short notice.

³ The term “user-defined” is used loosely; the creation of new shared data attributes is subject to appropriate data governance controls.

User–Created Reports. Increasingly, business information systems provide “raw data” (an Excel spreadsheet) and data sources (for example in a SAP BusinessObjects universe) to end users, who can then create their own custom reports. That said, the need for IT–developed production reporting has not gone away.

User–Controlled Scheduling. Information system users (from investment managers to compliance analysts) hate having to go through the same series of keystrokes to get the information they need on a regular basis. An adequate user-controlled scheduling mechanism is essential for both reporting and data integration purposes.

Flexible Production Reporting. User–controlled data overrides are sometimes required, especially with multiple or external data sources. Multi–language production reporting is expected of a business with a global presence.

6 Conclusions and Future Work

Our experience of utilising a BPM suite is based on a relatively low-risk, incremental and evolutionary approach. Applying machine–supported business process management throughout, or building a complete service architecture has not been an end in itself, but a long–term strategic vision. Progress has been made in discrete business–sponsored projects of manageable size, each one delivering business benefits – new functionality, further automation. So far, these projects have been successful, and we continue expanding our use of the webMethods BPM suite. Looking back, we have been fortunate to avoid major pitfalls and issues with BPM adoption experienced by others [1].

We are now at a stage where the business and IT can collaborate in the design of new processes, and the optimisation of existing ones, with the tools which the BPM suite provides. In our experience, such collaboration can best be achieved by following an agile project methodology.

The management of data in the enterprise – *Enterprise Data Management* (EDM), or *Master Data Management* (MDM) – has been identified as a priority area for the next couple of years. One reason for the renewed focus on data may be the expected increase in information required by financial services regulatory bodies. Within the industry, work on developing an ontology for the financial domain is well underway [2]. The challenge for us will be to keep the momentum in business process management, and apply it to achieving the objectives of data management. Semantic business processes [4] and related tools ([3], [7]) can be expected to play an important role in meeting this challenge.

References

1. Bandara, W., Indulska, M., Chong, S., Sadiq, S.: Major Issues in Business Process Management: An Expert Perspective. In: Proceedings ECIS 2007 - The 15th European Conference on Information Systems, St Gallen, Switzerland, pp. 1240–1251 (2007)

2. Enterprise Data Management Council. EDM Semantics Repository, <http://www.edmcouncil.org>
3. Dimitrov, M., Simov, A., Stein, S., Konstantinov, M.: A BPMO Based Semantic Business Process Modelling Environment. In: Hepp, M., Hinkelmann, K., Karagianis, D., Klein, R., Stojanovic, N. (eds.) CEUR Workshop Proceedings, Innsbruck, Austria, vol. 251 (2007)
4. Hepp, M., Leymann, F., Domingue, J., Wahler, A., Fensel, D.: Semantic business process management: A vision towards using semantic web services for business process management. In: IEEE International Conference on e-Business Engineering, ICEBE 2005, pp. 535–540. IEEE, Los Alamitos (2005)
5. Hohpe, G., Woolf, B.: Enterprise integration patterns: Designing, building, and deploying messaging solutions. Addison-Wesley Longman Publishing Co., Inc., Boston (2003)
6. Ouyang, C., Dumas, M., Breutel, S., ter Hofstede, A.: Translating standard process models to BPEL. In: Advanced Information Systems Engineering, pp. 417–432. Springer, Heidelberg (2006)
7. Stein, S., Stamber, C., El Kharbili, M.: ARIS for Semantic Business Process Management. In: Ardagna, D., Mecella, M., Yang, J. (eds.) Business Process Management Workshops. Lecture Notes in Business Information Processing, vol. 17, pp. 498–509. Springer, Heidelberg (2009)
8. van der Aalst, W.M.P., ter Hofstede, A.H.M., Weske, M.: Business Process Management: A Survey. In: van der Aalst, W.M.P., ter Hofstede, A.H.M., Weske, M. (eds.) BPM 2003. LNCS, vol. 2678, pp. 1–12. Springer, Heidelberg (2003)

The Logistics Service Engineering and Management Platform: Features, Architecture, Implementation

Christopher Klinkmüller, Robert Kunkel, André Ludwig, and Bogdan Franczyk

Information Systems Institute, Faculty of Economics and Management,
University of Leipzig, Grimmaische Straße 12, 04109 Leipzig
{klinkmueller, kunkel, ludwig, franczyk}@wifa.uni-leipzig.de

Abstract. The logistics service sector is faced with a growing complexity which needs to be handled by cooperating logistics providers aligning their services in a network. This paper introduces the Logistics Service Engineering and Management platform supporting the Fourth Party Logistics Provider business model that aims at establishing a coordinator of such a network. Hence the idea to employ the service oriented design paradigm at the software and at the business level along with the main features of the platform is presented. Furthermore the basic architecture is explained and a closer look at some implementation details is presented.

Keywords: Logistics, Service Orientation, Electronic Service, Business Service.

1 Introduction

Companies are constantly faced with changing market conditions and threats, new competitive pressures in terms of cost, time, and flexibility, and ever changing regulations that require compliance. In order to cope with those conditions, companies outsource internal applications to external service providers in order to focus on the growth of their core activities and competencies. As a consequence, value creation increasingly takes place collaboratively by several organisations forming value creation networks in which business activities are executed by the most capable member.

This holds particularly true in the logistics service sector. Outsourcing logistics services includes business functions such as warehousing, transportation, transshipment, order management, etc., but also computing functions such as enterprise software applications and logistics information systems. However, outsourcing logistics services does usually not only involve a single provider but rather a set of highly specialised companies that need to integrate their services into an end-to-end offering towards the outsourcer. Since integrating and aligning several logistics services and managing their provided operations are not trivial tasks, a rather new business model evolved that concentrates on these tasks and provides according services. The so called *4th party logistics service provider* (4PLP) business model offers services for integration and management of complex value added logistics

services [1]. A 4PLP acts as a requester of rather simple logistics services from different providers, organises those services in a logistics system and provides the resulting value added service to its customers. A central goal of a 4PLP is to set up the corresponding logistics system in a way that it is optimised towards different dimensions, i.e. time, costs, environmental protection, and allows to gain economies of scale among the offered contracts. Figure 1 illustrates how the 4PLP sources, integrates and offers logistics functions by means of *business services* (BS) following the idea to apply the service oriented design paradigm at the business level [2].

Being an intermediate the 4PLP acts as the central information hub in the value added logistics system. Information is provided by customers as well as providers and processed for planning, configuring, monitoring and optimising purposes by the 4PLP. But not only the 4PLP should use the information to set up the logistics system, moreover it has to be shared among the partners to ensure an aligned service execution and to enable a well-orchestrated logistics system. That is why an appropriate information integration and management platform supporting the operations of the 4PLP and its partners respectively customers is needed.

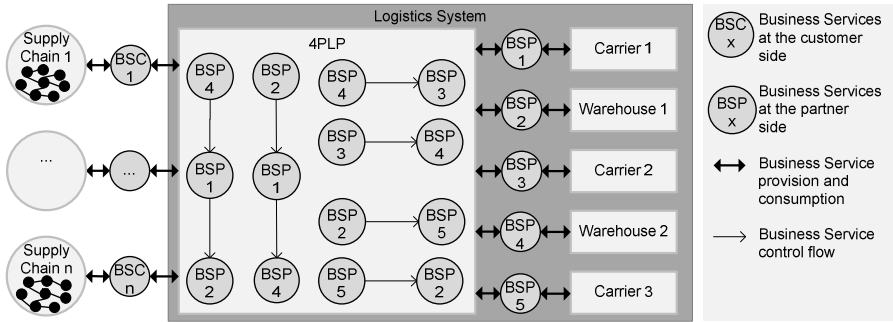


Fig. 1. The usage of partner business services to provide customer business services

This paper introduces such a platform, namely the *Logistics Service Engineering and Management* (LSEM) platform which aims at establishing the service oriented design paradigm at the business and the software level in order to bridge the gap between both domains. The goal of this paper is to provide the architectural foundation of this platform which future efforts to provide appropriate tool support can ground on. More precisely this paper outlines the main features (section 2) and presents the basic layered architecture (section 3). Implementation details (section 4) are presented as well and refer to the current version of the platform that comprises general integration solutions. An overview of the related (section 5) and the future work (section 6) concludes this paper.

2 Features

This section introduces the top level features of the LSEM platform. In doing so the service oriented paradigm is applied at the business and the software level. Hence the features are introduced by investigating the phases of the service lifecycle introduced

by Kohlborn et al. [3] at both levels. The lifecycle consists of the phases: analysis, design, implementation, publishing, operation and retirement which are examined with regard to the business and software level in the following subsections.

2.1 Features at the Business Level

Against the consideration of Kohlborn et al. the BS of the 4PLP are designed with regard to an individual contract and are not provided for mass consumption. Therefore the publishing phase is not part of this lifecycle. In the following each phase of the lifecycle is explained with regard to the tasks the 4PLP has to conduct. The features of the LSEM platform arise from these tasks by providing supportive tools including models and methods.

Analysis: During this phase the 4PLP creates a specification of the required BS based on a tender. The aspects that must be reflected include a description of the service functionality which refers to one of the business service types transport, storage, transshipment or value added. It also can refer to a combination of them. The kind of products and materials that need to be handled also determines the functionality. Furthermore the interaction protocol between the 4PLP and the customer needs to be specified. The terms of payments linked with service level agreements declaring quality aspects and legal aspects are another important part. Delivery reliability, delivery quality and deliver flexibility are vital indicators that should be negotiated at this point. The same specification format is used in this phase to create a pool of BS offered by logistics providers. These BS descriptions are the input for the next phase. Another task during the phase is the conduction of feasibility analysis to check risks and capabilities for the implementation of the demanded BS.

Design: Having the specification at hand the 4PLP starts to design the implementation of the BS. Because of the need to find synergy effects the BS should not be implemented separately, but in the context of the logistics system. The pool of partners created during the analysis phase describing their logistics capabilities constitutes a central point of information at this point. The design of the logistics system needs to orient itself by the demanded flows of goods and information. Quality aspects, legal aspects, terms of payment and coupling of processes restrict the composition. In order to examine the logistics system with regard to its behaviour and changing conditions, it is necessary to enable the simulation of the logistics system and this way to check the system's sustainability. Due to the complex requirements towards the logistics system versioning and archiving of different models should be supported. This phase and the analysis phase accompany the negotiation process between the 4PLP, its partners and customers. If the negotiation is not successful the lifecycle stops here. Otherwise detailing the logistics system model might be a further needful step in this phase.

Implementation: While the first two phases were mainly concerned with negotiation and planning this phase is busy with the preparation of the service execution. Hence the tasks for the 4PLP encompass the deployment of the service on the business level which is done by establishing the designed processes through the integration of the partners and the adaption of the existing logistics systems. Furthermore appropriate IT

support needs to be deployed. This on the one hand ensures the establishment of the flow of information enabling a smooth coordination of partners during the service operation. On the other hand a monitoring environment must be installed that allows the 4PLP to supervise the service operation. For that reason the introduction of radio-frequency identification and sensor based devices is necessary to make events occurring in the real world automatically processible. The related requirements at the electronic level are provided in the next subsection.

Operation: The actual service execution during the operation phase is left to the partners of the 4PLP. Having established the processes and IT systems needed for the operation phase, the 4PLP is supervising the service execution in this phase with regard to defined quality parameters and legal aspects. Monitoring enables the 4PLP to recognise faults and exceptions and to react to the problems. It also allows for analysing the processes over a longer period, so that a revision linked with process improvements and optimisations can be done. Depending on the result of this revision tasks from the analyses, the design and the implementation phase can be triggered.

Retirement: After the contract expires the 4PLP needs to terminate the service. It therefore abandons or changes collaborations with partners depending on the involvement of the partners in other services. In case of need parts of the logistics system have to be redesigned because potential for optimisation arises at this point. Therefore a detailed analyses of the whole lifecycle of the determined service becomes relevant. It also is important to archive this information and to shut down the deployed software environment.

2.3 Features at the Software Level

The features at the software level are concerned with *electronic services* (ES) as the counterpart of the BS. According to the business level the features comprise tools, methods and models for managing the lifecycle of ES. As contrasted with the BS lifecycle the ES lifecycle includes the publishing phase which is used to make ES available in the runtime environment. This runtime environment is used for hosting the tools, the operation systems used by the 4PLP to conduct its operational business and to connect the systems with the use of ES. Hence not only the lifecycle phases but also the runtime environment are examined in the following.

Analysis: During this phase artefacts for analysing the needs at the business levels must be provided. Those artefacts support the derivation of detailed requirements from the BS and logistics system models. Furthermore they allow for adding requirements not captured by these models and for conducting feasibility analyses which support those at the business level.

Design: After the requirements were recorded the design of the ES and workflows supporting the BS is performed. This includes the construction of various models, e.g. models for data management, workflows, user interfaces, system interfaces, business logic or ES interfaces. During this phase principles of designing *Service Oriented Architectures* (SOA) [4] like loose coupling, abstraction, reusability etc. must be considered to ensure well-orchestrated ES.

Implementation: After the design phase all necessary artefacts need to be implemented. The artefacts provided by the platform have to focus on a high degree of automation including generation and configuration of artefacts [5] to support the implementation. Applying those methods is not restricted to this phase, instead it has to be considered in all phases of the ES lifecycle and also in parts of the BS lifecycle.

Publishing: During this phase the implemented artefacts are integrated into the working environment and the developed ES are made available for the use inside the platform.

Operation: While the ES are used they and the whole runtime environment need to be monitored in order to recognise problems during the execution. If any problems occur appropriate countermeasures have to be captured to recover the functioning of ES or to ensure the quality of services. This can involve the integration of new features and the development of new service versions. At this point it is useful to be able to run different service versions in parallel. To keep track of the whole system a sophisticated management environment needs to be set up.

Retirement: After a contract expires or new service versions were put into operation ES and linked artefacts have to be put out of operation so that the resources of the runtime environment are released and can be allocated to other business cases.

Runtime environment: As already mentioned the runtime environment hosts the tools, the application systems and the solution developed on its bases. Reversely the tools on the software level are used to configure this environment. The following list presents the core capabilities of this runtime environment which among others comprise capabilities of an *enterprise service bus* (ESB) [6].

- Hosting of Tools and Systems: The runtime environment makes available the application servers, web servers and frameworks to host and run the operational systems of the 4PLP, the tools provided by the platform and the solutions developed during the BS and ES lifecycles. Thereby the environment must provide solutions for overcoming heterogeneity, e.g. in platforms, programming languages and technical protocols, for enabling the distribution of the platform onto globally allocated servers and for ensuring scalability and security.
- Interfacing of Tools and Systems: It is also necessary to integrate the software with each other. Therefore the runtime environment provides solutions for running components that serve as interfaces towards the software. These components make available information and functionality of software systems. In order to facilitate reuse and traceability during all phases it must be possible to register the defined ES at a central directory where descriptions of the ES are stored, too. It needs to be straightened out that not only the hosted software will be integrated into the platform. Moreover systems that run on servers of partners or customers also need to be integrated presenting challenges to security issues. Sometimes ES have to be developed from scratch to provide auxiliary functionality not available so far.
- Deployment of Information Flow: As mentioned in section 1 a vital point that the platform needs to address is to share information among the involved parties. With ES at hand the access to the information is granted. But to ensure that the information is available at the point where it is needed the deployment of an

information flow must be supported. Therefore the runtime environment has to provide tools that allow for execution of flow logic. A sophisticated messaging system should additionally take care about secure and reliable data transmission according to the flow logic. Beyond that the different data formats of the systems need to be matched, this also holds true for the models that need to be aligned on the syntactic as well as on the semantic level.

- **Provision of User Interfaces:** To ensure access to services from different platforms it is necessary to have the possibility for providing self developed user interfaces. A typical example therefore is the provision of user interfaces on mobile devices that are not able to display conventional web-based user interfaces.
- **Monitoring:** The environment needs to provide capabilities allowing for supervising the business and the operation of the software. This on the one hand demands tools for inspecting the information flow. On the other hand tools for examining service level agreements at the software level are demanded. In both cases it must be possible to evaluate predefined rules that are used to recognise violations or exceptions. Furthermore it should be possible to run scripts that present countermeasures or signal the problems.

3 Layered Architecture

The previous section outlined the top level features of the LSEM platform which provides a runtime environment for setting up software solutions and a collection of tools allowing to work with the runtime environment and to manage the business.

A software architecture is understood as a description of the subsystems and components of a software system and their relationships. It is furthermore specified via different views [7]. The paper thereby focuses on the development view of the 4+1 view model [8] presenting the logical organisation of components in different layers and their relation. This view was chosen because it serves as a basis for detailed requirements determination and for the allocation of work. This allows to refine requirements and to isolate work packages.

The architecture of the platform is introduced incrementally. At first the runtime environment as the bases for the platform is outlined. Afterwards the runtime environment is extended by the tools that the platform provides.

3.1 The Architecture of the Runtime Environment

The runtime environment is the grounding for the whole platform. Software solutions set up to support the 4PLP's operational business as well as the tools build upon this environment. It therefore provides central integration and hosting capabilities. As clarified in section 2 service orientation is chosen to implement the features of this environment. The S3 reference architecture [9] constitutes a starting point for the design of the architecture which has to be adapted. Figure 2 shows the result of the adaptation which lead to an architecture consisting of eight layers organised in two dimensions.

The layers consumer, business process, electronic services, electronic service components and operational systems belong to the first dimension reflecting the

functional integration of systems and humans. The three layers in the second dimension are the integration, the quality of services and the information architecture. This dimension is mainly used for ensuring basic integration capabilities and non-functional requirements. Below the layers are described with respect to their purpose and their LSEM platform specific components.

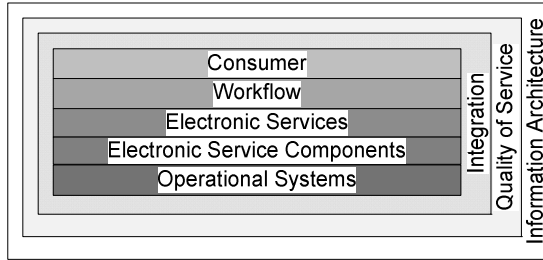


Fig. 2. The runtime space of the LSEM platform (referring to [9])

The operational systems layer summarises all application systems. These systems can be distinguished by the owner who typically is the 4PLP itself or one of its partners respectively customers. Hence the systems can be internal or external to the platform.

The electronic service components layer [10] deals with the implementation of ES in form of service components by interfacing the application systems. Hence there are artefacts for storing, deploying and invoking service components. Because of application systems relying on different technologies there need to be several types of adaptation components helping to interface the application systems' functionality and to run the electronic service components.

The electronic services layer exposes ES to the consumer and the business process layer by providing interfaces to the functionality of the electronic service components. An ES repository stores this interfaces and further information about the ES, so that a retrieval of the repository is possible during design and runtime.

The workflow layer [11] enables the technical representation of business processes as workflows. The workflow engine and the workflow repository realise this layer inside the runtime environment. While the engine enables the workflow execution, the workflow repository supports the storage of workflow definitions.

In the context of the LSEM platform the consumer layer consists of components ensuring provision of user interfaces [12] and of clients for application systems. The presentation view component is the runtime environment for graphical user interfaces. There are different types of it because the views should potentially be available on different devices. Furthermore there is the presentation controller which runs the logic for the views. Analogical to the presentation view there are different types of it. Both components are supported by a consumer profile component that stores preferences of users. These preferences are used to provide customised user interfaces. The human interaction component enables the binding of workflows to user interfaces. Finally the adaptation components are used to bind the application systems to ES and are similar to the ones in the electronic service component layer.

The integration layer possesses typical ESB capabilities and enables hosting and connecting distributed systems. At the centre there is a message oriented middleware facilitating asynchronous and reliable data exchange. This layer also comes along with application servers for hosting components and application systems. Additionally diverse environments for several programming languages are part of this layer. This way it is possible to integrate systems, components and tools basing on a wide range of technologies. The access manager is the central place of the platform regarding security issues like authentication and authorisation. It stores and checks the rights of actors regardless if they are humans or machines.

The quality of service layer empowers the platform to determine if the developed solutions meet the technical requirements. The logging module therefore collects necessary data during runtime. This data is used by the observation module which checks the adherence to defined rules. At last the notification module signals abuses to the responsible persons.

Finally the information architecture layer contains the canonical data format. This data format defines a standard vocabulary. It reflects the business terms and ensures a platform wide uniform understanding. Furthermore the format serves as an intermediary when integrating systems with different data formats. To enable the mapping of other data formats onto the canonical data format, this layer provides a data transformation module. Lastly this layer allows to monitor and analyse business related information spread across the platform. Besides a logging, an observation and a notification component for monitoring business processes it includes a business intelligence component enabling the detailed analysis of the 4PLP's business.

It should be noted that the governance and policies layer as a part of the S3 reference architecture and responsible for supervising the compliance to policies and strategic issues is not considered to be part of the runtime environment. Instead the tasks of this layer are included in the LSEM tools introduced in the next subsection.

3.2 The Integration of the LSEM Tools into the Runtime Environment

The previous subsection presented the architecture of the runtime environment as the core of the LSEM platform. Only some of the runtime environment's components are domain specific, e.g. the canonical data format, while the rest of the components can be deployed domain independently, e.g. the workflow engine. The tools enabling the 4PLP to implement the ES and BS service lifecycle by contrast must be tailored to the logistics domain and the 4PLP business model. These tools are organised in two layers that extend the runtime environment as illustrated in figure 3.

According to the service lifecycle the business service tools layer possesses one sublayer for each phase. Together with the application systems these tools enable the 4PLP and its partners to conduct their work. Hence the business service tools layer is a sublayer of the operational systems layer.

The electronic service tools layer instead allows for configuration and use of all components of the runtime environment to implement individual solutions for the 4PLP and its partners. It therefore extends the second dimension of the architecture of the runtime environment. Like the business service tools layer the electronic service tools layer consists of sublayers representing the electronic lifecycle's phases.

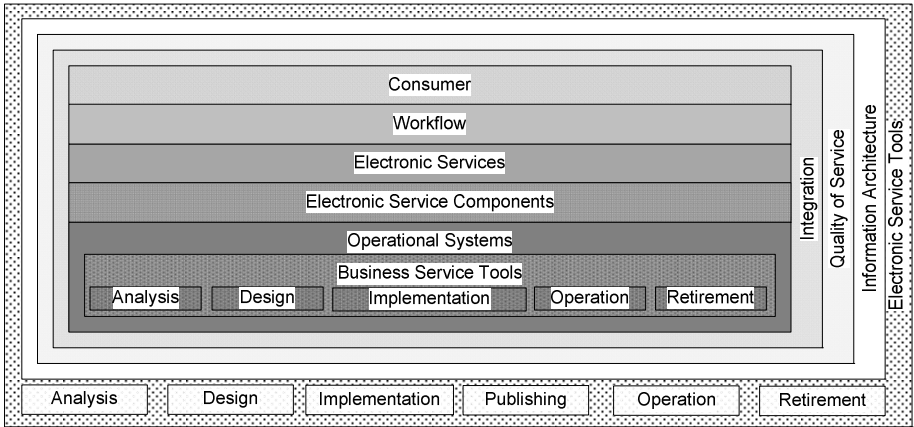


Fig. 3. The Runtime and Tooling Architecture

Alongside these layers a further component in the information architecture layer is introduced. The model alignment component enables the mapping of models in order to ensure consistent models describing different aspects of services

4 Implementation Details

Having the architecture of the platform a first version of the platform was implemented and some business scenarios were set up on this basis to examine the platform's overall behaviour and to illustrate the application of the platform. The main focus during this first iteration was on the core integration capabilities neglecting the tooling environment. Therefore different products from the open source market were examined with regard to their integration capabilities and the platform's requirements. After a set of components was chosen the integration and configuration of them was conducted. As one result of this procedure table 1 provides a brief overview of the main components employed in the first version. First prototypical scenarios were implemented on that basis as well. So a brief examination of the interaction of the platform's components and of the overall behaviour could be conducted. Nevertheless a detailed evaluation of the platform is subject to future work.

At this point it should be noted that many of the technologies are open source tools that were originally produced under the patronage of Sun. Due to the takeover of Sun by Oracle and the associated stagnation in the development of these technologies at this moment it is unclear if the LSEM platform needs to be revised in respect of the basic components. For that reason and because of the limited space a detailed description of the implementation decisions including aspects like scalability, security, data transformation and workflow definition is not provided here.

Beyond that first approaches covering various aspects of the tools are under way at the time of writing. This among others includes an environment for the alignment of models within the platform [13], the alignment of BS and ES specifications [14] and the management of ES [15].

Table 1. Components of the platform's first version

Layer	Components
Operational Systems	OpenERP, MS Dynamics NAV, SAP TM, RFID based warehouse management system
Electronic Service Components	XML-RCP for PHP, Equinox, Connectors for Java, .Net, SAP NetWeaver
Electronic Services	jUDDI, SwordFish Service Registry, JAX-WS, JAX-B
Workflow	Sun BPEL Engine
Integration	Liferay Portal, WS-Import
Quality of Service	OpenESB, Apache Camel, OpenMQ, GlassFish Server, Jboss AS, Java EE, SwordFish
Information Architecture	GS1 BMS, ebXML
Tools	Eclipse SOA, Eclipse Modeling, Eclipse RT, NetBeans DIE

5 Related Work

This section introduces work that is related to the presented architecture. The first work to be mentioned here is the S3 reference architecture presented in [9] which the LSEM platform is based on. Arsanjani et al. introduce an architecture consisting of nine layers, but do not suggest components for these layers. Because of the structure allowing to reflect all of the goals of the platform this architecture was taken as a starting point. Ideas about needed components were among others derived from [10], [11] and [12]. Each of these works deals with one of the layers of S3. In [16] the service-oriented modelling and architecture approach is introduced which extends the S3 architecture by a method helping to develop a solution based on S3 providing hints for the adaption of the reference architecture.

In [17] Mos et al. distinguish two spaces inside a SOA. These two spaces are the design and the runtime space where the design space represents the modelling tools needed to develop a solution while the runtime space includes monitoring tools. This mainly aims at the platform's tools and neglects the runtime environment, but this distinction is considered as too coarse grain for the LSEM platform's tools cause it does not fully reflect the service lifecycles. Nevertheless it is possible to divide the platform into a design and a runtime space by changing the understanding of the distinction. The runtime space then is equal to the runtime environment and the design space comprises all tools.

Cheesman and Ntinolazos explain their SOA Reference Model in [18]. Like it is done in the platform's context they also distinguish between BS and ES. The reference model also includes guidelines for implementing an SOA. Unfortunately a reference architecture is missing, so that this reference model gave hints during the design, but is not reflected by the platform's architecture.

Another layer structure is presented by Emig et al. in [19]. This layer structure includes six layers which can be compared to the five layers of S3's first dimension. But due to the lack of the second dimension this architecture does not allow for a detailed separation of features as the S3 reference architecture does.

All these approaches are general ones without restricting the domain of application. Among others [20] and [21] present architectures that focus on certain domains.

These architectures show how to utilise SOA to develop domain specific software solutions based on services. Nevertheless they do not consider the business level and do not focus on logistics which limits their application, e.g. in choosing a canonical data format.

Finally there are some proprietary platform's that focus on the provision of software solutions for logistics companies, e.g. Axit's AX4, TRANSPOREON's Tisys and Elemica's Elemica. But these platforms do not consider the specifics of the 4PLP business model. Due to the proprietary character of these platforms it was not possible to examine them at the software level.

6 Future Work

This paper introduced the LSEM platform and focused on the features and the architecture. The implementation so far considered the runtime environment and only some of the platform's tools are already designed and implemented. That is why the biggest efforts will be made with regard to the tools. In the near future the focus is on tools supporting the analysis and the design phase on both the business and the software level. This includes appropriate planning tools enabling a 4PLP to create and configure logistics systems and to incorporate logistics providers in it. Linked to that on the software level the acceleration of the implementation process is to the fore.

Nevertheless the runtime environment will be subject to future work, too. Because of the problems with some of the basic components as described in section 4 this includes a revision of the current functionality. It furthermore comprises the enrichment of the runtime environment by software as a service features that allow the 4PLP to provide software systems for logistics providers on demand. Like the conventional partner systems it will also be possible to integrate such software solutions into the platform.

Acknowledgement

The work presented in this paper was partly funded by the German Federal Ministry of Education and Research under the projects InterLogGrid (BMBF 01IG09010F) and Logistics Service Bus (BMBF 03IP504).

References

1. Kutlu, S.: Fourth Party Logistics: The Future of Supply Chain Outsourcing? Best Global Publishing, Brentwood (2007)
2. Nayak, N., Nigam, A., Sanz, J., Marston, D., Flaxer, F.: Concepts for Service-Oriented Business Thinking. In: IEEE International Conference on Services Computing, Chicago, USA (2006)
3. Kohlborn, T., Korthaus, A., Rosemann, M.: Business and software service lifecycle management. In: IEEE International Conference on Enterprise Distributed Object Computing, Auckland, New Zealand (2009)
4. Erl, T.: SOA Principles of Service Design. Prentice Hall, Upper Saddle River (2007)

5. Karakostas, B., Zorgios, Y.: Engineering service oriented systems: a model driven approach. IGI Global, Hershey (2008)
6. Chappell, D.A.: Enterprise Service Bus. O'Reilly Media, Sebastopol (2004)
7. Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., Stal, M.: Pattern-Oriented Software Architecture: A System of Patterns. Wiley, West Sussex (2009)
8. Kruchten, P.: The 4+1 View Model of Architecture. *IEEE Software* 12, 42–50 (1995)
9. Arsanjani, A., Zhang, L.-J., Ellis, M., Allam, A., Channabasavaiah, K.: S3: A Service-Oriented Reference Architecture. *IT Professional* 9, 10–17 (2007)
10. Zhang, L.-J., Zhang, J.: Design of Service Component Layer in SOA Reference Architecture. In: International Computer Software and Applications Conference, Seattle, USA (2009)
11. Zhang, L.-J., Zhang, J.: Componentization of Business Process Layer in the SOA Reference Architecture. In: IEEE International Conference on Services Computing, Bangalore, India (2009)
12. Zhang, L.-J., Zhang, J., Allam, A.: A Method and Case Study of Designing Presentation Module in an SOA-based Solution Using Configurable Architectural Building Blocks (ABBs). In: IEEE International Conference on Services Computing, Hawaii, USA (2008)
13. Augenstein, C., Müller, H., Franczyk, B.: Developing a unified service model for collaborative modeling of logistics services. In: Advanced Information Technologies for Management, Wrocław, Poland (2010)
14. Kluge, R.: Preselection of Electronic Services by Given Business Services Based on Semantic Concept Correspondence. In: International Workshop on Service Oriented Computing in Logistics, San Francisco, USA (2010)
15. Belter, R., Spies, T., Ludwig, A., Franczyk, B., Eicker, S.: Towards Information Transparency in the Context of Service Management. In: IEEE International Conference on Service-Oriented Computing and Applications, Perth, Australia (2010)
16. Arsanjani, A., Ghosh, S., Allam, A., Abdollah, T., Ganapathy, S., Holley, K.: SOMA: A method for developing service-oriented solutions. *IBM Systems Journal* 47, 377–396 (2008)
17. Mos, A., Boulze, A., Quaireau, S., Meynier, C.: Multi-layer perspectives and spaces in SOA. In: International Conference on Software Engineering, Leipzig, Germany (2008)
18. Cheesman, J., Ntinolazos, G.: The SOA Reference Model. *CBDI Forum Journal* 3 (2004)
19. Emig, C., Langer, K., Krutz, K., Link, S., Mayerl, C., Momm, C., Abeck, S.: The SOA's Layers. Research Report, University of Karlsruhe, Germany (2006)
20. Gao, Z., Kang, X., Zhang, R., Deng, S.: Research and Implementation of Campus Application Integration Based on SOA. In: International Conference on Advanced Computer Theory and Engineering, Phuket, Thailand (2008)
21. He, X., Li, H., Ding, Q., Wu, Z.: The SOA-Based Solution for Distributed Enterprise Application Integration. In: International Forum on Computer Science-Technology and Applications, Chongqing, China (2009)

Cooperative Semantic Document Management

Joerg Leukel¹, Michael Schuele², Andreas Scheuermann²,
Dominic Ressel¹, and Wiltrud Kessler¹

¹ University of Hohenheim, Information Systems 2,
70599 Stuttgart, Germany
{joerg.leukel,dominic.ressel,wiltrud.kessler}@uni-hohenheim.de

<http://wi2.uni-hohenheim.de>

² Jesselle GmbH
70599 Stuttgart, Germany
{michael.schuele,andreas.scheuermann}@jesselle.de

<http://www.jesselle.de>

Abstract. Document management concerns the storage, retrieval, and presentation of documents being created from multiple sources and delivered to different users. Semantic approaches for document management are based on enriching metadata and deriving semantic document models. However, the detection of relationships between documents is constrained by the metadata quality as well as the underlying domain ontology. This paper proposes a software architecture for cooperative semantic document management. Its rationale is to separate the semantic representation of single documents from knowledge about domain-specific relationships in two architectural layers. The applicability and utility is demonstrated in an use case scenario from civil engineering.

Keywords: Document Management, Information Extraction, Multi-agent Technology, Ontology, Semi-structured Information.

1 Introduction

Documents provide, in a literal sense, a fragmented documentation of tasks, people, organizations, and information being involved in business processes. As such, a rich set of documents can be regarded as an important source of knowledge existing in and across organizations. The goal of business information systems is making this knowledge available to end-users or application systems.

A core *problem* in document management is the identification of relationships between documents. Such relationships are often implicit and manifold, e.g., related to time, location, organization, task, process, or project. Current document management systems (DMS), however, fall short of identifying these relationships. The reason is an insufficient quality of metadata, which significantly aggravates the integration of diverse documents. A promising approach to overcoming this situation is adopting semantic technologies, i.e., providing a semantically richer document representation, which allows reasoning about large

sets of documents. This avenue of research revises current DMS architectures, often by adding a semantic layer.

This paper proposes a novel two-tier architectural extension of DMS for cooperative semantic document management. The rationale is to separate knowledge about business documents in two layers. First, basic relationships are identified and added to a knowledge base. Second, the enforcement of domain-specific integrity over documents is subject of cooperative software agents. These agents supervise documents that emerge over time, check integrity rules, and in case of violation trigger the notification of end-users or application systems. Cooperative means that these agents share a common goal, which is mutually beneficial, and maintain a global knowledge base of relationships. We develop this architecture from the perspective of knowledge engineering by adopting ontology-based information extraction and multiagent engineering. We demonstrate the applicability and utility of our approach in an use case scenario from civil engineering. The contribution of this research is the architecture that combines two AI technologies for document management.

The rest of the paper is organized as follows. Section 2 reviews related work. Section 3 provides a basic model for capturing business documents. Section 4 proposes the architectural extension of DMS. In section 5, we evaluate our proposal. Finally, we draw conclusions and point to future research.

2 Related Work

2.1 Semantic Document Management

Semantic document management aims at integrating documents by enriching their metadata quality, so that searching for and navigating in large document sets can be improved. It depends on the expressivity of the language used, e.g., RDF or description logic. An ontology serves as the underlying schema and is then used for annotating documents. The representation of electronic documents is amended whereas the original document remains unchanged.

Several architectures for semantic document management have been proposed. They can be described by criteria such as document formats, annotation support, annotation storage, ontology support, and ontology language. Early Semantic Web research put emphasis on enriching Web documents. Respective architectures and implementations concentrate on manually and semi-automatically annotating Web documents. A detailed review of 27 annotation tools can be found in [1]. For instance, AktiveDoc provides graphical annotation support for HTML pages [2]. Annotations are stored as RDF triples in an SQL database, whereas the expressivity is confined by the RDF model. More advanced architectures support ontology languages of higher expressivity (e.g., OWL DL). For instance, the PDFTab extension to Protégé [3] allows to annotate multiple PDF documents. The annotations as well as the ontology are stored in the PDF document. Such semantic documents are self-contained, but not suitable for document management applications which require a unified storage and access to semantic documents. Therefore, it has been proposed to store documents fully as RDF triples

in a dedicated RDF repository. Nesic [4] describes a semantic DMS based on the NEPOMUK Social Semantic Desktop. This system provides not only annotation and retrieval support, but also makes recommendations for possible annotations and builds upon a domain-independent document ontology.

For the purpose of identifying relationships, current architectures are limited and rely often on a document ontology, which then can be used for querying the knowledge base. A respective DMS is therefore passive in a sense that all implicit relationships are subject to reasoning capabilities and will be detected by end-users only, if they submit respective queries. This time-consuming task depends largely on the user's domain knowledge. Our research addresses right this problem.

2.2 Agent-Based Document Management

Agent technology has been adopted in document management, since it provides capabilities for considering the distributed nature of document sets (at least logically, with documents being created by different parties).

An early stream of research studies software agents that represent document management functionality such as document import, monitoring, notification etc. MANTHA is a system for managing distributed hypermedia documents [5]. Agents exist for typical roles found in document management. A more elaborate approach is taken in [6], which develops an agent-based model from the perspective of the document life-cycle. It is concerned with organizational issues, such as collaborative document usage. These works, though, do not help unfolding implicit relationships between documents.

Another direction is representing single documents or related sets as agents. By amending documents with agents, some parts of the document information are shifted to agents. These agents solve specific problems such as finding related documents respectively agents. Agent-to-agent communication and protocols enable such behaviour. Reed et al. [7] propose agent-based cluster analysis. Software agents represent subclusters and clusters, thus sets of documents, which then perform cluster analysis to populate the dynamic clusters with documents. The results indicate that this approach is computationally less expensive than hierarchical agglomerative clustering. With regard to document relationships, it determines is-a relations only, thus is severely limited.

3 Basic Model

This section defines the core elements of electronic documents in business. The document model will be used and extended in the subsequent section. We also define assumptions of our work.

An electronic document is constituted by a multitude of symbols and media to represent a set of information, which can be stored, retrieved, and presented electronically [8]. It contains information of three categories [9]: (1) Content information is the actual information relevant to the domain, (2) structure information is about the arrangement of content and thus determines the document's skeletal structure, and (3) presentation information is about formatting

both structure and content information for readers. The latter solely pertains to the appearance of a document and therefore is not relevant for our purpose. Structure information is necessary to be able to identifying content information correctly (e.g., by separating the items of a bill-of-material). Acknowledging the existence of these categories, the goal of document analysis is to abstract a logical document model from large sets of heterogeneous documents.

Definition 1 (document set). A document set is a directed graph $DS = (D, R, DT, RT, CD, CR, DG, G)$ consisting of documents $d \in D$ as nodes being connected by directed relationships $r \in R$ as edges, i.e., $R \subseteq (D \times D)$. DT is a set of document types. RT is a set of relationship types. CD is a function $CD \in D \rightarrow DT$, which maps each d to one document type $dt \in DT$. CR is a function $CR \in R \rightarrow RT$, which maps each r to one relationship type $rt \in RT$. DG is a set of document groups. G is a (mathematical) relation $G \in D \rightarrow DG$, which maps each d to none, one or more document groups $dg \in DG$.

Example 1. Assume a document set from a construction project, which differentiates three document types by $DT = \{contract, specification, invoice\}$ and categorizes specifications and invoices according to their processing state into $DG = \{received, approved, rejected\}$. The documents are as follows: $D = \{d_1, d_2, d_3, d_4, d_5\}$ with $C(d_1) = \{contract\}$, $C(d_2) = \{specification\}$, $C(d_3) = \{invoice\}$, $C(d_4) = \{invoice\}$, and $C(d_5) = \{invoice\}$. The processing states are $G(d_2) = \{approved\}$ and $G(d_3) = \{rejected\}$. d_2 is a supplement for contract d_1 , thus $r_1 = (d_1, d_2)$ and $rc_1 = \{suppement_{t\phi}\}$. d_3 is the invoice for d_1 , thus $r_2 = (d_1, d_3)$ and $rc_2 = \{successor\}$. d_4 is the corrected version of invoice d_3 , thus $r_3 = (d_4, d_3)$ and $rc_3 = \{replacement\}$.

The problem is that current DMS technology does not provide the type of each document, i.e., function CD , and even more important the relationships between documents, i.e., R and CR . Determining these is just the objective of our work. Our assumption is that these relationships can be identified by extracting relevant information from documents. That is why we need a document model as follows.

Definition 2 (document). A document $d \in D$ consists of atomic content components $cc \in CC$ and assigned values $cv \in CV$ by $d = (cc_1, cv_1 \dots, cc_n, cv_n)$.

Example 2. A specification consists of seven content components: address, contract, itemreference, itemdescription, itemquantity, itemprice, validity. Then, document d_2 with $C(d_2) = \{specification\}$ is defined as $d_2 = (\{address\}, \{University\ of\ Southampton\}, \{contract\}, \{A3526\}, \{itemreference\}, \{001\}, \{itemdescription\}, \{Radiator\ Universal\ 3024\}, \{itemquantity\}, \{12\}, \{itemprice\}, \{1, 998\}). \{validity\}, \{2011 - 05 - 31\})$.

4 Architectural Design

4.1 Rationale

The rationale is to separate knowledge about business documents as follows: (1) The conceptualization is subject of a document ontology, which is used for

semantic document representation. Basic relationships between documents are identified by means of ontology-based information extraction and then expressed using constructs of the ontology. (2) The enforcement of richer integrity constraints is subject of cooperative software agents. These agents supervise documents that emerge over time. Each agent represents a domain-specific dynamic set of inter-related documents, which are organized in document groups *DG*. It implements rules describing integrity constraints over two or more documents (e.g., order of document types considering their state). In case of integrity violation (e.g., during specification review whether to accept or reject), the agent adds a respective assertion to the knowledge base, which then can be used for, e.g., giving recommendations for action to end-users. These agents are cooperative, because they share a common goal, which is mutually beneficial (identification of relationships), and maintain a global knowledge base of relationships. Non-cooperative agents would hide particular relationships in their local knowledge base.

The architectural extension consists of two layers (Fig. 1). These layers are built upon a conventional DMS, which stores documents as files and provides some limited metadata.

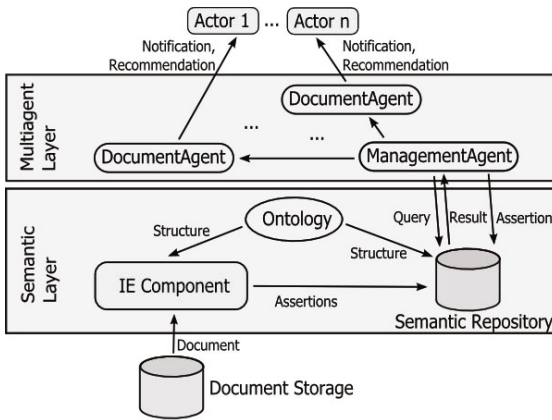


Fig. 1. Architectural Extension of DMS

The *semantic layer* retrieves documents and metadata from the storage layer; the only requirement is that some form of XML serialization is ensured. Each new document inserted into the document storage is processed by the information extraction (IE) component that extracts additional information by parsing the document text. This information as well as the original metadata is added as assertions into the semantic repository. This repository is structured by means of a business document ontology.

The *multiagent layer* is composed of ManagementAgent and DocumentAgent. The ManagementAgent is responsible for querying the semantic repository for new documents, either triggered by events or time, and adding assertions. It thus serves as an interface between the repository and the document agents.

For each document group, a DocumentAgent is dynamically generated which governs this group and maintains integrity constraints over these documents, other documents, and document groups.

4.2 Semantic Layer

Ontology-based Information Extraction (OBIE). IE automatically extracts structured information such as entities, relationships between entities, and attributes describing entities from unstructured sources [10]. Diverse methods [10] for rule-based as well as statistical IE exist. Rule-based methods use extraction rules incorporating linguistic knowledge and domain knowledge. In OBIE, an ontology guides the extraction process. OBIE is in particular adequate for specific domains, which both narrow the space of potential information and require dedicated domain knowledge for extracting the right facts. This is true for the business documents considered here.

We ground the semantic layer on the general architecture of OBIE as proposed in [11]. Its main constituents are preprocessor, IE rules, ontology, ontology editor, and semantic repository plus query answering system. Since our research is concerned with document relationships, we focus on IE rules and employ standard methods for preprocessing. An overview of the OBIE architecture is given in figure 2.

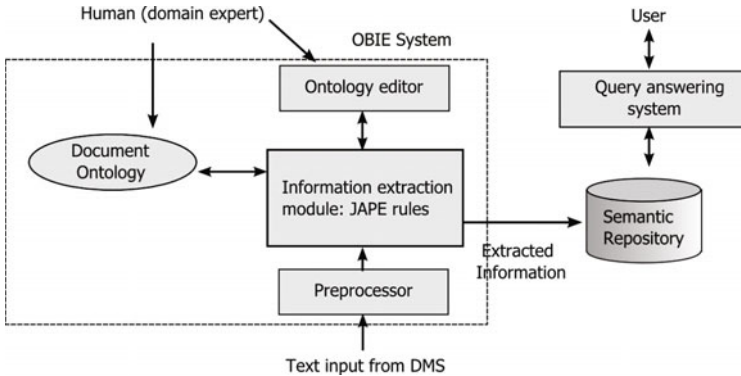


Fig. 2. OBIE architecture adopted from the general architecture in [11]

Information Extraction by Rules. The extraction rules are based on regular expressions. We make use of JAPE (Java Annotation Patterns Engine) rules [12]. A grammar includes one or more rules. The grammar definition begins with the input it processes. This could be annotations either from preprocessing or from the output of a previously applied rule. It is also possible to refer to an ontology instance identified in the document. The example shown in Tab. 1 contains two domain-specific rules for construction projects: the first rule determines the heat conductance of heating elements (e.g., as described in a contract or specification). The latter second rule determines the document type.

Table 1. Example JAPE rules

Rule: insulationValueWindow	Rule: documentypeInSubject
Phase: insulationValue	Phase: classification
Input: Classification Lookup Token	Input: Lookup Subject
Options: control = first	Options: control = brill
(Lookup.URI == "Window") (Token)*	(Lookup within Subject, URI== "contractForWork"
(Token.string == "heat conductance"	Lookup within Subject, URI=="specifications"
Token.string == "heat conductivity"	Lookup within Subject, URI=="order"
Token.string == "thermal conductivity")	Lookup within Subject, URI=="acceptanceProtocol"
Token.string == "ug-value"	Lookup within Subject, URI=="invoice"
Token.string == "ug value")	Lookup within Subject, URI=="changeOfOrder")
(Token)*	:classification ->
(Token.string !~ "[0-9,.]")?	:classification.Classification = rule = "documentypeInSubject"
((Token.string =~ "[0-9,.]")?)	
:insulationValue	
(Token.string !~ "[0-9,.]")?	
-> :insulationValue.InsulationValue =	
rule = "insulationValueWindow"	

Document Ontology. In OBIE, it is essential to rely on an ontology as a formal and explicit specification of a shared conceptualization [13]. Modeling domain knowledge requires the application of a methodology for ontology engineering and an ontology language of adequate expressivity, while retaining computational completeness and decidability, such as description logic [14].

We define a core document ontology, which builds on and is inspired both by well-grounded enterprise ontologies, in particular the TOVE Ontology [15] and the Enterprise Ontology [16], and concepts originating from document management. The ontology is at least constituted by four key concepts: *Actor*, *Project*, *Activity/Component*, and *Document*.

- **Actor:** An *Actor* refers to an individual or a corporative actor both being created for business ventures.
- **Project:** The concept *Project* refers to the notion of a finite set of temporal and factually logical related dynamic or static elements (e.g., activity, components) being closed in their content and meeting a business purpose.
- **Activity/Component:** The concepts *Activity* and *Component* respectively correspond to dynamic and static elements of projects. An activity or the construction of a component consumes resources, has a start time and end time and further characteristics related to other concepts.
- **Document:** The concept *Document* is the top-level concept of one or more document hierarchies that build document types based on different discriminators. Each document type can be characterized by a finite set of content components and assigned values providing a fragmented documentation of *Actors*, *Projects*, and *Activities/Components*.

4.3 Multiagent Layer

Adoption of the BDI Architecture Model. The multi-agent system is designed and implemented according to the *belief-desire-intention* (BDI) architecture model. A BDI architecture consists of (1) concepts of beliefs, representing information about the agent’s current environment, (2) desires, representing the agent’s goals, and (3) intentions, representing the agent’s current focus, that leads to concrete actions. The decision process, i.e., which action to perform in

order to achieve the goals, is based on the philosophical approach of practical reasoning [17]. Practical reasoning consists of two activities, deliberation and means-end reasoning. Deliberation stands for selecting and prioritizing multiple, potentially conflicting goals that the agent aims to achieve. Means-end reasoning denotes the process of inferring how these goals can be achieved [18]. We use the practical reasoning mechanism as a means for both representing and enforcing rules over documents. Thus, we represent the relevant document information as beliefs and rules as goal conditions that have to be true in order to achieve a goal. The code snippet below shows an example of a rule. If the heat conductance of the offered product (document: specification) is lower than as agreed in the contract, then the goal of maintaining a sufficient heat conductance is not fulfilled. The agent will add an assertion that defines a relationship between the specification and contract (type: product change). In addition, the agent will conclude that this product change has an effect on other construction elements (here: window); this is due to the underlying domain ontology, which defines concepts and roles for heat requirements calculation.

Rule example

```
<performgoal name="heat_conductance" exclude="never">
  <contextcondition>
    $beliefbase.insulationValue_contract <=
      $beliefbase.insulationValue_specification
  </contextcondition>
  <dropcondition>
    $beliefbase.insulationValue_specification >
      $beliefbase.insulationValue_contract
  </dropcondition>
</performgoal>
```

ManagementAgent. The ManagementAgent has control functions and serves primarily as a source of new document agents. Two main functions are assigned to the ManagementAgent: (1) initialization of the multi-agent system and (2) monitoring of the semantic repository for changes, i.e., new documents or facts. At the start of the runtime of the multi-agent system, the only agent is the ManagementAgent. The initialization involves the bundling of existing documents to document groups and the creation of one DocumentAgent for each document group. During runtime, the agents monitor changes in the repository. At the arrival of a new document, the ManagementAgent determines whether the document can be assigned to an existing document group (agent) or a new document group must be created, and thus a new agent initialized.

DocumentAgent. The DocumentAgent represents a document group. Representing *each* document by a dedicated agent would inhibit the scalability of the agent system due to inter-agent communication effort and the need for resources, i.e., memory. Relevant document information is loaded into each agent's local knowledge base. The agent's basic task is to monitor the documents and

document groups and their direct relationships to other documents, document groups, and indirect relationships due to relationships to other elements (such as project, activity, component, actor). For this purpose, these agents maintain a set of predefined rules and can actively react in case of violation of integrity. The analysis of documents and associated annotations involves two steps: (1) an agent-internal analysis of the document group and (2) a multi-agent, cooperative analysis of various potentially dependent document groups. If a rule is activated, a notification about a detected relationship is sent to the repository, accompanied with a recommendation for actions to be taken by an end-user. The presentation and messaging is subject of the GUI layer, which is not in the scope of this paper.

5 Evaluation

This section provides a preliminary evaluation of the proposed architecture by presenting a prototype implementation, conducting a set of experiments, and reporting its results.

5.1 Experimental Setup

The example scenario chosen originates from civil engineering and respective documents that emerge during the planning and construction phase. This domain is characterized by a high degree of division of labor, thus multiple actors providing services - being documented in construction documents - to accomplish the construction project's mission. Therefore, documents are created, processed, and used by a variety of roles. A construction project is divided into several crafts (like heating, storefront, etc.), each being documented by a typical chain of documents, ranging from planning to auditing. All these characteristics contribute to a high number and variety of relationships between documents, but these relationships are only partially represented in DMS.

We define the set of document types as: $DT = \{requestforproposal, offer, order, acceptancecertificate, invoice\}$.

The content components of interest are date (e.g., agreed construction start and end) and amount (e.g., agreed and charged prices). This information is extracted by JAPE rules from incoming documents. Rules also exist for mapping documents to the correct document group. For the purpose of evaluation, we construct an experimental document set, which contains in each document group exactly one deviation, i.e., related to date or amount. For example, the due date of a craft being violated. This deviation is in our terms a time-related deviation relationship between two documents (here: order and acceptance certificate). The OBIE component applies 10 rules for extracting information from the document's header (e.g., actor, address), body (e.g., subject line, invoice items, due dates), and determining the right craft. The document agents maintain 5 more complex rules concerned with deviations. The extraction rules and the document basis are kept rather concise to ensure a high recall and precision. Note that extraction optimization is not in the focus of this work.

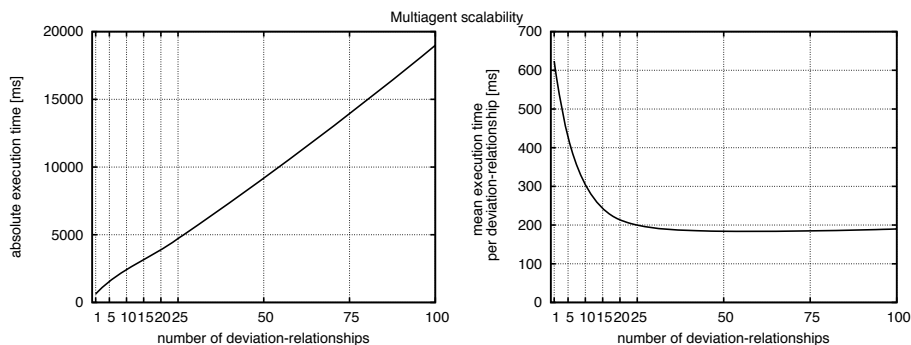


Fig. 3. Execution time and mean execution time per deviation with varying number of deviation-relationships

We conduct nine experiments with varying the number of deviation relationships, i.e., 1, 5, 10, 15, 20, 25, 50, 75, and 100. Since each group contains 5 documents, the total number of documents ranges from 5 to 500.

5.2 Implementation and Results

The proposed architectural extension has been implemented in a software prototype, which is used for conducting the evaluation experiments. We use Sesame¹ and BigOWLIM² for setting up the semantic repository. The IE component as well as the preprocessor is based on GATE³. The multiagent layer is based on the JADEX BDI Agent System. Each architectural layer runs on a separate machine: The machine for the semantic layer is actually a virtual machine (Xenserver) with CentOS 5.5 as operating system, which runs on 3 CPUs (2.133 GHz, 3 GB RAM). The multiagent layer was installed on a Mac system, with MacOSX 10.6, 2 CPUs 2.66 GHz, and 4GB RAM.

The experimental results are shown in Fig. 3. The execution time required for processing all documents is made up of the time for initializing the multi-agent system (e.g., generating respective agents) and the actual time for applying rules over documents. However, the initialization time is negligible for more than 20 deviation relationships. We observe a mean execution time of about 0.18 seconds per document. The results suggest an almost linear complexity.

6 Conclusions

This paper proposed a novel, two-tier architectural extension of DMS to better allow for identifying relationships between documents. Its key idea originates from combining a semantic document representation with cooperative software

¹ <http://www.openrdf.org/>

² <http://www.ontotext.com/owlim/>

³ <http://gate.ac.uk/>

agents that specifically cater for sets of documents instead of single documents. This architectural approach was implemented for the purpose of evaluation. It shows evidence of both the architecture's applicability (i.e., identification of relationships) and scalability (with first indications of almost linear complexity).

The current research is limited in a sense that it has not yet been tested for large-scale document sets that can be found in industrial applications. Such applications often comprise tens of thousands of documents. For this purpose, we conduct our research in a cooperative project, which involves a software company that provides a DMS to be extended as proposed. We implemented an interface for retrieving real-world documents and are currently in the process of setting up larger experiments.

Ontologies are an essential cornerstone for semantic document management. Whereas the general idea of separating the semantic representation and identification of document relationships (OIBE/JAPE and cooperative agents/JADEX) has been identified as feasible, we still need to explore the dividing line between the two layers in more detail to come up with a clear discriminator or at least a set of guidelines for potential users. We acknowledge that the codification of domain-specific rules that indicate relationships as well as integrity constraints that trigger notifications is also a knowledge-intensive task. Our future work will thus address the range of relationships and rules as well as alternative formal languages for their specification.

Acknowledgements. The work presented in this paper was partly funded by the German Federal Ministry of Education and Research under the project ProBauDok (BMBF 01IS09023). We wish to thank Dirk Hanselmann (CTO Balzuweit GmbH, Stuttgart) for providing the storage layer.

References

1. Uren, V., Cimiano, P.: Semantic annotation for knowledge management: Requirements and a survey of the state of the art. *Web Semantics: Science, Services and Agents on the World Wide Web* 4(1), 14–28 (2006)
2. Lanfranchi, V., Ciravegna, F., Petrelli, D.: Semantic web-based document: Editing and browsing in aktiveDoc. In: Gómez-Pérez, A., Euzenat, J. (eds.) *ESWC 2005*. LNCS, vol. 3532, pp. 623–632. Springer, Heidelberg (2005)
3. Eriksson, H.: An annotation tool for semantic documents. In: Franconi, E., Kifer, M., May, W. (eds.) *ESWC 2007*. LNCS, vol. 4519, pp. 759–768. Springer, Heidelberg (2007)
4. Nesic, S.: Semantic document model to enhance data and knowledge interoperability. In: Sharda, R., Voß, S., Devedæic, V., Gaëvic, D. (eds.) *Web 2.0 & Semantic Web*. *Annals of Information Systems*, vol. 6, pp. 135–160. Springer, US (2009)
5. Roberto, V., Mea, V.D., Gaspero, L.D., Conti, A.: MANTHA: Agent-Based Management of Hypermedia Documents. In: *Proceedings of the 1999 IEEE International Conference on Multimedia Computing and Systems*, pp. 814–818 (1999)
6. Ginsburg, M.: An agent framework for intranet document management. *Autonomous Agents and Multi-Agent Systems* 2, 271–286 (1999)

7. Reed, J.W., Potok, T.E., Patton, R.M.: A multi-agent system for distributed cluster analysis. *IEE Seminar Digests* 2004(916), 152–155 (2004)
8. Sprague, R.H.: Electronic document management: Challenges and opportunities for information systems managers. *MIS Quarterly* 19(1), 29–49 (1995)
9. Glushko, R., McGrath, T.: Document engineering for e-business. In: *Proceedings of the 2002 ACM Symposium on Document Engineering*, pp. 42–48. McLean, Virginia (2002)
10. Sarawagi, S.: Information Extraction. *FoT Databases* 1(3), 261–377 (2008)
11. Wimalasuriya, D.C.: Ontology-based information extraction: An introduction and a survey of current approaches. *Journal of Information Science* 36(3), 306–323 (2010)
12. Cunningham, H., Maynard, D., Tablan, V.: Jape: a java annotation patterns engine. Technical report, University of Sheffield, Department of Computer Science (2000)
13. Studer, R., Benjamins, R., Fensel, D.: Knowledge engineering: Principles and methods. *Data and Knowledge Engineering* 25(1-2), 161–197 (1998)
14. Horrocks, I., Kutz, Sattler, U.: The even more irresistible sroiq. In: *Proceedings of the Tenth International Conference on Principles of Knowledge Representation and Reasoning* (2006)
15. Fox, M.S.: The tove project: A common-sense model of the enterprise. In: Belli, F., Radermacher, F.J. (eds.) *IEA/AIE 1992*. LNCS, vol. 604, pp. 25–34. Springer, Heidelberg (1992)
16. Uschold, M., King, M., Moralee, S., Zorgios, Y.: The enterprise ontology. *Knowledge Engineering Review* 13, 31–89 (1998)
17. Bratman, M.: *Intentions, Plans, and Practical Reason*. Harvard University Press, Boston (1987)
18. Wooldridge, M.J.: Intelligent agents. In: Weiß, G. (ed.) *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, pp. 27–77. MIT Press, Cambridge (1999)

Deploying an Agent Platform to Automate the IT Infrastructure Auditing Process

Ana-Maria Ghiran, Gheorghe Cosmin Silaghi, and Nicolae Tomai

Babeş-Bolyai University, Dept. of Business Information Systems,
Str. Theodor Mihali 58-60, 400591, Cluj-Napoca, Romania
{Anamaria.Ghiran,Gheorghe.Silaghi,Nicolae.Tomai}@econ.ubbcluj.ro

Abstract. IT audit is employed in business organizations to demonstrate they hold the control for the correct and efficient functioning of their IT infrastructure. It is slowly moving from a completely practitioners' concern into a research domain. There is a need for identifying new methods that could facilitate an objective, real time and cost-effective assurance. This paper proposes a method to automate the IT audit process. Our approach is based on ontologies for formalizing the vast audit knowledge and on intelligent agents for real-time audit and risk assessment.

Keywords: IT audit, agent systems, ontologies, rule-based reasoning, network management services.

1 Introduction

Most organizations are dependent on technology, and especially on the risks associated with it. This implies that companies not only need to have control over the IT infrastructure but also to prove it, often through *an audit report*. In some countries, this is a strong recommendation in achieving IT Governance, part of Corporate Governance or it is even a legal requirement for public companies, like the Sarbanes Oxley Act [1] or the Health Insurance Portability and Accountability Act [2]. Similar legislation has been adopted in many countries world wide. Companies need to develop an internal IT continuous audit in order to comply with such regulations. Managing the IT has become a burden and soon was clear that we needed "technology to manage technology" [3]. IT audit refers to the assurance regarding how IT is managed and could be or not continuous or can be done with or without technology. This paper focuses on eliminating the manual procedures in an IT audit and incorporating some autonomic tools, not only to improve the process efficiency, but also the accuracy and the range of the audit applicability.

Using traditional (manual) techniques in IT audit makes process reusability difficult and real time assurance impossible. We propose to overcome the aforementioned limitations in the following ways: (i) by delegating auditing tasks to intelligent agents. We strive to improve the efficiency and to minimize the inconsistency or inaccuracy which may occur due to subjective human judgments or

potential errors. Using intelligent agents we can deliver almost real time assurance, (ii) by using ontology based semantic description of auditor's knowledge. As expert knowledge is captured and formalized explicitly, process reusability is enhanced and (iii) by applying automated reasoning techniques. New knowledge can be generated through deduction.

The main contribution of this paper is to show that a multi agent system can be successfully applied in automating the IT audit. We will propose a generic agent-based IT audit process and will show how an agent platform can be deployed to accomplish the main functions of the IT audit process. We will detail on the essential characteristics of every important item of the auditing agent-based system.

This paper evolves as follows. Section 2 presents some background and related work, section 3 presents an overall view of the agent-based IT audit process, section 4 presents the deployment of a JADE agent system for the IT audit process described in section 3, while section 5 concludes the paper.

2 Background and Related Work

In this section we shortly introduce the concepts used in this paper. First, we develop about the IT audit and its scope. Next, we will introduce concepts like agent platforms, IT infrastructure management services and ontologies.

2.1 IT Audit

IT audit sometimes is identified with information systems audit or is formally known as electronic data processing (EDP) audits. Progressively, the audit span has enlarged and now includes evaluation of data, applications and infrastructure. The auditor have to estimate the controls implemented by managers and to issue a report. While managers' role is *to ensure*, the auditors are supposed *to assure* [4]. IT auditors started to use different computer assisted audit tools and techniques (CAATs) specialized for each sort of the audit. Center for Internet Security (CIS) [1] describes a variety of IT audit tools that can be used for benchmarking the operating systems (CIS-CAT Configuration Audit Tool), network systems (RAT Router Audit Tool), web servers (Apache Benchmark Tool), etc. For vulnerability assessments auditors can use applications like Nessus [2] or Retina [3]. In vulnerability assessments usually the tests regard software flaws or misconfigurations. MITRE Corporation [4] is offering up-to-date descriptions of known software vulnerabilities - Common Vulnerabilities and Exposures (CVE) and OS/Applications misconfiguration - Common Configuration Enumeration (CCE).

NIST produces a list of security settings for operating systems like Windows XP and Vista known as Federal Desktop Core Configuration (FDCC)

¹ <http://www.cisecurity.org>

² <http://www.nessus.org>

³ <http://www.eeye.com>

⁴ <http://www.mitre.org/>

that have been adopted as *common security configurations* which should be enforced in order to provide a basic security level. One wishing to prove compliance with FDCC, can employ tools based on Security Content Automation Protocol (SCAP)⁵ that enables the automation of the compliance testing.

All these approaches use various standards to describe settings, risks and vulnerabilities in the system and the software tools are strongly dependent on these standards. Usually, an auditor uses such a variety of auditing tools and bases her conclusions on how well she can deal with those tools. With our automated agent-based approach we intend to offer an auditing system which requires as little intervention as possible from the human auditor. The auditor only needs to encode her knowledge by describing various ontologies and the intelligent agents will perform most of the tasks on behalf of the human auditor.

2.2 Agent Platforms

An agent is a software object that can receive tasks and, in addressing them, can autonomously initiate, receive, reject or execute messages towards to or from other agents [5]. An agent platform is a software environment that contains agents and enables them to function in order to achieve their tasks. The agent platform must deliver the following functionalities: (i) agent management: creation, destruction, migration of agents, (ii) agent communication, (iii) agent surveillance, (iv) error notifications, (v) security mechanisms. Currently, several agent platforms are available, like JADE [6], JACK [7] or Aglet [8].

For the scope of this paper we selected JADE, mainly because of its various advantages: compatibility of agents with the FIPA standards (which enables the agent's portability on different platforms), the existence of recent downloadable versions of the platform and the availability of documentation. Very important is the fact that JADE base language is Java, which enables us to enrich the created agents with different capabilities in order to access different CAATs, management services or ontology repositories and reasoning tools.

2.3 IT Infrastructure Management Services

When auditing the IT infrastructure of a company, a management service and protocol is required in order to acquire various information from the managed devices. Examples of such management services are the Simple Network Management Protocol (SNMP) [9], Common Management Information Protocol (CMIP) [10] or Web Services Distributed Management. (WSDM) [11]

SNMP offers a simple method to access information about a certain managed device or even to modify it, implementing a reduced number of commands. Each managed device has incorporated as part of its manufacturing process, a structure called Management Information Base (MIB) that retains the name of the variable (object identifier) and its value, which can be interrogated with SNMP. CMIP is a much complex information protocol, being able to access managed

⁵ <http://scap.nist.gov/>

resources as objects and to obtain the relationships between them. WSDM is promoted by OASIS to allow web service based information management.

As the scope of this paper does not reside on device information management and we need only a simple way to extract auditing information from managed devices, we will use SNMP, because of its simplicity and its compliance with new devices as well as legacy systems.

2.4 Ontologies and Reasoning Capabilities

Ontologies are very effective in defining, sharing and reusing knowledge from a specific domain. Ontologies can make use of reasoning tools like Pellet⁶ or Racer⁷ that allow ontology verification for inconsistencies, classification of concepts and type inference for certain instances. Sometimes the power of ontologies only and ontology reasoners are not enough to express aspects like the value of a concept which is known only at run time (e.g. a variable value) or inferring new concepts from previous ontology information. For this, ontologies have been enhanced with *if-then rules*, and the dominant language in expressing them is the Semantic Web Rule Language (SWRL). Ontologies have already proved their applicability in representing knowledge in the information security domain or risk management field and are continuously increasing in use. Tsoumas et al. [12] developed an ontology to perform management of network security. It models assets and all objects that are manageable extending the Common Information Model (CIM) with concepts related to risk assessment. Fenz et al. [13] proposed an ontology for risk management domain, incorporating security concepts like assets, vulnerabilities, threats, controls. Their ontology is inspired by the security relationship model described by NIST [14].

Ontologies are described in a specific language like OWL. In order that Java-based agents to interact with knowledge concepts from ontology, we can employ one of the following APIs: Jena Ontology API, OWL API or Protege OWL API. All these are based on Java and provide mechanisms to load and save ontologies, create OWL constructs, axioms and run inferences. The first one is meant to be more flexible as it covers all of RDF and OWL. However, it does not support SWRL. The other two are addressing mainly OWL constructs and also allow working with SWRL rules. As we are also creating our ontology in Protege, and since the Protege OWL API is optimized for the implementation of graphical user interfaces, we choose to use it as the Java library with our agents.

Ontologies are crucial in order to automate the auditing process. Intelligent agents deployed for various small auditing tasks will use the ontologies and the reasoning capabilities associated with them to infer auditing results. Ontologies encode the expert knowledge. Thus, to keep the automated auditing system up-to-date, one only needs to update the expert ontologies, from time to time.

⁶ <http://clarkparsia.com/pellet/>

⁷ <http://www.racer-systems.com/>

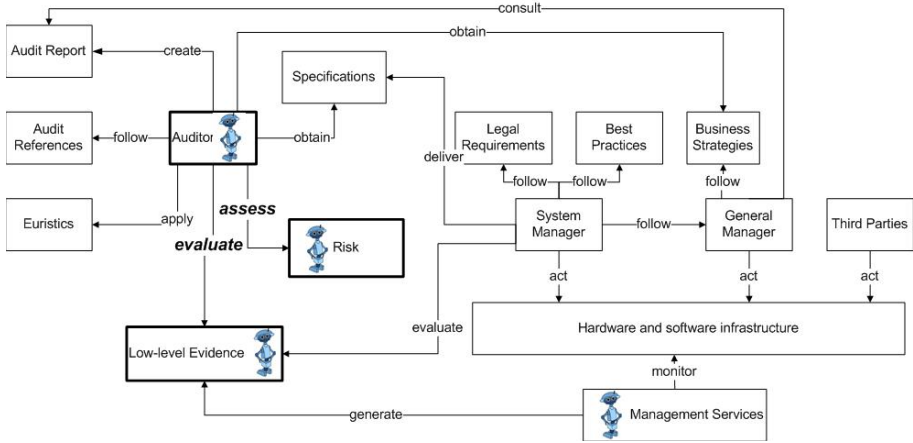


Fig. 1. The IT Process model

3 The IT Audit Model

In this section we develop a generic IT audit model for a company, with the help of intelligent agents. The overall view of this process is presented in figure 1.

System Managers are in charge to supervise the hardware and software infrastructure following directives from the **General Manager**, best practices or other legal requirements. The **General Manager** at her turn, is following the Business Strategies. System Managers, other managers, employees or other third parties are all acting on the IT Hardware and software infrastructure producing different changes. These changes are observed by different **Management Services** which record evidences.

An **IT Auditor** obtains the specifications she needs in order to assess the IT infrastructure. These specifications give the high level requirements about how the IT functioning should be implemented and what its purpose should be in the organization. Audit references give models of the ideal situation and also guidelines in conducting the audit. The auditor follows these references (standards and good practices) to apply her heuristics and, in the end, issues an audit report.

The actual audit process consists in two steps: *assessing* the risk level and *evaluating* the low level evidences, which are also checked by System Managers through network management applications. These steps and roles are highlighted in the fig. 1 with embossed boxes and printings. For these roles (assessing and evaluating), we devise intelligent agents to automate the process. These agents are managed inside the adopted agent platform (in our case JADE) and they run and perform their duties autonomously, on behalf of their users. The layered architecture of the agent-based auditing system is presented in figure 2.

The main user of this system is the IT auditor. The IT auditor has the **Auditor Agent** on its behalf, and this agent presents the auditor a configuration module and a report module. In the first step, the IT auditor uses the configuration

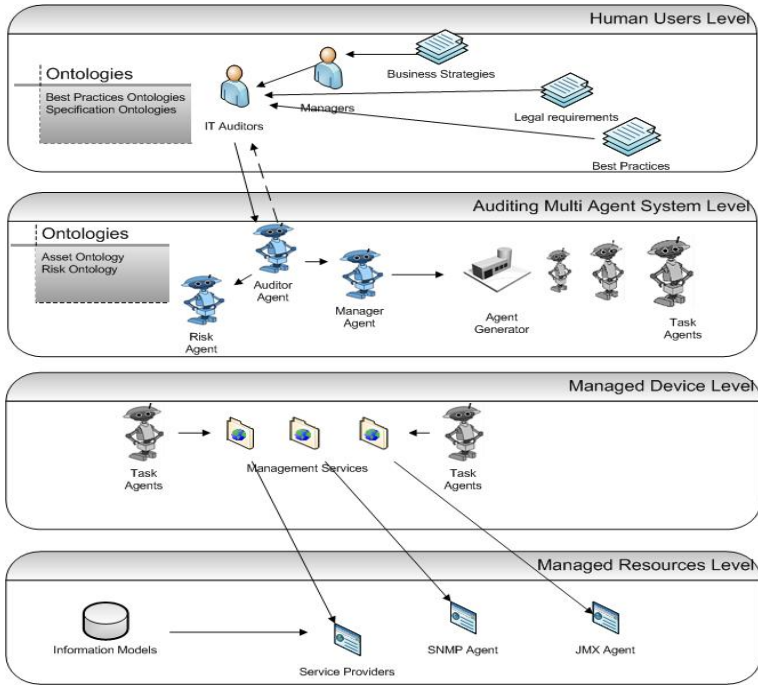


Fig. 2. The architecture of the agent-based auditing infrastructure

module for establishing the criteria and the domain of the audit applicability. In choosing the criteria for the audit process, the IT auditor can select a specific framework against which the audit will be performed or stipulates the corporate governance policies (the specifications). The next step consists in establishing the granularity view of the audit process: from general to specialized, or just a few elements from the IT infrastructure. Considering the elements selected to participate in the audit process, there will be a refinement of the control objectives from the first step that are applicable.

After deciding over the configuration for the IT audit, the auditor transmits a request to the **Auditor Agent** and waits for the report. The report module presents the report of the audit process together with possible suggestions. Suggestions can be regarding the following aspects: improving the controls implemented for risk reductions, treatments of risks by avoidance, transfer or just risk acceptance in some cases when the cost of insurance against the risk would be greater over time than the total losses.

The **Auditor Agent** receives the configurations for the audit process and translates them into specific requirements. It delegates to the **Risk Agent** the task of assessing the risk level and to the **Manager Agent** the task of evaluating the current state of the IT infrastructure. As soon as it receives responses regarding the risk level and the proofs of controls implemented, the **Auditor Agent** is able to match them and issue the report towards the human auditor.

The **Risk Agent** is in charge of assessing the risk level, i.e. to categorize risks as high, medium or low. Generally the risk is the potential lost and the probability of its occurrence. Complying with the NIST recommendations [15], the **Risk Agent** performs the following steps:

- identification and evaluation of assets, which includes all physical elements, applications and data. This information will be extracted from **Asset Ontology** which is continuously populated by the agents who interact with management services. The evaluation of physical goods should be done at the replacement costs. For applications and data, the impact that would result if the information was unavailable, destroyed, disclosed or modified must be considered. User interaction is required in order to establish the values which will be initially stored in the **Asset Ontology** for each asset.
- evaluation of threats and vulnerabilities for each of the previous identified assets; probability of their occurrences is calculated. A separate ontology has been created to describe the threats, vulnerabilities, countermeasures. The ontology used for the current prototype is just for demonstration purposes and is not intended to be comprehensive for IT risk assessment.
- recommendation of countermeasures

The **Manager Agent** will receive the request from an **Auditor Agent** and will transform it into even more specific tasks. It will consult the **Directory Facilitator**, a look up service offered by the platform agents, in order to find out agents already on the managed devices that could accomplish the specific tasks. In case no such agents are identified, the **Manager Agent** will use the **Agent Generator** in order to instantiate the **Task Agent** for the certain job. The **Agent Generator** searches among the agent templates existing in its repository for the one corresponding to the specific tasks. After the **Task Agents** are created, they migrate on the managed device where they interact with the management services. Some of the tasks might need a certain period of time for computing indicators regarding the current state of the assets. As soon as they decide on the results, the **Manager Agent** is informed, which will validate them against the answers acquired from the **Risk Agent**. The final conclusion is presented in a report and also written in a log file.

The **Specifications Ontology** formalizes different control frameworks and is used in order to obtain the applicable control objectives that need to be verified.

The **Asset Ontology** classifies the IT resources starting from the recommendation used in Cobit Framework [16]. Fig. 3 presents the foundation of the **Asset Ontology**, which can be further extended by integrating and describing novel devices.

The **Assets** concept integrates concepts like: **Infrastructure**, **Applications**, **Information**, and **People**, which are further detailed into other classes. For instance **Infrastructure** is composed of **Network_Service**, **Network**, **Devices**

⁸ Cobit contains guidelines for IT audit professionals, managers or other IT users for choosing the best control objectives to follow in accordance to their organization

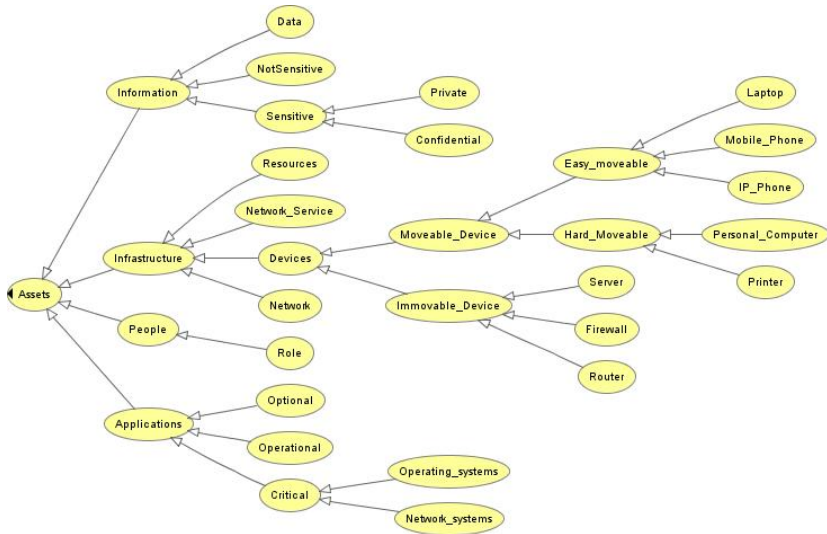


Fig. 3. The Asset Ontology

and **Resources**. **Network_Service** describes services identified to be offered by the network like: Internet service, mail service, file transfer, VPN connections etc. **Network** identifies the composition of the organizational network into subnets. **Devices** are divided into Moveable devices and Immovable.

The **Risk Ontology** (fig. 4) separates concepts based on the likelihood of occurrence and its impact. The **Likelihood** provides an inside regarding the **Threats**, **Vulnerabilities** and **Countermeasures** for each asset. The **Threats** are classified as **Natural** or **Human Threats**, depending on the **hasThreatSource** property. Further, **Natural** threats are classified as **Random** or **Predictible**; **Human Threats** as **Accidental** or **Intentional**. **Vulnerabilities** are defined as **Software**, **Hardware** and **Management Vulnerabilities**, and are populated from the National Vulnerability Database (NVD)⁹. Extremely important for our ontology are definitions of current software vulnerabilities given by the CVE entries, nomenclatures of systems configuration issues like CCE and the scores for each known vulnerability instance given by the Common Vulnerability Scoring System (CVSS)¹⁰. **Countermeasures** correspond to controls and they are in close relationship with vulnerabilities as the lack of appropriate controls indicates the presence of vulnerabilities. **Controls** are sorts of assets; therefore various assets' instances are found under the **Controls** concepts. The main difference between a **Control** and an **Asset** is the following: **Controls** are implemented to minimize the likelihood of vulnerability exploitation by a threat. Thus, we implemented **Controls** as separate **Assets**. The **Impact** represents the changes

⁹ NVD is a comprehensive vulnerability database integrating all USA publicly available vulnerability resources <http://nvd.nist.gov/>

¹⁰ <http://www.first.org/cvss/>

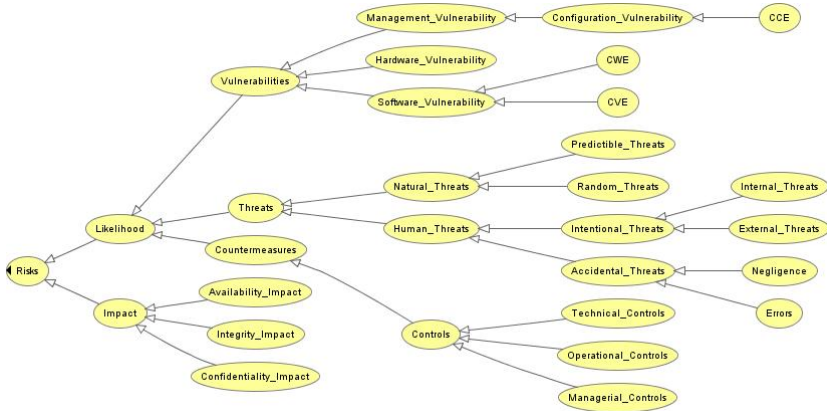


Fig. 4. The Risk Ontology

in the Confidentiality, Integrity and Availability of resources, when a Threat exploits a Vulnerability.

4 Deploying JADE Agents for IT Audit

In this section we show how to deploy JADE agents to accomplish the IT auditing functionality described in the previous section. Figure 5 presents the sequence diagram of the JADE agent-based system, comprising all message exchange between various agents of the system.

Launching in execution the infrastructure implies the start of the agent platform JADE, distributed on multiple hosts. Every host will hold a container where agents can execute their tasks. The Main Container of the JADE platform will reside on the host (named Auditing Station in fig. 5) with the user interface used to communicate with the human IT auditor. The Main Container has 2 special agents that start automatically namely AMS (Agent Management System) and DF(Directory Facilitator), which are part of the JADE platform. These are required in order to deliver the JADE platform’s main features like agent management or agent discovery. AMS is not depicted in fig. 5 because its role is only to initialize the JADE platform and launch the execution of the system. Besides these special agents, several specific agents are constructed: an GUI interface agent UI GUI, the Auditor Agent, the Risk Agent, the Manager Agent and the Agent Generator described in the previous section. As part of the initialization of the system, the Manager Agent deploys an Initialization Agent to all containers that already joined the platform. Also, every time a new instance of the container joins the platform, the Manager Agent delivers an Initialization Agent to it. This agent’s task is mainly to collect data and to instantiate some of the ontology concepts in the Asset Ontology. These concepts regard the network topology and structure, applications discovery and sensitive data identification.

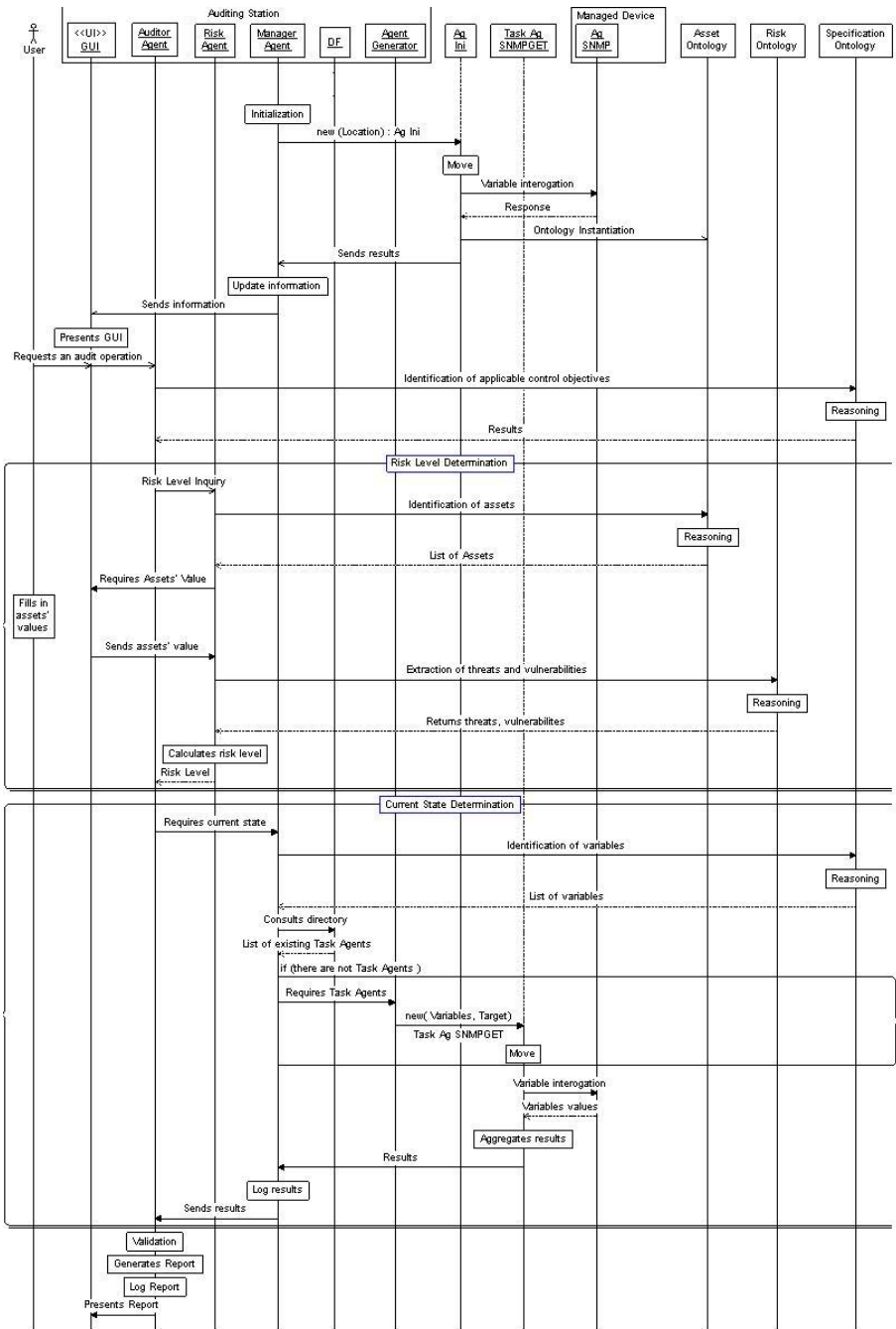


Fig. 5. The sequence diagram of the JADE agent system

Currently, JADE agents are not initially equipped with reasoning capabilities over rules. However, as advised by the JADE documentation, it is possible to implement an agent behaviour integrating reasoning capabilities, like Jess engine. Therefore, the **Risk Agent** and the **Auditor Agent** integrate in their behavior the Jess reasoning capability, in order to reason over the Risk and Asset ontologies.

As presented in section 3, there are 2 important use cases for the human auditor: to determine the risk level associated with an asset and to determine the current state of the asset. The process of determining the risk level is started by the **Auditor Agent** by an inquiry sent to the **Risk Agent**. The **Risk Agent** employs the Jess engine to perform reasoning over the **Asset Ontology** in order to extract the list of assets. This list of assets is presented to the user, via the UI GUI who is interrogated for exemplification of their value. The user can start the interrogation of the risk associated with a given asset from the list, which, in turn, is solved by the **Risk Agent** with the help of the Jess reasoning over the **Risk Ontology**. Finally, the **Risk Agent** computes the risk level according with its encoded rules and presents it to the human auditor.

The second important process, determining the current state, starts with identifying the state variables of a given device, by interrogating the **Specification Ontology**. This operation is performed by the **Auditor Agent**, who, using the Jess engine, extracts the list of the variables of interest. This list of variables is returned to the **Manager Agent**, which identifies whether **Task Agents** exist for these variables (from previous system executions), using the DF. If a **Task Agent** is available, this agent is deployed on the Managed Device. If not, a new **Task Agent** is instantiated and deployed on the device. The **Task Agent** inspects the device using the Management Service (in our case SNMP - as presented in section 2.3), aggregates the results, log them and return them to the **Auditor Agent**. The **Auditor Agent** perform results validation, generates the auditing report and presents it to the human auditor on the GUI interface.

Here we note the importance of the Management Service (MS). All devices of the audited system should accomplish with the employed MS, because, if a new device join in and it does not accomplish the supported MS, the system will not be able to create a proper **Task Agent** for that device. Thus, it will be impossible to audit that device. This property is crucial for the extensibility and the auto-adaptability of the auditing agent-based system. But luckily, as presented in subsection 2.3, there are enough management services available to cover all sorts of infrastructure devices, at various level of expressivity.

5 Conclusion

This work contributes with a method for transforming the IT audit process from the manual-based approaches to a much automated procedure. We introduced a generic model of the IT audit process and layered architecture of an agent-based system for it. Further, we showed how a JADE agent platform can be deployed to automate the described IT auditing.

Further research activities would detail the description of the audit knowledge formalized as ontologies and the possibilities of extending it. Also, integration with other (web-based) management services is needed, as their adoption will soon become prevalent. Alongside with management services, an equal importance should be given to the assimilation of publicly available sources of threats and vulnerabilities or best practices.

Acknowledgments. This work is supported by the Romanian Authority for Scientific Research under project IDEL1598.

References

1. Sarbanes-Oxley Act of 2002. Public Law 107-204. U.S. Government Printing Office (2002),
<http://www.gpo.gov/fdsys/pkg/PLAW-107publ204/content-detail.html>
2. The Health Insurance Portability and Accountability Act. U.S. Government Printing Office (1996), <http://www.hipaa.org>
3. An Architectural Blueprint for Autonomic Computing. IBM Autonomic Computing White Paper (June 2006)
4. Gallegos, F., Senft, S.: Information Technology Control and Audit, 3rd edn. Auerbach Publications (2008)
5. Wooldridge, M.: An Introduction to MultiAgent Systems, 2nd edn. John Wiley and Sons, Chichester (2009)
6. Bellifemine, F.L., Caire, G., Greenwood, D.: Developing Multi-Agent Systems with JADE. Wiley, Chichester (2007)
7. Jack white paper: An agent infrastructure for providing the decision-making capability required for autonomous systems,
www.aosgrp.com/downloads/JACK_WhitePaper_UKAUS.pdf
8. Lange, D., Oshima, M.: Programming and Deploying Java Mobile Agents with Aglets. Addison-Wesley, Reading (1998)
9. Verma, D.C.: Principles of Computer Systems and Network Management. Springer, Heidelberg (2009)
10. Black, U.: Network Management Standards: SNMP, CMIP, TMN, MIBs, and Object Libraries. McGraw-Hill, New York (1994)
11. WSDM 1.1 OASIS Standard Specifications. OASIS Consortium (2006),
<http://www.oasis-open.org/committees/wsdm/>
12. Tsoumas, B., Gritzalis, D.: Towards an ontology-based security management. In: Proceedings of the 20th International Conference on Advanced Information Networking and Applications, AINA 2006, vol. 01, pp. 985–992. IEEE Computer Society, Los Alamitos (2006)
13. Fenz, S., Ekelhart, A.: Formalizing information security knowledge. In: Proceedings of the 4th International Symposium on Information, Computer, and Communications Security, ASIACCS 2009, pp. 183–194. ACM, New York (2009)
14. Special Publication 800-12: An Introduction to Computer Security - The NIST Handbook. NIST (1995),
<http://csrc.nist.gov/publications/nistpubs/800-12/800-12-html/index.html>
15. Stoneburner, G., Goguen, A., Feringa, A.: Risk Management Guide for Information Technology Systems. NIST SP 800-30
16. Cobit 4.1. IT Governance Institute (2010),
<http://www.isaca.org/Knowledge-Center/COBIT/Pages/Overview.aspx>

Modeling Support for Confidentiality and Integrity of Object Flows in Activity Models

Bernhard Hoisl^{1,2} and Mark Strembeck^{1,2}

¹ Institute for Information Systems and New Media,
Vienna University of Economics and Business (WU Vienna), Austria

² Secure Business Austria (SBA) Research Center, Austria
{bernhard.hoisl,mark.strembeck}@wu.ac.at

Abstract. While the demand for an integrated modeling support of business processes and corresponding security properties has been repeatedly identified in research and practice, standard modeling languages do not provide native language constructs to model process-related security properties. In this paper, we are especially concerned with confidentiality and integrity of object flows. In particular, we present an UML extension called SecureObjectFlows to model confidentiality and integrity of object flows in activity models. Moreover, we discuss the semantics of secure object flows with respect to control nodes and provide a formal definition of the corresponding semantics via the Object Constraint Language (OCL).

Keywords: Activity Models, Modeling Security Properties, Process Modeling, UML.

1 Introduction

IT systems must comply with certain laws and regulations, such as the Basel II Accord, the International Financial Reporting Standards (IFRS), or the Sarbanes-Oxley Act (SOX). For example, adequate support for the definition and enforcement of process-related security policies is one important part of SOX compliance (see, e.g., [12]). Moreover, corresponding compliance requirements also arise from security recommendations and standards such as the NIST security handbook [3], the NIST recommended security controls [4], or the ISO 27000 standard family (formerly ISO 17799).

While the demand for an integrated modeling support of business processes and corresponding security properties has been repeatedly identified (see, e.g., [5,6]), different types of problems arise when modeling process-related security properties. First, contemporary modeling languages such as BPMN (Business Process Model and Notation, [7]) or UML activity models (Unified Modeling Language, [8]) do not provide native language constructs to model secure object flows. A second problem is that the language used for process modeling is often different from (or not integrated with) the system modeling language that is used to specify the corresponding software system. This, again, may

result in problems because different modeling languages provide different language abstractions that cannot easily be mapped to each other. In particular, such semantic gaps may involve significant efforts when conceptual models from different languages need to be integrated and mapped to a software platform (see, e.g., [9,10]).

However, a complete and correct mapping of process definitions and related security properties to the corresponding software system is essential in order to assure consistency between the modeling-level specifications on the one hand, and the software system that actually manages corresponding process instances and enforces the respective security properties on the other.

In this paper, we are concerned with the modeling of secure object flows in process models – in particular UML activity diagrams. UML is a de facto standard for software systems modeling. It provides a family of integrated modeling languages for the specification of the different aspects and perspectives that are relevant for a software system. Therefore, to demonstrate our approach, we chose to define an extension to the UML metamodel that allows to specify confidentiality and integrity properties of object flows in activity models. Activity models have a token semantics, and object tokens are passed along object flow edges (for details see [8]). Thus, to ensure the consistency of the corresponding activity models, it is especially important to thoroughly specify the semantics of secure object flows with respect to control nodes (such as fork, join, decision, and merge nodes). Therefore, we use the Object Constraint Language (OCL, [11]) to formally define the semantics of our extension. Corresponding software tools can enforce the OCL constraints on the modeling-level as well as in runtime models. Thereby, we can ensure the consistency of the extended activity models with the respective constraints.

The remainder of this paper is structured as follows. In Section 2 we present our UML extension for secure object flows in activity models. Subsequently, Section 3 discusses the semantics of secure object flows, with a special focus on the semantics arising from different types of control nodes. Next, Section 4 discusses related work, and Section 5 concludes the paper. ¹

2 UML Extension for Secure Object Flows

Thereby, *confidentiality* ensures that important/classified objects (such as business contracts, court records, or electronic patient records) which are used in a business process can only be read by designated subjects (see, e.g., [4,12]). *Integrity* ensures that important objects are in their original/intended state, and enable the straightforward detection of accidental or malicious changes (see, e.g., [3,13,14]).

To provide modeling support for confidentiality and integrity properties of object flows, we define a new package *SecureObjectFlows* as an extension to

¹ We provide an extended version of this paper on our Web page. In the extended version we re-inserted the text that we had to cut from the paper due to the page restrictions for the proceedings version.

the UML metamodel (see Fig. 1). In particular, we introduce *SecureNode*, *SecurePin*, *SecureDataStoreNode*, and *SecureActivityParameterNode* as new modeling elements. A secure object flow is defined as an object flow between two or more of the above mentioned secure object nodes. The *SecureNode* element is defined as an abstract node, and the *SecurePin*, *SecureDataStoreNode*, and *SecureActivityParameterNode* represent specialized secure nodes. In particular these three nodes inherit the properties from their corresponding parent object nodes as well as the security related properties from *SecureNode* (see Fig. 1).

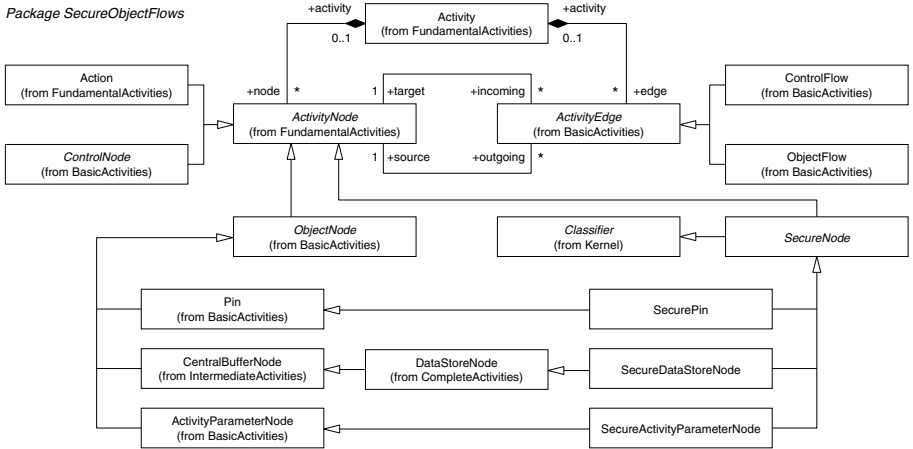


Fig. 1. UML metamodel extension for secure object flows

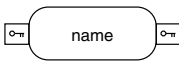
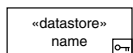
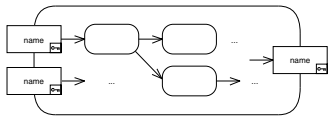
Below, we specify the attributes of the *SecureNode* elements defined via the metamodel extension. In addition, we use the OCL to formally specify the semantics of the *SecureObjectFlows* package. For the sake of readability, we decided to move the associated OCL constraints to Appendix A. However, these OCL constraints are a significant part of our UML extension, because they formally define the semantics of the new modeling elements. Therefore, each UML model that uses the *SecureObjectFlows* package must conform to these OCL constraints.

- *confidentialityAlgorithm* : Classifier [0..1]
References a classifier that provides methods to ensure confidentiality properties of the object tokens that are sent or received by a *SecureNode*, e.g. a class implementing DES (Data Encryption Standard) or AES (Advanced Encryption Standard) functionalities.
- *confidentialityKeyLength* : Integer [0..1]
Defines the key length of encryption method used, for example 256 bit.
- *confidentialityEnsured* : Boolean [0..1]
This Attribute is derived from the attributes *confidentialityAlgorithm* and *confidentialityKeyLength*. It evaluates to true if a *SecureNode* supports confidentiality-related security properties (see OCL Constraint 1).

- *integrityAlgorithm : Classifier [0..1]*
References a classifier that provides methods to ensure integrity properties of the object tokens that are sent or received by a SecureNode, e.g. a class implementing SHA-1 or SHA-384 (Secure Hash Algorithm) functionalities.
- *integrityEnsured : Boolean [0..1]*
This attribute is derived from the attribute *integrityAlgorithm*. It evaluates to true if a SecureNode supports integrity-related security properties (see OCL Constraint 2).

With respect to the attributes defined above, we specify that a secure object node either supports confidentiality properties, or integrity properties, or both (see OCL Constraint 3). Table 1 shows the graphical elements for SecureNodes.

Table 1. Notation of elements for modeling secure objects

Node Type	Notation	Explanation
<i>SecurePin</i> (attached to action)		A SecurePin attached to an action is shown as a UML Pin element that includes a key symbol.
<i>SecureDataStoreNode</i>		A SecureDataStoreNode is shown as a UML DataStoreNode element with a key symbol in the lower right corner surrounded by a small rectangle.
<i>Secure-Activity-Parameter-Node</i>		A SecureActivityParameterNode is shown as a UML ActivityParameterNode element with a key symbol in the lower right corner surrounded by a small rectangle.

3 Semantics of Secure Object Flows

The main element of an activity model is an activity. Actions define the tasks (steps) that are performed when executing the corresponding activity. Activity models have a token semantics, similar (but not equal) to petri nets (for details see 8). In general, two different types of tokens can travel in an activity model. Control tokens are passed along control flow edges and object tokens are passed along object flow edges. This means, each type of token is exclusively passed along edges of the corresponding edge type.

A decision node chooses between outgoing flows and, therefore, has one incoming and multiple outgoing edges. Decision nodes do not duplicate tokens. Therefore, each token arriving at a decision node can travel along exactly one outgoing edge. A merge node consolidates multiple incoming flows and thus has multiple incoming and one outgoing edge. However, merge nodes do not synchronize concurrent flows nor do they join incoming tokens. Thus, each token arriving at a merge node is offered to the outgoing edge. Both, decision and

merge nodes are represented by a diamond-shaped symbol respectively. A fork node splits a flow into multiple concurrent flows and thus has one incoming and multiple outgoing edges. Tokens arriving at a fork node are duplicated and passed along each edge that accepts the token. A join node synchronizes multiple flows and therefore has multiple incoming and one outgoing edge. A join node may join/combine incoming tokens (in contrast to merge nodes, see above). Both, fork and join nodes are represented via a thick line (for details see [8]).

To ensure the consistency of the corresponding activity models, it is especially important to thoroughly specify the semantics of secure object flows. Otherwise, a combination of ordinary object flows and secure object flows could result in inconsistencies. Therefore, Section 3.1 discusses the semantics of secure object nodes with respect to direct object flows, Section 3.2 discusses the semantics with respect to decision and merge nodes, and Section 3.3 with respect to fork and join nodes.

3.1 Semantics of Secure Object Nodes Regarding Direct Object Flows

We use the term *direct object flow* to refer to an object flow that directly connects object nodes without intermediate control nodes. Fig. 2 shows three example configurations of direct object flows involving SecureNodes. All statements and OCL constraints referenced below refer to SecureNode and therefore apply for each subtype of SecureNode (see Fig. 1).

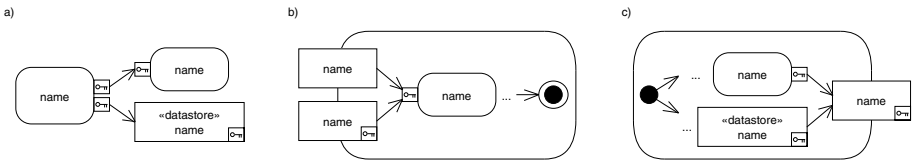


Fig. 2. Examples of direct object flows between secure nodes

Fig. 2a shows a configuration where two SecurePins attached to an action serve as data sources for two other secure object nodes. To ensure a secure object flow, we define that if an object node receives an object token from a SecureNode, the target node must also be a SecureNode (see OCL Constraint 4). Otherwise, a secure object flow could have a SecureNode as its source and an ordinary object node as its target – which would result in an inconsistency because ordinary object nodes cannot ensure the confidentiality or integrity of object tokens.

Because each subtype of SecureNode does also inherit the properties of the corresponding ordinary UML object node (see Fig. 1), it can process ordinary object tokens as well as secure object tokens. Fig. 2b shows a configuration where an ordinary ActivityParameterNode and a SecureActivityParameterNode serve as source nodes for a SecurePin. In such a configuration, the target node must be a SecureNode (see OCL Constraint 4) and the target node (here a SecurePin) must

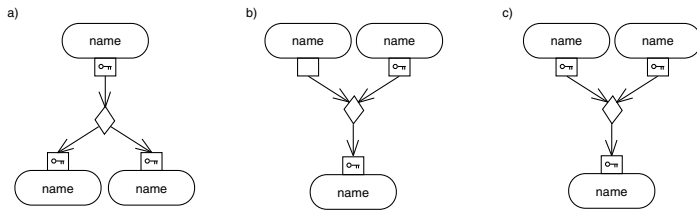


Fig. 3. Secure object flows with decision and merge nodes

support the same security properties as the corresponding secure source node (here a `SecureActivityParameterNode`). This requirement is formally specified via OCL Constraint 5.2. This constraint guarantees that the security properties of object tokens sent by a certain source node can be checked and ensured by the corresponding target node(s).

Fig. 2c shows a configuration where a `SecurePin` and a `SecureDataStoreNode` serve as source nodes for a `SecureActivityParameterNode`. Thus, according to OCL Constraint 4, the target node must also be a `SecureNode` (here it is a `SecureActivityParameterNode`) and it must support all security properties that are supported by the respective source nodes (see OCL Constraint 5). Moreover, we define that all source nodes must provide compatible security properties (see OCL Constraint 6). Otherwise, the source nodes could use, for example, different cryptographic algorithms or different key lengths – which could again result in inconsistencies and in a violation of OCL Constraint 5.

3.2 Semantics of Secure Object Flows Regarding Decision and Merge

Fig. 3 shows examples of the different configuration options of secure object flows that include decision or merge nodes. Fig. 3a shows a configuration where a decision node has an incoming secure object flow and presents the corresponding object tokens to multiple outgoing edges. As the source of the incoming object flow is a `SecureNode` (here it is a `SecurePin`) both target nodes must also be secured (see OCL Constraint 7). Otherwise, a secure object flow could have a `SecureNode` as its source and an ordinary object node as its target – which would result in an inconsistency because ordinary object nodes cannot ensure confidentiality or integrity of object tokens. Furthermore, target nodes of a secure object flow must support the same security properties as the respective source node (see OCL Constraint 8). This constraint ensures that security properties cannot be lost when traversing a decision node and that the target node(s) are able to check and ensure the corresponding security properties.

² Note that the OCL invariants from Appendix A complement each other.

³ For the sake of simplicity, Fig. 3 as well as Fig. 4 show only two incoming/outgoing flows for the respective control nodes. However, the corresponding OCL constraints apply for an arbitrary number of incoming/outgoing edges, of course.

Fig. 3b shows a configuration where a merge node brings together alternate flows – one of which is a secure object flow. For such a configuration, we define that if a merge node receives at least one secure object flow, the target node of this merge node must also be a SecureNode (see OCL Constraint 9). This constraint guarantees that each secure object token passing a merge node can be checked and processed by the corresponding target node.

Fig. 3c shows a configuration where a merge node brings together alternate secure object flows. According to OCL Constraint 9 the target must be a SecureNode. Furthermore, we define that all source nodes must provide compatible security properties (see OCL Constraint 10). In addition, the target node must support all security properties of the respective source nodes (OCL Constraint 11). Otherwise, incompatibilities could emerge if the security properties supported by the source nodes are different from the security properties supported by the target node.

3.3 Semantics of Secure Object Flows Regarding Fork and Join

Fig. 4 shows examples of the different configuration options of secure object flows that include fork or join nodes. Fig. 4a shows a configuration where a fork node splits a secure object flow into multiple concurrent flows. Because the tokens arriving at a fork node are duplicated, all target nodes must be SecureNodes (see OCL Constraint 12). Furthermore, the target nodes must support the same security properties as the corresponding source node (see OCL Constraint 13). This constraint ensures that security properties cannot be lost when traversing a fork node and that the target node(s) are able to check and ensure the corresponding security properties.

Fig. 4b shows a configuration where a join node synchronizes multiple object flows – one of which is a secure object flow. Because in this case the join node receives secure as well as ordinary object tokens, we define that the tokens cannot be combined (see OCL Constraint 14). Moreover, we define that if a join node receives at least one secure object flow, then the target node of this join node must also be a SecureNode (see OCL Constraint 15). This constraint guarantees that each secure object token passing a join node can be checked and processed by the corresponding target node.

Fig. 4c shows a configuration where a join node synchronizes multiple secure object flows. As defined in OCL Constraint 15 the target must be a secure node.

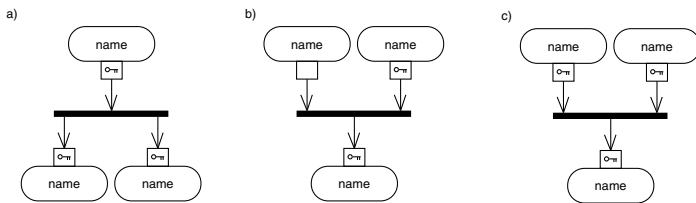


Fig. 4. Secure object flows with fork and join nodes

Furthermore, all source nodes must support compatible security properties (OCL Constraint [16]). In addition, the target node must support all security properties of the corresponding source nodes (see OCL Constraint [17]). Otherwise, inconsistencies could emerge if the security properties supported by the source nodes are different from the security properties supported by the target node.

4 Related Work

Several approaches exist to integrate process models with specific types of security policies and/or constraints on different abstraction levels. Jensen and Feja present an approach to specify three types of security properties (access control, confidentiality, and integrity) in Event-driven Process Chains [15]. Another related approach is UMLsec [16]. In essence, it provides a UML profile for the definition and analysis of security properties for software systems. For example, UMLsec is used to define and verify cryptographic protocols. However, UMLsec aims at a lower abstraction layer than our SecureObjectFlows extension. Therefore, UMLsec is well-suited to be combined with our approach. SecureObjectFlows would then be used to model business processes and process-level security properties, while UMLsec would be used to specify the fine-grained system-level procedures for encryption and integrity checking in a particular software system. Furthermore, Basin et al. [17] present a sophisticated approach called model-driven security. They demonstrate their approach with an UML profile for RBAC (Role-Based Access Control) called SecureUML. Here, the focus is on integrating security aspects with a model-driven development approach rather than modeling of business processes and process-related security properties. In fact, the model-driven security approach of SecureUML and our SecureObjectFlows package are well-suited to be combined in a complementary fashion.

5 Conclusion

A complete and correct mapping of process definitions and related security properties to the corresponding software system is essential in order to assure consistency between the modeling-level specifications on the one hand, and the software system that actually manages corresponding process instances and enforces the respective security properties on the other hand.

UML activity models provide a process modeling language that is tightly integrated with other model types from the UML family (such as class models, state machines, or interaction models). In this paper, we presented SecureObjectFlows as an integrated approach to model confidentiality and integrity properties of object flows in UML activity diagrams. The semantics of our extension are formally defined via the OCL. Corresponding software tools can enforce these invariants on the modeling-level as well as in runtime models. Thereby, we can ensure the consistency of our SecureObjectFlows models with the respective constraints. Moreover, our extension can be applied to supplement other UML-based approaches and can be integrated in UML-based software tools.

References

1. Damianides, M.: How does SOX change IT? *Journal of Corporate Accounting & Finance* 15(6) (2004)
2. Mishra, S., Weistroffer, H.R.: A Framework for Integrating Sarbanes-Oxley Compliance into the Systems Development Process. *Communications of the Association for Information Systems (CAIS)* 20(1) (2007)
3. National Institute of Standards and Technology: An Introduction to Computer Security: The NIST Handbook. Special Publication 800-12 (1995), <http://csrc.nist.gov/publications/nistpubs/800-12/handbook.pdf>
4. National Institute of Standards and Technology: Recommended Security Controls for Federal Information Systems and Organizations. NIST Special Publication 800-53, Revision 3 (2009), http://csrc.nist.gov/publications/nistpubs/800-53-Rev3/sp800-53-rev3-final_updated_errata_05-01-2010.pdf
5. Botha, R.A., Eloff, J.H.P.: Separation of Duties for Access Control Enforcement in Workflow Environments. *IBM Systems Journal* 40(3) (2001)
6. Wainer, J., Barthelmes, P., Kumar, A.: W-RBAC - A Workflow Security Model Incorporating Controlled Overriding of Constraints. *International Journal of Cooperative Information Systems (IJCIS)* 12(4) (December 2003)
7. Object Management Group: Business Process Model and Notation (BPMN) - Version 2.0 - Beta 2 (2010), <http://www.omg.org/spec/BPMN/2.0/Beta2/PDF>
8. Object Management Group: OMG Unified Modeling Language (OMG UML), Superstructure - Version 2.3 (2010), <http://www.omg.org/spec/UML/2.3/Superstructure/PDF/>
9. Axenath, B., Kindler, E., Rubin, V.: AMFIBIA: A Meta-Model for the Integration of Business Process Modelling Aspects. In: Leymann, F., Reisig, W., Thatte, S.R., van der Aalst, W. (eds.) *The Role of Business Processes in Service Oriented Architectures*. Dagstuhl Seminar Proceedings, vol. 06291 (2006)
10. Zdun, U.: Patterns of Component and Language Integration. In: Manolescu, D., Voelter, M., Noble, J. (eds.) *Pattern Languages of Program Design 5* (2006)
11. Object Management Group: Object Constraint Language - Version 2.2 (2010), <http://www.omg.org/spec/OCL/2.2/PDF>
12. Committee on National Security Systems: National Information Assurance (IA) - Glossary (2010), http://www.cnss.gov/Assets/pdf/cnssi_4009.pdf
13. National Security Agency: Information Assurance Technical Framework (2000), <http://handle.dtic.mil/100.2/ADA393328>
14. Sandhu, R.S.: On Five Definitions of Data Integrity. In: *Proceedings of the IFIP WG11.3 Working Conference on Database Security VII* (1993)
15. Jensen, M., Feja, S.: A Security Modeling Approach for Web-Service-based Business Processes. In: *2009 16th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems*, pp. 340–347. IEEE, Los Alamitos (2009)
16. Jürjens, J.: *Secure Systems Development with UML*. Springer, Heidelberg (2005)
17. Basin, D., Doser, J., Lodderstedt, T.: Model Driven Security: From UML Models to Access Control Infrastructures. *ACM Transactions on Software Engineering and Methodology (TOSEM)* 15(1) (January 2006)

A Constraints for Secure Object Flows

This section provides the complete list of OCL-expressed derived values and invariants for the UML extension specified in Section 2.

OCL Constraint 1 *The confidentialityEnsured attribute of the SecureNode classifier is derived from the confidentialityAlgorithm and confidentialityKeyLength attributes and evaluates to true if confidentiality-related security properties are supported.*

```
context SecureNode::confidentialityEnsured : Boolean
derive: if confidentialityAlgorithm->notEmpty() and
        confidentialityKeyLength->notEmpty()
        then true else false
endif
```

OCL Constraint 2 *The integrityEnsured attribute of the SecureNode classifier is derived from the integrityAlgorithm attribute. It evaluates to true if an integrity-related security property is supported.*

```
context SecureNode::integrityEnsured : Boolean
derive: if integrityAlgorithm->notEmpty()
        then true else false
endif
```

OCL Constraint 3 *A secure object node must ensure either confidentiality, or integrity, or both.*

```
context SecureNode inv:
    self.confidentialityEnsured or
    self.integrityEnsured
```

OCL Constraint 4 *Any target of a secure object flow must also be a secure object node.*

```
context ObjectNode inv:
    if self.incoming->exists(i | i.source.oclIsKindOf(SecureNode))
    then self.oclIsKindOf(SecureNode) else self.oclIsKindOf(ObjectNode)
endif
```

OCL Constraint 5 *The downstream secure object node must support at least all security properties supported by corresponding upstream secure object nodes.*

```
context SecureNode
inv: self.incoming->forall(i |
    if i.source.oclIsKindOf(SecureNode) and i.source.oclAsType(SecureNode).confidentialityEnsured
    then self.confidentialityAlgorithm = i.source.oclAsType(SecureNode).confidentialityAlgorithm and
        self.confidentialityKeyLength = i.source.oclAsType(SecureNode).confidentialityKeyLength
    else true endif)
inv: self.incoming->forall(i |
    if i.source.oclIsKindOf(SecureNode) and i.source.oclAsType(SecureNode).integrityEnsured
    then self.integrityAlgorithm = i.source.oclAsType(SecureNode).integrityAlgorithm
    else true endif)
```

OCL Constraint 6 *All secure object nodes having the same target node must support identical security properties.*

```
context SecureNode
inv: self.incoming->forall(i1,i2 |
    if i1.source.oclIsKindOf(SecureNode) and i2.source.oclIsKindOf(SecureNode) and
        i1.source.oclAsType(SecureNode).confidentialityEnsured and i2.source.oclAsType(SecureNode).confidentialityEnsured
    then i1.source.oclAsType(SecureNode).confidentialityAlgorithm = i2.source.oclAsType(SecureNode).confidentialityAlgorithm and
        i1.source.oclAsType(SecureNode).confidentialityKeyLength = i2.source.oclAsType(SecureNode).confidentialityKeyLength
    else true endif)
inv: self.incoming->forall(i1,i2 |
    if i1.source.oclIsKindOf(SecureNode) and i2.source.oclIsKindOf(SecureNode) and
        i1.source.oclAsType(SecureNode).integrityEnsured and i2.source.oclAsType(SecureNode).integrityEnsured
    then i1.source.oclAsType(SecureNode).integrityAlgorithm = i2.source.oclAsType(SecureNode).integrityAlgorithm
    else true endif)
```

OCL Constraint 7 *If a decision node has a secure source node, all target object nodes must also be secured.*

```
context DecisionNode inv:
  if self.incoming->exists(i | i.source.ocIsKindOf(SecureNode))
  then self.outgoing->forAll(o | o.target.ocIsKindOf(SecureNode))
  else true endif
```

OCL Constraint 8 *Target secure nodes of a decision node must support identical security properties as the corresponding source node.*

```
context DecisionNode
inv: self.incoming->forAll(i |
  if i.source.ocIsKindOf(SecureNode) and i.source.ocIsType(SecureNode).confidentialityEnsured
  then self.outgoing->forAll(o |
    o.target.ocIsType(SecureNode).confidentialityAlgorithm = i.source.ocIsType(SecureNode).confidentialityAlgorithm and
    o.target.ocIsType(SecureNode).confidentialityKeyLength = i.source.ocIsType(SecureNode).confidentialityKeyLength)
  else true endif)
inv: self.incoming->forAll(i |
  if i.source.ocIsKindOf(SecureNode) and i.source.ocIsType(SecureNode).integrityEnsured
  then self.outgoing->forAll(o |
    o.target.ocIsType(SecureNode).integrityAlgorithm = i.source.ocIsType(SecureNode).integrityAlgorithm)
  else true endif)
```

OCL Constraint 9 *If a merge node has at least one secure source node, the target must also be a secure node.*

```
context MergeNode inv:
  if self.incoming->exists(i | i.source.ocIsKindOf(SecureNode))
  then self.outgoing.target.ocIsKindOf(SecureNode)
  else true endif
```

OCL Constraint 10 *All secure source nodes that serve as input to a merge node must support the same security properties.*

```
context MergeNode
inv: self.incoming->forAll(i1,i2 |
  if i1.source.ocIsKindOf(SecureNode) and i2.source.ocIsKindOf(SecureNode) and
  i1.source.ocIsType(SecureNode).confidentialityEnsured and i2.source.ocIsType(SecureNode).confidentialityEnsured
  then i1.source.ocIsType(SecureNode).confidentialityAlgorithm = i2.source.ocIsType(SecureNode).confidentialityAlgorithm and
  i1.source.ocIsType(SecureNode).confidentialityKeyLength = i2.source.ocIsType(SecureNode).confidentialityKeyLength
  else true endif)
inv: self.incoming->forAll(i1,i2 |
  if i1.source.ocIsKindOf(SecureNode) and i2.source.ocIsKindOf(SecureNode) and
  i1.source.ocIsType(SecureNode).integrityEnsured and i2.source.ocIsType(SecureNode).integrityEnsured
  then i1.source.ocIsType(SecureNode).integrityAlgorithm = i2.source.ocIsType(SecureNode).integrityAlgorithm
  else true endif)
```

OCL Constraint 11 *The secure target node of a merge node must be capable of supporting all security properties of corresponding source nodes.*

```
context MergeNode
inv: self.incoming->forAll(i |
  if i.source.ocIsKindOf(SecureNode) and i.source.ocIsType(SecureNode).confidentialityEnsured
  then self.outgoing.target.ocIsType(SecureNode).confidentialityAlgorithm =
    i.source.ocIsType(SecureNode).confidentialityAlgorithm and
    self.outgoing.target.ocIsType(SecureNode).confidentialityKeyLength =
    i.source.ocIsType(SecureNode).confidentialityKeyLength
  else true endif)
inv: self.incoming->forAll(i |
  if i.source.ocIsKindOf(SecureNode) and i.source.ocIsType(SecureNode).integrityEnsured
  then self.outgoing.target.ocIsType(SecureNode).integrityAlgorithm = i.source.ocIsType(SecureNode).integrityAlgorithm
  else true endif)
```

OCL Constraint 12 *If a fork node has a secure source node, all target nodes must also be secured.*

```
context ForkNode inv:
  if self.incoming.source.ocIsKindOf(SecureNode)
  then self.outgoing->forAll(o | o.target.ocIsKindOf(SecureNode))
  else true endif
```

OCL Constraint 13 *Secure target nodes of a fork node must support the same security properties as the corresponding source node.*

```
context ForkNode
inv: if self.incoming.source.ocIsKindOf(SecureNode) and self.incoming.source.ocAsType(SecureNode).confidentialityEnsured
then self.outgoing->forall(o |
  o.target.ocAsType(SecureNode).confidentialityAlgorithm =
    self.incoming.source.ocAsType(SecureNode).confidentialityAlgorithm and
  o.target.ocAsType(SecureNode).confidentialityKeyLength =
    self.incoming.source.ocAsType(SecureNode).confidentialityKeyLength)
else true endif
inv: if self.incoming.source.ocIsKindOf(SecureNode) and self.incoming.source.ocAsType(SecureNode).integrityEnsured
then self.outgoing->forall(o |
  o.target.ocAsType(SecureNode).integrityAlgorithm = self.incoming.source.ocAsType(SecureNode).integrityAlgorithm)
else true endif
```

OCL Constraint 14 *If both, secure object nodes and ordinary object nodes are input to a join node, this join node must not combine the corresponding tokens.*

```
context JoinNode inv:
self.incoming->forall(i1,i2 |
  if i1.source.ocIsKindOf(SecureNode) and
  i2.source.ocIsKindOf(SecureNode) = false
then self.isCombineDuplicate = false
else true endif)
```

OCL Constraint 15 *If a join node has at least one secure source node, the corresponding target node must also be secured.*

```
context JoinNode inv:
if self.incoming->exists(i | i.source.ocIsKindOf(SecureNode))
then self.outgoing.target.ocIsKindOf(SecureNode)
else true endif
```

OCL Constraint 16 *All secure source nodes of a join node must support the same security properties.*

```
context JoinNode
inv: self.incoming->forall(i1,i2 |
  if i1.source.ocIsKindOf(SecureNode) and i2.source.ocIsKindOf(SecureNode) and
  i1.source.ocAsType(SecureNode).confidentialityEnsured and i2.source.ocAsType(SecureNode).confidentialityEnsured
then i1.source.ocAsType(SecureNode).confidentialityAlgorithm = i2.source.ocAsType(SecureNode).confidentialityAlgorithm and
  i1.source.ocAsType(SecureNode).confidentialityKeyLength = i2.source.ocAsType(SecureNode).confidentialityKeyLength
else true endif)
inv: self.incoming->forall(i1,i2 |
  if i1.source.ocIsKindOf(SecureNode) and i2.source.ocIsKindOf(SecureNode) and
  i1.source.ocAsType(SecureNode).integrityEnsured and i2.source.ocAsType(SecureNode).integrityEnsured
then i1.source.ocAsType(SecureNode).integrityAlgorithm = i2.source.ocAsType(SecureNode).integrityAlgorithm
else true endif)
```

OCL Constraint 17 *The secure target node of a join node must be capable of supporting all security properties of corresponding source secure nodes.*

```
context JoinNode
inv: self.incoming->forall(i |
  if i.source.ocIsKindOf(SecureNode) and i.source.ocAsType(SecureNode).confidentialityEnsured
then self.outgoing.target.ocAsType(SecureNode).confidentialityAlgorithm =
  i.source.ocAsType(SecureNode).confidentialityAlgorithm and
  self.outgoing.target.ocAsType(SecureNode).confidentialityKeyLength =
  i.source.ocAsType(SecureNode).confidentialityKeyLength
else true endif)
inv: self.incoming->forall(i |
  if i.source.ocIsKindOf(SecureNode) and i.source.ocAsType(SecureNode).integrityEnsured
then self.outgoing.target.ocAsType(SecureNode).integrityAlgorithm = i.source.ocAsType(SecureNode).integrityAlgorithm
else true endif)
```

Inventory Management with Dynamic Bayesian Network Software Systems

Mark Taylor¹ and Charles Fox²

¹ Management School

² Adaptive Behaviour Research Group
University of Sheffield, UK
`charles.fox@sheffield.ac.uk`

Abstract. Inventory management at a single or multiple levels of a supply chain is usually performed with computations such as Economic Order Quantity or Markov Decision Processes. The former makes many unrealistic assumptions and the later requires specialist Operations Research knowledge to implement. Dynamic Bayesian networks provide an alternative framework which is accessible to non-specialist managers through off-the-shelf graphical software systems. We show how such systems may be deployed to model a simple inventory problem, and learn an improved solution over EOQ. We discuss how these systems can allow managers to model additional risk factors throughout a supply chain through intuitive, incremental extensions to the Bayesian networks.

Keywords: inventory, supply chain, risk management, Bayesian networks, economic order quantity, Markov decision process.

1 Introduction

Supply chain risk management (SCRM, [6]) involves modelling and optimising the flow of goods between suppliers, customers and warehouses. Broadly construed, the scope of SCRM includes factors such as transportation risk, supplier failure and customer order models, at all levels of a supply chain. Traditionally, inventory theory has been applied at the level of the single organisation, but can now be viewed as a component of larger multi-echelon supply chain models.

Economic Order Quantity (EOQ, [3]) is still a common approach to practical inventory management, determining the order quantity of stock for a particular item [12, p. 374], and is used to determine optimal ordering quantity given a large number of assumptions [8, p. 275]. It is these assumptions that give EOQ its limitations. EOQ's limitations have previously been addressed using mathematical formulations of Markov Decision Processes, known as Stochastic Inventory Theory [2, 4, 11, 15]. These approaches, while optimal, require complex modelling and dynamic programming mathematics specific to each particular case and are not available to non-specialist managers.

The present paper provides a new approach to practical inventory and supply chain risk management using off-the-shelf Bayesian network software. This approach is highly general and may relax most of the standard assumptions made

by EOQ theory. It is highly extensible, as Bayesian networks allow and encourage incremental qualitative and quantitative modelling of details about the world in an intuitive and graphical manner by non-specialist managers. Our long-term goal is to model entire supply chains and detailed risk factors with these tools, however this paper presents our initial proof-of-concept results on a standard inventory problem, then discusses extensions to larger supply chain models. We show here that Bayesian networks can first replicate the standard EOQ case which is a familiar reference point for many managers, then show how they can extend EOQ by relaxing most of its assumptions and finding dynamic policies using reinforcement learning. These policies solve the Markov Decision Process of Stochastic Inventory Control but – unlike in previous work – the learning is wrapped inside a software package leaving the manager free to specify increasingly detailed world models with intuitive visual modelling and historical or subjective probability data. Our aim is not to introduce new mathematics but to illustrate the beginnings of a software-based methodology for highly general, integrated inventory and supply chain risk management by non-specialists.

1.1 Standard EOQ

We make a distinction between the ‘EOQ world’ and the ‘EOQ policy’. The *EOQ world* is the set of assumptions made in EOQ theory, and various inventory policies may be tested in simulations of this world. The *EOQ policy* is the particular policy found by EOQ theory. We will also introduce a *relaxed world* which relaxes most of the assumptions of the EOQ world. (As well as the EOQ policy, we will use a reinforcement learning software system to find optimal policies in the EOQ world and in the relaxed world).

The *EOQ world* assumes [8, p. 275]: (1) Demand is certain, constant and continuous over time; (2) The quantity ordered is constant over ordering times; (3) Ordering cost is constant and independent of quantity ordered; (4) Lead time is fixed; (5) The cost of holding a unit of stock is independent from the quantity in stock; (6) The purchase price of the item is constant; (7) There are no limits on order size.

The *optimal EOQ policy* is defined algebraically by [8, p. 277]

$$EOQ = \sqrt{\frac{2ca}{h}}, \quad (1)$$

where c is the acquisition cost of each order (i.e. the administrative and overhead costs of the order, not the price of the goods themselves); a is the annual usage in units; and h is the holding cost per unit per year.

The following is an example application of EOQ which will be referenced throughout the rest of this study. Assume a demand of $a = 6000$ items per year; potential ordering points at the start of each month; a monthly holding cost of $h = \$0.10$ per item and an ordering cost of $c = \$15$ per order. Using the EOQ formula above, the EOQ-optimal policy is to order 1341 items, with $4.47 = a/1341$ orders per year, which gives an order every $2.7 = 12/4.47$ months.

1.2 Bayesian Networks

Bayesian networks provide a formalism for reasoning about partial beliefs under conditions of uncertainty. These parameters are combined and manipulated according to the rules of probability theory [10]. Let us consider n discrete random variables x_1, x_2, \dots, x_n , a directed acyclic graph with n nodes, and suppose the j th node of the graph is associated to the x_j variable. Then the graph is a Bayesian network, representing the variables x_1, x_2, \dots, x_n , if

$$P(x_1, x_2, \dots, x_n) = \prod_j P(x_j | \text{parents}(x_j)),$$

where $\text{parents}(x_j)$ denotes the set of all variables x_i such that there is an arc from node x_i to x_j in the graph. The probability terms in the product are described by Conditional Probability Tables (CPTs) which may be set by hand or learned from data. Standard algorithms such as junction trees [1] exist to perform inference on Bayesian networks.

Dynamic Bayesian Networks (DBNs) allow the modelling of entities in a changing environment where the values of variables change over time [1, 9]. Functionally, DBNs capture the process of variable values changing over time by representing multiple copies of network nodes with one copy for each time step [1]. Visually, they may be displayed using two copies of each recurrent node representing the current, t , and previous, $t - 1$ states.

Decision Networks (DNs) [14] – also known as ‘influence diagrams’ [5] – are Bayesian Networks with the addition of two types of nodes: Decision Nodes and Utility Nodes. Inferences can be made about the best actions to take to maximise utility.

Utility nodes u_i – conventionally represented as diamonds in pictures of Bayesian networks – may have multiple parents and no children. Here they have finite, integer-valued state space, and are associated with Utility Tables specifying their values as a deterministic functions of the state of their parents. The utility of a whole network is given by the sum of all utility nodes, $\sum_i u_i$. When there is uncertainty in the network, the expected value of the total utility is used, $\langle \sum_i u_i \rangle$. When used in DBNs, utility nodes are discounted by the time value of utility γ to give an expected discounted present utility objective,

$$\langle U \rangle = \sum_{t=1}^{\infty} \gamma^t \left(\sum_{i=1}^N \langle u_i \rangle \right).$$

Action nodes – conventionally represented as squares in pictures of Bayesian networks – are here taken to be finite discrete variables, whose values are selected dynamically by a policy, as a function of the states of their parent nodes [4]. The

¹ The present paper considers only cases where the parents of action nodes are observed. For unobserved parents the actions must be function of the *distribution* of belief rather than the state and the task is known as a Partially Observable MDP.

goal of policy optimisation is to find the policy that maximises the expected discounted present utility.

The networks in fig. 1 and 2 are dynamic Bayesian decision networks, including recurrent connections, action and utility nodes. There are many off-the-shelf software systems that allow Bayesian networks to be constructed graphically by end-users, for example BayesiaLab (www.bayesia.com), Netica (www.norsys.com) and Hugin (www.hugin.com). The examples in this paper are produced using BayesiaLab and show a novel application to inventory and supply chain risk management.

1.3 Reinforcement Learning Policy

To find optimal policies in the relaxed world – as opposed to EOQ policies – our software system uses internally a form of reinforcement learning [13]. It maintains a Q -value for each (action, state) pair, where the state ranges over the combined states of the parents of the action node. These values become estimates,

$$\langle U | s_t, a_t \rangle \approx Q(s_t, a_t) \approx \langle (\sum_i u_i)_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) \rangle, \tag{2}$$

when updated by

$$Q(s_t, a_t) \leftarrow (1 - \alpha)Q(s_t, a_t) + \alpha \langle (\sum_i u_i)_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) \rangle. \tag{3}$$

If Q is known exactly, or a best estimate finalised from learning, then the optimal policy is to select actions,

$$a_t = \arg_{a_t} \max Q(s_t, a_t). \tag{4}$$

Learning can be performed by drawing actions from annealed distributions based on successive estimates of Q ,

$$P(a_t) = \frac{1}{Z} Q(s_t, a_t)^{1/T}. \tag{5}$$

If T is reduced sufficiently slowly over many Monte Carlo Markov Chain samples from the DBN [9], then Q converges to a locally optimal policy, with the probability of reaching a global optimal increasing with the slowness. MCMC sampling and reinforcement learning are standard DBN algorithms which may be performed with off the shelf software systems and knowledge of their details is not required by the end-user.

We emphasise that in contrast to EOQ, which computes all order sizes and times in advance, DBNs can learn dynamic policies, where the ordering action at each time is a function of the state s_t of the network at that time. Such policies can for example increase the number of orders when a backlog state is large.

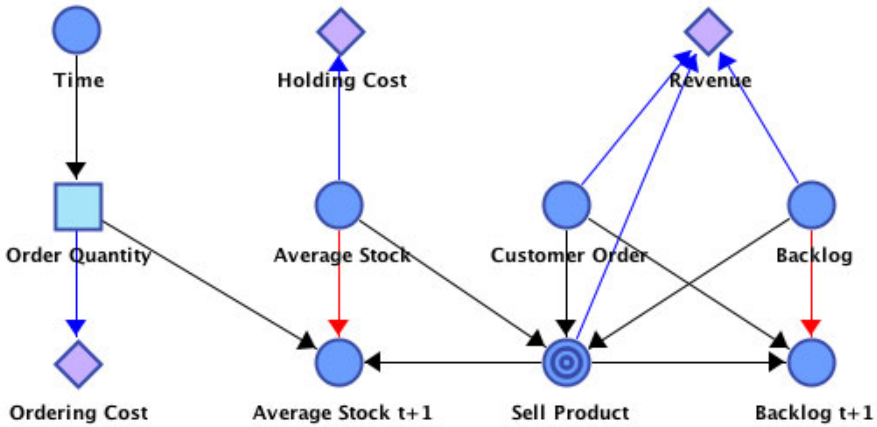


Fig. 1. Dynamic Bayesian network implementation of the EOQ world

2 Methods

We work with two Bayesian network models and two policy types. The first model is of the EOQ world, where customer demand is constant over time. The second is a relaxed world in which customer demand is a random variable conditioned on time of year. The first policy type is the EOQ policy described above. The second policy is the reinforcement learning policy, found automatically as described below. All models are created in BayesiaLab using a graphical interface to design the network, then deterministic equations or probabilities to specify the CPTs. Importantly, specialist knowledge of Bayesian inference and reinforcement learning is not required by the user, who needs only specify the world model using a graphical interface.

2.1 DBN Model of the EOQ World

We first aim to demonstrate how a DBN can model the standard EOQ world. This will serve as a basis for the more complex relaxed model later. Fig. 1 shows the DBN structure. We use discrete time steps, of one month in length. Time-dependent CPTs are discretised into months, and stock quantity CPTs are discretised into integers between 1 and 20 ‘units’ of stock, where one unit comprises 500 items.

As the demand is constant and known (6000 items per year), the *CustomerOrder* node is deterministic and constant, giving the number of customer orders, in units, at each time step to be

$$CustomerOrders = (6000/500)/12 = 1$$

which can be entered in the software as a specification of the CPT.

The number of units ordered for our own inventory each month is modelled by the *OrderQuantity*, an action node whose policy is to be learned.

There are two utility nodes modelling the ordering and holding costs, where the ordering cost is the administrative cost, \$15, of actually placing an order rather than the price of the units ordered, and it costs $-\$0.1 * 500 = -\50 to hold each unit,

$$OrderCost = -15$$

$$HoldingCost = -50 * AverageStock$$

The *Revenue* node models the reward for completing unit sales, being the difference between the purchase and sale price, which we set to be \$200 (this quantity is not required in standard EOQ), multiplied by the number of unit sales in the month,

$$Revenue = 200 * SellProduct$$

We model the size of the inventory and/or order backlog over time by introducing recurrent deterministic nodes,

$$AverageStock_{t+1} = AverageStock_t + OrderQuantity_{t+1}$$

$$Backlog_{t+1} = Backlog_t + CustomerOrder_{t+1}.$$

The quantity sold each month is the minimum of the stock level and the number of orders,

$$SellProduct = \min(AverageStock, Backlog + CustomerOrders).$$

2.2 DBN Model of a Relaxed World

The relaxed world model in fig. 2 violates most of the EOQ assumptions. In particular, (1) the demand (*CustomerOrders*) are no longer constant each month but now depend stochastically on an underlying demand level which is a function of the time of year. This leads to an inherently dynamic model as this uncertainty propagates through the DBN; (2) Non-constant order quantities are now allowed, which may be functions of the backlog size; (3) the order cost utilities may now depend on order size; (4) Dynamic ordering policy allows non-fixed lead times to be modelled: if a component is not received when ordered, the backlog increases which may allow a re-order at the next step; (5) the holding cost utility may now be a function of the stock size (this is common in practice, for example filling a warehouse and needing to build a new one introduces a large non-linearity)². The variable costs used to illustrate (3) and (5) are listed in tables 1a and 1b.

² In the present network we retain (6) fixed purchase prices, although their dependency on time or random variability could easily be added to the DBN model. A new limitation introduced by the DBN formalism is that (7) we must use some upper limit on available order size, although in practice this can be made arbitrarily large, at the expense of computation time, until it becomes unimportant.

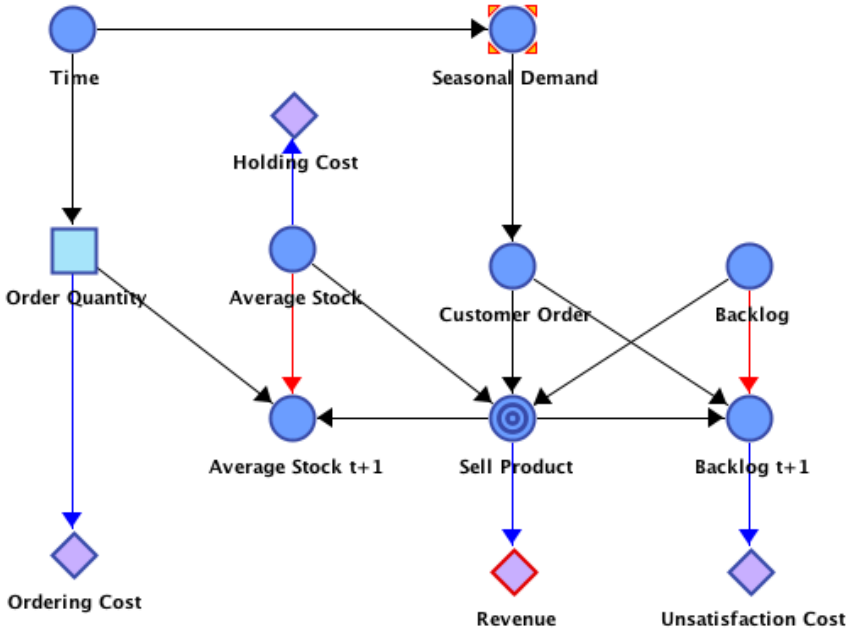


Fig. 2. Dynamic Bayesian network implementation of the relaxed world

Table 1. (a) Non-constant and dependent ordering cost; (b) Quantity dependent stock cost

Order quantity	Ordering cost	Average stock	Holding cost
10	-15	10	-420
9	-16	9	-320
8	-17	8	-290
7	-18	7	-260
6	-19	6	-230
5	-19	5	-200
4	-19	4	-120
3	-21	3	-90
2	-22	2	-60
1	-23	1	-30
0	0	0	0

3 Results

Having introduced time and backlogs into model, we first found that it was necessary to add a negative utility to model customer dissatisfaction if an order is late – this concept does not appear in the original EOQ world, and we found that it can lead to computational instabilities such as infinite backlogs if not included in the model. The *UnsatisfactionCost* node in figure 2 adds a \$10 penalty for each unit-month of delay which stabilises the model.

3.1 Bayes Nets Can Model EOQ World

To show that DBNs have the power to model the EOQ world, and to check for DBN model assumptions effects on solution optimality, we first run the standard EOQ policy on the EOQ DBN world model. If it produces similar results to the standard EOQ model then evidence is obtained that the model is good. In particular, the DBN introduces dynamic state over time, and discretizes time steps, both of which could lead to small deviations from the EOQ solution. Table 2 shows the results of the standard EOQ model against the DBN model of the EOQ world and EOQ policy. It can be seen that there is a small difference in the profit. To test whether this was due to the discretisation of time (which could be improved by using smaller time intervals in the DBN) or to other assumptions of the DBN, we computed similar results for a discretised version of standard EOQ, rounding the EOQ policy to integer months and the same order size bins as used in the DBN model. This result was very close to the original EOQ, suggesting that the rest of the discrepancy in the DBN model is due to other assumptions which do not appear in the EOQ model.

Table 2. Expected Average Profit

	Order freq.	size	cost	Hold. cost	Goods cost	Revenue	Profit
Standard EOQ model	4.47	1341.6	67.04	67.08	1200	2400	1065.88
DBN model of EOQ world	4	1500	60	100	1200	2400	1040
Discretised EOQ model	4	1500	60	75	1200	2400	1065

3.2 RL Policy Beats EOQ in Relaxed World

Table 3 shows results from running the EOQ policy in the relaxed world, which includes stochastic customer demand, and variable holding and order costs. It also shows the results of a policy learned by reinforcement learning – built into our software system and usable by non-specialist modellers. Although the overall customer demand level is the same as the EOQ world, its fluctuations over time give rise to backlogs and unsatisfaction costs. The RL policy is able to adapt to these fluctuations and make more profit than the rigid EOQ policy. (The overall profit is lower than in the EOQ *world* because the demand fluctuations make inventory management into a harder problem in general).

Table 3. EOQ vs RL policy, in Relaxed world

	Holding cost	Unsatisfaction	Revenue	Profit
EOQ policy, on relaxed world	120.5	959.87	2399.63	61.28
RL policy, on relaxed world	116.15	825.60	2200.33	112.76

4 Discussion

The first set of results in table 2 show that the DBN formalism – recently made accessible to non-specialist managers through off-the-shelf software systems – has the power to model the standard EOQ world. The results show that the DBN simulation gives similar results to the discretised version of the EOQ policy, which in turn gives a close match to the performance of the continuous EOQ policy. In practice discretisation is often a real-world requirement, with stock being ordered in large standardised unit sizes. The discrepancy between the discrete EOQ math model and the DBN simulation is small, but should be explained.

We believe the discrepancy is due to latency factors relating to ordering of node updates. For example, demonstrating EOQ policy over a series of 12 time steps causes a cost discrepancy where a unit of stock is held until it can be sold fulfilling a customer order. The EOQ policy requires the ordering 3 units of stock four times per 12 time steps. Ordering of stock starts at time period 1. At time period 1 there are only two customer orders available (one customer order at time period 0, and one customer order at time period 1). This requires the third unit of stock to be held until another constant customer order is received at time period 2, at which point it is sold, and there are no units of stock left. This pattern therefore repeats every third time step and incurs additional holding cost over the EOQ model at each repetition. Latency factors could be made arbitrarily small by using smaller time steps in the model, at the expense of computation time.

Table 3 shows that EOQ policy is poor in the more realistic, relaxed world. Sticking to a predetermined, deterministic policy gives poor results when the customer orders and backlog are dynamic. In contrast, the profit is increased in this world when a dynamic, reinforcement learning policy is used, which can alter its ordering behaviour as a function of these variables.

Together, our results give a proof-of-concept example that shows how Dynamic Bayesian Network software systems can improve over standard EOQ inventory methods. Unlike previous Markov Decision Process approaches, such software systems do not require the application of advanced math by end-users, rather they allow graphical representations of the Bayesian networks to be constructed by managers in an intuitive ‘point and click’ manner. This allows the manager to focus on constructing a realistic world model rather than on the details of inference and training algorithms. In contrast to recent fuzzy inventory approaches which solve a similar task, [7], DBNs have precise probabilistic semantics; in contrast to (s,S) inventory systems, DBNs are more general, allowing the policy to depend on factors other than the current stock level such as time of year.

We found that a problem with practical Bayesian network reinforcement learning is that it requires large amount of computation time to find good policies. The simple inventory policy used in table 3 took several hours to learn on an Apple iBook laptop, and the time typically grows exponentially with the number of parent nodes of the action nodes and the number of states in those parents.

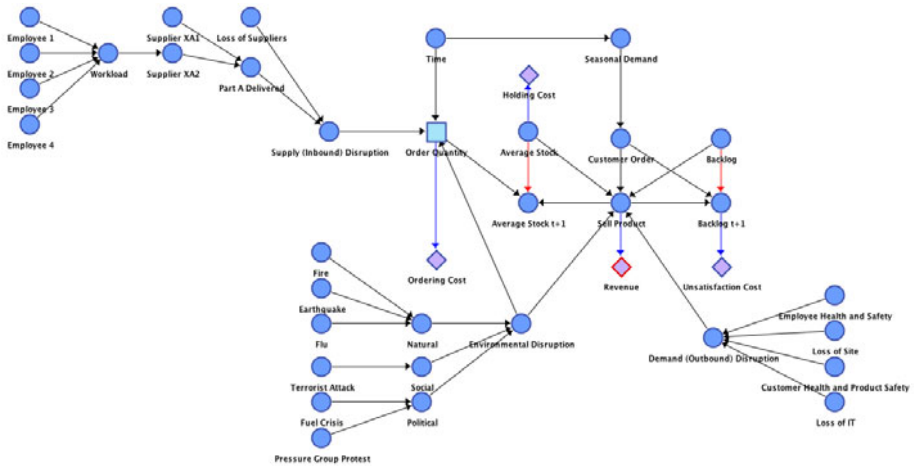


Fig. 3. Example of an extended SCM Dynamic Bayesian network

This is because the number of possible policies scales with these values and any learning algorithm needs to search amongst these policies for the best one. However for economically important decisions it is possible to buy supercomputer or cloud computing time to reduce the search time by arbitrary amounts.

A further advantage of dynamic Bayesian network software systems over existing MDP methods is the potential for extensibility of the models, both within the manager's organisation and beyond it into its supply chain partners. Bayesian networks allow incremental model refinement by replacing generic prior nodes with more detailed world models. As an example of this process, fig 3 shows an extended, supply chain risk version of the inventory model. Three sources of uncertainty have been modelled in detail: inbound, outbound, and environmental factors. Inbound disruption can be caused by one or more suppliers failing to deliver one or more parts required to manufacture. In this example, supplier XA2 is a collaborating supply chain partner who has provided internal data about its individual employees workloads. By including this information into the Bayesian network we can made more detailed inferences about the supply side, and find better policies depending on them, such as distributing orders across multiple suppliers as a function of the dynamic internal states of those suppliers. On the outbound side, we can model potential problems with our client's state such as loss of site or IT facilities which would affect their ability to take our deliveries. Internal to our own manufacturing processes, we can model the effects of potential disruptions such as earthquakes, fuel crises or epidemics. Each of these nodes contains a CPT table, whose values can be estimated from historical data (e.g. the number of earthquakes in the location of our factory) or by experts (e.g. political analysts forecasts of terrorist risks). Unlike EOQ and standard MDP approaches to inventory, the Bayesian network formalism, as illustrated here, allows seamless integration of such detailed risk models with inventory policy, both within the organisation and across the supply chain partners. Bayesian network

software systems thus provide a common language through which to integrate supply chain management, enterprise management and inventory policy, which can be spoken and used by non-specialist inventory managers through intuitive graphical tools.

References

1. Bayesian Networks. In: Arbib, M.A. (ed.) *The Handbook of Brain Theory and Neural Networks*, MIT Press, Cambridge (2003)
2. Buffet, S.: A Markov model for inventory level optimisation in supply chain management. In: *Eighteenth Canadian Conference on Artificial Intelligence*, pp. 133–144 (2005)
3. Erlenkotter, D.: Ford Whittman Harris and the economic order quantity model. *Operations Research* 38(6), 937–946 (1990)
4. Giannoccaro, I., Pontrandolfo, P.: Inventory management in supply chains: a reinforcement learning approach. *International Journal of Production Economics* 78, 153–161 (2002)
5. Howard, R.A., Matheson, J.E.: Influence diagrams. *Decision Analysis* 2(3), 127–143 (2005)
6. Kayne, D.: *Managing Risk and Resilience in the Supply Chain*. BSI British Standards Institutio (2008)
7. Kofjač, D., Kljajić, M., Škraba, A., Rodič, B.: Adaptive fuzzy inventory control algorithm for replenishment process optimization in an uncertain environment. In: Abramowicz, W. (ed.) *BIS 2007. LNCS*, vol. 4439, pp. 536–548. Springer, Heidelberg (2007)
8. Lysons, K., Gillingham, M.: *Purchasing and Supply Chain Management*, 6th edn. Prentice-Hall, Englewood Cliffs (2003)
9. Murphy, K.P.: *Dynamic Bayesian networks: Representation, inference and learning*. Technical report, University of California, Berkeley (2002)
10. Pearl, J.: *Probabilistic reasoning in intelligent systems: networks of plausible inference*, 2nd edn. Morgan Kaufmann Publishers, San Francisco (1988)
11. Porteus, E.L.: *Foundations of stochastic inventory theory*. Stanford University Press (2002)
12. Slack, N., Chambers, S., Johnston, R.: *Operations Management*, 5th edn. Prentice Hall, Financial Times (2007)
13. Sutton, R., Barto, A.: *Reinforcement Learning*. MIT Press, Cambridge (1998)
14. Zhang, N.L., Qi, R., Poole, D.: A computational theory of decision networks. *International Journal of Approximate Reasoning* 11, 83–158 (1994)
15. Zhao, Q., Chen, S., Leung, S., Lai, K.: Integration of inventory and transportation decisions in a logistics system. *Transportation Research Part E* 46, 913–925 (2010)

Author Index

- Bhiri, Sami, 86
Bodenstaff, Lianne, 98
Buchmann, Robert Andrei, 218
Busch, Bastian, 62
- Cheng, Ran, 13
- de Leoni, Massimiliano, 37
Derguech, Wassim, 86
Döhring, Markus, 25
Dolean, Cristina Claudia, 172
- Fahland, Dirk, 37
Feja, Sven, 50
Feldmann, Marius, 148
Fellmann, Michael, 62
Fox, Charles, 290
Franczyk, Bogdan, 242
Frasincar, Flavius, 185
- Ghiran, Ana-Maria, 266
Giannoulis, Constantinos, 111
- Heerschop, Bas, 185
Hogenboom, Alexander, 185
Hoisl, Bernhard, 278
- Indulska, Marta, 13
- Karg, Lars, 25
Katz, Philipp, 148
Kazienko, Przemysław, 197
Kessler, Wiltrud, 254
Kirkman, Ryan, 136
Klinkmüller, Christopher, 242
Koetter, Falko, 74
Kunkel, Robert, 242
Kurz, Peter, 207
- Leukel, Joerg, 254
Lotyzc, Aneta, 50
Ludwig, André, 242
Lunze, Torsten, 148
- Maier, Bernhard, 160
Mendt, Tamara, 1
- Meza, Radu, 218
Mican, Daniel, 172
Michalski, Radosław, 197
Mintchev, Sava, 230
Mitschang, Bernhard, 123, 160
- Niedermann, Florian, 123, 160
- Palus, Sebastian, 197
Petrusel, Razvan, 172
Pulcher, Delia, 218
Pulvermüller, Elke, 50
- Radeschütz, Sylvia, 123, 160
Reichert, Manfred, 98
Ressel, Dominic, 254
Röhrborn, Dirk, 148
- Sadiq, Shazia, 13, 136
Scheuermann, Andreas, 254
Schill, Alexander, 148
Schleicher, Daniel, 74
Schuele, Michael, 254
Schwarz, Holger, 160
Setiawan, Mukhammad Andri, 136
Shahzad, Khurram, 111
Sikorski, Andrzej, 207
Silaghi, Gheorghe Cosmin, 266
Sinz, Carsten, 1
Speck, Andreas, 50
Strembeck, Mark, 278
- Taylor, Mark, 290
Thomas, Oliver, 62
Tomai, Nicolae, 266
Tveretina, Olga, 1
- van der Aalst, Wil M.P., 37
Vanderfeesten, Irene, 172
van Dongen, Boudewijn F., 37
- Weidmann, Monika, 74
Witt, Sören, 50
Wombacher, Andreas, 98
- Zimmermann, Birgit, 25