Roland Kaschek
Christian Kop
Claudia Steinberger
Günther Fliedl (Eds.)

LNBIP 5

Information Systems and e-Business Technologies

2nd International United Information Systems Conference
UNISCON 2008
Klagenfurt, Austria, April 2008, Proceedings

Springer

Roland Kaschek Christian Kop
Claudia Steinberger Günther Fliedl (Eds.)

Information Systems and e-Business Technologies

2nd International United Information Systems Conference
UNISCON 2008
Klagenfurt, Austria, April 22-25, 2008
Proceedings

Springer

Volume Editors

Roland Kaschek
Massey University
School of Engineering and Advanced Technology
Private Bag 756, Wellington, New Zealand
E-mail: R.H.Kaschek@massey.ac.nz

Christian Kop
Claudia Steinberger
Günther Fliedl
Alpen-Adria-Universität Klagenfurt
Institute of Applied Informatics
Research Group Application Engineering
Universitätsstraße 65-67, 9020 Klagenfurt, Austria
E-mail: {chris,claudia,fliedl}@ifit.uni-klu.ac.at

Library of Congress Control Number: 2008924791

ACM Computing Classification (1998): H.4, J.1, D.2

ISSN	1865-1348
ISBN-10	3-540-78941-3 Springer Berlin Heidelberg New York
ISBN-13	978-3-540-78941-3 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12250912 06/3180 5 4 3 2 1 0

Preface

Welcome to the proceedings of UNISCON 2008! UNISCON 2008 was an experiment – additionally to a number of peer-reviewed conference papers two keynote presentations and a substantial number of invited presentations were offered at the conference and are included in these proceedings. Making this experiment was a consequence of our desire to honor with UNISCON 2008 our teacher, colleague, boss, and friend Heinrich C. Mayr on occasion of his 60th birthday. A large number of well-known researchers agreed that Heinrich's birthday was a very welcome occasion for meeting and working together. All of them came to Klagenfurt to celebrate Heinrich's birthday in the way we know he likes best: meeting, talking to each other and extending and perfecting one's knowledge. Heinrich in fact is a very rare individual, at least in the European information systems community. As very few others he has contributed to informatics in industry (as CEO of a successful software vendor), research (as co-author of more than 150 papers), research politics, organization, and administration (as twice elected president of the GI, co-founder of EMISA, vice chancellor of his university, dean of his school, head of his department, and organizer of important conferences such as the Entity-Relationship conference). He has provided visions for the development of the Alpen-Adria-Universität and its scientific environment, and established ties between that university and local as well as international industry. Any recollection of his credits in this preface will be incomplete.

UNISCON 2008 was also a repetition of a bundling of events, the organization of which Heinrich previously as organizer and co-founder was involved in, i.e., the ECOMO workshop series and the ISTA conference series. We obtained a sensible paper acceptance rate of about 30%. In addition to full papers we had some short paper submissions, which our Program Committees found worthy of being published. These are included in this volume and are not specifically marked.

We thank each of the many friends, colleagues, and collaborators who helped in preparing, organizing, and running UNISCON 2008. In particular we thank the office staff and our colleagues (Marko Anzelak, Wolfgang Ebner, Stefan Ellersdorfer, Doris Gälle, Jörg Kerschbaumer, Gert Morak, Klothilde Puschl, Christine Seger, Maria Semmelrock-Picej, Volodymyr Sokol, Kateryna Sokol, Anita Toschkov, Jürgen Vöhringer, Martin Waiguny, Karl Wiggisser), students who helped us throughout the conference, the authors of the papers in these proceedings, and of course the colleagues from "Springer" who made it possible to have this volume ready for distribution at the beginning of the conference. We also thank the sponsors of UNISCON 2008.

Most of all we thank Heinrich for inspiring us, for the role model he has provided, and for the counsel he has given when he was asked to do so. We hope that he is going to be with us for a long time to come.

Finally, we thank every UNISCON 2008 attendee, and everyone else for purchasing this volume, for reading papers in it, and for sharing their views with us.

April 2008

Roland Kaschek
Christian Kop
Claudia Steinberger
Günther Fliedl

UNISCON 2008 Organization

eCOMO Steering Committee

Heinrich C. Mayr	Alpen-Adria-Universität Klagenfurt, Austria
Bernhard Thalheim	Christian-Albrechts-Universität zu Kiel, Germany

ISTA Steering Committee

Dimitris Karagiannis	Universität Wien, Austria
Steven W. Liddle	Brigham Young University, USA
Heinrich C. Mayr	Alpen-Adria-Universität Klagenfurt, Austria
Mikhail D. Godlevsky	National Technical University "KhPI", Ukraine

eCOMO Co-chairs

Christian Kop	Alpen-Adria-Universität Klagenfurt, Austria
Claudia Steinberger	Alpen-Adria-Universität Klagenfurt, Austria
Willem Jan v. d. Heuvel	Tilburg University, The Netherlands
Gernot Mödritscher	Alpen-Adria-Universität Klagenfurt, Austria

Ukrainian Workshop Chair

Vladimir Shekhovtsov	National Technical University "KhPI", Ukraine

ISTA Co-chairs

Roland Kaschek	Massey University, New Zealand
Günther Fliedl	Alpen-Adria-Universität Klagenfurt, Austria
Mykola Tkachuk	National Technical University "KhPI", Ukraine

Industrial Program Co-chairs

Robert Grascher	Integranova GmbH, Geschäftsstelle Österreich
Michael Stark	Comm-Unity EDV GmbH
Dietmar Wuksch	CICERO Consulting GmbH

Liaison Managers

Tilmann Reuther	Alpen-Adria-Universität Klagenfurt, Austria
Volodymyr Sokol	Alpen-Adria-Universität Klagenfurt, Austria

Public Relationship Management

Maria Semmelrock-Picej	Industriestiftungsinstitut eBusiness, Austria
Anna Margarete Landes	Alpen-Adria-Universität Klagenfurt, Austria
Karin Waldher	Alpen-Adria-Universität Klagenfurt, Austria

Organization Co-chairs

Doris Gälle	Alpen-Adria-Universität Klagenfurt, Austria
Jürgen Vöhringer	Alpen-Adria-Universität Klagenfurt, Austria

Conference Desk

Christine Seger	Alpen-Adria-Universität Klagenfurt, Austria
Klothilde Puschl	Alpen-Adria-Universität Klagenfurt, Austria

Organization

Stefan Ellersdorfer	Alpen-Adria-Universität Klagenfurt, Austria
Jörg Kerschbaumer	Alpen-Adria-Universität Klagenfurt, Austria
Marko Anzelak	Industriestiftungsinstitut eBusiness, Austria
Wolfgang Ebner	Industriestiftungsinstitut eBusiness, Austria
Gert Morak	Industriestiftungsinstitut eBusiness, Austria
Kateryna Sokol	Industriestiftungsinstitut eBusiness, Austria
Anita Toschkov	Industriestiftungsinstitut eBusiness, Austria
Martin Waiguny	Industriestiftungsinstitut eBusiness, Austria
Vladimir Shekhovtsov	National Technical University “KhPI”, Ukraine

eCOMO Program Committee

Fahim Akhter	Zayed University, United Arab Emirates
Peter Bellström	Karlstad University, Sweden
Anthony Bloesch	Microsoft Corporation, USA
Cinzia Cappiello	Politecnico di Milano, Italy
Antonio di Leva	University of Turin, Italy
Vadim A. Ermolayev	Zaporozhye State University, Ukraine
Jörg Desel	Katholische Universität, Eichstätt, Germany
Ulrich Frank	University Duisburg-Essen, Germany
Doris Gälle	Alpen-Adria-Universität Klagenfurt, Austria
Marcela Genero	University of Castilla-La Mancha, Spain
Athula Ginige	University of Western Sydney, Australia
Martin Glinz	University of Zurich, Switzerland
Jon Atle Gulla	Norwegian University of Science and Technology, Norway

Jan Jürjens	The Open University, UK
Bill Karakostas	UMIST Manchester, UK
Roland Kaschek	Massey University, New Zealand
Stephen Liddle	Brigham Young University, USA
Jacques Kouloumdjian	Lyon, Insa, France
Hui Ma	Massey University, New Zealand
Zakaria Maamar	Zayed University, United Arab Emirates
Norbert Mikula	Intel Labs, Hillsboro, USA
Günther Müller Luschnat,	FAST GmbH München
Oscar Pastor	Valencia University of Technology, Spain
Sudha Ram	University of Arizona, USA
Reind van de Riet	Vrije Universiteit Amsterdam, The Netherlands
Matti Rossi	Helsinki School of Economics, Finland
Joachim Schmidt	TU Hamburg-Harburg, Germany
Michael Schrefl	University of Linz, Austria
Elmar Sinz	Universität Bamberg, Germany
Maria Semmelrock-Picej	Industriestiftungsinstitut eBusiness, Austria
Il-Yeol Song	Drexel University, Philadelphia, USA
Vijay Sugumaran	Oakland University, USA
Peter Tabeling	HPI Potsdam, Germany
Bernhard Thalheim	Christian-Albrechts-Universität Kiel, Germany
Farouk Toumani	Blaise Pascale University, France
Jos van Hillegersberg	Erasmus University, The Netherlands
Tatjana Welzer	University of Maribor, Slovenia
Carson Woo	UBC Vancouver, Canada
Jian Yang	Macquarie University Sydney, Australia

ISTA Program Committee

Witold Abramowicz	The Poznan University of Economics, Poland
Marko Anzelak	Industriestiftungsinstitut eBusiness, Austria
Stephane Bressan	National University of Singapore, Singapore
Ruth Breu	University of Innsbruck, Austria
Andrew Burton-Jones	University of British Columbia, Canada
Lois Delcambre	Portland State University, USA
Anatoly Doroshenko	Institute of Software Systems, Ukraine
Vadim Ermolayev	Zaporozhye State University, Ukraine
Alexander Gelbukh	National Polytechnic Institute, Mexico
Robert Goodwin	Flinders University, Adelaide, Australia
Remigijus Gustas	Karlstad University, Sweden
Wolfgang Hesse	Philipps University Marburg, Germany
Roland Kaschek	Massey University, New Zealand
Jörg Kerschbaumer	Alpen-Adria-Universität Klagenfurt, Austria
Myoung Ho Kim	KAIST, Korea
Kinshuk	Athabasca University, Canada

Christian Kop	Alpen-Adria-Universität Klagenfurt, Austria
Stephen W. Liddle	Brigham Young University, Provo, USA
Elisabeth Metais	Laboratory CEDRIC, Paris, France
Boris A. Novikov	St. Petersburg State University, Russia
Andreas Oberweis	University of Frankfurt, Germany
Maria E. Orlowska	Polish-Japanese Institute of IT, Warsaw, Poland
Wolfgang Reisig	Humboldt-Universität zu Berlin, Germany
Gunter Saake	Otto-von-Guericke-Universität, Magdeburg, Germany
Demetrios G. Sampson	University of Piraeus, Greece
Johannes Siedersleben	sdm München, Germany
Vladimir Shekhovtsov	National Technical University “KhPI”, Ukraine
Keng Siau	University of Nebraska-Lincoln, USA
Markus Stumptner	University of South Australia, Australia
Yuzura Tanaka	Hokkaido University, Japan
Helmut Thoma	IBM Global Services, Switzerland
Alexei Tretiakov	Massey University, New Zealand
Jürgen Vöhringer	Alpen-Adria-Universität Klagenfurt, Austria
Georg Weber	Infineon Technologies, Austria
Jože Zupančič	University of Maribor, Slovenia

Table of Contents

Keynotes

Standardizing Methodology Metamodelling and Notation: An ISO Exemplar 1
Brian Henderson-Sellers and Cesar Gonzalez-Perez

Dependable Systems – Wishful Thinking or Realistic Expectation? (Abstract) 13
Andreas Reuter

Invited Talks

Interoperability Governance for e-Government 14
Witold Abramowicz, Andrzej Bassara, Marek Wisniewski, and Pawel Zebrowski

Foundational Data Modeling and Schema Transformations for XML Data Engineering 25
Reema Al-Kamha, David W. Embley, and Stephen W. Liddle

Integration of Semantic XForms and Personal Web Services as a Tool to Bridge the Gap between Personal Desktops and Global Business Processes 37
Amin Anjomshoaa and A. Min Tjoa

A Slicing Technique for Business Processes 45
Eike Best and Astrid Rakow

Public Adoption of Digital Multimedia – Why Is It Lagging behind Expectations? (Extended Abstract) 52
Laszlo Böszörmenyi

Model-Driven Security Engineering of Service Oriented Systems 59
Ruth Breu, Michael Hafner, Frank Innerhofer-Oberperfler, and Florian Wozak

Towards Truly Flexible and Adaptive Process-Aware Information Systems 72
Peter Dadam, Manfred Reichert, Stefanie Rinderle, Martin Jurisch, Hilmar Acker, Kevin Göser, Ulrich Kreher, and Markus Lauer

From Human Knowledge to Process Models 84
Jörg Desel

Temporal Consistency of View Based Interorganizational Workflows 96
Johann Eder and Amirreza Tahamtan

Engineering Design Performance (Extended Abstract) 108
Vadim Ermolayev, Wolf-Ekkehard Matzke, and Richard Sohnius

Integration – Reflections on a Pivotal Concept for Designing and Evaluating Information Systems 111
Ulrich Frank

Aspect Based Conceptual Modelling of Web Applications 123
Athula Ginige

The Influence of Mobile Computing to Information Systems Technology (Abstract) 135
Otthein Herzog, Michael Lawo, Hendrik Witt, and Michael Boronowsky

Engineers Discovering the “Real World” – From Model-Driven to Ontology-Based Software Engineering 136
Wolfgang Hesse

Holistic Engineering of Ultra-Highspeed Mobile Information and Communication Systems 148
Matthias Jarke

A Framework for Building Mapping Operators Resolving Structural Heterogeneities 158
Gerti Kappel, Horst Kargl, Thomas Reiter, Werner Retschitzegger, Wieland Schwinger, Michael Strommer, and Manuel Wimmer

Metamodeling: Some Application Areas in Information Systems 175
Dimitris Karagiannis, Hans-Georg Fill, Peter Höfferer, and Martin Nemetz

Collaborative Model Driven Software Development for SOA-Based Systems 189
Thomas Karle and Andreas Oberweis

A Meta-model-Driven Tool Integration Development Process 201
Felix Klar, Sebastian Rose, and Andy Schürr

Information and Communication Systems for Mobile Emergency Response 213
Janine Lachner and Hermann Hellwagner

Tacit Knowledge Management: Do We Need a Re-orientation of Traditional KM Approaches? 225
Franz Lehner

Collaborative Software Development: A Case Study of Model Interdependencies 232
Peter C. Lockemann

Conceptual Modelling in an Evolving World 246
Roland T. Mittermeir

An Ontological-Based Approach to Analyze Software Production Methods 258
Óscar Pastor, Sergio España, and Arturo González

Towards a Theory of Services 271
Wolfgang Reisig

Evolving and Implanting Web-Based E-Government-Systems in Universities 282
Dirk Reiss, Bernhard Rumpe, Marvin Schulze-Quester, and Mark Stein

Facets of Media Types 296
Klaus-Dieter Schewe and Bernhard Thalheim

e-Skills and Standards – Prerequisites to Fully Exploit the Potential of ICT in Europe 306
Wolffried Stucky and Peter Weiß

ASM Foundations of Database Management 318
Bernhard Thalheim and Klaus-Dieter Schewe

Software Quality – Experiences in System Development and Operation 332
Helmut Thoma

Adaptive Macro-designing Technology for Complex Process Control Systems 340
Mykola Tkachuk

Mokum for Correctness by Design in Relation to MDA 352
Reind van de Riet

Conceptual Model – Meeting of Cultures 365
Tatjana Welzer, Zala Volčič, Boštjan Brumen, Izidor Golob, Aida Kamišalić, and Marko Hölbl

Operational Business-IT Alignment in Value Webs 371
Roel Wieringa

Towards Virtual Sales Assistants: State-of-the-Art and Research Challenges (Abstract) 379
Markus Zanker

ISTA Papers

An Investigation into Improving the Load Balance for Term-Based Partitioning 380
Ahmad Abusukhon, Mohammad Talib, and Michael P. Oakes

A UML-Rewriting Driven Architectural Proposal for Developing Adaptive Concurrent IS 393
Nasreddine Aoumeur and Gunter Saake

A Framework for Designing and Recognizing Sketch-Based Libraries for Pervasive Systems 405
Danilo Avola, Fernando Ferri, Patrizia Grifoni, and Stefano Paolozzi

A Conceptual Framework for Handling Complex Administrative Processes in E-Government 417
Stefka Dzhumalieva and Markus Helfert

Fuzzy Time Intervals for Simulating Actions 429
Vadim Ermolayev, Natalya Keberle, Wolf-Ekkehard Matzke, and Richard Sohnius

Conceptual Design of a Method to Support IS Security Investment Decisions 445
Heinz Lothar Grob, Gereon Strauch, and Christian Buddendick

A Query Language for Rational Tree Structures 457
Markus Kirchberg, Faizal Riaz-ud-Din, and Klaus-Dieter Schewe

A Learning Object Composition Model 469
Victor H. Menendez and Manuel E. Prieto

Gathering Preference Data from Restricted Natural Language 475
Johann Mitlöhner

Patterns for Screenography 484
René Noack and Bernhard Thalheim

A Model of Self-organized Criticality in Emergent Web Systems 496
Christian Russ and Mathias Lux

Towards Prototyping-Based Technology for Adaptive Software Development 508
Mykola Tkachuk, Andriy Zemlyanoy, and Rustam Gamzayev

Management of Multi-services Structures Through an Access Control Framework 519
Luigi Ubezio, Enrico Valle, and Claudia Raibulet

Contract-Based Exception Handling Process Patterns 531
Jelena Zdravkovic and Vandana Kabilan

Mobile Applications for the Academic Environment 544
Răzvan Daniel Zota, Ileana Stănescu, Emil Stănescu, Raluca Stănculescu, and Laura Stănescu

eCOMO Papers

Modeling Trust Relationships in Collaborative Engineering Projects 555
Carlo Argiolas, Nicoletta Dessì, and Mariagrazia Fugini

Loosening the Hierarchy of Cross-Company Electronic Collaboration ... 567
Christoph Schroth

(Meta-)Models, Tools and Infrastructures for Business Application Integration 579
Ralf Kutsche and Nikola Milanovic

The Impact of Structuredness on Error Probability of Process Models 585
Ralf Laue and Jan Mendling

Author Index 591

Standardizing Methodology Metamodelling and Notation: An ISO Exemplar

Brian Henderson-Sellers[1] and Cesar Gonzalez-Perez[2]

[1] Faculty of Information Technology, University of Technology, Sydney, PO Box 123, Broadway, NSW 2007, Australia
[2] Instituto de Estudios Galegos Padre Sarmiento, Consejo Superior de Investigaciones Científicas, San Roque, 2, 15704 Santiago de Compostela, Spain
brian@it.uts.edu.au, cesargon@verdewek.com

Abstract. Standardization within a discipline often reflects its maturity. Within software engineering, standardization occurs in many areas – here we focus on a recent ISO standard that has been developed for a methodology metamodel: the Software Engineering Metamodel for Development Methodologies, ISO/IEC 24744. Since its publication as a pure metamodel (represented by several UML-style class diagrams) in February 2007, a follow-on project has been established to provide a complementary notation for all the methodological elements, both within the method domain and the endeavour domain. Here, we discuss not only the technical details but also the process by which standardization occurs.

Keywords: Metamodelling, methodology, notation, standards.

1 Introduction

Maturity in a discipline is often reflected when standardization occurs. The prime international standardization body is the International Organization for Standardization or ISO, headquartered in Geneva in Switzerland.

Standardization of methodological elements within software engineering assists development teams in following the same approach leading to interoperability at all levels and an increase in their efficiency and productivity. Often companies using an ISO standard have a head start in obtaining contracts, particularly with government departments worldwide, since adherence to an ISO standard can be indicative of a quality-focussed approach to software development. While there are many ISO standards relevant to software engineering, here we will focus on just one of these: the Software Engineering Metamodel for Development Methodologies, ISO/IEC 24744 [1].

All standards go through a rigorous development process. In ISO, software engineering standards mostly fall under the purview of Subcommittee number 7 (SC7) of the Joint Technical Committee 1 of ISO and IEC (International Electro-technical Commission). Within SC7, each standardization project is attributed to one of the several working groups for development. Development stages have a six-month duration during which work is undertaken to develop an initial, raw proposal into a standard acceptable to a very wide community. At each of these six-monthly stages, a vote is taken by National Bodies before the embryonic international standard can proceed to the next stage, the voting community growing larger at each six month milestone. Final approval for this particular standard (24744) occurred after the November 2006 meeting in Kyoto and the standard was published February 13 2007.

In this paper, we focus on the technical details of the ISO/IEC 24744 International Standard. We first introduce metamodelling basics (Section 2) and then explain how these are reflected in the International Standard itself (Section 3). In Section 4, we go a little beyond the published standard to describe a notation that is currently the topic of a New Work Item (NWI) within SC7. This, in due course, is aimed at created an addendum to 24744 that will describe a recommended notation for describing methods and processes conformant to the metamodel in the existing standard.

2 Metamodelling Basics

Metamodelling is cognitively challenging and often ill understood (see examples in [2]). A metamodel is a model of models [3,4] where a model is *a statement about a given subject under study (SUS), expressed in a given language* [5]. The SUS that is the focus of our current study is that of methodologies so we are concerned with metamodels that are models of methodologies. These models provide the underpinning (quasi-formalism) for methodologies that exist in the real world e.g. the methodology used by a particular company on their projects (or endeavours in the wider sense). The relationship between a model and a metamodel is thus that the metamodel elements *represent* the model elements [6]. Together, the elements in the metamodel are the modelling language that can be used to describe such conformant models.

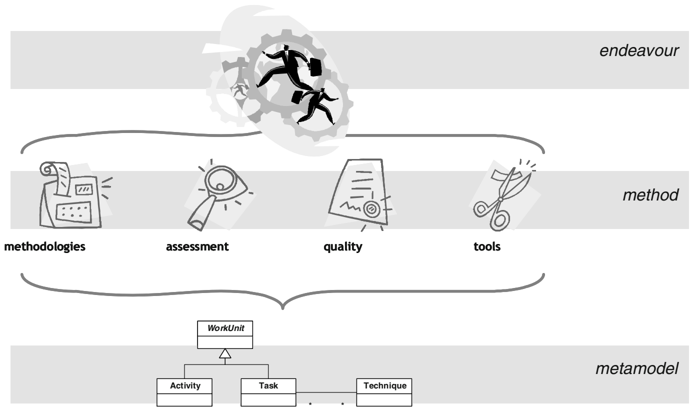


Fig. 1. Three layer architecture of recent standards (after [7])

This suggests some sort of layering or multi-domain representation – the metamodel domain, the methodology domain and, hinted at above, the endeavour domain. The metamodel domain is usually composed of standardized conceptual constructs, the method domain contains real-world methodology elements (methods, tools, coding standards) while the endeavour domain represents actual processes in use by the people on a particular project/endeavour (Figure 1). Typically, standardization occurs in the metamodel domain.

3 An Exemplar – The ISO/IEC 24744 International Standard

ISO/IEC 24744 [1] is an International Standard that defines a metamodel for development methodologies. Although it is geared towards *software* development methodologies, there is nothing in it that prevents it from being applied to systems development methodologies or even other areas. The standard exists in the metamodel domain of Figure 1 and contains a number of elements that model entities in the method domain *plus* a number of element that model entities in the endeavour domain. This is unlike other method/process-focussed standards that only model entities in the method domain.

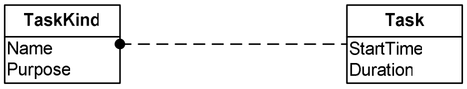


Fig. 2. Task/TaskKind powertype pattern in ISO/IEC 24744 (after [8])

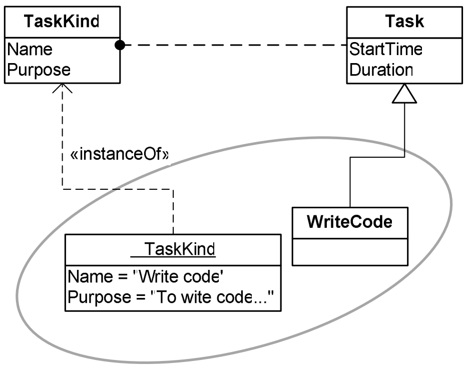


Fig. 3. The "instantiation" of a powertype pattern. A regular object is instantiated from the TaskKind class and a regular class is obtained by subtyping the Task class. Together these form a clabject (depicted by the ellipse) (after [8]).

To accomplish this duality, as discussed in [8] in more detail, ISO/IEC 24744 eschews the strict metamodelling paradigm and the well-known instantiation-related, multiple layering architecture of the Object Management Group and introduces the powertype concept of Odell [9] as a core model element in the metamodel. Pairs of concepts, one representing entities in the method domain and one representing entities in the endeavour domain, are abstracted to form powertype patterns [10] (Figure 2). The endeavour-focussed element (Task in Figure 2) represents elements (actual tasks) in the endeavour domain, while the method-domain element is modelled, in the example, by a conceptual class called TaskKind that represents all kinds of tasks that could possibly exist in the method domain. In the method domain, an instance of

TaskKind is also a subtype of Task (where TaskKind and Task are both in the metamodel domain) (Figure 3). This means it has both class-like and object-like characteristic. Such an entity has been called a "clabject" by Atkinson [11].

ISO/IEC 24744 makes wide usage of clabjects and powertype patterns. Its scope includes work units, work products, producers, stages and model units (for details see relevant subsection in Section 4). The overall architecture of ISO/IEC 24744 is shown in Figure 4. Powertype patterns are used for subtypes of Template and Endeavour elements, whilst regular instantiation semantics are sufficient for the subtypes of Resource. The differentiation between templates and resources is practical, with clear semantics (see also [12]). Nevertheless, it is appropriate for both template and resource concepts to be implemented in the metamodel equally, as regular classes. It is their usage that is different. Furthermore, we must emphasize that the instances of resources and templates (collectively called methodology elements) reside in the Method domain, in contrast to the instances of endeavour elements, which reside in the Endeavour domain.

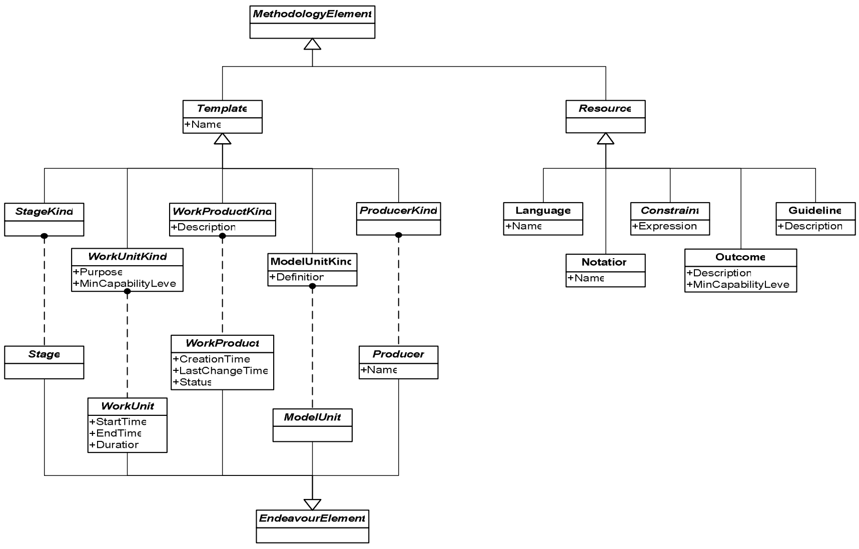


Fig. 4. Overall architecture of ISO/IEC 24744 (after [1])

4 Adding a Notation

While the published International Standard [1] does not contain any recommended notation (but just an abstract syntax and semi-formal semantics), at the time of writing, a new project under Working Group 19 of the SC7 subcommittee of ISO, has been commenced to standardize a notation especially designed to depict ISO/IEC 24744 concepts. Here is a summary of the progress to date on this soon-to-be-standardized set of symbols.

The proposed notation is mainly graphical and supports most of the template concepts found in 24744 (left-hand side of Figure 4).

Although the metamodel of ISO/IEC 24744 contains classes that represent concepts from the Method domain and classes that represent concepts from the Endeavour domain, the notation so far gives more support for the former. It has been designed to depict concepts from the Method domain in order to help the methodologist or method engineer represent method fragments and complete methodologies. The notation does not yet comprehensively cover the Endeavour domain, i.e. it is not capable of depicting concepts pertaining to specific enactments of any methodology. There is an exception to this, namely enactment diagrams. Enhancement of the notation presented here to cover Endeavour domain concepts is for future consideration in the ISO standardization process.

The notation has been designed to be easy to draw by hand as well as using a drawing package on a computer. Special care has been taken to follow semiotic principles, as suggested in [13, 14] in choosing symbols that convey the underlying concept, at least in most situations and to readers of most cultures and backgrounds. In addition, the symbols adopted by the notation exhibit visual resemblance (based on shapes and colours) to each other that mimic the structural relationships of the underlying concepts in the metamodel, establishing common "visual themes" for closely related concepts. Colour is extensively used by this notation, since it helps identify symbols and symbol patterns with ease when displayed on a computer display or a colour printout. However, care has been taken to guarantee that greyscale and black and white versions of the same symbols are perfectly readable and identifiable. In this regard, colour does enhance diagram readability but can be avoided without great loss.

This notation introduces the novel concept of *abstract symbols*, i.e. symbols that depict instances of abstract classes. In principle, most notations only include symbols to depict instances of concrete classes, since abstract classes do not have direct instances. However, it is suggested that in some scenarios it is convenient to represent an entity in a diagram for which only the abstract type is known. For example, consider the case where a work product kind representing a certain system must be depicted in a diagram. A notation with only concrete symbols would force the designer to choose a specific concrete type of work product (document, model, software item etc.) in order to depict it. This notation thus includes an "abstract work product kind" symbol that allows the designer to depict the above-mentioned system without specifying whether it is software, hardware, composite etc. Abstract symbols usually consist of the simple shape from which all the concrete symbols in a visual theme are generated.

4.1 Stages

A stage is a managed time frame within an endeavour. Stages are partitioned into stage kinds by the StageKind class (Figure 4). In addition to the abstract StageWithDurationKind, three of its subtypes are covered by this notation: TimeCycleKind, PhaseKind and BuildKind. These are represented by broad symbols that can contain other elements. A rectilinear theme has been chosen. Colours, when used, for all these symbols belong to the blue-purple range (Figure 5).

The symbol used to depict a time cycle kind is composed of two navy blue horizontal brackets with their right-hand side end bent outwards, simulating a truncated arrow head. These brackets delimit a quasi-rectangular area within which symbols for other stage kinds can be shown. This symbol tries to convey the meaning that a time cycle kind comprises a collection of other stage kinds - hence the bracket analogy. A similar argument underpins the shape for PhaseKind and the more iteratively focussed BuildKind with its double point, resembling dual arrow heads. It too can serve as a container for other elements, which can be shown inside it.

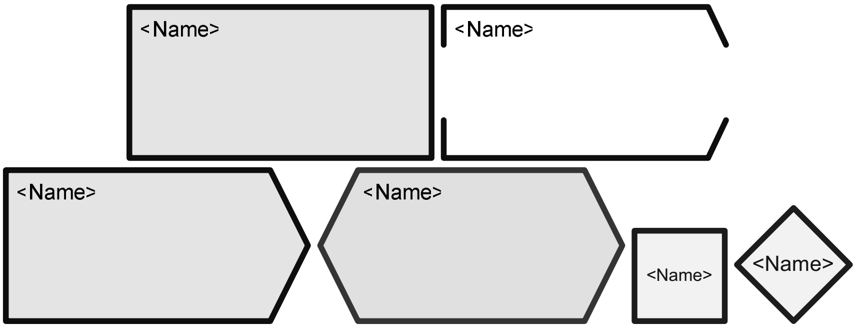


Fig. 5. The "stage family" has six symbols representing StageWithDurationKind, TimeCycleKind, PhaseKind, BuildKind, InstantaneousStageKind and MilestoneKind

In contrast, an instantaneous stage is a managed point in time within an endeavour. InstantaneousStageKind is thus another abstract class, depicted by an abstract symbol (a square). This symbol tries to convey the idea of a point in time, hence the similarity with other stage-related symbols (overall rectangular shape) but a smaller area. Since instantaneous stages are points in time rather than time spans, no other symbols need to be shown inside this one.

The symbol used to depict a milestone kind is a small square rotated 45 degrees. This symbol tries to convey the idea of an event-marking point in time, hence the similarity with other stage-related symbols (points facing left and right) but a smaller area. It also resembles the symbol used by Microsoft PowerPoint™ and other project management software tools to depict milestones. Since milestones are points in time rather than time spans, no other symbols will need to be shown inside this one.

Temporality is represented by the "pointed" nature of the left and/or right sides of these symbols – other than the two symbols (rectangle and square) representing the abstract supertypes (StageWithDurationKind and InstantaneousStageKind).

4.2 Work Units

A work unit is a job performed, or intended to be performed, within an endeavour. Work units are partitioned into work unit kinds by the WorkUnitKind class according to their purpose within the endeavour (Figure 4).Three subtypes of WorkUnitKind are covered by this notation: ProcessKind, TaskKind and TechniqueKind. None of these concepts are involved in whole/part relationships that may need nesting of symbols, so the symbols chosen to depict them are basic curvilinear shapes and easily resizeable to accommodate long names or abbreviations. However, since work unit symbols often occur on the same diagram, as well as their contrasting shape outlines, an additional colour coding can be added so that the user can readily discriminate between them in process diagrams that support colour (see Figure 6).

A process is a large-grained work unit that operates within a given area of expertise within the endeavour. The symbol used to depict a process kind is a rounded rectangle or “roundangle”. When colour is available, line colour is medium green and fill colour is light green. In some contrast, a task is a small-grained work unit that focuses on what must be done in order to achieve a given purpose within the endeavour. The symbol used to depict a task kind is an ellipse with more intense shades of green than ProcessKind. TechniqueKind, shown with similar colours to ProcessKind, details *how* tasks are to be accomplished. The minimum capability level of work unit kinds is optionally shown inside a circle toward the top or left as indicated (“n” in the figure).

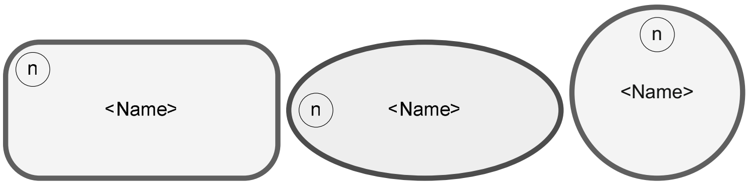


Fig. 6. The WorkUnitKind family of icon shapes and colours: ProcessKind, TaskKind and TechniqueKind

4.3 Work Products

A work product is an artefact of interest for the endeavour. Work products are partitioned into work product kinds by the WorkProductKind class according to the nature of their contents and the intention behind their usage. Five subtypes of WorkProductKind are covered by this notation: DocumentKind, ModelKind, SoftwareItemKind, HardwareItemKind and CompositeWorkProductKind. All of them are represented by tall symbols with colours in the red-pink range. The former four correspond to “simple” work products, and therefore are represented by reasonably simple rectangular shapes, but with different adornments. CompositeWorkProductKind, on the other hand, uses a symbol that tries to convey a sensation of depth to represent multiplicity.

The most general is the abstract symbol for WorkProductKind itself. This is depicted by a vertically-oriented rectangle. Line colour is red and fill colour is pale red/pink (Figure 7). This symbol captures the overall shape used for the other (concrete) types of work product kinds. A document is a durable depiction of a fragment of reality. Documents are partitioned into document kinds by the DocumentKind class according to their structure, type of content and purpose. The symbol used to depict a document kind is a vertical rectangle with a dog-eared top right corner. This symbol depicts a sheet of paper.

A model is an abstract representation of some subject that acts as the subject’s surrogate for some well-defined purpose. Models are partitioned into model kinds by the ModelKind class according to their focus, purpose and level of abstraction. The symbol currently used to depict a model kind consists of a vertically-stacked pair of horizontal rectangles linked by a short vertical line. This symbol is reminiscent of two nodes and an arc in a model.

A software item is a piece of software of interest to the endeavour. Software items are partitioned into software item kinds by the SoftwareItemKind class. The symbol currently used to depict a software item kind is a square with a bottom left chamfered corner and a smaller rectangle adjacent to the bottom side. This symbol depicts a floppy disk, commonly used to represent the concept of software

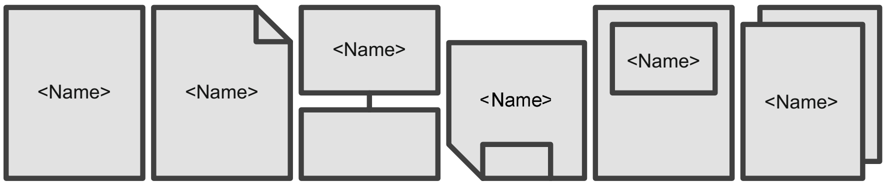


Fig. 7. Symbols for WorkProductKind, DocumentKind, ModelKind, SoftwareItemKind, HardwareItemKind and CompositeWorkProductKind, respectively

A hardware item is a piece of hardware of interest to the endeavour. Hardware items are partitioned into hardware item kinds by the HardwareItemKind class according to their mechanical and electronic characteristics, requirements and features. The symbol currently used to depict a hardware item kind is a vertical rectangle with a small rectangle nested in its upper half.

A composite work product is a work product composed of other work products. Composite work products are partitioned into composite work product kinds by the CompositeWorkProductKind class. The symbol used to depict a composite work product kind is a pair of vertical rectangles "stacked" along the z-axis, simulating perspective. This symbol tries to convey the idea of composition, i.e. a work product made of multiple components.

4.4 Producers

A producer is an agent that has the responsibility for executing work units. Producers are partitioned into producer kinds by the ProducerKind class according to their area of expertise. Three subtypes of ProducerKind are covered by this notation: TeamKind, RoleKind and ToolKind. All of them are based on a schematic depiction of a torso using half an ellipse with colours in the orange-yellow range (Figure 8).

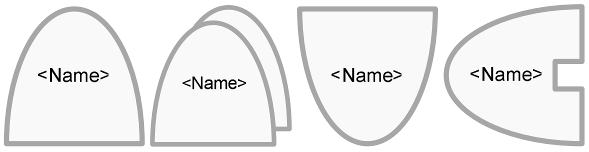


Fig. 8. Symbols for ProducerKind, TeamKind, RoleKind and ToolKind, respectively

The most generic is the abstract symbol for the superclass, ProducerKind. The symbol used to depict a work product kind is half an ellipse standing on its flat side.

A team is an organized set of producers that collectively focus on common work units. Teams are partitioned into team kinds by the TeamKind class according to their responsibilities. The symbol used to depict a team kind is a pair of half ellipses standing on their flat side and "stacked" along the z-axis, simulating perspective. This symbol depicts multiple human torsos, and hence the team.

A role is a collection of responsibilities that a producer can take. Roles are partitioned into role kinds by the RoleKind class according to the involved responsibilities. The symbol used to depict a role kind is half an ellipse standing on its round tip – a "mask" symbol, used to show a person who is playing a (theatrical) role.

A tool is an instrument that helps another producer to execute its responsibilities in an automated way. Tools are partitioned into tool kinds by the ToolKind class. The symbol used to depict a tool kind is a vertical half ellipse with its round tip pointing leftwards and a square indentation in the centre of its flat side - depicting the head of an open-end wrench or spanner, a prototypical tool.

4.5 Other Groups of Symbols

There are other supporting symbols proposed as part of the ISO/IEC 24744 notation, including Actions and other similar relationships linking pairs of methodology elements. For example, an action is a usage event performed by a task upon a work product. Actions are partitioned into action kinds by the ActionKind class according to their cause (the specific task kind), their subject (the specific work product kind) and their type of usage (such as creation, modification etc.). An action is depicted (Figure 9) by an arc that goes from the symbol for the associated task kind to the symbol for the associated work product kind. The arc is a plain line with a small circle on the end of the work product kind. The type of usage is specified inside the small circle using an abbreviation ("t" in the figure – there are various options not listed here – see [15]). The role of the work product kind for this particular action kind, if any, can be shown close to the work product end. The optionality of the action kind can be shown using a deontic marker ("d" in the figure).

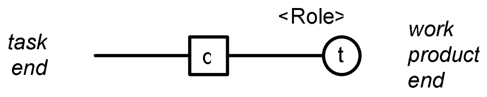


Fig. 9. Notation for actions

Similarly, a work performance is an assignment and responsibility association between a particular producer and a particular work unit. Work performances are partitioned into work performance kinds by the WorkPerformanceKind class according to the purpose of their inherent assignment and responsibility association. The symbol used to depict a work performance kind is an arc that goes from the symbol for the associated producer kind to the symbol for the associated work unit kind. A plain arc is used. The recommended assignment of the work performance kind can also be shown using a deontic marker.

4.6 Diagram Types

As with any kind of modelling using a graphical notation, various views of the total model can be made. In each a different aspect of the system is stressed. With ISO/IEC 24744, we can identify the need for the following diagram types:

- Lifecycle diagrams, which represent the overall structure of a method. They can depict both the temporal aspects (reading them left to right as time passes) and the content aspects (Figure 10).

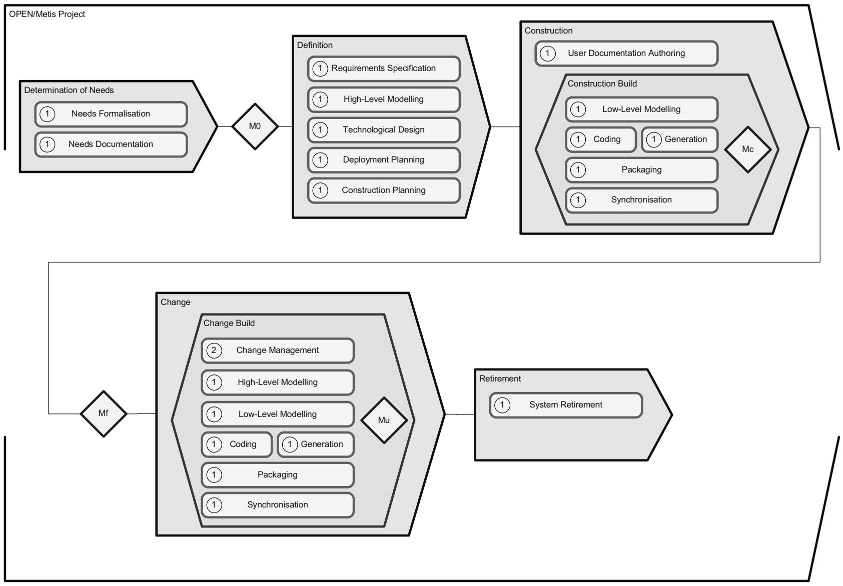


Fig. 10. A lifecycle diagram showing the content structure as well as the temporal structure of a method. Notice how stage kinds can contain both nested stage kinds (such as "Construction Build" inside "Construction") and process kinds ("User Documentation Authoring").

- Process diagrams, which describe the details of the processes used in a method. These may include the relationships between process kinds, the links between process kinds and task kinds, and the producer kinds assigned to the work unit kinds by the appropriate work performance kinds. Process diagrams, therefore, focus strongly on the "*what*" aspect of a method, being also able to show the "*who*" (Figure 11).
- Action diagrams (Figure 12), which show the usage interactions between task kinds and work product kinds. Action diagrams represent the usage that task kinds make of work product kinds. These usages are basically modelled as action kinds in ISO/IEC 24744. Action diagrams, therefore, serve to visualize the bridge between the process and product sides of a method.
- Task-Technique diagrams, which some developers may find useful to formalize which techniques are useful for which tasks and vice versa. Such diagrams would provide an alternative to the more well-established textual descriptions such as a deontic matrix [17].

The notation proposed so far for ISO/IEC 24744 focusses on representing the method domain. In the endeavour domain, diagram types are also needed. The only one so far tentatively proposed is the "enactment diagram", which represent a specific enactment of a method (or part of a method) and its relationship to the method specification.

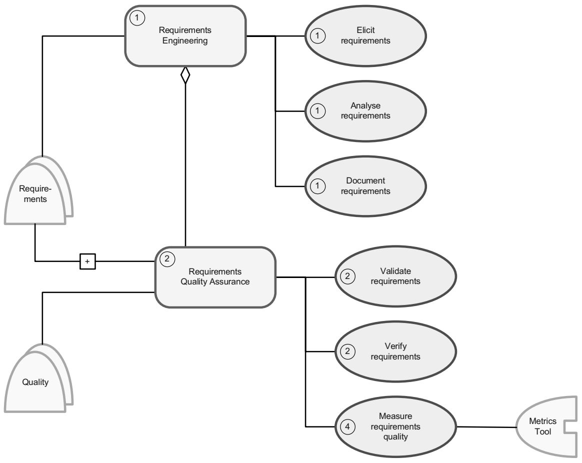


Fig. 11. A process diagram showing the details of the "Requirements Engineering" and "Requirements Quality Assurance" processes

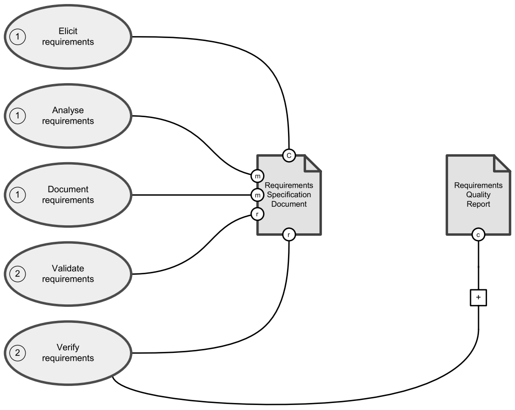


Fig. 12. An action diagram showing how requirements-related task kinds interact with requirements-related work products

5 To the Future

The ISO process for the notation to accompany the metamodel [1] will continue over the next several years with a likely final publication in 2009/10. During that process, elements of shape, topology, colour etc. may well change from those suggested above. In addition, it is anticipated that diagram types will be suggested to support the endeavour domain. At each six-monthly stage, international feedback is sought via the National Bodies that each have voting rights on ISO/IEC standards under SC7. Best practice is achieved by exposure to a wide-ranging audience, initially just software engineering experts in each country worldwide and then over a wider ISO electorate.

References

1. ISO/IEC: 24744: Software Engineering - Metamodel for Development Methodologies. International Organization for Standardization/International Electrotechnical Commission, Geneva (2007)
2. Henderson-Sellers, B.: On the Challenges of Correctly Using Metamodels in Method Engineering. In: Fujita, H., Pisanelli, D. (eds.) New Trends in Software Methodologies, Tools and Techniques. Proceedings of the sixth SoMeT 2007, pp. 3–35. IOS Press, Amsterdam (2007)
3. Flatscher, R.G.: Metamodeling in EIA/CDIF – Meta-metamodel and Metamodels. ACM Trans. Modeling and Computer Simulation 12(4), 322–342 (2002)
4. OMG: MDA Guide Version 1.0.1. OMG document omg/03-06-01 (2003)
5. Gonzalez-Perez, C., Henderson-Sellers, B.: Modelling Software Development Methodologies: A Conceptual Foundation. J. Systems and Software 80(11), 1778–1796 (2007)
6. Seidewitz, E.: What do Models Mean? IEEE Software 20, 26–31 (2003)
7. Henderson-Sellers, B.: Method Engineering: Theory and Practice. In: Karagiannis, D., Mayr, H.C. (eds.) Information Systems Technology and its Applications. 5th International Conference ISTA 2006, Gesellschaft für Informatik, Bonn. Lecture Notes in Informatics (LNI), vol. P-84, pp. 13–23 (2006)
8. Gonzalez-Perez, C.: Supporting Situational Method Engineering with ISO/IEC 24744 and the Work Product Pool Approach. In: Ralyté, J., Brinkkemper, S., Henderson-Sellers, B. (eds.) Situational Method Engineering: Fundamentals and Experiences. Proceedings of the IFIP WG 8.1 Working Conference, Geneva, Switzerland, September 12-14, 2007. IFIP Series, vol. 244, pp. 7–18. Springer, Boston (2007)
9. Odell, J.: Power Types. Journal of Object-Oriented Programming 7(2), 8–12 (1994)
10. Henderson-Sellers, B., Gonzalez-Perez, C.: Connecting Powertypes and Stereotypes. J. Object Technol. 4(7), 83–96 (2005)
11. Atkinson, C.: Metamodelling for Distributed Object Environments. In: First Int. Enterprise Distributed Object Computing Workshop (EDOC 1997), Brisbane, Australia (1997)
12. Gonzalez-Perez, C., Henderson-Sellers, B.: Templates and Resources in Software Development Methodologies. J. Obj. Technol. 4(4), 173–190 (2005)
13. Constantine, L.L., Henderson-Sellers, B.: Notation Matters: Part 1 - Framing the Issues. Report on Object Analysis and Design 2(3), 25–29 (1995)
14. Constantine, L.L., Henderson-Sellers, B.: Notation Matters: Part 2 - Applying the Principles. Report on Object Analysis and Design 2(4), 20–23 (1995)
15. Gonzalez-Perez, C., Henderson-Sellers, B.: Metamodelling for Software Engineering. J. Wiley & Sons, Chichester (2008)
16. Henderson-Sellers, B., Edwards, J.M.: BOOKTWO of Object-Oriented Knowledge: The Working Object, p. 594. Prentice-Hall, Sydney, Australia (1994)

Dependable Systems – Wishful Thinking or Realistic Expectation?

Andreas Reuter

European Media Laboratory GmbH, Heidelberg
University of Kaiserslautern

Abstract. A system is dependable if you can trust it to work. This seems to a completely obvious, almost trivial definition. But the question of what it means for a system to "work" is influenced by the type of system and the perspective of the user – among other things. Depending on the function, reliability can be an important criterion, but in other cases it can be throughput, response time, accuracy of computations, immunity against malicious attacks –this list could be continued for a while.

In addition to that, the discussion of dependability does not stop at the boundaries of a technical system. The interaction between (human) actors and a technical system creates a new, larger system, for which dependability needs to be defined, too. Thus dependability is not static property, determined by a set of design decisions, but a behavioural trait that strongly depends on external influences. In particular, different users groups can have different perceptions of what they regard as the system's dependability.

The presentation will start out by precisely defining dependability as a user-dependent phenomenon. Based on this, we will explore the technical means available for supporting the implementation of highly dependable systems by adequate methods and corresponding tools.

Interoperability Governance for e-Government

Witold Abramowicz, Andrzej Bassara, Marek Wisniewski, and Pawel Zebrowski

Department of Information Systems,
Poznan University of Economics,
ul. Niepodleglosci 10, Poznan, Poland
W.Abramowicz@kie.ae.poznan.pl

Abstract. The problems concerning interoperability have been addressed in the literature a number of times. Unfortunately, most of existing interoperability frameworks have been developed in a separation from existing approaches for architecture development. On the other hand, enterprise architecture frameworks and architecture governance in general are not directly concerned with interoperability issues.

In this paper we investigate enterprise archtecture frameworks and present recommendations for developing an interopeable architectures in the sector of public administration. As a result, parties involved in the application of interoperability-intensive IT solutions can effectively manage the scope, results and potential benefits.

Keywords: Governance, architecture, e-government, interoperability.

1 Introduction

Interoperability defined as ,,the ability of information and communication technology (ICT) systems and of the business processes they support to exchange data and to enable sharing of information and knowledge" [2] plays crucial role in modern organizations. Unfortunately, reasons for developing interoperable system are inherently different for business and government institutions. Therefore, special care must be taken during creation of interoperability: guidelines, recommendations or policies for governmental institutions.

In business, the need for seamless communication is driven by business goals which are usually defined towards increasing company value by permanent self improvement, by searching for and creating opportunities. Sample business goals that are, for instance, focused on shortening time to market or setting up more robust product distribution system require an ability to dynamically integrate organizations, their business processes and information systems. Interoperability is therefore an intrinsic and necessary characteristic of a modern business.

In governmental institutions, the interoperability is driven by increasing demand on better quality of service. On the one hand, the need for interoperability is imposed by business and citizens which are obliged to contact with governmental bodies. On the other, government institutions (or more general, public administration) pursuit for better way of providing its services and execute public tasks.

One of principles of modern government is to provide public services according to the needs of citizens and businesses. As a result, huge effort is put into policies and plans that emphasize the importance of one-stop e-governments. In such joined-up governments citizens and businesses expect to be offered with services that handle with cases that need aggregated information and effort from different public authorities. Furthermore, it brings also the demand to reuse existing information not only within the same but also in different administration divisions.

The rationale comes from the real life. In case of businesses, the most willingly provided example is the compulsion to visit several agencies to supply the same basic information in order to set up a business. For citizens, very annoying is an obligation to pop at several authorities to inform them on change in the address.

Provision of integrated services via one stop shops requires much more collaboration within and across public authorities. In the effort to modernize public authorities and the way public services are provided, the interoperability plays a significant role.

2 Interoperability

The assurance of interoperability requires addressing vast number of issues. The common pitfall is to think of interoperability only in terms of technical issues. Interoperability should address many mutual dependent aspects. In literature one may find several frameworks that focus on different levels of interoperability. One of the most recognized and well accepted in Europe's public administrations is the approach presented in EIF 1.0 [4]. According to this view, interoperability should be perceived on three layers (Fig. 1) that, apart from technical issues, distinguishes the semantic and organizational domains.

Organizational Interoperability – The organizational structures of Europe's administrations vary significantly. Furthermore, they tend to have closed structures that do not easily allow for operating with external units. The need for the reorganization of internal structures arises.

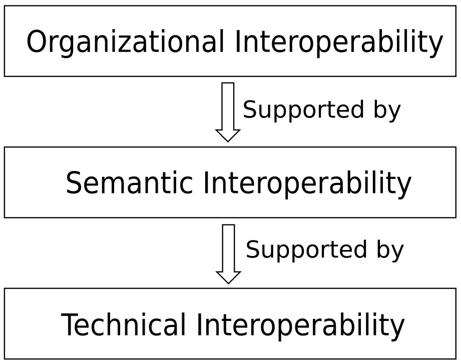

Organizational Interoperability
Supported by
Semantic Interoperability
Supported by
Technical Interoperability

Fig. 1. Interoperability Layers [5]

Organizational interoperability is about the preparation of organizational structures and business processes to be able to coexist with the increased need for seamless communication. Organizational structures needs to be shifted from paper-centric to information-centric structures.

Semantic Interoperability – Semantic interoperability is to ensure that the precise meaning of exchanged information is understandable by any other application not initially developed for this purpose [3]. In this definition, the focus is raised from data to information which means that systems, not only have to know how to handle a specific piece of data, but also need to understand what this data is about.

Dealing with semantic interoperability means that one has to consider problems of structural and semantic nature. The structural problems are due to the variety of model representations, whereas semantic problems have to do with incoherency in meaning. Semantic interoperability issues have been widely addressed in the activities related to the Semantic Web vision. The goal of Semantic Web is to enable an understanding of data through the introduction of appropriate infrastructure, the common and shared conceptualization (ontologies) and intelligent agents [1].

Technical Interoperability – Technical interoperability in e-Government applications requires addressing issues on several distinct levels. Each of these levels is a standardization effort on its own. Web services standardize the heterogeneous hardware, operating systems and communication protocols. The standardization effort allows a certain degree of abstraction to the technical interoperability stack. Data formats are standardized by the efforts such as, XML SOAP or Web services again. There is however, no agreement on the representation languages, although, OWL and WSMO family languages gaining the most attention. In the past, parties had to conform to the technical details of each interaction. Nowadays, some degree of technological heterogeneity is assumed, yet still general frameworks need to be established.

This well known three-layer interoperability model has been further revised in Gartner's report providing enhancements to European Interoperability Framework in order to allow faster deployment of public services. The new framework (EIF 2.0) [7] provides new Interoperability Reference Model that maps EIF v1.0 dimensions of interoperability. Gartner points out that organizational dimension of interoperability lacks explicitly distinguished notion of processes. Furthermore, the EIF 2.0 is to provide more details in technical dimension of interoperability, especially in new data, application and hardware sub-layers.

Several other studies indicate a need for interoperability governance dimension. According to EPAN [6], governance of interoperability refers to coordination and alignment of business processes and various aspects of information architectures. These include ownership, definition, development, maintenance, monitoring and promotion of standards, protocols, policies and technologies defined within interoperability architecture.

3 Governance

The problem of interoperability is often simplified in two dimensions. Firstly, its outcome is defined, for instance, as a set of standards that systems must be designed to support. Secondly, it is often perceived as a single action or process. Unfortunately, as highlighted in the introduction, to achieve interoperability in dynamic and changing environment one must establish a governance process.

Regardless actual subject of the governance, it is concerned with definition of:

– rules, processes and procedures guiding strategic decisions;
– roles, relationships and responsibilities of people/organizations involved;
– objective evaluation metrics of performance.

Enterprise governance is a broad concept that spans across much of the enterprise assets. As a general rule, enterprise governance can be perceived at the four distinct levels (see Fig. 2):

Corporate Governance – constitutes the highest level of governance. Corporate governance deals with guiding the enterprise at the corporate level, including motivation system or work schemes (e.g. Management by Objectives (MBO) system).

Technology Governance – deals with the both tangible and intangible technology assets of an enterprise. The division between technology and IT assets in the enterprises varies. Especially in domains where IT technology is not pervasive, the technology governance gains an important role.

IT Governance – is a subset of technology governance. In enterprises with high IT pervasiveness, it is extracted from the technology governance. Contrary to other technological problems, issues related to IT often exceeds technical competences of a board of management. Therefore, comptetences are

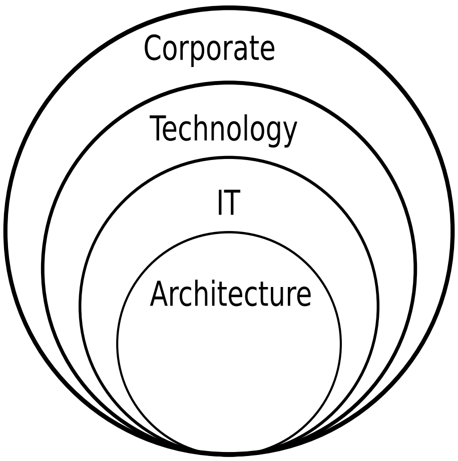


Fig. 2. Governance Levels

delegated. In consequence, key and strategic decisions regarding IT infrastructure are undertaken by IT analysts what in turn leads to the situation where this infrastructure is not fully aligned with the business and their influence on achieving the corporate goals are unknown.

Architecture Governance – IT architecture is a set of building blocks that specify the services required by the business and the model of dependencies among building blocks. Additionally it should contain the list of architectural decisions that influenced the choice of forming building blocks.

Architecture governance is a set of practices and methods to allow for the management of enterprise architectures. An architecture governance framework should provide the means to create, monitor, implement and evolve IT architectures. Frameworks such as TOGAF [8], and in particular the TOGAF ADM method provide the architecture development cycle that aids the IT architect with the task of architecture governance.

Architecture Governance may be treated as part of IT governance (as depicted above). However, care must be taken when interpreting this relation. Architecture is not only data, technology and information systems. Architecture is foremost organization, its goals and only then applications supporting these goals.

4 Interoperability Governance

From the above descriptions of various levels of governance it should be already clear, that the issue of interoperability is integral and important part of architecture governance. If architecture governance is to ensure that existing systems support business goals and this support will continue in the changing environment, then the enterprise architecture should not only define structure and behavior of the organization and its information systems, but most importantly should prove their readiness to interoperate.

It cannot be absolutely stated, that this information is never present in enterprise architectures. It is usually scattered among multiple architecture views, is imprecise or incomplete. Quick analysis of available architecture frameworks and their views definition shows that, maybe with exception of FEA [9], neither of them is even partially focused on this issue. It is not easy to use architecture description prepared according to these views in order to assess the ability of an organization and its information systems to interoperate with external bodies.

We will support our argument with the Zachman Framework [11], which for the vast number of enterprise architecture professionals is de facto standard for documenting the architecture. It has one appealing feature, due to its definition, it is often used to prove the completeness (or at least to show the scope) of architecture description. For instance, one may define which cells of the net are covered by which documents or models.

The Zachman Framework is organized around the concept of perspectives and aspects (Fig. 3). Each perspective defines an abstraction layer over the whole architecture. Such an abstraction layer provides only details which are of special

	What (Data)	How (Function)	Where (Locations)	Who (People)	When (Time)	Who (Motivation)
Scope	List of things important to the business	List of processes that the business performs	List of locations in which the business operates	List of organizations important to the business	List of things important to the business	List of things important to the business
Enterprise Model	e.g. Semantic Model	e.g. Business Process Model	e.g. Business Logistics System	e.g. Workflow Model	e.g. Master Schedule	e.g. Business Plan
System Model	e.g. Logical Data Model	e.g. Application Architecture	e.g. Distributed System Architecture	e.g. Human Interface Architecture	e.g. Process Structure	e.g. Business Rule Model
Technology Model	e.g. Physical Data	e.g. System Design	e.g. Technology Architecture	e.g. Presentation Architecture	e.g. Control Structure	e.g. Rule Design
Detailed Representation	e.g. Data Definition	e.g. Program	e.g. Network Architecture	e.g. Security Architecture	e.g. Timing Definition	e.g. Rule Definition
Functioning System	e.g. Data	e.g. Function	e.g. Network	e.g. Organization	e.g. Schedule	e.g. Strategy

Fig. 3. Zachman Framework. Available at: http://www.zifa.com/

importance to certain group of stakeholders. Usually perspectives of planner (business), owner, designer, builder and implementer are used.

The aspect on the other hand limits the architecture view only to certain concerns, which are important from design process point of view. These aspects include: data, function, network, people, time and motivation.

he architecture view is defined as an intersection of a perspective and an aspect. For instance, a view may be defined as a designer perspective on data, which, depending on the application of the architecture work, may be: a logical data model, a class diagram or a documentlogical definition.

Assuming that interoperability plays a central role in the enterprise architecture, then it should be possible to answer interoperability-oriented questions based exclusively on defined views. For example:

- Are two different systems administered by two different organizations interoperable?
- What is the highest level of interoperability they can achieve?
- What is the effort required to achieved demanded level of interoperability?
- How to govern the evolution of information systems in order to achieve interoperability?

The answer to these questions should be present in six views (Fig. 4). We are concerned only with two aspects of the whole design process: data (what?) and function (how?). Of course this limitation is highly oversimplified, while networking or timing aspects are also important. They do not impose, however, any significance challenge compared to data and function.

We also limit our attention to following perspectives: owner (business model), designer (system model) and builder (technology model). As it will become clear in the following subsections, they may be tightly connected with the already identified interoperability levels.

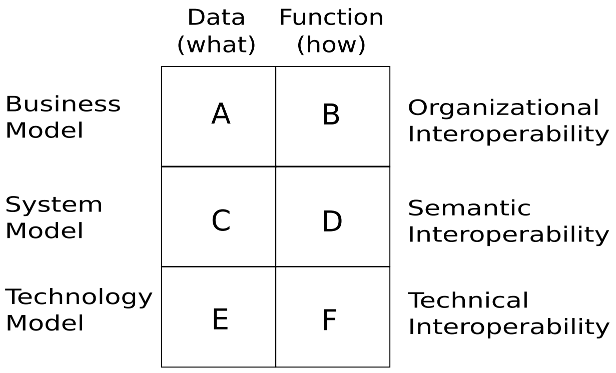

Data (what)
Function (how)
Business Model
System Model
Technology Model
A
B
C
D
E
F
Organizational Interoperability
Semantic Interoperability
Technical Interoperability

Fig. 4. Part of Zachman Framework with associated Interoperability Levels

4.1 Organizational Interoperability

Organizational interoperability is dependent on a common understanding of business processes (at least their relevant parts). Under this notion of common understanding we infer that all parties involved agree on definition of workflow including definition of services to be offered, roles to be undertaken and data to be processed.

In business environment, definition of business process is often considered as a source of company competitive advantage and due to its value is kept secret. There are no libraries of available business processes. Publicly available processes definitions exist only for trivial cases. Companies coin their processes related to the core business, through years of gradual improvements through reengineering.

The situation is, or at least should be, drastically different in public administration. Processes that govern administration are imposed by law. It is law that defines what services, to whom, by whom should be delivered. Often legislative acts define how these services should be implemented presenting de facto administrative process definition.

Of course, these definitions are not properly formalized (we would like to see BPMN or EPC diagrams, not a law jargon). Often there is also a certain degree of allowed interpretation. Therefore, public administration institutions provide their own formalized definitions, which is not a bad situation, as long as these definitions are not protected and are able to interoperate with the process definitions from commercial sector. Even if created by local administration unit, administrative processes belong to a public domain and should be made publicly available.

This is not only the matter of law but also a common sense. Business processes in public administration are characterized by very high level of recurrence. A business process defined for one organization unit or for interchange of processes between two organization units may be often successfully applied without any modification for other organization units at the same level.

Let us take into consideration the process of Personal Income Tax (PIT) forms submission. In Poland, such form may be submitted to one of a few hundreds of local tax offices. It is pointless to define such a process from scratch for every office. It is also unreasonable to whole enterprise architecture for each office – business, application and data architectures shouldn't exhibit large variations among different units.

Therefore, there is a strong need for a centralized repository of architectural resources (models, documents, recommendations, best practices etc.). At the level of organizational interoperability, the repository should be arranged around the concept of service. Processes which may be defined as a flow of services invocations should also be exposed as services.

The source of information on services should be twofold. Definitions may be specified globally or they may be submitted for approval by any party that uses the repository. After approval such a definition becomes publicly available.

In order to promote reusability each service description should be convenient to find. Therefore each service should be properly described and the description should cover:

Service details – textual description of the service accompanied by the classification according to predefined schema (preferably multiple), details of the service delivery (duty hours, price of the service).

Service description details should be separated from service implementation details, while each service may be provided by multiple providers. For instance, PIT form submission service is provided by numerous of local tax office. All service implementations posses certain common characteristics (for instance price), but some of them may be specialized locally (for instance duty hours).

Repository should allow for linking of service description with their implementations. Properties that are not specified for service implementation should be inherited from service description. Information on the implementation should be accompanied by the detailed information on the provider (organization unit, localization, contact details, contact person etc.);

Context – each entity to which a service may be offered is described by a certain context. If the same context definition is used to describe services, then these entities are able to automatically locate services which they may execute.

For instance, if a service has a constraint on a legal status, inclusion of this information in the service description allows for a convenient filtering of services.

Context dimensions that seem to be useful include: geographical (area to which service is offered), subject (legal status, occupation, age, gender, offered services, produced goods etc.) and life situation (building a house, getting married etc.). Each of these dimensions and they attributes (except for life situations) can be easily expresses with existing classification schemes;

Prerequisites and results – a service may be also defined in terms of its input and output artifacts (which in case of administration are mostly documents).

Service may be then described by input and output documents. In this case a taxonomy of documents is needed.

If above assumptions are met, the business model of an administration unit is relatively straightforward. The description of functions (cell B from Fig. 4) consists only of a set of references to services descriptions provided by the central repository. These references may of course refine these descriptions or provide additional information. The data model (cell A from Fig. 4) is also a set of references to documents defined in the central repository.

This approach allows not only for easy comparison of organizations capabilities, but above all, allows for impact of change tracking.

4.2 Semantic Interoperability

Unfortunately, it is not sufficient to name offered services and processed documents. What is missing, is the precise definition of processed documents. Each document, information it is supposed to convey, should be defined in sufficient details (cell C from Fig. 4).

In public administration documents are mostly forms, which can easily by implemented, for instance, in XML Schema. This layer, for reasons provided in the subsequent section, should be technology agnostic. The document should be defined as a list of information elements with some multiplicity constraints.

The biggest challenge at this stage is a promotion of reusability, which would lead to increased semantic interoperability. That means that new information elements, or new forms may only be defined, if existing ones are not sufficient. This may be achieved only if information elements are easily discoverable (again information must be properly classified in the repository) and information elements are defined in context dependent way. CCTS [10] contains a set of good practices that help to achieve this goal.

There is also one additional aspect of document definition that often is neglected. Each form, before it can go for further processing, must be checked for errors. Therefore, it is essential to define and reuse (again in technology independent way) evaluation rules. Each rule may check if the data provided in the field is valid or if the value is properly correlated with the value of other field or fields (based on arithmetic or symbolic computations).

4.3 Technical Interoperability

Definitions provided in the organizational and semantic layer are still insufficient for performing cross organization system integration, while they lack details concerning the actual implementation of information systems. No information on services interfaces, communication protocols, document encoding and structure is present.

This information is not important on business or even the system model. The life-time of technology in comparison to life-time of organization and its processes is relatively short. The change in technology very often allows to perform some actions faster, more securely or implement something more easily, but also very often does not invalidate organizations business processes.

It is very desirable to have all of the architectural resources built up in a platform independent way. According to Model-driven Architecture approach concrete implementation should be derived automatically based on the set of transformations. Change in technology should be reflected in a change of transformation rules.

There is one simple real-life example which explains why implementation specific details should not be created manually. There are four well defined paradigms of structuring XML Schema. Actually, schemas usually do not obey strictly these guidelines. Therefore, the probability that two persons will implement XML Schema for the same document in identical way is rather low. If the rules of transformation from technology independent form were defined formally this problem would be not present. In a best possible scenario, they could be executed automatically.

Assuming that the target technology is set of standards defined for service oriented architectures (as defined for instance in WS-I profiles) then the functionality (cell E from Fig. 4) is simply defined in WSDL document and the document is defined in XML Schema (cell F from Fig. 4) – both derived by transformation rules.

Unfortunately, it is not as simple as that. Additional standards are required. As mentioned above, the formalization for validation rules is required. Most of rules cannot be expressed in XML Schema. More expressive formalisms are required, e.g. Schematron or Relax-NG.

In public administration each document must be archived and recoverable. This requires standard for meta-data description. Although, there exist commonly accepted standard for annotation data (Dublin Core), the way it should be implemented is not that obvious. Guidelines for connecting meta-data to documents as well as the rules for assigning meta-data values should be formalized.

There are also additional aspects of electronic documents that must be specified and specification should be formal. Especially important issues covers: visualization – documents must be visualized in the same form, regardless the platform and security – issues related to electronic signature (which part of document should be signed, by whom, how to sign multiple documents etc.).

5 Summary

The problem of interoperability has been addressed in the literature a number of times. Unfortunately, most of existing interoperability frameworks have been developed in separation from existing approaches for architecture development. On the other hand, enterprise architecture frameworks and architecture governance in general are not particularity concerned with interoperability issues.

In this paper we link abovementioned frameworks and present recommendations for developing an interopeable architectures in the sector of public administration. In the approach presented in the paper the issues of the interoperability in e-Government are completed with selected aspects and perspectives from the Zachman Framework.

In an enterprise architecture life-cycle the presented solutions are only a starting step. Definition of the scope and development (in terms of contracting) of a particular IT assets in e-Government institutions is often perceived as sufficient. However, we state that in public adminsitration the same principles as in businesses should apply. Businesses gain their competitive advantage by continuous management and self-improvement of IT assets. In public authorities we would like the same principle to be applied. Interoperability is not only one-time requirement for information system. It should be perceived rather as long-lasting process within IT architecture governance life cycle.

The introduction of interoperability issues requires the mutual involvement of particular interests groups. Business aspects imply the interest of business analysts, technical aspects require the involvement of technical staff while semantics aspects requires particular attention from designers. The usual role for the coordination of enterprise architecture is enterprise architect that communicates and manages all aspects. Enterperprise architects should be aware of the issues raised within this paper.

References

1. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific American 284(5) (May 2001)
2. European Commission. The role of egovernment for europe's future' communication from the commission to the council. Technical report, the European Economic and Social Committee and the Committee of the Regions (2003)
3. European Commission. Eif and ida elink: advancing e-government interoperability at pan-european level. Technical report (2004)
4. European Commission. European interoperability framework for pan-european egovernment services. Technical report (2004)
5. Peristeras, V., Liotas, N., Tambouris, E., Tarabanis, K.: D2.7 study on interoperability at local and regional level. final version – version 2.0. Technical report, Modinis Lot 2 (2007)
6. European Public Administration Network eGovernment Working Group. Key principles of an interoperability architecture. Technical report (2004)
7. Gartner: Preparation for update european interoperability framework 2.0. Technical report (2007)
8. The Open Group. The Open Group Architecture Framework Version 8.1.1, Enterprise Edition. Van Haren Publishing (2007)
9. U.S. Office of Management and Budget. Federal enterprise architecture (2007), `http://www.whitehouse.gov/omb/egov/a-1-fea.html`
10. UN/CEFACT. Core component technical specification (ccts). Technical report (2003)
11. Zachman, J.A.: A framework for information systems architecture. IBM Syst. J. 26(3), 276–292 (1987)

Foundational Data Modeling and Schema Transformations for XML Data Engineering

Reema Al-Kamha[1], David W. Embley[2], and Stephen W. Liddle[3]

[1] Informatics Department, Damascus University, Syria
[2] Department of Computer Science, Brigham Young University, USA
[3] Information Systems Department, Brigham Young University, USA

Abstract. As XML data storage and interchange become ubiquitous, analysts and data engineers increasingly need tools to model their data and map it to XML schemas and to reverse engineer XML documents and schemas in support of evolution and integration activities. For effective data management, model transformations require guarantees of properties of interest including guarantees of information and constraint-preservation, redundancy-free and compactness guarantees, and assurances about readability and maintainability. In this paper, we make foundational observations about XML data management, including conceptual modeling for XML data, transformations to and from XML Schema and XML data models, and transformation guarantees concerning properties of interest, and we provide resolutions for conceptual mismatches between XML data management and more traditional data management. Our implemented prototype tools show that these observations and insights can provide a strong foundation for XML data engineering.

1 Introduction

Because XML has become a standard for data representation, there is a need for a simple conceptual model for XML-based data engineering. But this is not enough—engineers also need a suite of design and development tools to map conceptual designs into implementable designs and to reverse-engineer legacy implementations to conceptual designs. In addition to facilitating these activities, the tools should guarantee certain desirable properties about generated implementations and should warn developers if such properties do not hold.

In building a suite of XML design and development tools, we face several interesting challenges. (1) Creating a conceptual model is a delicate balance between providing enough but not too many high-level conceptualizations without introducing low-level, implementation detail; making the model formal but easily understandable; and having a notation that is easily understood by developers and customers alike. (2) Once a conceptual model exists, the challenge becomes defining equivalence transformations to and from XML Schema—a nontrivial task because of the large conceptual mismatch. (3) Beyond just having transformations, XML data engineering demands certain guarantees. As a minimum, the translations must preserve information content and, to the extent possible, preserve constraints. These guarantees should also enable developers to guarantee that forward translations yield storage structures that are redundancy-free and thus free of update anomalies, and that reverse translations yield faithful and understandable conceptual models.

In a keynote address [4] Carey challenged the conceptual-modeling community to develop conceptual models usable in XML data engineering. Several researchers have contributed to making this challenge a reality. Many have attempted to create or define characteristics for conceptual models for XML (e.g., [5,6,10,11,12,14,15,16]), but all have fallen short of capturing some interesting features of XML Schema. Some have attempted to provide transformations or design guidelines based on standard conceptual models or on XML-augmented conceptual models (e.g., [5,10,11]), but few address design properties such as being redundancy free or address them in a way to provide mapping guarantees, and none provide transformations for XML reverse-engineering.

In this paper, we build on our earlier work [2,3,8] and describe our implemented algorithms for conversions between a generic conceptual model and XML Schema. Based on these implementations, we explain how to meet the challenges of creating a suite of design and development tools for XML. In particular, we make the following contributions: (1) We argue for a few augmentations to traditional conceptual models to accommodate XML (Section 2). (2) We provide implemented equivalence transformations between XML-augmented conceptual models and XML Schema (Sections 3 and 4). (3) We show how to guarantee properties of interest in XML design and development (Sections 3 and 4).

2 C-XML: Conceptual XML

We show here how to extend traditional conceptual models for XML-based data engineering. In our prototype implementation, we have extended the conceptual modeling language OSM [7] for use with XML, resulting in *C-XML* (*Conceptual XML*). Figure 1 shows an example of a graphical rendition of a particular C-XML model instance for a small part of a student database.

All who have addressed the issue of creating a conceptual model for XML Schema argue for augmenting conceptual models with XML-like *sequence* and *choice* features. Although we disagree with previously-suggested ways of including *sequence* and *choice* features, we agree that both should be included. Some may argue that *sequence* and *choice* constitute low-level, implementation detail. We, and most others who have faced this problem, disagree. Order is a natural, high-level conceptualization among some entities (e.g. to represent component parts of addresses and start-end values for chromosome sequences), and alternatives are natural for others (e.g. to represent various forms of international addresses). We do, however, recommend only appropriate and conceptual use of *sequence* and *choice*—not the inappropriate and artificial use often seen in XML Schema because of its lack of modeling alternatives.

Fundamentally, a C-XML model instance is a hypergraph whose nodes are *object sets* and whose edges are *relationship sets*, which are often binary, but

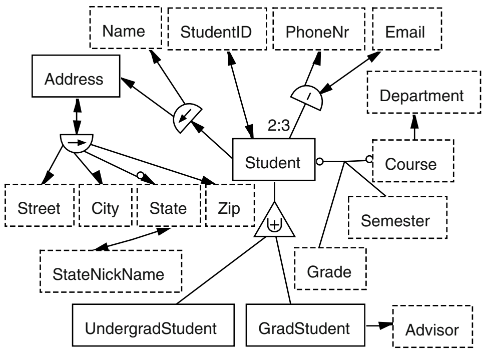


Fig. 1. Given C-XML Model Instance

may be n-ary ($n > 2$). Beyond a generic hypergraph, C-XML provides notation to declare various *constraints* over object and relationship sets.

Rendered graphically, object sets are boxes, relationship sets are lines, and constraints are decorations. A dashed box denotes a *lexical* object set whose object values are printable types, and a solid box denotes a *non-lexical* object set whose object values are object identifiers. In Figure 1, for example, *Name* and *StudentID* are lexical object sets, and *Student* and *Address* are non-lexical object sets. A *participation constraint* (e.g., *2:3*) indicates how many times an object in a connected relationship may participate in the relationship set. C-XML uses decorations for common constraints: (1) an arrowhead specifies a *functional constraint* and (2) an “o” on a connection designates *optional participation.* A triangle represents a generalization/specialization relationship among object sets, and C-XML allows for these relationships to be constrained (e.g., $\cup$, $+$, and $\uplus$ respectively denote union, mutual-exclusion, and partition constraints).

A bounded half circle with a directional arrow represents a *sequence* structure. Sequenced child object sets connect to the curved side, and the parent object set connects to the flat side. The representation for the *choice* structure is similar, but instead of an arrow a vertical bar indicates *choice.* In Figure 1, for example, each student has an address that is in a one-to-one correspondence with an ordered 4-tuple (*Street*, *City*, *State*, *Zip*), and each student has two or three ways to be contacted, each of which is either a phone number or an email address. Students may share phone numbers, but email addresses are unique as indicated by the functional relationship from *Email* to the *choice* structure.

C-XML satisfies the requirements for conceptual modeling for XML presented by others who have studied the problem of creating a conceptual model for XML [14,15,16]. These requirements include: a graphical notation, a formal foundation, structural independence, reflection of the mental model, n-ary relationship sets, views, logical-level mapping, cardinality for all participants, ordering, allowance for irregular and heterogeneous structure, and document-centric data. The property of having a formal foundation is particularly important, and we thus point out that a C-XML model instance is precisely an abstract representation for a particular set of predicates and predicate-calculus formulas. Each object set maps to a one-place predicate, and each n-ary relationship set maps to an n-place predicate. Each constraint maps to a closed predicate-calculus formula. A properly populated C-XML model is therefore a reformulation of a model in first-order theory [9].

3 Mapping C-XML to XML Schema

In the translation from C-XML to XML Schema we must consider several challenging issues.

– XML Schema has a hierarchical structure, while a particular conceptual-model instance may have no explicit hierarchical structure. Converting non-hierarchical structure to hierarchical structure presents some interesting challenges especially if we wish to be able to guarantee properties such as making the hierarchical structures as large as possible without introducing redundancy.
– XML Schema often does not mesh well with conceptual-modeling structures. Translations resulting in a valid XML-Schema instance sometimes need extra, unwanted artifacts to satisfy XML Schema's syntactic requirements.
– Because of XML-Schema limitations the translation sometimes cannot capture all the constraints of a C-XML model instance (e.g. some cardinality constraints). To preserve constraints in the translation, we add them as special comments that auxiliary constraint-checking software can read and enforce.
– A conceptual-model instance may contain a variety of conceptualizations: hypergraphs representing interrelated object and relationship sets; generalization/specialization hierarchies with union, mutual-exclusion, and partition constraints; hierarchies of *sequence* and *choice* structures; and mixed textual/conceptual structures. Translating these conceptualizations individually presents some challenges, and translating conceptual-model instances with a mixture of these conceptualizations is harder.

3.1 Build Scheme-Tree Forest

Our translation starts by applying an algorithm to convert a conceptual-model hypergraph to a forest of scheme trees [13], which we refer to as the *HST* algorithm (Hypergraph-to-Scheme-Tree translation algorithm). By observing many-one cardinality constraints, this algorithm finds hierarchical structures, making them as large as possible without introducing potential redundancy in stored XML data instances. Observing mandatory/optional constraints, this algorithm also ensures that all values populating a C-XML model instance are representable in the XML data instance.

An application of the HST algorithm to the C-XML model instance in Figure 1 generates the forest of scheme trees in Figure 2. The algorithm lets us build a root node in a scheme-tree beginning with any object set. By default it chooses

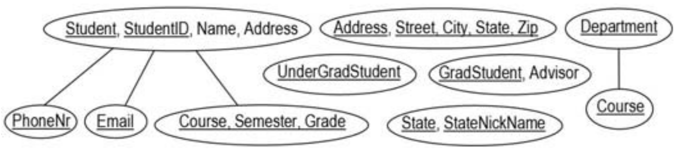


Fig. 2. Scheme Trees

an object set that is the root of the largest natural hierarchy as determined by the one-many relationship sets. In Figure 1, *State* is one-many with *Address* which is one-many with *Student* and thus would be the default choice. Often, however, it is best to start with the most important node, measured as one with the maximum number of incident edges—*Student* in our example. A starting object set for a node becomes a key for the node. For our example, we start with *Student*, place it in the first root node as Figure 2 shows, and underline it to indicate that it is a key for the node. The HST algorithm then builds the rest of the scheme-tree forest in Figure 2.

The details are beyond the scope of this paper, but in essence, the HST algorithm ensures that each relationship instance appears at most once in any populated scheme tree it generates, which ensures the redundancy-free property, and thus also ensures that storage structures are free from update anomalies. At the same time, from a given starting node, it grows scheme trees as large as possible without introducing potential redundancy. This ensures that the representation does not introduce unnecessary trees, and thus ensures that the representation is compact [13].

3.2 Generate Nested Object Containers

After generating the forest of scheme trees, we construct the XML-Schema instance. Since each scheme-tree node denotes a set of tuples, we generate a container for the set of tuples as well as a container for an individual tuple. Each container requires a name, and although we could use arbitrary names or let the user select names, we attempt to select a reasonable name for each container automatically. Since a key for a set of tuples identifies individual tuples, we choose keys as names for individual tuples and plurals of these names for sets of tuples. For example, Figure 3 shows part of the XML-Schema instance generated for the *Student* scheme tree in Figure 2, and Figure 4 shows part of an XML document that complies with the XML-Schema instance in Figure 3.

Each container has some content. Thus, the container element has *complexType* content. The container element for a node must provide for a set of tuples. We introduce them with a *sequence* structure, even though the *sequence* structure has the extra, perhaps unwanted, constraint of requiring its children to be ordered. The *sequence* structure is the only choice that suffices in this case. The container element for an individual tuple, on the other hand, contains at most one instance of each object set and each child node of the tuple. We thus use the *all* structure for the elements representing the child nodes of the node being

```
<xs:element name="Students">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Student" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:sequence>
              <xs:element name="Name" />
              <xs:element name="Address">
                <xs:complexType>
                  <xs:attribute ref="AddressOID"
                    use="required" />
                </xs:complexType>
              </xs:element>
            </xs:sequence>
            <xs:choice minOccurs="2" maxOccurs="3">
              <xs:element name="Emails">
...
          <xs:attribute name="StudentOID" type="xs:string"
            use="required" />
          <xs:attribute name="StudentID" use="required" />
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:key name="StudentOID-Key">
    <xs:selector xpath="./Student" />
    <xs:field xpath="@StudentOID" />
  </xs:key>
  <xs:key name="StudentID-Key">
...
</xs:element>
```

Fig. 3. Generated XML-Schema Instance

```
<Students>
  <Student StudentOID="Student1" StudentID="0000-1">
    <Name>Alice</Name>
    <Address AddressOID="Address1"/>
    <Emails>
      <Email Email="alice@university.edu/>
      <Email Email="alice@gmail.com/>
    <Emails>
    <Course-Semester-Grades>
      <Course-Semester-Grade
        Course="CS100" Semester="Fall" Grade="A"/>
      ...
    </Course-Semester-Grades>
  </Student>
  <Student StudentOID="Student2" StudentID="0000-2">
    <Name>Bob</Name>
  ...
</Students>
```

Fig. 4. Complying XML Document

built, where possible. It is not possible when *sequence* and *choice* structures are present, as they are for *Student* in our example, and thus in Figure 3 we introduce the elements of the *Student* tuple with a *sequence* structure.

The tuple itself consists of the key object set (or object sets in case of a compound key)—*Student* in our example; other object sets in the node being constructed—*StudentID*, *Name*, and *Address* in our example; and the container element for child nodes of the node being constructed—*PhoneNrs*, *Emails*, and *Course-Semester-Grades* in our example. We generate the object sets as attributes, when possible, and otherwise as elements, and we generate the child-node containers as elements. Thus, as Figure 3 shows, *Name* and *Address* are elements since they appear under a *sequence* structure, and *StudentID* and *StudentOID* are attributes.

Each non-lexical object set has an "OID" attribute, which allows us to explicitly represent the entity. If designers prefer not to have explicit representations for non-lexical entities, they can "lexicalize" the C-XML model instance before transforming it to an XML-Schema instance. To lexicalize a non-lexical object set, we replace it with a lexical object set with which it has a one-to-one correspondence or by a group of lexical object sets with which it has a one-to-one correspondence. To retain original names for generating more pleasing tag names for XML documents, we can rename the lexicalized nodes in a special way: *X (of Y)* where *X* is the name of the lexical object set and *Y* is the name of the non-lexical object set. Thus, in our example we would have *StudentID (of Student)*, *StudentID (of UnderGradStudent)*, and *StudentID (of GradStudent)*. With this construction we would retain the names *Students* and *Student* for node and tuple containers, and we would omit the attribute *StudentOID*, retaining *StudentID* as an attribute and as the key.

Each XML-Schema instance must have a single root element. When the number of scheme trees in the generated forest is one, we do not generate a root element because the container element for the set of tuples for the root node in that scheme tree can serve as the root element. When the number of scheme trees in the generated forest is more than one, we generate a root element, call it *Root*, and nest the elements beneath it that represent the sets of tuples for each generated scheme tree.

3.3 Add Constraints

The structure of the generated scheme trees plus the optional constraints of the input C-XML model instance dictate the cardinality constraints. Every XML-Schema element declaration specifies its cardinality with respect to its parent as a *minOccurs* value and a *maxOccurs* value, and every XML-Schema attribute declaration specifies whether it is "required" or "optional" in its *use* attribute. Since there is exactly one instance of the container element for a set of tuples for a node, the assigned values for *minOccurs* and *maxOccurs* are both *1*, the default values. A set of individual tuples may contain one or more tuples. Thus, for a container for individual tuples, the *minOccurs* value is *1* and *maxOccurs* is *unbounded*. The *Student* element in Figure 3, for example, shows these cardinalities. The value for *use* is *required* unless the conceptual model constrains the *use* to be *optional*. In Figure 3, for example, the *use* for the attribute for *StudentID* is *required*.

We observe that since XML Schema has a hierarchical structure, we can only capture participation constraints in the conceptual model instance for the parent element. By default, the nesting structure in an XML-Schema instance makes the minimum participation constraint for a child element be *1* and the maximum constraint be *unbounded*. XML Schema provides no way to capture any constraint other than this default constraint. We can, however, capture constraints that differ from the default in special *pragma* comments. We prefix pragma comments with *C-XML* so that we can know to process them if we wish to enforce the constraint or if we wish to restore the original C-XML model instance from the XML-Schema instance. So that we can know what constraint to enforce or restore, we write the constraint formally using predicate-calculus syntax. (All constraints in C-XML have equivalent predicate-calculus expressions [7].) For example, to declare that the participation of *Course* in the *n*-ary relationship set in Figure 1 among *Student*, *Course*, *Semester*, and *Grade* is optional, we write the comment

forall x(Course(x) => exists [0:] <y, z, w>*
  (Course(x)Student(y)Semester(z)Grade(w)))

which establishes the constraint that each element *x* in *Course* may have zero or more tuples *<y, z, w>* in the relationship set among *Course*, *Student*, *Semester*, and *Grade*.

For each key within a node we determine the uniqueness constraints and express them in the generated XML-Schema instance. Every key for each node is unique within the container element for that node. Keys within child nodes, however, are only known to be unique within their parent node. *Course*-*Semester*-*Grade* tuples, for example, are only unique for each *Student*. Figure 3 shows the generated key constraint for *StudentOID*. Its *selector* declaration is *./Student* since its declaration is within *Students*, and its *field* declaration is @*StudentOID* since *StudentOID* is an attribute.

Generalization/specialization is not a native construct in XML Schema. Nevertheless, with judicious use of XML-Schema's *keyref* constraint, we can make XML Schema enforce basic generalization/specialization constraints. The main idea in generalization/specialization is that a generalization object set is a superset of each of its specialization object sets. Using *keyref* enables us to specify that the set of values in a specialization element is a subset of the set of values in a generalization element. Figure 5 shows these basic subset declarations for *UnderGradStudent* and *GradStudent*. Observe that these declarations force the set of *UnderGradStudentOID*s and the set of *GradStudentOID*s to be a subset of the set of *StudentOID*s.

When a generalization/specialization hierarchy has a *union*, a *mutual-exclusion*, or an *intersection* constraint, we generate a special pragma comment to capture the constraint. Since our example in Figure 1 has a partition constraint, we generate both a *union* constraint and a *mutual-exclusion constraint* as follows.

forall x(StudentOID(x) => GradStudentOID(x)
  or UnderGradStudentOID(x)
forall x(GradStudentOID(x) =>
  not UnderGradStudentOID(x))

```
<xs:keyref name="UnderGradStudentOID-Keyref"
    refer="StudentOID-Key">
  <xs:selector
    xpath="./UnderGradStudents/UnderGradStudent" />
  <xs:field xpath="@UnderGradStudentOID" />
</xs:keyref>
<xs:keyref name="GradStudentOID-Keyref"
    refer="StudentOID-Key">
  <xs:selector
    xpath="./GradStudents/GradStudent" />
  <xs:field xpath="@GradStudentOID" />
</xs:keyref>
```

Fig. 5. Generated Subset Declarations

3.4 Property Guarantees

Having explained the basic idea of our translation of a C-XML model instance to an XML-Schema instance, we now show that the translation preserves both information and constraints. We also show that the translation is redundancy-free and is reasonably compact.

Definition 1. *Let T be a transformation from a model instance M to a model instance M' that not only derives M' from M but also derives a populated model instance M'_p from a populated model instance M_p. T* preserves information *if for any properly populated model instance M_p, there exists an inverse transformation T^{-1} that maps $T(M_p)$ to M_p such that $T^{-1}(T(M_p)) = M_p$.*

Theorem 1. *The transformation from C-XML to XML Schema preserves information. (We have omitted all proofs due to space constraints.)*

Definition 2. *Let C be the constraints of a model instance M, and let C' be the constraints of a model instance M' obtained from M by a transformation T. T* preserves constraints *if $C' \implies C$.*

Theorem 2. *Allowing for pragma constraints, the transformation from C-XML to XML Schema preserves constraints.*

Definition 3. *A value or object instance i in a populated model instance M_I is* redundant *with respect to a set of constraints C if i is uniquely determinable from C and the values and object instances in M_I other than i.*

If, for example, we store the $StateNickName$ with every student's address, then for any two student instances S_1 and S_2 whose address is in the same state, we can uniquely determine the state nickname of S_1 from the information in S_2's address and from the constraint that $StateNickName$ and $State$ are in a one-to-one correspondence.

Theorem 3. *Let C be a canonical C-XML model instance whose declared functional constraints are its functional edges and whose declared multivalued constraints are its non-functional edges. The transformation from C-XML to XML Schema for C yields an XML-Schema instance whose complying XML documents* *have no redundant value or object instances with respect to the functional and multivalued constraints declared in C.*

Observe that we have not claimed redundancy-free with respect to the inclusion dependencies in a generalization/specialization hierarchy. Indeed, because of the way we store and reference object identifiers or values in generalization/specialization hierarchies, every value or object in a generalization is stored redundantly. XML system developers can avoid this redundancy by collapsing all generalization/specialization hierarchies to their roots before generating XML-Schema instances. In Figure 1 collapsing the generalization/specialization hierarchy to the root consists of discarding the object sets *UnderGradStudent* and *GradStudent* and attaching *Advisor* optionally through a functional relationship set to *Student*. Implicitly, those students who have advisors are graduate students and those who do not have advisors are undergraduate students. The disadvantage of this approach is that the generalization/specialization constraints (the partition in our example) are lost as are all the specialization names.

As it turns out, the HST algorithm does not always produce the fewest number of scheme trees both because a developer may choose to start the algorithm at a node that does not initially carve out the largest natural hierarchy and because of some pathological cases that occur only when the hypergraph is cyclic. Normally, however, the HST algorithm yields the fewest number of scheme trees and thus yields a compact representation.

Although not provable, we believe that XML-Schema instances obtained by transforming C-XML model instances are reasonably readable and thus maintainable. As Figure 4 indicates, they provide reasonable tags for both sets of objects (e.g. *Students*) and individual objects (e.g. *Student*), and they appropriately nest associated information nicely inside of the scope of objects (e.g. *Address*, *Email*, and *Course-Semester-Grades* inside the scope of student).

4 Mapping XML Schema to C-XML

The basic translation strategies for mapping XML Schema to C-XML are straightforward, although some parts of the translation require some sophisticated manipulation. In the translation, elements and attributes become object sets. Elements that have simple types become lexical object sets, while elements that have complex types become non-lexical object sets. Attributes become lexical object sets since they always have a simple type. Built-in data types and simple data types for an element or an attribute in XML Schema are specified in the data frame associated with the object set representing the element or the attribute. XML parent-child connections among elements and XML element-attribute connections both become binary relationship sets in C-XML. The constraints *minOccurs* and *maxOccurs* translate directly to participation constraints in C-XML.

Unfortunately, not everything is straightforward. Translations for keys, extension, restriction, substitution groups, and mixed content are all quite interesting. The translation also involves a myriad of detail extending to over 40 pages in

[1]. One general observation about the translation is that it is often difficult and sometimes impossible to express modeling constraints of interest.

– XML Schema provides no good way to nest elements that are not natural sequences of, not natural alternatives of, and not functionally dependent on a parent. Although designers usually declare sequences for these nestings, it is difficult to tell whether a *sequence* is merely an artifact, required by XML Schema, or whether it is a conceptual sequence, as address elements would be. Since a translation algorithm does not know whether a sequence is meaningful, however, we must faithfully generate all declared sequences. We can, and do, handle the special case of a *sequence* of one nested element.
– Generalization/specialization, especially large hierarchies with complex constraints, is difficult and sometimes impossible to specify [3].
– Some functional constraints are difficult or impossible to capture.
– Lexical elements cannot be parent elements. In such cases, we must introduce another object set.

Because some unwanted artifacts appear and because some constraints are either difficult or impossible to declare, one interesting possibility is to reverse-engineer an XML Schema instance to an C-XML instance, add the missing constraints, remove unwanted artifacts, and then regenerate a more desirable XML Schema instance.

Theorem 4. *The transformation from XML Schema to C-XML preserves information.*

Theorem 5. *The transformation from XML Schema to C-XML preserves constraints.*

A main use of the transformation from XML Schema to C-XML is to reverse-engineer an XML Schema instance. We believe that the result is quite usable. We capture each XML concept as an object set and each nested connection among concepts as relationship set, and we also capture all representable constraints. Further, we believe that the representation is appropriately abstract and sufficiently high level in the usual way in which conceptual models are abstract and high level. Thus, we can reasonably claim that this reverse-engineering transformation can aid in understandability and thus maintainability and evolvability.

5 Concluding Remarks

We have implemented automatic conversions between C-XML and XML Schema that preserve information and constraints, that have guaranteed redundancy-free and compactness properties, and that yield reasonably understandable results and thus provide for maintainability and evolvability. Our prototype implementations and these observations and insights provide a solid theoretical foundation for XML data engineering.

References

1. Al-Kamha, R.: Conceptual XML for Systems Analysis. Phd dissertation, Brigham Young University, Department of Computer Science (June 2007)
2. Al-Kamha, R., Embley, D.W., Liddle, S.W.: Representing generalization/specialization in XML schema. In: Proceedings of the Workshop on Enterprise Modeling and Information Systems Architectures (EMISA 2005), Klagenfurt, Austria, October 2005, pp. 250–263 (2005)
3. Al-Kamha, R., Embley, D.W., Liddle, S.W.: Augmenting traditional conceptual models to accommodate XML structural constructs. In: Parent, C., Schewe, K.-D., Storey, V.C., Thalheim, B. (eds.) ER 2007. LNCS, vol. 4801, Springer, Heidelberg (in press, 2007)
4. Carey, M.: Enterprise information integration—XML to the rescue! In: Song, I.-Y., Liddle, S.W., Ling, T.-W., Scheuermann, P. (eds.) ER 2003. LNCS, vol. 2813, p. 14. Springer, Heidelberg (2003)
5. Choi, M., Lim, J., Joo, K.: Developing a unified design methodology based on extended entity-relationship model for XML. In: Sloot, P.M.A., Abramson, D., Bogdanov, A.V., Gorbachev, Y.E., Dongarra, J., Zomaya, A.Y. (eds.) ICCS 2003. LNCS, vol. 2660, pp. 920–929. Springer, Heidelberg (2003)
6. Conrad, R., Scheffner, D., Freytag, J.C.: XML conceptual modeling using UML. In: Laender, A.H.F., Liddle, S.W., Storey, V.C. (eds.) ER 2000. LNCS, vol. 1920, pp. 558–571. Springer, Heidelberg (2000)
7. Embley, D.W., Kurtz, B.D., Woodfield, S.N.: Object-oriented Systems Analysis: A Model-Driven Approach. Prentice Hall, Englewood Cliffs, New Jersey (1992)
8. Embley, D.W., Liddle, S.W., Al-Kamha, R.: Enterprise modeling with conceptual XML. In: Atzeni, P., Chu, W., Lu, H., Zhou, S., Ling, T.-W. (eds.) ER 2004. LNCS, vol. 3288, pp. 150–165. Springer, Heidelberg (2004)
9. Enderton, H.B.: A Mathematical Introduction to Logic. Academic Press Inc., Boston, Massachussets (1972)
10. Feng, L., Chang, E., Dillon, T.: A semantic network-based design methodology for XML documents. ACM Transactions on Information Systems 20(4), 390–421 (2002)
11. Liu, H., Lu, Y., Yang, Q.: XML conceptual modeling with XUML. In: Proceedings of the 28th International Conference on Software Engineering, Shanghai, China, May 2006, pp. 973–976 (2006)
12. Mani, M.: Conceptual models for XML. In: Bellahsène, Z., Milo, T., Rys, M., Suciu, D., Unland, R. (eds.) XSym 2004. LNCS, vol. 3186, pp. 128–142. Springer, Heidelberg (2004)
13. Mok, W.Y., Embley, D.W.: Generating compact redundancy-free XML documents from concptual-model hypergraphs. IEEE Transactions on Knowledge and Data Engineering 18(8), 1082–1096 (2006)
14. Nečaský, M.: Conceptual modeling for XML: A survey. In: Proceedings of the Annual International Workshop on Databases, Texts, Specifications and Objects (DATESO 2006), Desna, Czech Republic, April 2006, pp. 45–53 (2006)
15. Sengupta, A., Wilde, E.: The case for conceptual modeling for XML. Technical report, Wright State University (WSU) and Swiss Federal Institute of Technology (ETH) (February 2006)
16. Wilde, E.: Towards conceptual modeling for XML. In: Proceedings of the Berliner XML Tage 2005 (BXML 2005), Berlin, Germany, September 2005, pp. 213–224 (2005)

Integration of Semantic XForms and Personal Web Services as a Tool to Bridge the Gap between Personal Desktops and Global Business Processes

Amin Anjomshoaa and A. Min Tjoa

Institute of Software Technology and Interactive Systems,
Vienna University of Technology, Austria
{anjomshoaa,amin}@ifs.tuwien.ac.at

Abstract. A large portion of Internet advances owes the human-computer interaction and data exchange via web forms. Current complex Internet applications are demanding more advanced features that are not covered by traditional forms. New technologies such as Semantic Web and XForms are responses to today's business requirements and expectation. In this paper the possible potential of combining XForms with Semantic Web technology are explored and an integration model for enterprise solutions is presented. The new model is supposed to bridge the gap between desktops as the nodes of global business networks representing the individuals and the involved web business processes. The paper shows that combining Semantic Web concepts with XForms can provide elegant holistic solutions for lifetime data consistency, integration of personal services, and checking of data validity.

Keywords: Semantic Web, Ontology, XForms, SOA, Interoperability, Web Applications.

1 Introduction and Related Works

Over two decade history of World Wide Web, the Internet has been evolved from a unidirectional information stream to an enterprise application framework. The traditional web forms are simple interfaces that aim to transfer data between server side components and browser application. Despite the advances in design and implementation of web applications, the web development is still a frustrating task. As a matter of fact, the difficulty of developing and deploying commercial web applications increases as the number of technologies they use increases and as the interactions between these technologies become more complex [1]. One of the reasons for this complexity relies on the distributed nature of web applications. As an example consider the browser side errors that are hardly traceable to back end server modules. Another challenge in web programming is the fact that web programming languages such as JSP, PHP and ASP are inevitably mixing the HTML tags with the code that should be executed on server side. Also the proposed solution to separate the presentation content from the web application logic have not simplified the situation that much and has added some other complexities and generally the web programs are not easily readable and understandable for the human user.

The W3C' solution to the mentioned problem is XForms that offers separation of the form's purpose from its presentation. An XForm allows processing of data to occur using a declarative model composed of formula for data calculations and constraints. It also equipped with a view layer composed of intent-based user interface controls that are bound to the model and finally XForms provide an imperative controller for orchestrating data manipulations, interactions between the model and view layers, and data submissions. Thus, XForms accommodates form component reuse, fosters strong data type validation, eliminates unnecessary round-trips to the server, offers device independence and reduces the need for scripting [2].

Another interesting aspect of XForms is the role that it might play in the integration of desktop information (user's world) with other business processes using Semantic Web [3] technologies. More precisely, the Semantic Web should bridge the gap between user information and external processes by mapping the user resources to those needed by a specific web form. As an example consider an online e-shopping system that requires the payment information from the user. Such data should be provided by each shopping (or once per e-shopping system); however, this could be avoided by integrating the user information which reside on the user's desktop. The combination of the XForms' data model with the user ontology and the appropriate mapping it to an upper ontology will improve the logical design of application and may have more software engineering benefits too.

Meanwhile the Service Oriented Architecture (SOA), has emerged as a communication paradigm for extending the interaction between applications whereas the applications may run on heterogeneous systems. Web services allow automatic and dynamic interoperability between systems to accomplish business tasks by putting together their functionalities. Another trend that has influenced the web applications in the recent years is the Web 2.0 technology that is going ahead hand in hand with SOA. Basically Web 2.0 is much more than adding a nice facade to old web application rather it is a new way of thinking about software architecture of web applications. In comparison to traditional web applications, the application logic of modern Web 2.0 applications is more concentrated on the client side and the business processes are considered as remote services. As a matter of fact it is a step toward the cloud computing where business services are presented on Internet and developers should select and weave them together to create new compound services. The SOA's Achille's heel in our belief is its lack of the semantic description of Web Services. Without the explicit semantic context, the process of assembling “pieces of functionality” into complex business processes is still unthinkable without significant human involvement and exactly here is the point that Semantic Web can fit, to facilitate the application of web services to modern web applications that use XForms.

XForms 1.1 was announced as an official W3C Recommendation on 29th November 2007, and has attracted the attention of developers and researchers. The most interesting part of XForms is the introduction of the data models described in XML that can be used for presentation and reasoning purposes and opens the gates to lots of possible extensions. Peng Yew Cheow et al. [4], propose a logical frame work, which maps the semantics of the XForms documents to knowledge bases in Description Logic. This framework can be used as a basis for reasoning and validity checks on XForms.

With the emergence of complex business processes, there is a growing need to embed the web forms into user's information context. In other words the generic web forms should be personalized according to user history and context information. As addressed in the previous section, Semantic Web can glue up the open world processes to user's information in a machine-processable way. Our SemanticLIFE project is an effort to realize a Personal Information Management (PIM) system using Semantic Web technologies, with the aim of creating a semantic repository of all personal data from a variety of sources like emails, contacts, running processes, web browsing history, calendar appointments, chat Sections, and other documents. This PIM system acts as a digital memory and provides the “personal profile” for acquired persons [5]. To our belief, the emergence of Semantic PIM systems plus the ever-increasing processing capability of desktops makes the integration of personal desktops into real world business processes, conceivable.

In this paper, some possible solutions based on XForms and Semantic Web are presented and the feasibility of implementation of corresponding use cases is discussed in details.

2 Methodology

As explained before the the XForms data model can play an important role to connect web forms to the back-end processes. The more known roles of the XForms' data model are:

- Providing a declarative data structure that document's visual elements will refer to.
- Decoupling data, logic from user interface rendering features of the target device (handheld, television, desktop browse, etc).
- Adding data calculations and dependency of form elements.
- Adding data types and other constraints.
- Defining the form submission parameters and behavior and also providing an XML instance data at submission time.

The focus of this paper is to explore the role where XForms' model can connect an XHTML form to other available services by using Semantic Web technologies to extend the functionalities of XHTML forms via ontologies. In the rest of this section, the feasibility of such a solution will be discussed.

An XForms model is an XML structure that is included in the header part of an XHTML document and where its elements are conforming with the documents' name spaces. Listing below shows a typical XHTML document that contains a simple data model with three elements: first name, last name and city name.

```
<html
      xmlns="http://www.w3.org/1999/xhtml"
      xmlns:sample="http://www.sampleinfo.net/"
      xmlns:xf="http://www.w3.org/2002/xforms">
 <head>
     <title>Testing XForms</title>
     <xf:model>
        <xf:instance>
           <DataElement>
              <sample:FirstName/>
              <sample:LastName/>
              <sample:City/>
           </DataElement>
        </xf:instance>
     </xf:model>
 </head>
 <body>
    ...
 </body>
 </html>
```

The interesting part of the above listing is the name-spaces that attach the definition of the model elements to a known name-space. So every element can be specified with its fully qualified URI and this URI can be selected to be identical with the name space of a domain ontology. For instance the first name item of above XHTML document has the URI of “http://www.sampleinfo.net/FirstName” and this can be assumed to be the name space of a domain ontology. As a result the form elements can be easily mapped to the relevant domain ontologies.

Web forms contain atomic elements that can be better processed in comparison to web pages and natural text. Elements that appear on a typical form usually convey a logical relationship between elements. For example if a web form contains a city name and country name, probably the city should be located in the specified country. Traditionally, such relation can be identified only by human users and the semantic of the elements' relationships is missing. With the current advances of Semantic Web technologies, the machines can also detect the concepts and their relationships. In other words the form elements are mapped to an ontology via the XHTML's embedded model. As soon as we arrive at ontology level, the application would benefit from all advantages of Semantic Web world.

For the scenarios that will follow in next section, we are especially interested in Semantic Inference and Ontology mapping. The scenarios will be classified into two main groups. First are those scenarios that are aimed to combine the web forms with business services on the web and secondly are the scenarios that are using the Semantic Desktop as a basis for connecting the user's world to web forms and customize the forms and their content data according to user's profile. The following figure shows these two methods and how the ontology is filling the gap between services and web forms.

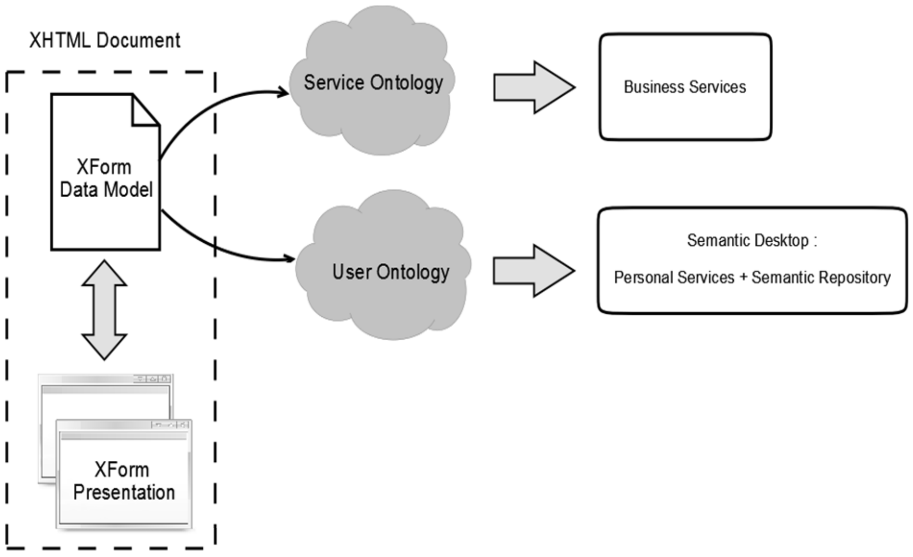


Fig. 1. XForm's Service integration methodology

3 Demonstration Scenarios

In this section some scenarios for integrating the web forms with business and personal processes are presented. Some presented scenarios can leverage the productivity of both web form end-users and web form providers/programmers. The web form end-users would be able to save time in filling out forms by connecting the forms to their Semantic Desktop using personal ontologies. They can also be warned about the information that might be inferred from the data that they submit through different forms to a specific service provider. The usage of XForms can also be beneficial for service providers and programmers because of the increasing productivity of software development processes.

3.1 Unwanted Information Disclosure

The information gathered by service providers on the Internet, might lead to the disclosure of some information which from the end-user's point of view should not be exposed to that organization. A simple example to illustrate this is date of birth. If the user prefer to disclose his/her date of birth, then he/she will probably ignore the fields that are asking directly about date of birth, however the provider can ask year, day, and month of the birth separately during a period of time and then put the pieces together and hack the date of birth.

In some cases service providers can use rules to perform reasoning and deduce new information. For example by providing the Zip code and country name, and the name of the company that has built your house the service provider can conclude the city name from zip code and country and then locate your building among a smaller set of possibilities. As a proposed solution personal ontologies can capture the user intent and consult the history of user's interaction with a specific service provider, to avoid completion of such chains. The XForms' data model is the key for joining the web form elements to domain ontology.

3.2 Lifetime Data Consistency

In the long periods of time the user might forget the detail of his/her interactions with a service provider and submit some data that is in conflict with previously provided data. For example, user's name might be spelled differently (e.g. Special characters in European languages) and it can cause ambiguities at service provider side. An XForms' data model that is mapped to a personal ontology can help to add a consistency check before submitting the data. So the user's history can be consulted to avoid such inconsistencies.

3.3 Avoiding Multiple Data Entries

Similar Internet applications usually require a common set of data that should be provided by the end-user. For example to register in an online shop, a user needs to provide the payment information, the delivery address, etc. The Semantic Desktop can help the user by auto completion of common fields. Again the XForms' data model that is mapped to user ontology, can query the Semantic Store and extract the required information.

3.4 Integration of Personal Services

In addition to services that are available on Internet as web services, the user's desktop can also provide personal services. Semantic Desktop systems, such as SemanticLIFE, contribute some personal services that can be used in service composition scenarios. These services are enriched with semantic information and can be queried according to service parameters and/or service intent. Personal services are distinguished with two pivotal factors: first they are closely related to the entities that are typically used in everyday life like emails, appointments, documents, etc and secondly are customized for their owner and vary not only from user to user, they also vary depending on the context of use.

To clarify the issue, consider an appointment matching service that is presented by user's Semantic Desktop. A non-semantic service will simply consult the user's calendar and says whether a specific time slot can be assigned for a meeting. The drawback of a “non-semantic” service is that the service is not able to conceive the location concept and check if the meeting is feasible according to the ontology of appointments in the days before and after. So the appointment matching service is dependent on user's and appointment's context and additionally it deals with appointment items that are highly related to user's daily desktop applications.

Personal services can be also categorized according to their internal complexity. Some services perform a simple one step action whereas more complex services may be composed of multiple actions and additional conditions and service calls. BPEL processes [6] fall under the category of more complex processes. A Semantic Desktop that supports BPEL processes might be seen as a data source that extract and process the data from different resources and hand in the useful information to external world.

As an example of a more complex BPEL process let us consider an online shopping use case (figure 2), where the shopping price limitation should be first checked against your bank account and credit information. A Semantic Desktop BPEL process is able to call external bank web services in a trusty way and calculate your shopping limit based on user's cache amount in the banks minus the planned monthly loans.

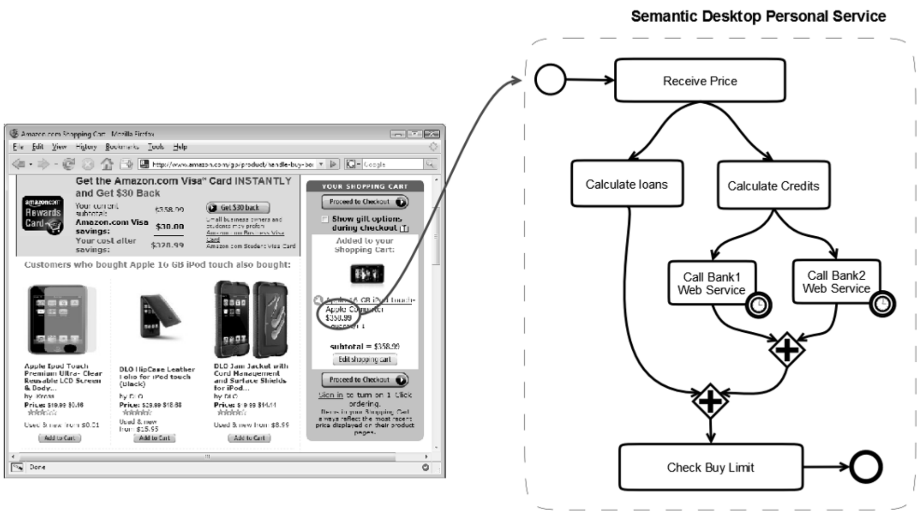


Fig. 2. Integration of Semantic Desktop Personal Services

In the Semantic Desktop environment, where services are described in a semantic way, the XForms' data model can be coupled to a personal service, via user ontology and service profiles. As a result the personal services can be connected to the global business processes and ease the user interaction with open world.

3.5 Checking the Data Validity and Data Consistency

During the development of a web application, the software designer and/or programmer should add constraints and checks to the submitted data to guaranty the data validity and consistency. Moreover adding and removing fields to the web form after the first design is a common issue. After applying such changes the data validation routines should be reviewed and adopted with new changes.

XForms might again be helpful in connecting the form elements to service and domain ontologies. First of all, domain ontology can deliver the information about validity checks that should be accomplished for the combination of the fields, appearing on the web form. Additionally the logic of the check program can be deduced from the domain ontology and enables the selection of the appropriate service from the service ontology. The XForms' logic might force the user to correct the data before submission. In this case the generated validation component that is added on the client side will contact relevant web services to confirm validity of data. In this way the data validity is checked at the very early stages of relevant business processes.

4 Conclusion

XForms as the next generation of web forms have the potential to realize a bunch of business scenarios more efficiently. Joining the structured models of XForms and the ever-increasing capabilities of desktop computers facilitates the implementation of many business scenarios. On the other hand, the emergence of Semantic Web technologies has leverage the process interoperability that in turn boosts the business processes. In this paper, the feasibility of combining the XForms, Semantic Desktops, and SOA was discussed and some possible scenarios were explored. In future we will focus on the integration of useful personal services in the Semantic Desktop environment and apply them to open world business processes via connectors such as XForms.

References

1. Cardone, R., Soroker, D., Tiwari, A.: Using XForms to simplify Web programming. In: 14th International World Wide Web Conference (WWW), Chiba, Japan (2006)
2. W3C Consortium, W3C Candidate Recommendation (November 2007), http://www.w3.org/TR/2007/CR-xforms11-20071129/
3. W3C Consortium, W3C Semantic Web Activity (2001), http://www.w3.org/2001/sw/
4. Cheow, Peng, Y., Governatori: Guido, Representing and Reasoning on XForms Document. In: Fifteenth Australasian Database Conference (2004)
5. Anjomshoaa, A., Nguyen, T.M., Shayeganfar, F., Tjoa, A.M.: Utilising Web Service Based Business Processes Automation by Semantic Personal Information Management Systems – The SemanticLife Case. In: Reimer, U., Karagiannis, D. (eds.) PAKM 2006. LNCS (LNAI), vol. 4333, pp. 1–10. Springer, Heidelberg (2006)
6. OASIS, Web Services Business Process Execution Language Version 2.0 (April 2007), http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf

A Slicing Technique for Business Processes

Eike Best and Astrid Rakow

Parallel Systems, Department of Computing Science
Carl von Ossietzky Universität Oldenburg, D-26111 Oldenburg, Germany
{eike.best,astrid.rakow}@informatik.uni-oldenburg.de

Abstract. A slicing technique is described for Petri net models of business processes. By this technique, a Petri net can be reduced to a smaller one with less states, such that verification can be done on the smaller net and transferred to the larger one. Three case studies are discussed.

1 Introduction

Program slicing [11] is a well-established technique for software analysis. More recently, slicing methods have been transferred to concurrent systems (e.g. in [2]), and in particular, to Petri nets [9].

Petri nets are a popular model for business processes and workflows [1], as they can model choice, split and join operations. Places, transitions and arcs also make a (not just fleeting, cf. [6]) connection to the concepts of thing-type, operation and connection-type in KCPM, the *Klagenfurt Conceptual Predesign Model* [7].

By implication, Petri net analysis methods, and in particular, slicing techniques, can be used for the validation of business processes. The algorithm described in [9] is interesting in this respect, because it is particularly useful in case a Petri net is not strongly connected. This is often true for business process models.

In this paper, we discuss the algorithm (Section 2) and explain it on three case studies (Section 3).

2 A Slicing Algorithm

Let Σ be a marked Petri net modelling some system, and let P be a non-empty set of places of Σ. P is called a *slicing criterion*. Typically, P contains places which are referred to by some temporal logic formula φ expressing a property that one wishes to prove or disprove for the system described by Σ.

The following algorithm constructs a subnet of Σ called $\mathit{slice}(\Sigma, P)$. The idea is that in order to prove φ on Σ, it suffices to prove it on $\mathit{slice}(\Sigma, P)$, while at the same time, the latter has less states than the the former. Thus, by constructing the slice first, one needs to explore less states than would be needed without constructing the slice.

```
1  $T$, $S_{done} := \emptyset$;
2  $S := P$;
3  while ( $\exists s \in (S \setminus S_{done})$ ) {
4     while ( $\exists t \in ((^\bullet s \cup s^\bullet) \setminus T) : W(s,t) \neq W(t,s)$ ) {
5        $S := S \cup {^\bullet t}$;
6        $T := T \cup \{t\}$ };
7     $S_{done} := S_{done} \cup \{s\}$ };
8  return the net generated by $S$ and $T$.
```

Here, $^\bullet s$ and $^\bullet t$ denote the preset of a place s and a transition t, respectively, while $s^\bullet$ denotes the postset of a place s, in Σ. Also, W denotes the arc (weight) function in Σ.

Starting with places in P, the outer loop in lines 3-7 checks whether more places of Σ need to be considered. For each place to be considered, the inner loop in lines 4-6 checks all of its surrounding transitions. If they are non-reading (determined by the condition $W(s,t){\neq}W(t,s)$), then they too need to be considered. In line 5, the set of places to be considered is extended only on the input places of t, not its output places. The idea is that output places are influenced by, but do themselves not influence, the property under consideration. More precisely, we have the following result:

Theorem: Let φ be a CTL$^*_{\text{-X}}$ formula such that the set of places referred to in φ is contained in P.

Then Σ models φ fairly with respect to T if and only if $slice(\Sigma, P)$ models φ.

Here, CTL$^*_{\text{-X}}$ denotes extended computation tree logic (comprising both CTL and LTL) without the nexttime operator.[1] Fairness w.r.t. T simply means that enabled actions from T cannot be neglected indefinitely.

By this theorem, we know that all the fair executions of the full net satisfy φ fairly towards the slice, provided it can be proved that the slice itself satisfies φ. We need fairness here, since otherwise we could have the case, for example, that all executions of the slice terminate properly while some execution of the full net, circumventing the slice, does not.

3 Case Studies

3.1 A Workflow Describing Electronic Trade Procedures

This case study is taken from [8], but since the Petri net described there is rather large, we show only part of it (cf. Figure 1). A consignee may order shipment of products from a shipper residing in a different country, say across the Atlantic ocean. In order to enhance trustworthiness of the process, two banks are involved, the issuing bank that provides a loan to (creditworthy) consignees, and the corresponding bank on the shipper's side of the ocean. On receiveing a

[1] It is beyond the scope of this paper to describe these logics, cf. [3]. We will show some example formulas and explain them in due course.

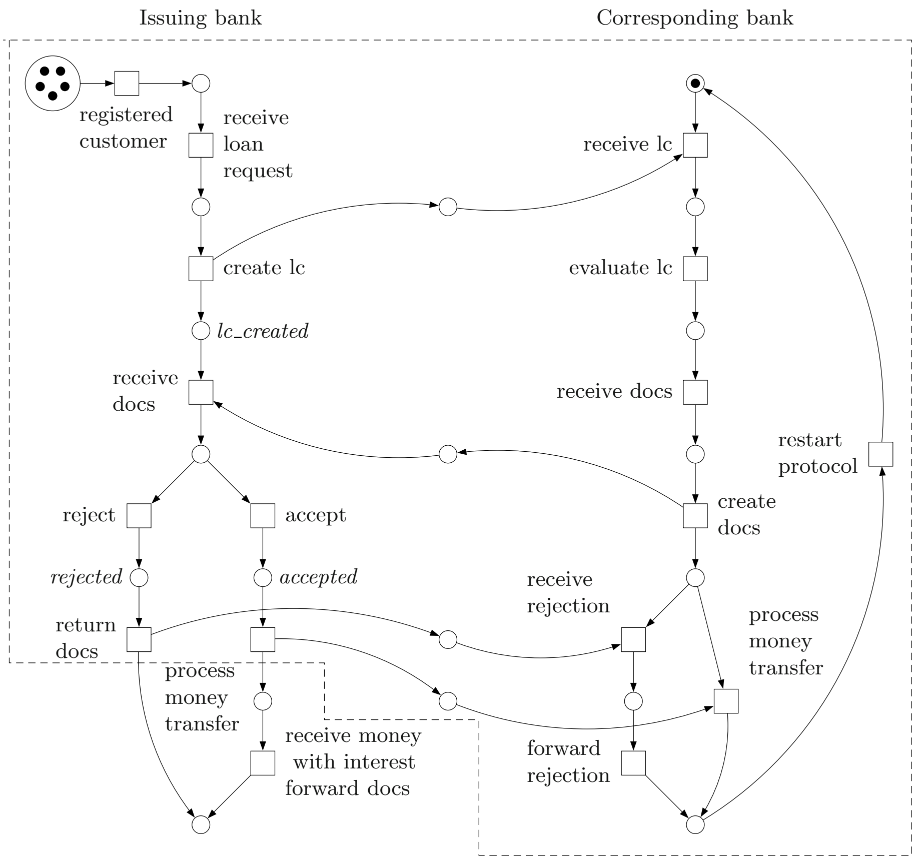


Fig. 1. An electronic trade system (excerpt)

buyer's loan request, the issuing bank informs the correponding bank by means of a letter of credit (lc) and requests shipment documents (bill of lading, etc.) in return. It then makes a decision to either reject the request, or to accept it and provide the credit. In the former case, the documents are returned to the corresponding bank, while in the latter case, money is transferred so that the shipper can be paid, and finally, payment with interest is received from the consignee.

The net shown in [8] is acyclic, since it essentially describes a choice between a few use cases. In order to make its analysis more complicated (and perhaps more interesting), we introduced the possibility that up to five registered customers may request a loan concurrently. Moreover, we set up the corresponding bank's activity as a cyclic deterministic protocol.

We may want to verify the following temporal logic formula:

$$\varphi_1 \quad = \quad \Box(\,(\mathit{lc\_created} > 0) \rightarrow \Diamond\,((\mathit{rejected} > 0) \vee (\mathit{accepted} > 0))\,),$$

according to which it is always the case ($\Box$) that if one of the customers is at *lc_created*, then at some later point in time ($\Diamond$) some customer (not necessarily the same one) will be at *rejected* or at *accepted*. We choose

$$P = \{lc\_created, rejected, accepted\}$$

and compute the slice according to the above algorithm, obtaining the subnet within the dashed borders. In the full net, 1778 states (i.e., markings) are reachable via 5096 state transitions, whereas the slice has only 896 states and 2275 state transitions. Thus verifying the formula on the slice is approximately twice as fast as verifying it on the full net. After verifying φ_1 for the slice, we know that it holds for the full net as well, since all of the latter's executions are fair towards the slice.

3.2 A Manufacturing Example

This example (cf. Figure 2) is taken from [5]. A customer asks the support department of a manufacturing company for information about a certain product. Support then files a request to the production department, asking for information about this product, and awaits the answer. Upon receiving the information, support may either continue to request for more information (asking, for instance, for some more details), or thank for the answer and proceed to provide the appropriate information to the customer.

Consider verifying the formula

$$\varphi_2 \quad = \quad \Box(P\_end \rightarrow (\Box\, \neg request)),$$

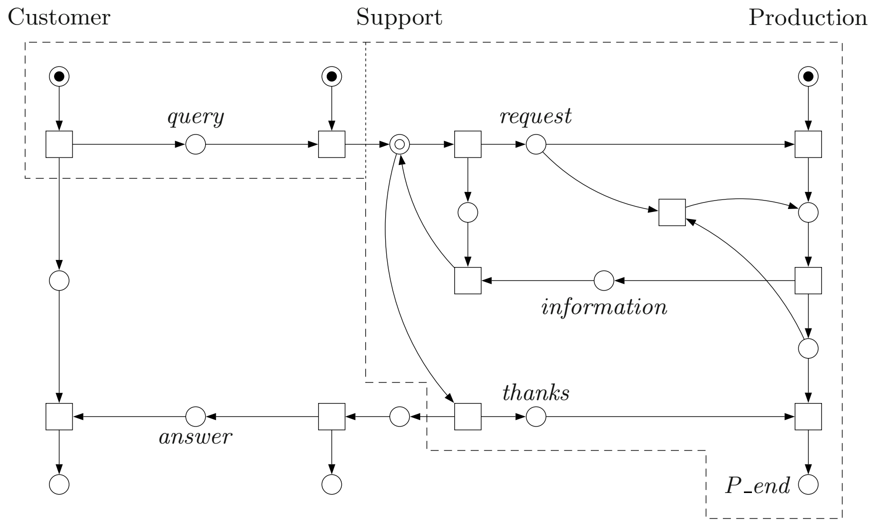


Fig. 2. A customer / support / production system

which means that whenever control is at P_end, there will be no more tokens on *request*. Slicing with $P = \{P\_end, request\}$ yields the subnet shown within dashed lines. The full net has 17 reachable states and 19 state transitions, while the slice has 11 states and 11 state transitions.

This example shows that slicing can usefully be combined with another reduction method [10] which is based on articulation points (that is, nodes of a graph that connect two or more otherwise unconnected parts of the graph). Note that the slice has articulation points, e.g. the place marked with the hollow token. Because of this, it can be further reduced, yielding a smaller net (to the right of the dotted line in Figure 2 and with the hollow token) with only 9 states and 9 state transitions, on which the formula can be checked and the result be transferred to the same formula on the full net. Note that the reductions cannot be done in the other order, since the full net does not have articulation points.

3.3 A Business Process for Dealing with Insurance Claims

The Petri net of Figure 3 models a workflow as described in [1]. An incoming claim is recorded first. A claim may be accepted or rejected, depending on the insurance cover. For a rejected claim, a rejection letter is written. If the claim is accepted, emergency measures, if necessary, are provided. After an assessment – possibly done by an expert – a settlement is offered to the customer, who may

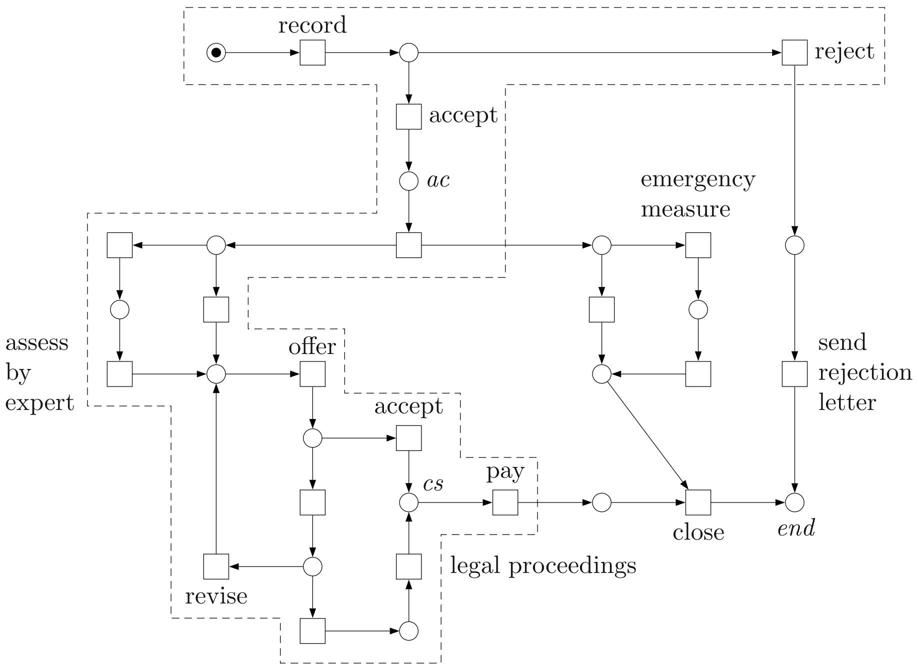


Fig. 3. Insurance claim

either accept or reject. A rejected offer may be followed by legal proceedings or a revision. If a settlement is agreed upon, money is paid.

We want to verify that every accepted claim is settled, i.e., in temporal logic terminology:

$$\varphi_3 \quad = \quad \Box\,(ac \rightarrow \Diamond\, cs).$$

The slice of this net, for $\{ac, cs\}$, is again shown as a subnet within dashed borders. In the full net, 29 states (i.e., markings) are reachable via 60 state transitions, whereas the slice has only 11 states and 14 state transitions. Here also, the algorithm described in [10] can be used to reduce the slice still further.

4 Concluding Remarks

The slicing algorithm works quite well for business process Petri net models, since they often are not strongly-connected. As a benchmark for the effect of slicing on other nets, we investigated the well-known collection of `Promela` examples of [4]. This collection comprises 75 case studies, some of which are scaled up instances of the same problem type (e.g., 6, 8, 10 and 12 dining philosophers). In total, there are 23 different types of examples. For our experiments the biggest instance of each scalable type has been removed to reduce the overall computation time. For each place of the net, a slice was generated. In case the slice is smaller than the original net, its reachability graph was compared with the reachability graph of the original net. We set an upper bound of three hours for the state space generation. With such a generic test setup we were able to analyse other test sets fully automatically, since we did not need to analyse each net to find a meaningful slicing criterion in terms of a formula. But since we did not provide such a formula, we filtered the generated slices. We considered slices with more than 20 states or 10% of the states, or with less than 85% of the states and state transitions as “interesting”, and slices that were not equal to the full net as “nontrivial”. Of the 23 types, almost all had nontrivial slices and 6 had interesting slices. In average, the latter covered 67% of the places (with a median of 69%). This was actually more than we expected. Moreover, as we have seen in this paper, the slicing algorithm can produce added value by being daisy-chained with more sophisticated net reductions.

Acknowledgement

We thank Jörg Desel for reading and commenting on the paper.

References

1. van der Aalst, W., van Hee, K.: Workflow Management - Models, Methods, and Systems, p. 62. The MIT Press, Cambridge (2002)
2. Brückner, I., Wehrheim, H.: Slicing Object-Z Specifications for Verification. In: Treharne, H., King, S., Henson, M.C., Schneider, S. (eds.) ZB 2005. LNCS, vol. 3455, pp. 414–433. Springer, Heidelberg (2005)

3. Clarke, E.M., Grumberg, O., Peled, D.A.: Model Checking. The MIT Press, Cambridge (1999)
4. Corbett, J.C.: Evaluating Deadlock Detection Methods for Concurrent Software. IEEE Transactions on Software Engineering 22(3), 161–180 (1996)
5. Kindler, E., Martens, A., Reisig, W.: Interoperability of Workflow Applications. In: van der Aalst, W.M.P., Desel, J., Oberweis, A. (eds.) Business Process Management. LNCS, vol. 1806, pp. 235–253. Springer, Heidelberg (2000)
6. Mayr, H.C.: Towards Business Process Modeling in KCPM. In: Desel, J., Oberweis, A., Reisig, W., Rozenberg, G. (eds.) Petri Nets and Business Process Management, Dagstuhl-Seminar-Report Nr. 217 (1998)
7. Mayr, H.C., Kop, C.: A User Centered Approach to Requirements Modeling. In: Glinz, M., Müller-Luschnat, G. (eds.) Proc. Modellierung 2002, GI edn. Lecture Notes in Informatics (LNI), vol. P-12, pp. 75–86 (2002)
8. Lee, R.M.: Documentary Petri Nets: A Modeling Representation for Electronic Trade Procedures. In: van der Aalst, W.M.P., Desel, J., Oberweis, A. (eds.) Business Process Management. LNCS, vol. 1806, pp. 359–375. Springer, Heidelberg (2000)
9. Rakow, A.: Slicing Petri Nets with an Application to Workflow Verification. In: Proc. SOFSEM 2008. LNCS, vol. 4910, Springer, Heidelberg (2008)
10. Rakow, A.: Slicing and Reduction Techniques for the Verification of Concurrent Systems (preliminary title). PhD Dissertation (2008)
11. Weiser, M.: Program Slicing. In: Proceedings of the 5th International Conference on Software Engineering, pp. 439–449. IEEE Press, Piscataway, NJ, USA (1981)

Public Adoption of Digital Multimedia – Why Is It Lagging behind Expectations? (Extended Abstract)

Laszlo Böszörmenyi

Department for Information Technology
Alpen-Adria Universität Klagenfurt
laszlo@itec.uni-klu.ac.at

Abstract. The talk will address a number of problematic issues in current multimedia technology and try to give some positive answers. The term multimedia is used for a mixture of modalities, at least one of them being continuous. In the center of interest are "natural" videos, taken by cameras.

Keywords: Distributed Multimedia, Streaming, Multimedia Delivery, Multimedia Presentation.

1 Multimedia Everywhere – A Dream or a Nightmare? Some Scenarios

1. I wish to watch a movie I do not know the title and authors exactly of; I know only some impressions, told by my neighbor. The dream: My video system finds me the required movie, which I can watch in perfect quality. The nightmares (there are many of them): (a) My system offers me 100 hours movies that could fit – however they do not. It lies on me, after how many hours do I give up. (b) The system noticed what kind of movies I seem to like and do not stop recommend me movies in many hours length. (c) I am watching the movie in perfect quality, but police bells at my door and brings me into jail because of watching movies illegally.
2. I have three 2-hour movies of my own. I am looking for two interesting, 2-minutes-long scenes, which I would like to present at a talk. The dream: My video system helps me to find, cut out and mix the two scenes in 5 minutes. The nightmare: I am spending the whole night by stepping fast forward and backward, resulting in cutting out only one of the required scenes, missing the start but taking longer than required.
3. I know exactly the URL I have to click on. The dream: I do so and watch the required clip. The nightmare: I do so; then wait 5 minutes to start; after 1 minute I get blurred pictures for a while, after further minutes the sound is breaking down etc.
4. I am watching a movie, everything goes well, but I have to leave. I would like to redirect the movie to my handheld device, and watching it on the train. The dream: I just let the session migrate to my handheld and resume after entering my train.

The nightmare: After migration the movie is resumed at the wrong position, with a wrong resolution and with a lot of interrupts.

5. I want to design and implement a new multimedia application, relying on existing software. The dream: I take the specification of the components I want to assemble in a certain way and have in a month a reliable new service, with the required quality of service. The nightmare: I start to try to understand several hundreds of thousands lines of C-code and give the project in two years up.
6. I have a mobile guide in my pocket, which helps me at traveling. I just arrive in Japan (not knowing a single word Japanese). The dream: My device tells me in my own language, how I can find the hotels, restaurants and museums I presumably like. The nightmare: My device switches to Japanese and tells me something I do not understand and shows me pictures I cannot see, because their format is not compatible to my device.

Everybody is invited to continue the game of finding new dreams – and the corresponding nightmares. By the way – the latter makes much more fun.

2 Motivation

Digital multimedia technology has developed considerably in recent years. Nevertheless, we still cannot speak from a real breakthrough, as for example it had been the case with graphics in the early Eighties. Graphics became in a few years an integral part of all computer systems, whereas multimedia – especially continuous media – are still staying rather aside.

At the end of the Seventies, the Alto computer was developed at Xerox PARC, which was not only one of the first usable personal computers, but it was the first computer providing pixel graphics. Graphics had been before something very expensive, for exceptional applications (such as architecture), supported by special hardware. Pixel graphics made the technology essentially software driven (first, with some microcode support), and thus cheap and flexible. Nevertheless, it took many years, until it became a pervasive technology. After a lot of research, the commercial breakthrough was brought due to the Apple-Macintosh. Many companies (such as IBM and DEC) were fairly long skeptical against graphics, and lost a lot due to this hesitation. It is important to note that it is not pixel graphics alone, which won this battle, but a combination – or much rather: integration – of three technologies: graphics, window-management and object orientation. They together could serve as a basis for some absolutely "killer" applications, such as a totally new way of user-machine interaction, and flexible document processing including drawings and fonts with variable size and shape (first in black and white, a bit later in color). What can we learn from this lesson? We can observe the following key points:

1. It was an integration of several technologies.
2. Most of these technologies were very innovative, even the reinvention of object-orientation can be regarded as such.
3. It took several years until the technology became widely expected.
4. The new (combination of) technologies were strong enough to enforce fundamental changes in hardware, operating systems and even applications.

Actually all applications had to change there view of interaction – instead of putting messages onto a statically defined screen, they had to adapt their outputs to the actual window size (usually given as the width and height parameters of a method, called "redraw").

If we take a look at the same points in case of multimedia, we can observe the following:

1. Integration of technologies
In this point the development goes even much further than with graphics. Probably this is exactly the reason for a slow-down in the dispersion. A really huge amount of technologies ought to be consolidated, beginning from film and video production, over compression, storage, delivery and presentation issues, up to questions like digital right management.

2. Innovation
Regarding innovation, the landscape is disappointing. Really new, "killer" applications are rare or rather not existing yet. Applications like YouTube have surely their merits, but they cannot be called really innovative. Soft-video players look very similar to video recorders from the Sixties, they do not offer essentially new functionality, but do offer mostly less – and still worse, varying – quality.

3. Dispersion time
The dispersion of multimedia takes definitely longer than that of graphics. This might cause some resignation in some researchers and companies. Five years ago many experts were fairly sure that the breakthrough comes soon. The same can be said today as well – however with less credit, due to the time went in between. Nevertheless, it is hard to believe that digital multimedia remains a side-effect.

4. Power to change existing technologies
Multimedia – especially video – needs huge storage capacity, and data paths providing guaranteed quality of service (QoS). The first point is very much in trend; the second one is, however, still at the beginning. The whole previous development of computer and network technology went into another direction. All usual operating systems and networks rely essentially on the "best effort" principle and fit therefore principally bad for video delivery. That video applications work at all is almost a miracle – and can be explained only due to massive over-allocation of resources. There is some movement on the network side (such as the integrated and differentiated services, implemented by some newer routers), but most working solutions apply more or less "nasty tricks" on the application level. In the area of operating systems almost nothing happens – even research is hesitating strongly in this issue. An operating system, which allows opening a stream and delivering it corresponding to a set of QoS parameters, is still a dream. Multimedia applications behave typically greedy. Even if they have some notion of QoS control, this is used only in the context of the given application, but hardly in a federative way. For this, a generally accepted cost model would be necessary.

There are some further challenges, that must be handled if multimedia should become pervasive.

5. Semantic gap
Digital graphics is based on computer-generated mathematical descriptions. Thus, the semantic gap between internal representation and human vision is not as big as in the case of "natural" multimedia. The internal representation has notions of geometrical objects, such as "circle" or "triangle". In the case of pictures taken by a camera, we have at first nothing than a matrix of pixels – resp. a sequence of them, in the case of a movie. The semantic gap is one of the main issues. It is a real challenge to find the proper video sequence in a large archive, in reasonable time.

6. Delivery
Even if we find the real video sequence, we are still confronted with the delivery problem. To deliver a video from any point of the world to any target, in exactly that quality as the user requires it, is a major challenge.

7. Presentation
Text processing tools, like Word, have found a way of presentation, which corresponds both to the content and to the user requirements sufficiently. Surely, this development took also fairly long, but can now be regarded as solved. The presentation of continuous data, especially of video, and their integration with other modalities is unsolved. The paradigm of the sequential video tape seems to be extremely resistant against any innovation. However, apart from pure entertainment this implies a number of technologically unnecessary restrictions.

8. Software engineering
One of the barriers in the development is the lack of good engineering methods and tools for QoS-aware software development. A lot of research has been done, but the state-of-the-art is by far not satisfactory. If applications behave greedy instead of federative, this is partly, because the programming is too cumbersome. We could imagine a programming language, in which we can not only assign a value to a variable but also a stream to a screen – under automatic QoS control.

9. Digital right management
Digital right management is not a new issue, but gets a new dimension with multimedia. This seems to be one of the hardest nuts in the area. Actually it is that hard that we do not consider it further in this talk.

3 Detailed, Exemplary Discussion of Three Issues

In the talk only four of the above topics will be discussed in detail.

3.1 The Semantic Gap

Pictures, seen by people, are *not* arrays of pixels. The pixel representation has significant advantages at the last stage in the life-cycle of pictures: at presenting them on a screen. In any other cases, this representation is rather disadvantageous. This is true even for pictures, used by robots and embedded systems. Currently, the only way to cope with the semantic gap is using metadata. However, detailed metadata are usually not considered as being an integral part of multimedia data. The slow or even missing acceptance of MPEG-7 and especially that of MPEG-21 show a good example for this. On the other hand, some innovative techniques, such as cooperative annotation, promise new insights into the possibilities of handling semantics as a product of human cooperation. A further interesting approach could be looking for higher-level mathematical representations of pictures – maybe as an additional level between pixels and metadata. Such techniques have been already addressed to a certain extent in different research directions, such as video retargeting or automatic scene generation.

3.2 Video Delivery, Proxy-to-Proxy (X2X) Streaming

Delivering videos in good quality can still be regarded as an unsolved problem. If we wish to stream a video from an arbitrary source to an arbitrary target then the perceived quality is typically a question of luck. In the general case; streaming over the internet, the chances for good luck are rather small. Moreover, different approaches can be hardly compared, as no generally accepted metrics exist.

The problem can be best characterized with the help of two extreme cases – which hardly occur in pure form in practice, but show well the essence of the difficulties.

1. Pure client/server, without caching.

Videos are *not replicated* and on each client request the video is streamed from the original source. *Sharing* of the content is maximal – all clients share the same single copy. Beside the well-known problems of the client/server model – poor scalability and single point of failure – we have an additional difficulty, in the case of videos, which could be called the *long distance problem*. Even if the server is only lightly loaded, streaming over a long path (let the length being measured simply in hops) may provide unacceptable poor quality.

2. Pure download and play.

Each client creates an own copy before playing back. The quality of the play-back is maximal – we may assume that the local disc and operating system have resources abundantly. However, beside the obvious problem of large *start-up delay* – each video has to be fully replicated before played back – the approach wastes a lot of resources, both at the clients and on the network. This is, because now, *sharing* of the content is minimal.

Improved video delivery methods try to find a good balance between these two extreme cases. They create a certain number of *replicas*, try to *locate* these near to clients and let the clients *share* the replicas. More formally stated; we are looking for an optimum in the space of replication and sharing possibilities: How many replicas should be placed, where and how long? Different, well-known techniques, such as Client/Server-, Contend Delivery Network- (CDN) and Peer-to-Peer- (P2P) based streaming can be placed differently in this optimization space. None of these approaches addresses the issue as a general optimization problem; rather they either postulate explicitly, or – still worse – assume implicitly some basic conditions, under which the method works well.

We define a model of the content delivery problem, based on a set of simple *metrics*. This allows us to model different kinds of behaviors (such as CDN-like and P2P-like) and to compare them quantitatively. The model assumes three kinds of components: (1) videos (2) proxies and (3) clients. Proxies serve as possible target locations for holding replicas, and form resource-sharing groups. Each component has a so-called *affinity* function, and "strives for a world" with maximal affinity. The affinity function is different for each component-type, but the affinity-based behavior is common: (1) Videos, entering the system, let themselves replicate at places, to which they have a high affinity (hoping many interested consumers there). (2) Proxies, entering the system join a group of proxies, to which they have maximal affinity – promising similar content, good network connections and well-balanced load. The "similarity" of content is not investigated further; we just assume that it is possible to differentiate between semantically near resp. far content. (3) Clients have a double connection to proxies. On the one hand, they have a "home" proxy, to which they are directly connected – typically, but not necessarily via a LAN. On the other hand, if they request for a video, they find that source proxy group, to which their request has the greatest affinity – i.e. with the highest chance not only possessing the required video, but also having it in the required quality and also having the necessary streaming capacity. By computing the streaming capacity, the possibility of multiple-source streaming is considered, if necessary.

Based on this model, we define a new approach, called Proxy-to-Proxy (X2X) streaming, which tries to make the best of CDN and P2P. Before getting into details let us describe shortly the basic restrictions of the well-known approaches and to show, where the improvements of X2X principally lie.

3.3 Non-linear Video Presentation and Browsing

With the dispersion of video technology, not only the number of videos is growing fast, but also the kinds of their usage. Besides entertainment, a large number of application domains, such as learning, health-care support, and disaster management rely more and more on audiovisual material. A common characteristic of these emerging applications is that videos are not just watched sequentially.

We are often not interested on the entire "movie", we rather need to find certain video segments as fast as possible. A very important special case is that we want to be able to identify very quickly a video we do *not* need. Tools for efficient and user-friendly navigation both in video archives and inside of single videos are urgently required. Even modern soft video players operate in a manner as invented for traditional VCR devices in the 1960s, mainly for entertainment. They provide such well known interaction features as play, pause, stop and fast forward/backward. Additionally, they usually offer a timeline which allows more or less precise jumping to a particular point in time within a video. Finding a specific segment within a large video file still remains a difficult and time consuming task.

To improve the interactivity of digital video, a lot of research has already been done in the area of video abstraction (also called summarization). The purpose is to create a short summary of a long video. Video abstraction is mainly used to facilitate browsing of large video-databases, but can also enable non-linear navigation within a single video sequence. However, video abstracts alone do not solve the interaction problems linked to video data. We present our concepts and work done so far for a flexible video navigation tool, which is a step towards more efficient and powerful interactivity in video data. Beyond its simple interaction mode which is similar to a common video player, it provides features, such as (1) skimming roughly through the video, (2) watching an on-demand created video abstract/skim of a particular sequence or of the entire video, and (3) several presentation modes of different scenes in parallel in time and/or in space. Moreover, the tool allows (1) to extract representative parts of the video (e.g. key frames) very efficiently, and (2) to dynamically switch between several presentations modes. Instead of enforcing any "best" presentation mode, we extend the currently running prototype to enable experiments with larger groups of non-expert end-users, in order to find different modes for different needs.

3.4 Adaptive Programming

Multimedia streaming and play-back is the periodical process of dealing with long sequences of data (e.g. video frames) under so-called "soft real-time" constraints. The implementation of this substantial pattern of common multimedia applications is cumbersome and error-prone. Instead of defining QoS and adaptation capability as an external aspect of an application, we suggest additional features at the programming language level. A set of minimal language extensions allows us to encourage implementation of quality-aware applications. Our work concentrates on the following feature extensions for a general-purpose system programming language (actually implemented for C#):

1. An additional dimension can be added to any data type. This n+1st dimension represents the time.
2. To take use of the time dimension, we introduce a special assignment operation (called streaming) and the new, streaming mode for passing method parameters.
3. A new declarative syntax, embedded in a common programming language is used to define QoS specifications.

If a streaming operator (assignment or parameter passing) violates a QoS specification then compiler generated code raises an exception automatically. The designer of an adaptive program can concentrate on the essential business logic, and QoS violations can be handled as usual exceptions. If e.g. the QoS constraint specifies a top and a bottom level for the frame rate, then an individual exception is generated if the actual frame rate is too high resp. too low. In the corresponding exception handler the programmer may simply slow down in the first case (a simple implementation of a leaky token bucket), or may switch to an alternative stream with lower bandwidth requirement in the second case. To our knowledge, no similarly simple method exists for handling this fairly complex issue. The language elements are used further to define common patterns for adaptive programming.

Model-Driven Security Engineering of Service Oriented Systems

Ruth Breu[1], Michael Hafner[1], Frank Innerhofer-Oberperfler[1], and Florian Wozak[2]

[1] Universität Innsbruck, Technikerstrasse 21a, 6020 Innsbruck, Austria
{ruth.breu,m.hafner,frank.innerhofer-oberperfler}@uibk.ac.at
[2] University of Health Sciences, Medical Informatics and Technology, 6060 Hall in Tirol, Austria
florian.wozak@healthatnet.at

Abstract. In this paper we present the main steps of a process for the systematic development of security critical inter-organizational applications. We employ models of the functional aspects of the system at different layers of abstraction for analyzing security requirements, risks and controls and for generating executable security services.

Keywords: Security engineering, security analysis, service oriented architecture.

1 Introduction

In the recent years the paradigm of service oriented systems has created visionary business scenarios enabling novel ways of cooperation between business partners, outsourcing parts of business processes, or sharing data in heterogeneous environments. IT applications of the new generation are highly interconnected and concurrent, involve electronic devices of many kinds and stakeholders in manifold roles and interrelations. As a consequence novel challenges to the engineering of such systems have emerged. This includes the increasing need for **validating compliance** and methods for the **systematic design of secure architectures**. In this paper we present a tool-supported framework for the development of secure service oriented solutions. Our approach is model-driven in the sense that we tightly integrate aspects of security with the functional views of the system ranging from the business level down to system design.

The two basic building blocks of our framework are a method for the **model-based security analysis** of service oriented systems and a development environment for the **model-based configuration of security services**. This paves the way for a traceable process from requirements engineering down to the high-level realization of secure solutions.

Sample applications for our method are health care networks. Health care networks support cooperation between stakeholders in the health care domain like hospitals, general practitioners and the patient. As running example we will use the system health@net, an Austrian initiative to develop concepts and an implementation of distributed cross-institutional health data records [21,19]. Other examples may be cooperative traffic scenarios (involving traffic control centres, drivers or car manufacturers) or applications in e-procurement.

More general, the following assumptions are characteristic for the kind of systems we consider. First, the networks are highly dynamic, both concerning stakeholder types, stakeholder instances and the workflows to be run. Second, the stakeholders are highly heterogeneous in their organizational structure, security requirements and security infrastructure (e.g. comparing hospitals and general practitioners). Third, there may be a high number of stakeholder instances (e.g. millions of patients, thousands of hospitals) requiring complex infrastructures and effective engineering techniques. Finally, many of the systems adhere to legal regulations and standards like the Data Protection Act and the IHE IT Infrastructure Technical Framework [3] in the health care domain.

From these basic assumptions we derive a set of major goals for our framework.

Modularity

– Different levels of abstraction can be analyzed independently of each other (e.g. separating organizational requirements from technical requirements)
– Different subdomains can be analyzed independently of each other (e.g. separating the analysis of the organizational structure of hospitals and general practitioners)
– The notions of requirements, risks and controls are clearly separated and may be considered independently of each other

Traceability

– Security aspects can be traced along the levels of abstraction starting with general security objectives (which may be derived from legal regulations) and arriving at the implemented security controls. Security controls may range from organizational rules (e.g. four eyes principle) to technical components (encryption, firewalls).
– The analyzer is provided with aggregated information about the state of the security analysis process at any time.

Model-Driven Configuration of Security Services

– Secure solutions can be realized based on high-level security patterns. The security patterns are attached with model elements of the functional system models and configure executable security services in a target architecture.

The key idea of our framework is that we put any security related aspect in the context of the functional system view and the functional model elements drive the whole analysis process. This enhances the security analysis to a level of preciseness and completeness which is beyond existing pragmatic approaches.

The remaining sections are organized as follows. In Section 2 we present the security analysis method ProSecO. Section 3 presents a short overview of related work and a conclusion is drawn in Section 4.

In this paper we give an overview of the security analysis method. For a detailed presentation of the SECTET platform for model-driven configuration of security services we refer to [11,6]. In the sequel we assume the reader to be familiar with general notions of software development and software process models like business processes, services and actors.

2 Security Analysis of Service Oriented Systems

Information security is generally defined by properties like confidentiality, integrity, availability, non-repudiation and authenticity of information [15,4]. For a thorough security analysis not only the information objects but also the applications, business processes and the organization supporting the creation and processing of information have to be considered. We recognize this fact by analyzing the entire information processing context including business and technical aspects. In our model all basic security concepts like security objectives, security requirements and risks are bound to model elements, thus enabling systematic tracing along dependencies.

In Subsection 2.1 we shortly present our system view targeted towards the modeling of service oriented systems. In Subsection 2.2 we enhance this view by security related concepts and sketch the analysis process in Subsection 2.3.

2.1 Functional Models

Functional system views in ProSecO are described by interrelated model elements according to the ProSecO meta model. The meta model elements are classified along two orthogonal categories.

- **Level of Interaction:** The *Global View* describes aspects related with the interaction of different stakeholders (i.e. autonomous partners in the network), whereas the *Local View* describes aspects related with the behaviour and structure of a specific stakeholder.
- **Level of Abstraction:** We distinguish three levels of abstraction in our meta model, the *business level* describes business requirements, whereas the *application level* and the *technical level* are concerned with the executable solution.

The ProSecO meta model is described in a two layered way where each layer consists of meta model elements of the three abstraction levels. Fig. 1 shows the *Global System Meta Model*, Fig. 2 the *Local System Meta Model*. The main concepts and their interrelationships are explained below.

Institution (Global System Model, Business Level)**.** An institution instance represents an autonomous partner in the service oriented application.
Examples: Hospital, General Practitioner, Pharmacy

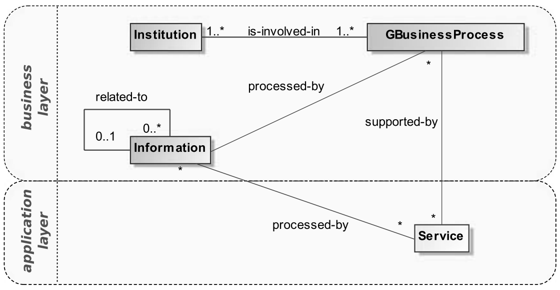


Fig. 1. Global System Meta Model

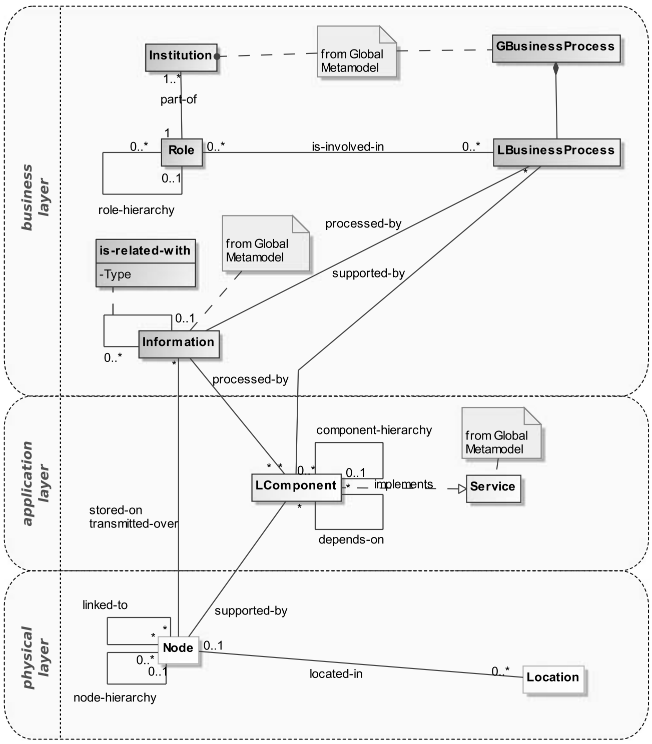


Fig. 2. Local System Meta Model

Role (Local System Model, Business Level)**.** A role represents an actor in the fine-grained view of an institution.
Examples: Physician, Nurse in a Hospital

Information (Global System Model, Business View)**.** Information is an abstract concept to specify information object types. Information classes may be related in various relation types such as association, composition or inheritance.

Examples: Patient, Health Record

Business Process (Business Level)**.** A Business Process is a type of interaction between stakeholders. We distinguish *Global Business Processes* modeling the interaction of institutions and *Local Business Processes* modeling more fine-grained interactions within one institution as part of a global business process.

Business processes are related with the type of information processed during the interaction (e.g. as input or output). Moreover, global business processes are related with the institutions involved, local business processes with roles and the global business process they are part of.

Examples: The Global Business Process *ReleasePatientData* describes the process of releasing patient related information to other stakeholders, the Local Business Process *Internal Release Process* describes the internal steps in a hospital necessary to release information

Service (Global System Model, Application Level)**.** Services are executable components available in the network. Services are related with input and output information object types.

Examples: The services *retrievePatientData* and *releasePatientData*

Local Component (Local System Model, Application Level)**.** A component represents an application residing on some organizational domain. A component may implement a service, may be part of other components or may depend on other components and processes information objects. Moreover, local components are linked with the local business processes they support.

Node (Local System Model, Technical Level)**.** A node represents technical or physical objects that are either used to store information objects (e.g. file server, usb stick, plain paper) or to run applications. The concept of a node may be hierarchically composed and nodes may be linked with other nodes. Moreover, nodes can be linked with the components they run and the physical **Location** they reside on.

2.2 Security Concepts

In our method model elements of the Functional Models are associated with security related information as shown in Fig. 3.

Security Objective. A Security Objective describes the overall security goals of the system, in particular general legal requirements, specific availability, integrity and confidentiality requirements. Security Objectives are defined for a set of model elements that are dependent on each other.

Security Requirement. A Security Requirement is a detailed context-dependent explication of a Security Objective. It breaks a Security Objective down in several more detailed descriptions. The context of a Security Requirement is derived from the model element for which it is defined. Security Requirements are linked to Security Objectives to depict their paths

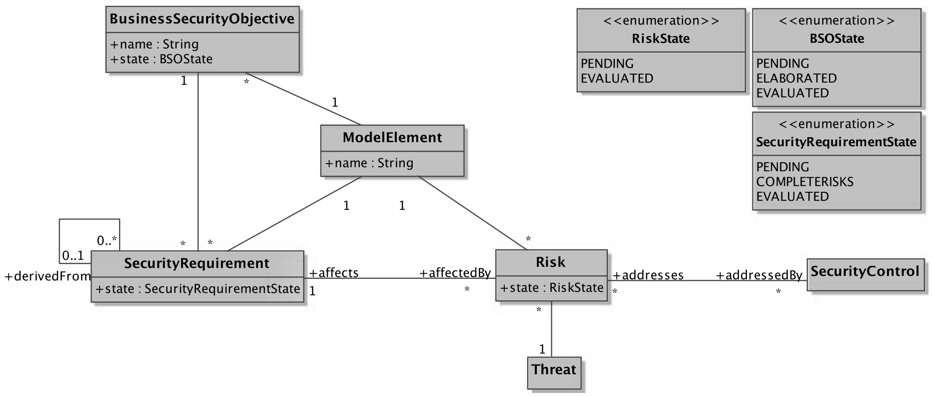


Fig. 3. Security Meta Model

of inheritance. In early stages security requirements are described informally by text.

Threat. A Threat is the description of an adverse event that is considered as potentially having a negative impact. A Threat by itself is not interesting for our analysis, it only becomes relevant, if we further identify a targeted model element and a related security requirement. Once the threat has been assessed and estimated regarding its impact, it becomes a risk.

Risk. A Risk is therefore defined as a triplet consisting of a targeted model element, a related security requirement and a threat that potentially undermines the requirement, including an assessment of its severity.

Security Control. A Security Control is any measure or safeguard that has been put in place to mitigate the identified risks.

Both business objectives, security requirements and threats have a state indicating the state of the security analysis process. More information on the state is given in the subsequent sections.

At each point of time during the security analysis, the system is described by a set of interrelated model elements, where these model elements either adhere to the Functional Model Types of Fig. 1 and Fig. 2 or to the Security Meta Model of Fig. 3. We call each such set of interrelated model elements a **Security Model**.

2.3 The Security Analysis Process

The task of the Security Analysis Process is to support the security analyst in developing, evolving and analyzing Security Models. The core of the ProSecO Analysis Process is depicted in Fig. 4.

Each instance of the process is executed in an iterative way where each iteration is initiated by a set of possible triggers, mainly time events (e.g. the security process is executed periodically) or a state change in the Security Model. Relevant state changes may be the creation of new model elements (e.g. new

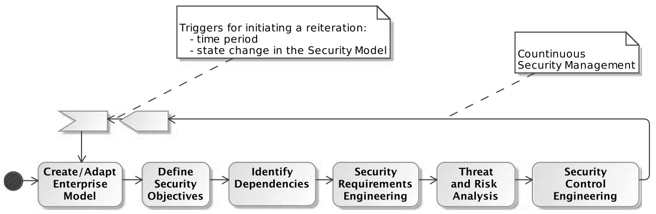

Triggers for initiating a reiteration:
- time period
- state change in the Security Model
Countinuous Security Management
Create/Adapt Enterprise Model
Define Security Objectives
Identify Dependencies
Security Requirements Engineering
Threat and Risk Analysis
Security Control Engineering

Fig. 4. The ProSecO Analysis Process

services), the identification of new threats or requirements and the implementation of security controls.

The security process depicted in Fig. 4 is a **micro-process** which means that it may not be executed on the whole Security Model but on submodels. We call such a sub–model a *security domain* and the responsible for the related micro–process the *security domain responsible*. Security domains related with the Functional Meta Model may be the Global System Model and submodels related with each institution.

Since each security micro–process may modify the state of related security information (e.g. new requirements and risks are added) and the model elements of different security domains in general are interrelated the micro–processes are not independent but interact with each other. As an example, the identification of a new risk at the local level of some institution is likely to have effects on the security evaluation of the whole network. The output of each security micro–process is the subsequent modification of the Security Model as described in the following subsections.

Create or Adapt Functional Model. The first step in each micro-process is the *creation* – or in the case of a reiteration cycle the *adaptation* — of the domain.

For instance, new services may be added and linked with other model elements or the technical infrastructure is changed. In this step it is not yet necessary to complete the links to other model elements. Correspondingly the state of new elements is set to *pending* or *elaborated*.

Define Security Objectives. A Security Objective can be based on legal requirements or on business considerations. Security Objectives are useful for establishing a clear goal that is understandable at all levels of an organisation. Typically such a Security Objective is attached to a model element of the business level (e.g. a business process).

The Security Objective's purpose is not only the communication of the goal of a security management effort, but it also serves as a guidance on the formulation of concrete security requirements. It is possible to define many Security objectives per domain.

New Security Objectives are associated with the state *pending*.

In our case study we define the Security Objective that research groups can have anonymized access to patient related health information and associate this Security Objective with *Research Group* as an instance of the *Institution* meta model element.

Identify Dependencies. With the definition of Security Objectives we have defined the areas of interest in the Functional Model. Each of the model elements that has a Security Objective attached is the root element of a separate scope called a *dependency graph*. The dependency graph is a non-cyclic graph of model elements. The dependencies are identified following a top-down-approach along the levels of abstraction of the Functional Meta Model. In the case of a global business process we

- identify the institutions involved in the process
- analyse ingoing and outgoing information and
- the services supporting the process
- relate the services with the implementing components
- relate these components with the hosting physical nodes and their location

In order to identify the dependencies the business processes may be refined based on common modeling techniques such as BPMN [2].

Fig. 5 shows a sample dependency graph starting with the *Research Group* institution leading to the business process *RetrievePatientData* and *Diagnosis* and *Prescription* as processed information types.

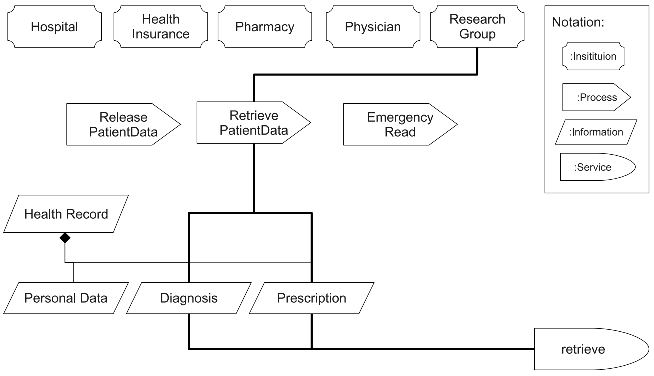
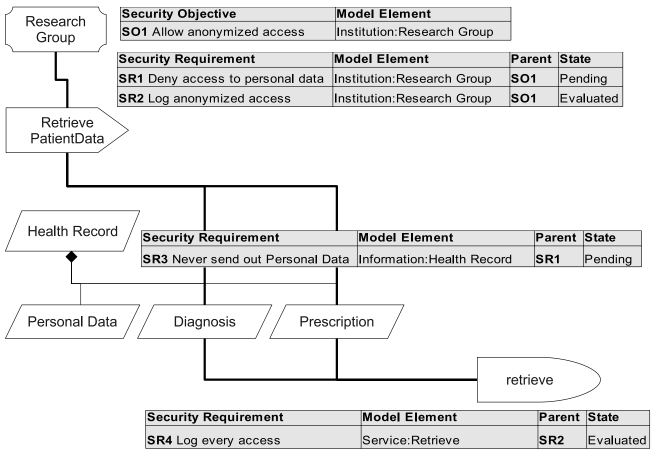


Fig. 5. Identified Dependencies

Security Requirements Engineering. During security requirements engineering a general security objective attached with some model element is broken down into concrete requirements based on the model element's dependency graph. Security requirements engineering is done in a top-down way where the security requirements of lower level model elements inherit the security requirements of upper level model elements.

In our case study we break the anonymization objective down to security requirements attached with institutions, information types and services as shown in Fig. 6.



Security Objective	Model Element
SO1 Allow anonymized access	Institution:Research Group

Security Requirement	Model Element	Parent	State
SR1 Deny access to personal data	Institution:Research Group	**SO1**	Pending
SR2 Log anonymized access	Institution:Research Group	**SO1**	Evaluated

Security Requirement	Model Element	Parent	State
SR3 Never send out Personal Data	Information:Health Record	**SR1**	Pending

Security Requirement	Model Element	Parent	State
SR4 Log every access	Service:Retrieve	**SR2**	Evaluated

Fig. 6. Sample Security Requirements

As a result of the elicitation of security requirements we obtain a non-cyclic graph of security requirements, where each requirement is attached with some model element and the parent-child relationships are induced by the dependency relations between the attached model elements. Note that security requirements at the same level are connected by a logical "and" which means that a parent security requirement enforces the fulfilment of all children security requirements. The requirements graph involves typically model elements at the business level.

At the stage of definition the state of a security requirement is set to *pending*. The state of the corresponding security objective is set to *elaborated*.

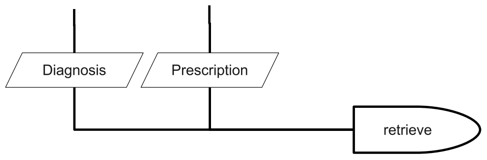
Threat and Risk Analysis. While the security requirements state what properties have to be guaranteed in the network, threats and risks state what kind of attacks may occur and what damage may be the consequence. Similarly to security requirements we associate each risk with some model element.

For a systematic threat analysis at the application or technical level threat catalogues such as the Baseline Protection Manual [9] or EBIOS [20] can be used. In this respect the drivers of threat analysis are the model elements at lower level of abstraction exposed to attacks from outside or inside the network.

In the second step threats are elaborated to risks. Each risk evaluates a threat in the context of a model element, an attached security requirement and relevant security controls. This means that we only consider threats related to security requirement. Moreover, the evaluation is based on the current level of security controls.

For the evaluation itself we use a qualitative approach estimating the probability and the impact of the damage. For an approach which uses key figures for evaluating risks we refer to [16].

In the case study (cf. Fig. 7) we consider threats related with the service *retrieve*. These threats are related with the requirement of logging access and evaluated in this context.



Security Requirement	Model Element	Parent	State
SR4 Log every access	Service:Retrieve	**SR2**	Evaluated

Threat	Model Element	Sec. Req.	Prob.	Impact	Risk	State
TH1 Overflow log system using a DOS Attack	Service:Retrieve	**SR4**	med.	high	med	Evaluated
TH2 Hacking and manipulating log system	Service:Retrieve	**SR4**	low	high	low	Evaluated

Fig. 7. Sample Threat and Risk Analysis

Concerning the state model a risk is first *pending* then *evaluated*. Security requirements qualify if the related list of risks is handled to be *complete*. As soon as all risks related with a security requirement are evaluated the state of the requirement is set to *evaluated*. Upward propagation of state information in the dependency graph takes place when all subordinate security requirements are set to *evaluated*. Moreover, a security requirement which has been re–set to *pending* or *complete* also causes an upward propagation.

Security Control Engineering. In the final step of the micro-process the security controls are chosen and documented in the security information network. Generally, security control engineering is a complex task ranging from the choice of appropriate countermeasures (including alternatives), their correctness check, analysis of cost-effectiveness compared to the reduction of risks, analysis of remaining threats and the whole procurement- and roll-out-process.

In this paper we only consider the small part of documenting implemented security controls. Each security control is related to the set of risks it reduces. Consequently, the risks have to be re–evaluated which means that the states of the related risks are re–set to *pending* and this state change is propagated up in the dependency graph as described above. In an extended framework also the planning stage of security controls should be supported.

3 Related Work

Related work exists in two directions – security requirements engineering and formal approaches to security engineering.

In the area of security requirements engineering the OCTAVE method [4] uses a three phase approach to identify and manage information security risks. This comprises the identification of critical assets, threat analysis and security strategy planning. OCTAVE provides strong support for the overall process and management aspects whereas our approach focuses on the systematic integration of modeling artifacts and security analysis. In this respect OCTAVE could perfectly be used complementarily to our approach.

An approach that is following a model-based risk analysis is CORAS [17]. CORAS uses UML models mainly for descriptive purposes to foster communication and interaction during the risk analysis process. A strength of CORAS are the methodological foundations on which it is built, like Failure Trees, Event Trees, HazOp and Failure Mode Effect Analysis (FMEA), that help to identify vulnerabilities and threats. To depict the identified assets, sources of threats and threats CORAS uses adapted diagrams inspired by UML. Our method uses text-based representation of threats and security requirements but supports a security analysis process driven by the functional system properties.

Approaches focusing on the elicitation and analysis of security requirements are the Tropos method with the ST-Tool [8] and [8,10,18,12].

Other approaches within the Security Engineering community deal with the specification of security requirements in the context of formal methods. The motivation is to provide frameworks for the (tool–supported) correctness proof of security solutions. Examples are the UML extensions UMLsec [13] and SecureUML [14,5], the PCL approach [7] and the SH Verification Tool [1]. So far our framework does not provide verification facilities and rather focuses on pragmatic security analysis and model-driven software development. In future steps the integration of a formal foundation is planned.

4 Conclusion

In the preceding sections we have presented the security analysis method ProSecO targeted towards the model-based security analysis of service oriented systems. The output of the method is a Security Model describing interrelated functional model elements, requirements, risks and controls. ProSecO is complemented by SECTET, a platform for the model-driven configuration of service oriented architectures based on security patterns. ProSecO and SECTET provide the backbone of a development process focusing on traceability, modularity and high-level realization of security-critical inter-organizational applications.

There remains a lot to be done in future steps. One of our major activities will be the elaboration of the ProSecO tool. In particular, our goals are to support information retrieval and aggregation within Security Models and to develop analysis patterns. Second, as indicated in section 3 we will equip our framework with a formal foundation. This will enable security proofs of security patterns in SECTET. Finally, we plan empirical studies to underpin the effectiveness of our approach.

Acknowledgement. This work was partially elaborated in the context of the health@net research project funded by "Tiroler Zukunftsstiftung" and the Federal Ministry of Economics and Labour, coordinated by CEMIT - Center of Excellence in Medicine and IT.

References

1. SHVT Manual, Fraunhofer Institute for Secure Telecooperation (2004)
2. Business Process Modeling Notation (BPMN) Information (2006), Available online: `http://www.bpmn.org`
3. IHE.net: IT Infrastructure Technical Framework (2006), Available online: `http://www.ihe.net`
4. Alberts, C.J., Dorofee, A.J.: Managing information security risks: the OCTAVE approach. Pearson Education, London (2002)
5. Basin, D., Doser, J., Lodderstedt, T.: Model driven security for process-oriented systems. In: Proc. 8th ACM Symposium on Access Control Models and Technologies, ACM Press, New York (2003)
6. Breu, R., Popp, G., Alam, M.: Model-based development of access policies. Journal for Software Tools and Technology Transfer (STTT) 9, 457–470 (2007)
7. Datta, A., Derek, A., Mitchell, J.C., Pavlovic, D.: A derivation system and compositional logic for security protocols. Journal of Computer Security 13, 423–482 (2005)
8. Giorgini, P., et al.: ST–Tool: A CASE Tool for Modeling and Analyzing Trust Requirements. In: Herrmann, P., Issarny, V., Shiu, S.C.K. (eds.) iTrust 2005. LNCS, vol. 3477, Springer, Heidelberg (2005)
9. Federal Office for Information Security (BSI). IT Baseline Protection Manual (2003), Available online: `http://www.bsi.bund.de/english/gshb/manual/index.htm`
10. Gutierrez, C., Fernandez-Medina, E., Piattini, M.: Security risk analysis in web services systems. In: Proc. SECRYPT 2006, pp. 425–430 (2006)
11. Hafner, M., Agreiter, B., Breu, R., Nowak, A.: SECTET - An Extensible Framework for the Realization of Secure Inter-Organizational Workflows. Journal of Internet Research (2006)
12. Hailey, C., Laney, R., Moffett, J., Nuseibeh, B.: Security requirements engineering, a framework for representation and analysis. IEEE Transactions on Software Engineering (to appear, 2007)
13. Jürjens, J.: Secure Systems Development with UML. Springer, Heidelberg (2004)
14. Lodderstedt, T., Basin, D., Doser, J.: SecureUML: A UML-Based Modeling Language for Model-Driven Security. In: Jézéquel, J.-M., Hussmann, H., Cook, S. (eds.) UML 2002. LNCS, vol. 2460, pp. 426–441. Springer, Heidelberg (2002)
15. Peltier, T.R.: Risk analysis and risk management. Information Systems Security 13(4), 44–56 (2004)
16. Innerhofer-Oberperfler, F., Breu, R., Yautsiukhin, A.: Quantitative assessment of enterprise security systems (submitted)
17. Raptis, D., Dimitrakos, T., Gran, B.A., Stølen, K.: The CORAS Approach for Model-based Risk Management applied to e-Commerce Domain. In: Proc. CMS-2002, pp. 169–181 (2002)

18. Rodriguez, A., Fernandez-Medina, E., Piattini, M.: Capturing Security Requirements in Business Processes Through a UML 2.0 Activity Diagrams Profile. In: Roddick, J.F., Benjamins, V.R., Si-said Cherfi, S., Chiang, R., Claramunt, C., Elmasri, R.A., Grandi, F., Han, H., Hepp, M., Lytras, M., Mišić, V.B., Poels, G., Song, I.-Y., Trujillo, J., Vangenot, C. (eds.) ER Workshops 2006. LNCS, vol. 4231, pp. 32–42. Springer, Heidelberg (2006)
19. Schabetsberger, T., Ammenwerth,, Breu, R., Hoerbst, A., Goebel, G., Penz, R., Schindelwig, K., Toth, H., Vogl, R., Wozak, F.: E-Health Approach to Link-up Actors in the Health Care System of Austria. Stud. Health Technol. Inform. 124
20. Service Central de la Securite des Systemes d'Information. Expression of needs and identification of safety objectives (EBIOS) (online, 1997)
21. Wozak, F., Ammenwerth, E., Breu, M., Penz, R., Schabetsberger, T., Vogl, R., Wurz, M.: Medical Data GRIDs as approach towards secure cross enterprise document sharing (based on IHE XDS). In: Proc. MIE 2006, pp. 377–383 (2006)

Towards Truly Flexible and Adaptive Process-Aware Information Systems*

Peter Dadam, Manfred Reichert, Stefanie Rinderle, Martin Jurisch, Hilmar Acker, Kevin Göser, Ulrich Kreher, and Markus Lauer

Ulm University, Institute of Databases and Information Systems,
James-Franck-Ring, 89081 Ulm, Germany
{peter.dadam,manfred.reichert,stefanie.rinderle,
martin.jurisch,hilmar.acker,kevin.goeser,ulrich.kreher,
markus.lauer}@uni-ulm.de

Abstract. If current process management systems shall be applied to a broad spectrum of applications, they will have to be significantly improved with respect to their technological capabilities. Particularly, in dynamic environments it must be possible to quickly implement and deploy new processes, to enable ad-hoc modifications of running process instances on-the-fly (e.g., to dynamically add, delete or move process steps), and to support process schema evolution with instance migration (i.e., to propagate process schema changes to already running instances if desired). These requirements must be met without affecting process consistency and by preserving the robustness of the process management system. In this paper we describe how these challenges have been addressed and solved in the ADEPT2 Process Management System. Our overall vision is to provide a next generation process management technology which can be used in a variety of application domains.

1 Introduction

More and more contemporary information systems (IS) have to be aligned in a process-oriented way. This new generation of IS is often referred to as Process-Aware IS (PAIS) [1]. Recently, numerous technologies and paradigms have emerged in this context such as Workflow Management, Business Process Management, Enterprise Application Integration, and Service-oriented Architectures (SOA). They all focus on the realization of PAIS [1]. By offering system-based support for implementing business processes, these technologies aim at an increased efficiency and adaptivity of enterprises regarding their internal processes. By combining process management with SOA the interaction between enterprises and their customers and partners shall be improved as well.

To provide effective process support, PAIS should capture real-world processes adequately, i.e., there should be no mismatch between the computerized processes and those in reality. To achieve this, PAIS enabling technologies must fulfill a number of requirements:

* The development of ADEPT2 was funded by the State of Baden-Württemberg as part of the AristaFlow project (see www.AristaFlow.de).

1. They must cover a broad spectrum of applications ranging from form- or document-centered workflows to complex production workflows (where application integration constitutes a major task).
2. They must allow for the rapid and cost-effective implementation of a large variety of business processes.
3. The implemented processes must run in a robust and stable manner. The overall objective should be "robustness by design".
4. PAIS must not lead to rigidity and freeze existing business processes. Instead, they must allow authorized users to flexibly deviate from the predefined processes as required (e.g., to deal with exceptions). Such ad-hoc process changes should be enabled at a high level of abstraction and without affecting robustness of the PAIS.
5. Due to process optimization or legal changes PAIS implementations evolve over time Respective process changes have to be accomplished in an easy and cost-effective way. For long-running processes the "on-the-fly" adaptation of already running process instances to the new process schema should be possible as well.

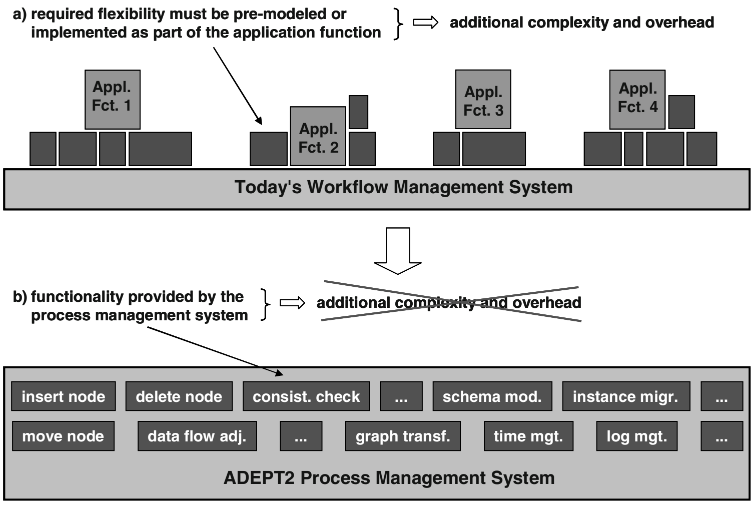

a) required flexibility must be pre-modeled or implemented as part of the application function
additional complexity and overhead
Appl. Fct. 1
Appl. Fct. 2
Appl. Fct. 3
Appl. Fct. 4
Today's Workflow Management System
b) functionality provided by the process management system
additional ~~complexity and~~ overhead
insert node
delete node
consist. check
...
schema mod.
instance migr.
...
move node
data flow adj.
...
graph transf.
time mgt.
log mgt.
...
ADEPT2 Process Management System

Fig. 1. Overhead caused by realizing system functions within the application programs is avoided by providing the required functionality as integral part of the ADEPT2 system

Off-the-shelf process management systems do not meet these requirements or offer restricted features [1,2]. Several vendors promise flexible process support, but are unable to cope with fundamental issues related to process change (e.g., correctness). Most systems, however, completely lack support for deviating from the predefined processes in an ad-hoc manner or for migrating process instances to a changed process schema. Thus, application developers are forced to "enrich" applications with respective process support functions to deal with these limitations. This, in turn, aggravates PAIS development and maintenance significantly and shifts the risk of errors and the task to deal with them to application developers or end-users.

This paper presents the ADEPT2 process management system – one of the leading technologies for realizing flexible and adaptive processes. Using ADEPT2, process-oriented applications can be composed out of existing application components in a plug & play like fashion, and then be flexibly executed at run-time. The ADEPT2 technology enables support for a broad spectrum of processes, ranging from simple document-centered workflows to complex production workflows, which integrate heterogeneous, distributed application components. We illustrate how ad-hoc changes of single process instances as well as process schema changes with (optional) propagation of these changes to the running instances are supported by ADEPT2 in an integrated, safe, and easy-to-use manner. In particular, application programmers and users of the ADEPT2 system are not confronted with the inherent complexity coming with dynamic changes (as indicated in Fig. 1a). Instead, this functionality is easy to use since it is an integral part of ADEPT2.

The remainder of this paper is structured as follows: Sect. 2 sketches how process composition is realized based on plug & play. In Sect. 3 we show how ad-hoc process adaptations can be accomplished by end users and how the interaction between the user and the ADEPT2 system looks like. In Sec. 4 we discuss process schema changes and the adaptations of already running process instances. Sect. 5 describes the current status of the ADEPT2 technology and Sect. 6 discusses current trends and related work. We close with a summary and an outlook in Sec. 7.

2 Process Composition by Plug & Play

A new process can be realized by creating a process template (also denoted as process schema). Such a template describes the planned order of the process steps (e.g., sequential, parallel, alternative paths, loops, etc.) as well as the data flow between them. It either has to be defined from scratch or an existing template is chosen from the process template repository and adapted as needed ("process cloning"). Afterwards application functions (e.g., web services, Java components, ERP functions, or legacy applications) have to be assigned to the process steps. When using the ADEPT2 process editor, these functions can be selected from the component repository and be inserted into the process template by drag & drop (cf. Figure 2). Following this, ADEPT2 analyzes whether the application functions can be connected in the desired order; e.g., we check whether the input parameters of application functions can be correctly supplied for all possible execution paths imposed by the process schema. Furthermore, additional checks are performed in order to exclude deadlocks, livelocks, etc. Only those process templates passing these correctness checks may be released and transferred to the ADEPT2 runtime system.

When dragging application components from the repository and assigning them to particular steps in the process template, the process designer does not need to have detailed knowledge about the implementation of these components. Instead the component repository provides an integrated, homogeneous view as well as access to the different components. Internally, this is based on a set of wrappers provided for the different types of application components. The chosen architecture will allow to add new wrappers if new component types shall be supported. Currently, the ADEPT2 Execution Environment Framework allows to integrate different kinds of application components like electronic forms, stand-alone executables, web services, Java library functions, and function calls to legacy systems. All these application components require different treatment when interacting with them.

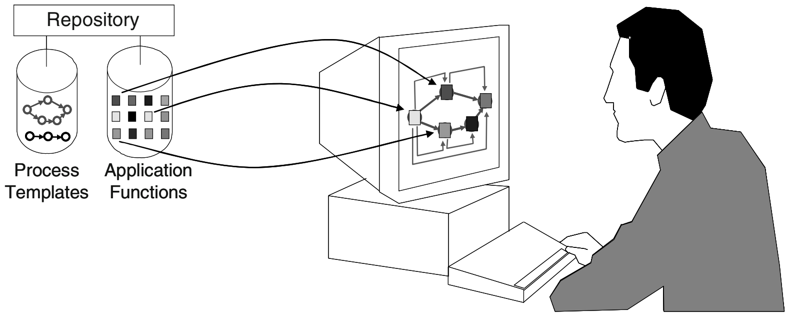


Fig. 2. Composition of correct processes using plug & play

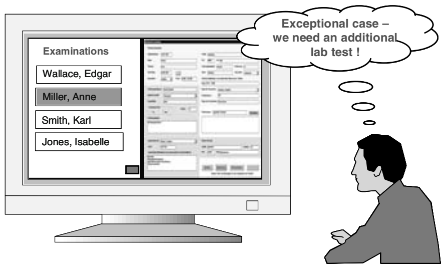
3 Support of Ad-Hoc Adaptations

Composing processes in a plug & play like fashion is very useful for developers since it allows for rapid implementation of new processes. However, composition support alone does not constitute a big technological progress when compared to the state-of-the-art. If process management technolgy shall become applicable to a broader spectrum of applications than today, it must allow for ad-hoc deviations from the pre-defined process schema as well. Such runtime changes must not violate workflow correctness. Further, authorized users should be able apply ad-hoc changes in an intuitive way to.

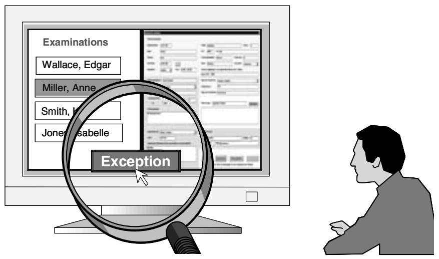
Figure 3a – h illustrate how the interaction between the ADEPT2 system and the end user may look like. In this example it is assumed that during the execution of a particular process instance (e.g., the treatment of a certain patient under risk) an additional lab test becomes necessary. Assume that this has not been foreseen at buildtime (cf. Fig. 3a). As a consequence, this particular process instance will have to be individually adapted if the change request is approved by the system. After the user has pressed the "exception button" (cf. Fig. 3b), he can specify the type of the intended ad-hoc change (cf. Fig. 3c). If an insert operation shall be applied, for example, the system will display the application functions that can be added in the given context (cf. Fig. 3d). These can be simple or complex application components (e.g., "write letter” or "send email” vs. application services), interactive or automatic functions, or even complete processes.

Now the user simply has to state after which step(s) in the workflow the execution of the newly added activity shall be started and before which step(s) it shall be finished (cf. Fig. 3e). Finally, the system checks whether resulting process instance adaptations are valid (cf. Figure 3f and Figure 3g).

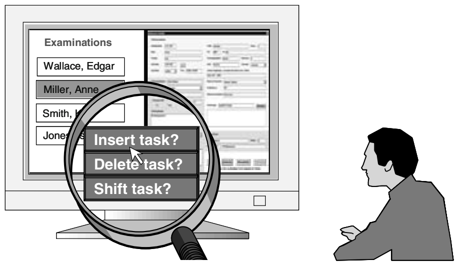
In this context, the same checks are performed as during the process design phase (e.g., absence of deadlocks or validity of actor assignment expressions). In addition, the current process instance state is taken into account when the instance is modified.



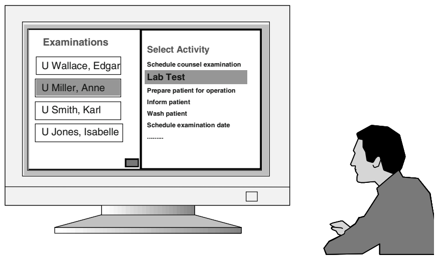
a) An exception occurs



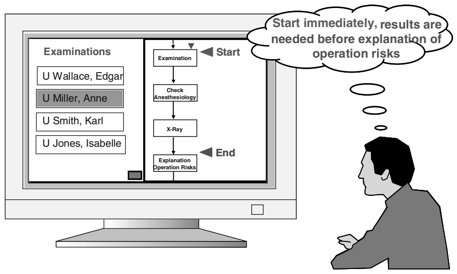
b) User presses the "exception button"



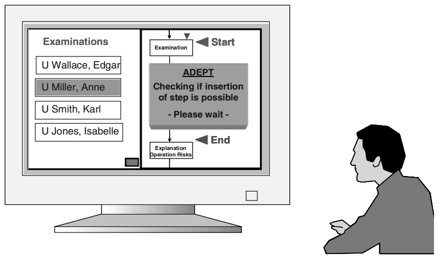
c) User selects type of the ad-hoc change



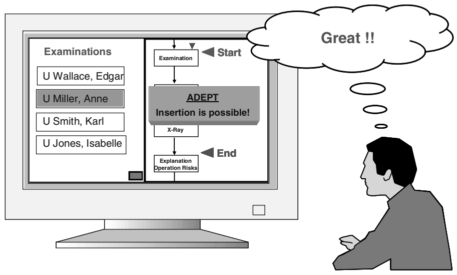
d) User selects step to be inserted



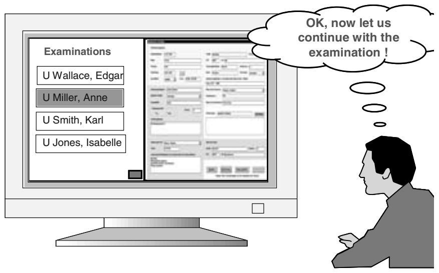
e) User specifies where to insert the step



f) System checks validity of the change



g) Change can be applied



h) User continues work

Fig. 3. Executing an ad-hoc modification from the end user's point of view

On the one hand this allows for modifications which would not be valid at design time (e.g., due to uncertainty at design time which execution branches will be taken). On the other hand the process state also restricts possible changes (see [3] for details).

All implemented change operations are also available via the ADEPT2 application programming interface. Furthermore, changes can be specified at a semantically high level of abstraction (e.g. "Insert Step X between Node Set 1 and Node Set 2"), which eases change definition significantly [2]. All these operations are guarded by preconditions which are automatically checked by the system when the operation is invoked. The related post conditions guarantee that the resulting process instance graph is again "problem-free". Users or application programs only interact via these high level API functions with ADEPT2. They never do have to manipulate system-internal states directly (see [4] for details).

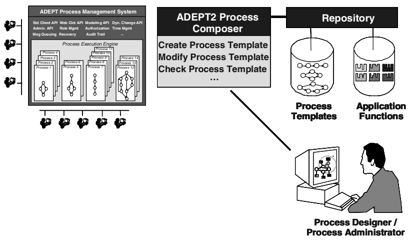
For several reasons (e.g., change traceability) ADEPT2 stores process instance changes within so called change logs. Together with the enactment logs, which capture the execution information of process instances, the structure and state of a particular process instance can be reconstructed at any time. This log information is also a valuable source for process optimizations because repeatedly performed ad-hoc modifications may be an indicator that the process has been not optimally designed [5].

4 On Supporting Process Schema Evolution

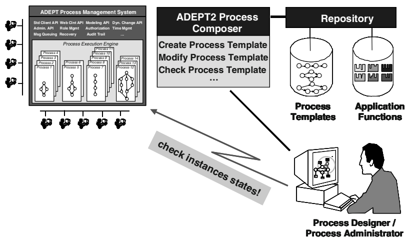
Though the support of ad-hoc modifications is very important, it is not yet sufficient. In the context of long-running business processes, it is often required to adapt the process schema itself (e.g., due to organizational changes in the company or because of business process optimizations). In this case all process instances based on this process schema may be affected by the change. If the processes are of short duration only, already running process instances usually can be finished according to the old schema version. However, this strategy will be not applicable for long running business processes. Then the old process version may no longer be applicable, e.g., when legal regulations have changed or when the old process reveals severe problems. One solution would be to individually modify each of the running process instances by applying corresponding ad-hoc changes (as described in the previous section). However, this would be too expensive and error-prone if a multitude of running process instances had been involved. Note that the number of active process instances may become very large; i.e., change propagation must be accomplished in a very efficient manner for hundreds or thousands of process instances.

An adaptive process management system must be able to support correct changes of a process schema and their subsequent propagation to already running process instances if desired. In other words, if a process schema is changed and thus a new version of this schema is created, process instances should be allowed to migrate to the new schema version (i.e., to be transferred and re-linked to the new process schema version). In this context, it is of particular importance that ad-hoc changes of single process instances and instance migrations do not exclude each other since both kinds of changes are needed for the support of long-running processes!

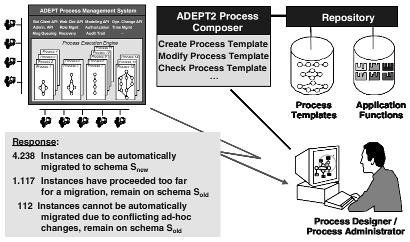
The ADEPT2 technology implements the combined handling of both kinds of changes. Process instances which have been individually modified can be also migrated to a changed process schema if this does not cause inconsistencies or errors in the sequel. All needed correctness checks (on the schema and the state of the



a) Process schema change

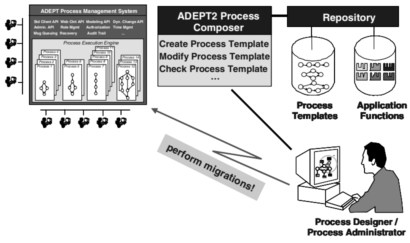


b) Check state of running process instances



Response:
4.238 Instances can be automatically migrated to schema S_{new}
1.117 Instances have proceeded too far for a migration, remain on schema S_{old}
112 Instances cannot be automatically migrated due to conflicting ad-hoc changes, remain on schema S_{old}

c) Result of checks



d) Execute instance migration

Fig. 4. Process schema evolution

instances) and all adaptations to be accomplished when migrating the instances to the new process schema version are performed by ADEPT2. The implementation is based on a comprehensive formal framework [3, 6]. Based on it, ADEPT2 can precisely state under which conditions a certain process instance can be migrated to the new process schema version. This allows for checking the compliance of a collection of process instances with the changed schema version in an efficient manner. Finally, concurrent and conflicting changes at the process type and the process instance level are managed in a reliable and consistent manner as well. In particular, long-running processes will benefit from this close integration of the different change levels.

Figure 4a – Figure 4c illustrate how such a process schema evolution is conducted from the user's point of view in ADEPT2. The process designer loads the process schema from the process template repository, adapts it (using the ADEPT2 process editor), and creates a new schema version (cf. Figure 4a). Then the system checks whether the running process instances can be correctly migrated to the new process schema version (cf. Figure 4b and Figure 4c). These checks are based on state conditions and structural comparisons. Furthermore, the system calculates which adaptations become necessary to perform the migration at the process instance level. The ADEPT2 system analyzes all running instances of the old schema and creates a list of instances which can be migrated as well as a list of instances for which this is not possible (together with a report which explains the different judgments). When pressing the "migration button" the system automatically conducts the migration for all selected process instances (see Figure 4d).

5 On Transferring the ADEPT2 Technology to Business

The vision of enabling ad-hoc modifications within the process management system in a correct and consistent manner was the starting point for our research and implementation work done within the ADEPT project more than 10 years ago [4]. The resulting technology has been integrated in the experimental ADEPT1 system which, to our best knowledge, is still leading in the field of adaptive process management today. The ADEPT1 technology has enabled ad-hoc deviations in a controlled, secure, and user-friendly manner. Unforeseen exceptions can be handled within the PAIS and not by bypassing it as often required when using commercial process management systems. The ADEPT1 system has been used in several national and international research projects.

From these projects we gained valuable insights into the practical needs for process management technology on the one side, and we learned many lessons about implementing a complex system such as ADEPT1 on the other side. Partly these "lessons learned" have been published covering topics like log management, system-internal representation of process schemas and process instances, and the transfer of selected implementation concepts to other application areas such as data warehouses [7-9]. One important perception is that the system design and architecture should offer powerful functionality at a semantically high level and hide the inherent complexity as much as possible from the user.

To transfer the ADEPT2 technology into industrial usage and business, the AristaFlow GmbH was jointly founded by members of our institute and industrial partners (see [10] for details). The major focus of this company is to implement a robust and scalable commercial version of the ADEPT2 system. ADEPT2 includes all functionality of the old ADEPT1 system. Ad-hoc flexibility is now based on a broader range of supported operations, however. In addition, ADEPT2 realizes the composition of processes based on plug & play techniques as sketched in Section 2. Furthermore, it implements the theoretical framework for process schema evolution as outlined in Section 4. Altogether, the design and implementation of such a powerful and innovative system constitutes a big challenge. However, we can now exploit our lessons learned from implementing the ADEPT1 system. Finally, we benefit very much from the cooperation with our industrial partners and the University of Mannheim within the AristaFlow project (see [10] for details).

6 Current Trends and Related Work

This section discusses current trends and approaches from literature and relates them to ADEPT2:

Plug & Play: Recently, lots of attention has been paid to the area of (dynamic) web service composition [11,13]. In particular, the emergence of WS-BPEL (Business Process Execution Language) has resulted in many research activities (e.g., on match making [12]) as well as new process composition tools and service flow engines (e.g., IBM WebSphere Process Server or SAP Netweaver). How to provide intelligent support for finding the right web services or partners, however, has not been a research topic in ADEPT2 so far. Nevertheless, the basic functionality for composing services in a process-oriented way is already provided by ADEPT2 as described in Section 2. In particular, ADEPT2 is able to invoke any kind of standard component (including web services and Java components); it further offers powerful interfaces to integrate arbitrary application components with little programming effort.

Process Choreography & Orchestration. Though a distributed variant of ADEPT1, which supports distributed process execution, has been developed [14, 15] and some attention was paid to inter-workflow coordination [16, 17], the focus of ADEPT2 is on orchestration; i.e., on the coordination and enactment of (business) processes from the viewpoint of one company. Opposed to that, conversation languages like WS-CDL or WSCI have been designed for defining the choreography of different partner processes (i.e., for global processes where each partner runs an internal process and provides a public view on it based on which it exchanges messages with partners).

A promising integration variant of ADEPT2 with such web service standards is conceivable: While ADEPT2 defines and manages internal (i.e., private) processes, WS-BPEL can be used for describing public process views and WS-CDL for defining the choreography of partner processes based on these public views (i.e., the global protocol based on which the processes of the different partners communicate). The advantage of this approach would be that with ADEPT2 any application component and particularly interactive process steps can be called within the internal processes. This is useful since internal processes typically comprise a mix of interactive and automated activities as well as a variety of different application functions such as Java components, Web services, but also complex ERP system functions. Using the ADEPT2 technology the correctness of the modeled (plugged) processes can be guaranteed based on the formal checks on, for example, control and data flow.

Adaptive Process Management: The flexible support of business processes has been a hot topic in research for a long time. Mostly, approaches either deal with ad-hoc deviations at process instance level or process schema evolution [3, 18, 19]. The same applies for commercial systems offering some, but very limited flexibility (e.g., Staffware or Ultimus Workflow). Only few approaches and prototypes [20, 21, 22] allow for both kinds of process changes, but in an isolated manner. To our best knowledge, ADEPT2 is the only adaptive PMS which supports ad-hoc deviations, process schema evolution, and their interplay based on a sound theoretical framework and within one implemented system.

Finally, several approaches exist that allow to define “placeholder activities” in a process model for which a concrete sub-process can be bound or modeled during runtime. Representatives of this system category include PocketsOfFlexibility [23] and Worklets [24]. Typically, late modeling has to be finished before creating the corresponding sub-process instance. Though late binding and late modeling increase process flexibility (see [2] for a detailed discussion), they do not allow for structural changes of already running instances. Recently a list of change patterns and change support features have emerged, which allow to systematically compare change and flexibility support in existing PAIS [2].

7 Summary and Outlook

The ADEPT2 technology meets major requirements claimed for next generation adaptive process management systems: it provides advanced functionality to support process composition by plug & play of arbitrary application components, it enables ad-hoc flexibility for process instances without losing control (i.e., without causing process execution errors or inconsistencies), and it supports process schema evolution in a controlled and efficient manner. As opposed to many other approaches all these aspects work in interplay as well. For example, it is possible to propagate process schema changes to individually modified process instances or to dynamically compose processes out of existing application components. All in all such a complex system requires a sound theoretical framework in order to avoid incomplete solutions and implementation gaps [3-5]. Finally, it is important to mention that most of our theoretical results have not been just kept "on paper", but have been implemented within the process management systems ADEPT1 and ADEPT2 respectively. As ADEPT2 additionally provides powerful tools and application programming interfaces its transfer to and its use in practice will be further eased.

In addition to the features presented in this paper, the ADEPT2 technology offers promising perspectives for process learning and continuous process optimization [5, 26]. In particular, audit data (i.e., process logs) become much more meaningful, since they do not only capture process execution events (e.g., start / completion of activities), but also contain insightful information on performed ad-hoc changes at the process instance level [8]. By mining the change logs related to a collection of individually modified process instances, we can (semi-)automatically derive potential process improvements (i.e., changes to be applied to the original process schema).

Recently, we have started working on process change mining, and first project results indicate the new opportunities emerging in this context [26]. Furthermore, process schema adaptations derived with respective mining techniques can be implemented with ADEPT2 in a much quicker and more cost effective way when compared to existing technology. Thus, continuous process evolution and full process lifecycle support become possible. Finally, other interesting perspectives arise when introducing adaptive process management to practice. This includes the support for emergent processes [30], the "outsourcing" of successfully applied exception handling procedures to knowledge management components [5, 27], the automatic, rule-based adaptation of process instances [28, 29], or the support of ad-hoc workflows.

In future work we will incorporate more semantic knowledge into the ADEPT2 framework, i.e., it shall become possible to specify semantical integrity constraints on business processes [25]. We also aim at providing efficient methods to check the validity of such semantic constraints when changing processes. Following such a constraint-based approach it becomes possible to not only guarantee that processes run correctly regarding their control and data flow, but also regarding the validity of the specified semantic constraints. As another important task we will elaborate the use of ADEPT2 in different practical settings and application areas (e.g., healthcare [31,32]).

References

1. Dumas, M., van der Aalst, W., ter Hofstede, A.: Process-aware Information Systems. John Wiley & Sons, Chichester (2005)
2. Weber, B., Rinderle, S., Reichert, M.: Change Patterns and Change Support Features in Process-Aware Information Systems. In: Krogstie, J., Opdahl, A., Sindre, G. (eds.) CAiSE 2007 and WES 2007. LNCS, vol. 4495, pp. 574–588. Springer, Heidelberg (2007)
3. Rinderle, S., Reichert, M., Dadam, P.: Correctness Criteria for Dynamic Changes in Workflow Systems - A Survey. Data and Knowledge Engineering 50(1), 9–34 (2004)
4. Reichert, M., Dadam, P.: ADEPTflex - Supporting Dynamic Changes of Workflows Without Losing Control. Journal of Intelligent Information Systems 10(2), 93–129 (1998)
5. Rinderle, S., Weber, B., Reichert, M., Wild, W.: Integrating Process Learning and Process Evolution – A Semantic Based Approach. In: van der Aalst, W.M.P., Benatallah, B., Casati, F., Curbera, F. (eds.) BPM 2005. LNCS, vol. 3649, pp. 252–267. Springer, Heidelberg (2005)
6. Rinderle, S., Reichert, M., Dadam, P.: Flexible Support Of Team Processes By Adaptive Workflow Systems. Distributed and Parallel Databases 16(1), 91–116 (2004)
7. Rinderle, S., Kreher, U., Lauer, M., Dadam, P., Reichert, M.: On Representing Instance Changes in Adaptive Process Management Systems. In: Proc. ProFlex 2006, First IEEE Workshop on Flexibility in Process-aware Information Systems, Manchester (2006)
8. Rinderle, S., Reichert, M., Jurisch, M., Kreher, U.: On Representing, Purging, and Utilizing Change Logs in Process Management Systems. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102, pp. 241–256. Springer, Heidelberg (2006)
9. Rinderle, S., Jurisch, M., Reichert, M.: On Deriving Net Change Information From Change Logs - The DeltaLayer Algorithm. In: Proc. BTW 2007, Datenbanksysteme in Business, Technologie und Web, Aachen. LNI, vol. 103, pp. 364–381 (2007)
10. http://www.AristaFlow.com (AristaFlow GmbH), http://www.AristaFlow.de (AristaFlow project)
11. Khalaf, R., Mukhi, N., Weerawarana, S.: Service-oriented composition in BPEL4WS. In: Proc. WWW 2003, Budapest (2003)
12. Wombacher, A., Fankhauser, P., Mahleko, B., Neuhold, E.: Matchmaking for business processes based on choreographies. IJWS 1(1), 14–32 (2004)
13. Yi, X., Kochut, K.J.: Process composition of web services with complex conversation protocols. In: Proc. Conf. on Design, Analysis, and Simulation of Distributed Systems Symposium at Advanced Simulation Technology, pp. 141–148 (2004)
14. Bauer, T., Reichert, M., Dadam, P.: Intra-Subnet Load Balancing in Distributed Workflow Management Systems. Int'l J. of Cooperative Information Systems 12(3), 205–223 (2003)
15. Bauer, T., Dadam, P.: A Distributed Execution Environment for Large-Scale Workflow Management Systems with Subnets and Server Migration. In: Proc. CoopIS 1997, Kiawah Island, South Carolina, USA, pp. 99–108 (1997)
16. Heinlein, C.: Synchronization of Concurrent Workflows Using Interaction Expressions and Coordination Protocols. In: Meersman, R., Tari, Z., et al. (eds.) CoopIS 2002, DOA 2002, and ODBASE 2002. LNCS, vol. 2519, pp. 54–71. Springer, Heidelberg (2002)
17. Heinlein, C.: Workflow and Process Synchronization with Interaction Expressions and Graphs. In: Proc. ICDE 2001, Heidelberg, Germany, April 2001, pp. 243–252 (2001)
18. van der Aalst, W.M.P., Basten, T.: Inheritance of Workflows: An Approach to Tackling Problems Related to Change. Theoretical Computer Science 270(1-2), 125–203 (2002)
19. Casati, F., Ceri, S., Pernici, B., Pozzi, G.: Workflow Evolution. Data & Knowledge Engineering 24(3), 211–238 (1998)

20. Kochut, K., Arnold, J., Sheth, A., Miller, J., Kraemer, E., Arpinar, B., Cardoso, J.: IntelliGEN: A distributed workflow system for discovering protein-protein interactions. Distributed and Parallel Databases 13(1), 43–72 (2002)
21. Weske, M.: Formal Foundation and Conceptual Design of Dynamic Adaptations in a Workflow Management System. In: Proc. HICSS-34, Maui, Hawaii, vol. 7, p. 7051 (2001)
22. Pesic, M., Schonenberg, M.H., Sidorova, N., van der Aalst, W.M.P.: Constraint-Based Workflow Models: Change Made Easy. In: Meersman, R., Tari, Z. (eds.) OTM 2007, Part I. LNCS, vol. 4803, pp. 77–94. Springer, Heidelberg (2007)
23. Sadiq, S., Sadiq, W., Orlowska, M.: Pockets of flexibility in workflow specifications. In: Kunii, H.S., Jajodia, S., Sølvberg, A. (eds.) ER 2001. LNCS, vol. 2224, pp. 513–526. Springer, Heidelberg (2001)
24. Adams, M., ter Hofstede, A.H.M., Edmond, D., van der Aalst, W.: Worklets: A service-oriented implementation of dynamic flexibility in workflows. In: Meersman, R., Tari, Z. (eds.) OTM 2006. LNCS, vol. 4275, pp. 291–318. Springer, Heidelberg (2006)
25. Ly, L.T., Rinderle, S., Dadam, P.: Semantic Correctness In Adaptive Process Management Systems. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102, pp. 193–208. Springer, Heidelberg (2006)
26. Günther, C.W., Rinderle, S., Reichert, M., van der Aalst, W.M.P.: Change Mining in Adaptive Process Management Systems. In: Meersman, R., Tari, Z. (eds.) OTM 2006. LNCS, vol. 4275, pp. 319–326. Springer, Heidelberg (2006)
27. Weber, B., Wild, W., Breu, R.: CBRFlow: Enabling Adaptive Workflow Management Through Conversational CBR. In: Funk, P., González Calero, P.A. (eds.) ECCBR 2004. LNCS (LNAI), vol. 3155, pp. 434–448. Springer, Heidelberg (2004)
28. Müller, R.: Event-oriented dynamic adaptation of workflows. Ph.D. Thesis, University of Leipzig, Germany (2002)
29. Greiner, U.: Quality-oriented Execution and Optimization of Cooperative Processes: Model and Algorithms. Ph.D. Thesis, University of Leipzig, Germany (2005)
30. Jørgensen, H.D.: Interactive Process Models. Ph.D. Thesis, Norwegian University of Science and Technology, Trondheim, Norway (2004)
31. Lenz, R., Reichert, M.: IT Support for Healthcare Processes - Premises, Challenges, Perspectives. Data and Knowledge Engineering 61, 39–58 (2007)
32. Dadam, P., Reichert, M., Kuhn, K.: Clinical Workflows - The Killer Application for Process-oriented Information Systems? In: Proc. 4th Int'l Conf. on Business Information Systems (BIS 2000), Poznan, Poland, pp. 36–59 (2000)

From Human Knowledge to Process Models

Jörg Desel

Angewandte Informatik, Katholische Universität Eichstätt, Germany
joerg.desel@ku-eichstaett.de
www.informatik.ku-eichstaett.de

Abstract. This contribution suggests a novel approach for a systematic generation of a process model in an informal environment. It is based on the claim that the knowledge about the process to be modelled is distributed in several involved people's minds. Some people have knowledge about the general process where the single activities are on a high level of abstraction and have to be refined. Other people only know something about some detail of the process, i.e., about the refinement of an activity of the general process which defines a subprocess. Moreover, it is assumed that these domain experts can more easily define instances, i.e. runs, of the general process (of a subprocess, respectively) than the process itself. The approach employs new techniques developed for process mining and Petri net synthesis and adapts these techniques to generate processes from example runs. It is based on a formal definition of partially ordered processes, which allows to proceed in a modular way: Subprocesses and general processes are generated independently. Finally, it is argued that the approach is a suitable first step in a general method for process definitions which is followed by validation techniques.

Keywords: Business Process Modelling, Petri nets, Synthesis.

1 Introduction

During the last decades, information system modelling emphasized the modelling of data as a prerequisite of data base design. The modelling of processes was also considered, but only in the last years process models received attention comparable to data models [21]. This development is closely connected to the raise of Business Process Management [4,34], of workflow management systems [1], of process-centric ERP systems and finally of web services.

It is well known that for both, data and process models, the first phase of modelling is of particular importance. Only if models that faithfully present the part of reality (or of intended reality) to be modelled are used in system design, the final system can be expected to behave correct according to the requirements. Conversely, any error in an early stage of modelling will cause very costly redevelopments in later phases.

For data models, suggestions for a systematic analysis of requirements have been developed since a long time and have proven to be very useful in practice [5,27]. In particular, the research done within the project KCPM (Klagenfurt

Conceptual Predesign Model) in Klagenfurt shows how to extract formal model components from natural language documents provided by the system experts and users [30].

For process models, there is no such generally agreed procedure. Again, the KCPM people made suggestions since a long time (see e.g. [28], where pre- and post-conditions of actions are extracted from text and used as local vicinity of corresponding transitions of Petri net models; more recent developments are given in [29,23,31]).

The purpose of this paper is to tackle the very same question: How can we derive a process model in/from an informal environment? So instead of deriving a process model from another model by any kind of transformation (where correctness is a matter of verification), we do not assume any previous representation of the model to be constructed but only distributed knowledge about the process in people‘s minds.

Very often, semi-formal modelling languages are suggested to support this first step in process modelling. However, there is no clear definition of "semi-formal". Sometimes, a language is called semi-formal if it has a defined syntax but no (or more than one) semantics. Whereas people often praise missing unique semantics as flexibility, I cannot see any advantage of semi-formality if not even author and recipient of a model are sure to communicate the same object. In particular, for this kind of conceptual models it should be clear what exactly is modelled and what is left open [12]. This approach, starting with an informal model and adding semantics in a fuzzy way will never guarantee that the final, formal model matches the initial intuition.

The approach sketched in this paper is different. It starts with formal objects from the beginning but keeps these objects as simple as possible. In the area of processes (including alternatives, variants etc.) the simplest concept is on the instance level: a single run of a process, sometimes called a case or a run of a case. We assume that the relevant persons can more easily describe runs than starting with a process description (which is on the type level and can be as complicated as an arbitrary algorithm). Whereas the definition of a run for a given process is often obvious, the converse direction is more involved: Given a set of runs, what is the corresponding process?

A related question is answered by Petri net synthesis theory [22,9,6,26] which either starts with a reachability graph or with a set of sequential runs or with a set of partially ordered runs. However, synthesis aims at a construction of precisely the process with the specified behavior or – if such a process does not exist – a process with minimal behavior including the specified one. Thus, often an unnecessarily complicated process is constructed.

Another related research area is process mining [2,32]. Mining techniques aim at constructing a simple process from a large set of recorded behavior, given by so-called process logs. This is also not exactly our setting. We start from some representative examples and try to construct a process which is as simple as possible, can behave like the example runs and does not have too much additional behavior. [3] shows how techniques from synthesis theory can be used for process mining. The same techniques will be applied in our approach to generate process models from sequential or from partially ordered runs.

A second assumption of this paper is that relevant people know something about the runs of the general process on an abstract level. In other words, the single activities occurring in the runs can be refined to subprocesses. Other experts might not know the entire process but have knowledge about runs of a subprocess.

An obvious way to deal with this kind of vertical modularity is to replace each occurrence of a coarse activity in a given run by one (or all) of the runs of the subprocess and then to apply synthesis techniques to the set of all runs refined in this way. Another way to deal with the same problem is to first generate subprocesses from their runs, then to generate the general process on the abstract level, and finally to refine its activities by the subprocesses. It could be assumed that in both ways we end up with the same process definition. However, this does not hold if only sequential runs are considered, see the following example: Assume we have to activities x and y which refine as follows: x has subactivities a followed by b whereas y has subactivities c followed by d. Assume moreover that x and y occur concurrently. In a sequential setting, the general process is specified by example runs $x\,y$ and $y\,x$. Specifying x by the run $a\,b$ and specifying y by the run $c\,d$ and replacing x and y by these runs yields the following possible runs of the refined process: $a\,b\,c\,d$ and $c\,d\,a\,b$. Taking the other approach, we first construct a process with two concurrent activities x and y. Replacing then these activities by processes allowing only for sequences $a\,b$ and $c\,d$ respectively leads to a process where the subactivities are mutual concurrent. So several more runs are possible, for example $a\,c\,b\,d$ and $a\,c\,d\,b$. Taking the Petri net formalism, the process is shown in Figure 1. We argue in this contribution that this problem is solved by using nonsequential semantics. Instead of restricting runs to sequences we consider partially ordered runs, i.e., runs where concurrent occurrences of activities are not ordered by a causality relation.

The following section will provide some necessary formal definitions. Section three shows that partial order semantics indeed solves the above problem. Section four uses this observation and introduces our approach to process modelling. In Section five we relate the approach to a previously introduced validation approach, i.e., we argue that both approaches nicely work together. Finally, the concluding Section six provides some details about our further research plan on this topic.

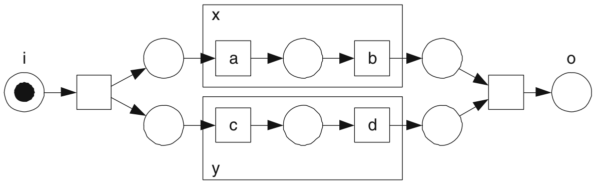


Fig. 1.

2 Formalities

We use the Petri net formalism (see e.g. [33]) to represent processes. Petri nets have been a standard formalism for the representation of processes since a long time (see e.g. [10]) and they are the underlying formalism for many other languages [15,19].

We use the usual notions and definitions of a Petri net (S, T, F, M_0). Capacity restrictions, arc weights and inhibitor arcs are not considered. A Petri net is called 1-bounded if no marking reachable from M_0 assigns more than one token to a place.

Definition 1. *A process is a Petri net (S, T, F, M_0) with two distinguished sets of input- and output-transitions $T_i, T_o \subseteq T$. A process is called safe if the Petri net*

$$(S \cup \{s\}, T, F \cup (\{s\} \times T_i) \cup (T_o \times \{s\}), M_0 \cup \{s, 1\})$$

is a 1-bounded Petri net, where s is a new place.

We assume for the rest of the paper that all processes are connected Petri nets.

1-boundedness of the place s implies that transitions of T_i and of T_o occur strictly alternatingly, starting with a transition of T_i. 1-boundedness of all other places implies that no transition can ever occur concurrently with itself: Since the net is connected, every transition has at least one place in its pre-set or in its post-set, which carries at most one token.

In the sequel we will distinguish main processes and subprocesses which are formally both processes in the above sense.

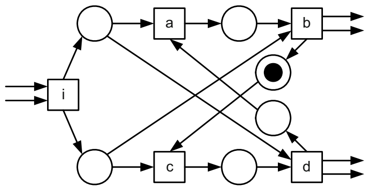


Fig. 2. Example process with input transitions $\{i\}$ and output transitions $\{b, d\}$

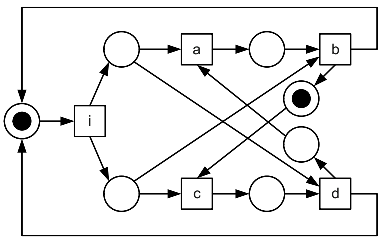


Fig. 3. The process is safe if this Petri net is 1-bounded

A main process is considered connected to an additional place s which is initially marked with one token. s has an empty pre-set. The post-set of s is T_i. Sometimes another place s' with pre-set T_o is considered.

A subprocess refines a transition of another process. It replaces the transition, the transition's input places are connected to the input transitions whereas the output transitions of the subprocess are connected to the post-set of the transition. For a formal definition, let us first notice that we will not distinguish isomorphic processes, i.e., we are free to consistently rename places and transitions of processes.

Definition 2. *Let $P = (S, T, F, M_0)$ be a process with a transition t and let $P^t = (S^t, T^t, F^t, M_0^t)$ be a process with input-transitions T_i^t and output-transitions T_o^t. Assume w.l.o.g. that the elements of P and of P^t are disjoint. The refinement of P w.r.t. transition t and process P^t is defined as*

$$(S \cup S^t, T \cup T^t \setminus \{t\}, F \cup F^t \cup ({}^\bullet t \times T_i) \cup (T_o \times t^\bullet), M_0 \cup M_0^t)$$

where ${}^\bullet t$ and $t^\bullet$ refers to the process P.

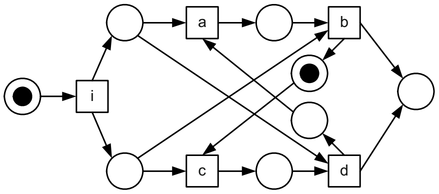


Fig. 4. The process of Figure 2 considered as a main process

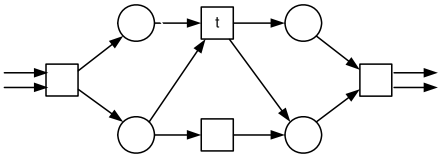


Fig. 5. An abstract process...

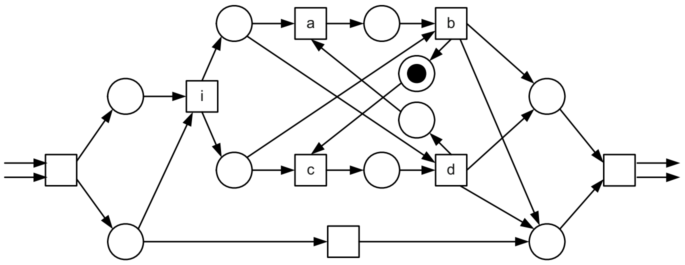


Fig. 6. ...and its refinement w.r.t. t and the process of Figure 2

Figure 4 shows a process and Figure 5 shows the same process when transition t is refined by the process of Figure 2.

Different transitions of a main process can be refined by the same subprocess. As mentioned above, we do not distinguish isomorphic processes and assume that instead disjoint copies of the subprocess are used.

Proposition 1. *Assume a process P with two transitions u and v and respective refinements P^u and P^v. Then the processes obtained by first refining u and then v and by first refining v and then u are isomorphic.*

Hence the order of refinement does not matter. So we can speak about refinement of all refinable transitions in one step. Since a subprocess can again have transitions to be refined, an arbitrary refinement hierarchy can be constructed.

The intuition of the behavior of a safe process is that first one of its input transition fires and after some internal behavior one of its output transition fires, provided the process does not run in a deadlock. It is not possible that two input transitions occur without an intermediate output transition. Hence the external behavior of the subprocess resembles the behavior of a transition, with the difference that first the input tokens are consumed and later output tokens are produced. Conversely, the surrounding net resembles the behavior of the additional place s shown in Figure 3 with the difference that the input tokens show up after the output tokens have been produced. This results in the following observation:

Proposition 2. *If processes P and P^t are safe, then the refinement of P w.r.t. one of its transitions t and P^t is safe, too.*

3 Concurrent Runs and Refinement

There are several suggestions how to define the behavior of a Petri net. The best known is based on sequential runs, formalized by occurrence sequences, where an occurrence sequence is a sequence of transition names that can occur one after the other. A marking graph can be viewed as a condensed version of all occurrence sequences, together with the intermediate markings, where additionally the branching behavior is represented.

Partial order semantics of Petri nets have originally be defined by means of so-called occurrence nets which are mapped to the original net (see e.g. [33]). An occurrence net together with this mapping is often called process. This instance use of "process" contradicts today's use of "process" in "business process" on the type level. Therefore, the term "run" is used instead for a single instance of a process in this paper.

The main aim of this section is to motivate and show that the diagram of Figure 7 commutes for partial order semantics. In the introduction it was shown that this is not the case for sequential semantics.

The following definition is nearly the usual one for partial order semantics. However, we require that each run leads to the end of the a process. In other

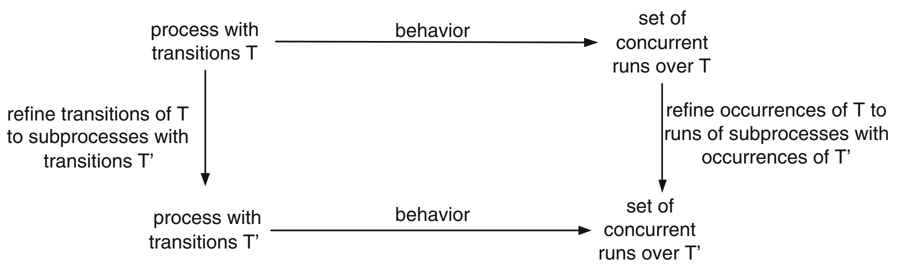

process with transitions T
behavior
set of concurrent runs over T
refine transitions of T to subprocesses with transitions T'
refine occurrences of T to runs of subprocesses with occurrences of T'
process with transitions T'
behavior
set of concurrent runs over T'

Fig. 7. Behavior and refinement

words, we consider maximal runs and not arbitrary prefixes and we rule out runs that run into a deadlock.

Definition 3. *Let $P = (S, T, F, M_0)$ be a safe process with input-transitions T_i. A (concurrent) run of P is a Petri net $R = (B, E, G, I)$ together with mappings $\rho_B : B \to S$, $\rho_E : E \to T$ satisfying the following properties:*

- *places of R are not branched, i.e., $|^\bullet b| \leq 1$ and $|b^\bullet| \leq 1$ for each b in B,*
- *R is acyclic, i.e., $G^* \cap id_{|B \cup E} = \emptyset$,*
- *the flow relation G of R agrees with the flow relation F of P, i.e., for each e in E, ρ_B induces bijections from $^\bullet e$ to $^\bullet \rho_E(e)$ and from $e^\bullet$ to $(\rho_E(e))^\bullet$,*
- *exactly one transition e in E is mapped by $\rho(e)$ to a transition of T_i,*
- *exactly one transition e in E is mapped by $\rho(e)$ to a transition of T_o,*
- *$I(b) = 1$ for each b satisfying $^\bullet b = \emptyset$ and $I(b) = 0$ for all other places b.*
- *initial places of the run are mapped to initially marked places of the process, i.e., ρ_B induces a bijection from $\{b \in B \mid {}^\bullet b = \emptyset\}$ to $\{s \in S \mid M_0(s) = 1\}$.*

Since the process P is assumed to be safe and by the general assumption that each transition has at least an input place or an output place, there are no transitions with empty pre-set except input transitions. Moreover, the fifth item could be relaxed to"at least one transition..." because safety ensures that no output transition occurs more than once.

We aim at comparing runs of a process with runs of its refinement. Places and transitions of runs are related to places and transitions of processes respectively by the mappings ρ_B and ρ_E. We also have to relate elements of a subprocess to the refined transition. Therefore, given a process P with a transition t and a process $\overline{P}$ resulting from a refinement of t by a subprocess, we define a mapping ϕ from elements of $\overline{P}$ to elements of P [7]. ϕ maps all elements of the subprocess to t and is the identity function for all other elements of $\overline{P}$. Abusing terminology, we will call the process constructed by several subsequent transition refinements still a refinement of the original process. Its associated mapping ϕ is defined as the composition of the mappings associated to the stepwise refinements. In other words, ϕ maps each element of the refinement to the associated transition of the original process.

Transitions in runs represent occurrences of transitions in processes. If a transition is refined by a subprocess, the corresponding refinement of the transition occurrence in the run is done by a run of the subprocess. Formally, we use the same definition as above. Therefore, we have do define runs as particular subprocesses. This is done in the obvious way: The set of input transitions consists only of the unique transition of the run which is mapped to an input transition, and likewise for output transitions.

Proposition 3. *Let P be a process with a transition t and let P^t be a refinement of t. Let $\overline{P}$ be the resulting process. Each run $\overline{R}$ of $\overline{P}$ is a refinement of a run R*

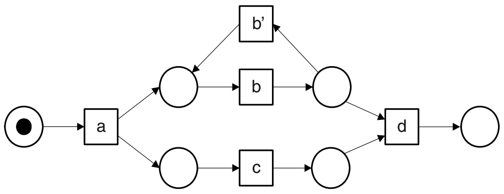


Fig. 8.

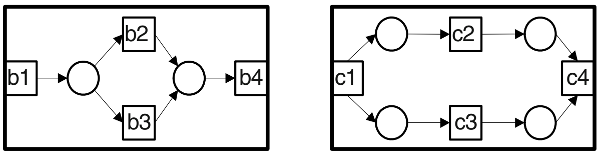


Fig. 9.

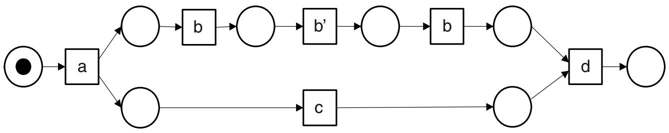


Fig. 10.

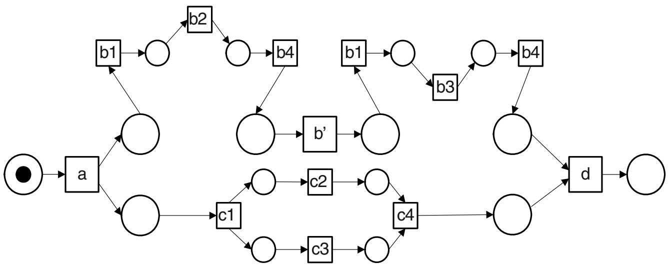


Fig. 11.

of P where each occurrence of t (i.e., each transition of R mapped to t by ρ_E) is refined by a run R^t of P^t. Conversely, each refinement of a run R of P with respect to all occurrences of t (refined by a run R^t of P^t each) is a run of $\overline{P}$.

In other words, the diagram given in Figure 7 commutes.

Figure 8 shows a process (considered a main process here, hence the input and output place), Figure 9 shoes refinements of transitions b and c, Figure 10 shows a run (where for the sake of completeness also a marked input place and an output place is added) and Figure 11 shows both, a run of the refined net and a refinement of the run of Figure 10.

There exist other notions of concurrent runs, given by partial languages. Each run of a process defines a partial order on the transitions occurring in the process in an obvious way: A transition t precedes a transition t' if there is a directed path leading from t to t'. Each partial order over the same transitions including this order is sometimes considered a possible execution of the process. It is known that linearizations of the partial orders given by all runs correspond to all occurrence sequences, when transitions are replaced by their respective images according to the mapping ρ_E. It is well known how to construct runs from a given process [33] and it is straight-forward to check whether a Petri net is a run of a process. The same is less obvious for partial language executions, but [24] (supported by the tool described in [25]) shows that there is an efficient way to answer this question, too. For runs as defined above, [20] (based on [13]) shows how to obtain a process by folding a given set of runs such that these runs are actually runs of the process. For partial languages, a synthesis algorithm is provided in [26].

4 Process Construction Using Synthesis and Mining Techniques

Although this section is the core of the paper, it only roughly describes how to construct a process from given runs of different abstraction levels, provided by experts. The motivation was given in the introduction.

1. First, identify start conditions, start actions and end actions of the process to be defined.
2. Let relevant people define runs of the process on an abstract level.
3. Agree on the abstract actions that occur in these runs.
4. Synthesize a process from the runs using the techniques from [20,26,3].
5. Validate this process by generating runs (see next section).
6. Identify actions that have to be refined.
7. Identify experts that can provide information (namely runs) for these actions.
8. Agree on actions that occur in these runs.
9. Synthesize a process from the runs using the techniques from [20,26,3].
10. Validate this process by generating runs (see next section).
11. Iterate the procedure if there is a need for a higher refinement hierarchy.
12. Construct the entire flat process by refining all transitions on the abstract level (possibly to be repeated for the next levels).
13. Analyze this process w.r.t. appropriate correctness criteria, such as liveness, using known techniques for Petri nets.

5 Validation by Generation of Runs

As seen in the previous section, the constructed processes are validated at various steps. Whereas verification means to automatically or semi-automatically prove desired properties, validation aims at checking correctness w.r.t. the reality to be modelled [16,17]. Starting with the VIP project [8], we have developed a validation technique which is based on the construction of runs from a given process. Additionally, desired properties can be edited in the process and checked for all generated runs. This way, the process as well as the specification of properties can be validated by the user of the VIP-tool. For example, he can see which runs are ruled out by a behavioral specification and which runs agree with the specification. If the process is modelled correctly, this way the specification is validated. If the specification can be assumed to hold for the process, this way the process is validated.

In our setting, the VIP approach is used to validate the generated processes on different levels. Users are the respective experts. If it turns out that the generated process can generate undesired runs then an according specification is formulated, validated as described above and finally implemented by an according modification of the process. This procedure is iterated as often as necessary.

Finally, the specifications used in the last step of the procedure described in the previous section is first validated by the VIP tool before it is used to prove correctness of the final process.

The process definition of this paper is closely related to the definition of workflow nets in [1]. However, an important difference is that workflow nets are assumed to have no memory. They are executed starting with an empty marking. In contrast, we assume that processes can remember their past. For example, the process of Figure 2 will execute c and d in its first run, and a and b in its second run. So also the specification of runs has to take additional conditions into account which determine which variant of the process has to be executed.

6 Conclusion

This contribution presents the idea of a systematic stepwise construction of a process net in a completely informal setting. Roughly speaking, this idea copied ideas from conceptual data modelling. In particular, I looked at the KCPM project and its predecessors.

Refinement and the relation between processes and related runs only rapport for partial order semantics, as illustrated in the paper. The idea also heavily depends on techniques to automatically derive processes from runs, techniques which have been developed for synthesis and for mining in the last years. Only recently, such techniques have been developed for partially ordered runs, which is necessary for this approach.

Since we only presented the idea, most of the work has still to be done. There are various technical questions. For example, it is important to reduce the complexity of the synthesis techniques. We are quite confident that the modular approach presented in this paper helps to keep the single processes sufficiently small. Whereas this modularity is vertical, a horizontal composition of modules would be useful in this setting, too.

Another questions which appears to be of importance is how to deal with different granularity of actions formulated by different experts and related terminology problems. Since these questions are not specific to process modelling, results from data modelling might be reusable.

The most important and last point to be mentioned is that the research shall not be guided by theoretical questions but rather by application examples and solid evaluation. Only this way the really important problems will show up, and only this way a really useful concept can arise.

References

1. van der Aalst, W.M.P., van Hee, K.: Workflow Management – Models, Methods and Systems. MIT Press, Cambridge (2002)
2. van der Aalst, W.M.P.: Finding structure in unstructured processes: the case for process mining. In: 7th International Conference on Application of Concurrency to System Design (ACSD), pp. 3–12. IEEE, Los Alamitos (2007)
3. Bergenthum, R., Desel, J., Lorenz, R., Mauser, S.: Process mining based on regions of languages. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 375–383. Springer, Heidelberg (2007)
4. Business Process Management. LNCS, vol. 1806, 2678, 3080, 4102, 4714. Springer, Heidelberg (2000-2007)
5. De Antonellis, V., Demo, G.B.: Requirements Collection and Analysis. Methodology and Tools for Data Base Design, pp. 9–24. North-Holland, Amsterdam (1983)
6. Darondeau, P.: Synthesis and control of asynchronous and distributed systems. In: 7th International Conference on Application of Concurrency to System Design (ACSD), pp. 13–22. IEEE, Los Alamitos (2007)
7. Desel, J.: On abstractions of nets. In: Rozenberg, G. (ed.) APN 1991. LNCS, vol. 524, pp. 78–92. Springer, Heidelberg (1991)
8. Desel, J., Oberweis, A.: Validierung von Informationssystemen durch Auswertung halbgeordneter Petrinetz-Simulationsläufe. Formale Grundlagen für den Entwurf von Informationssystemen. Informatik-Berichte der Universität Hannover Nr.03/94, pp. 132–138 (1994)
9. Desel, J., Reisig, W.: The synthesis problem of Petri nets. Acta Informatica 33, 297–315 (1996)
10. Desel, J., Oberweis, A.: Petri-Netze in der Angewandten Informatik. Wirtschaftsinformatik 38(4), 359–367 (1996)
11. Desel, J., Oberweis, A., Reisig, W., Rozenberg, G. (eds.): Petri nets and Business Process Management. Dagstuhl-Seminar-Report Nr. 217 (1998)
12. Desel, J.: Formaler Umgang mit semi-formalen Ablaufbeschreibungen. Angewandte Informatik and Formale Beschreibungsverfahren, Teubner-Texte zur Informatik 29, pp. 45–60, Teubner (1999)
13. Desel, J., Erwin, T.: Hybrid specifications: looking at workflows from a runtime perspective. International Journal of Computer System Science & Engineering 15(5), 291–302 (2000)
14. Desel, J.: Validation of process models by construction of process nets. In: van der Aalst, W.M.P., Desel, J., Oberweis, A. (eds.) Business Process Management. LNCS, vol. 1806, pp. 108–126. Springer, Heidelberg (2000)

15. Desel, J., Juhás, G.: What is a Petri net? In: Ehrig, H., Juhás, G., Padberg, J., Rozenberg, G. (eds.) APN 2001. LNCS, vol. 2128, pp. 1–25. Springer, Heidelberg (2001)
16. 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)
17. Desel, J.: Validierung and Verifikation von Prozessmodellen. In: Promise 2002 – GI-Workshop über Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen, Gesellschaft für Informatik. LNI, vol. P-21, pp. 78–80 (2002)
18. Desel, J., Juhás, G., Lorenz, R., Neumair, C.: Modelling and validation with VIP-tool. In: van der Aalst, W.M.P., ter Hofstede, A.H.M., Weske, M. (eds.) BPM 2003. LNCS, vol. 2678, pp. 380–389. Springer, Heidelberg (2003)
19. Desel, J.: Process modelling using Petri nets. In: [21], pp. 147–177 (2005)
20. van Dongen, B.F., Desel, J., van der Aalst, W.M.P.: Aggregating causal runs to workflow nets, Eindhoven University of Technology. BETA Working Paper Series, vol. WP 173 (2006)
21. Dumas, M., van der Aalst, W.M.P., ter Hofstede, A.H.M.: Process-Aware Information Systems – Bridging People and Software through Process Technology. Wiley, Chichester (2005)
22. Ehrenfeucht, A., Rozenberg, G.: Partial (set) 2-structures; Part I and II. Acta Informatica 27, 315–342 (1990)
23. Fliedl, G., Kop, C., Mayr, H.C.: From scenarios to KCPM dynamic schemas – aspects of automatic mapping. In: Natural Language Processing and Information Systems, 8th International Conference on Applications of Natural Language to Information Systems, Gesellschaft für Informatik. LNI, vol. P-29, pp. 91–105 (2003)
24. Juhás, G., Lorenz, R., Desel, J.: Can I execute your scenario in my net? In: Ciardo, G., Darondeau, P. (eds.) ICATPN 2005. LNCS, vol. 3536, pp. 289–308. Springer, Heidelberg (2005)
25. Lorenz, R., Bergenthum, R., Desel, J., Juhás, G.: Can I execute my scenario in your net? VipTool tells you! In: Donatelli, S., Thiagarajan, P.S. (eds.) ICATPN 2006. LNCS, vol. 4024, pp. 381–390. Springer, Heidelberg (2006)
26. Lorenz, R., Bergenthum, R., Desel, J., Mauser, S.: Synthesis of Petri nets from finite partial languages. In: 7th International Conference on Application of Concurrency to System Design (ACSD), pp. 157–166. IEEE, Los Alamitos (2007)
27. Mayr, H.C., Dittrich, K.R., Lockemann, P.C.: Datenbankentwurf. Kapitel 5. In: Datenbank-Handbuch, pp. 481–557. Springer, Heidelberg (1987)
28. Mayr, H.C., Schnattler, M.: Vorkonzeptuelle und konzeptuelle Dynamik-Modellierung. In: Petri-Netze im Einsatz für Entwurf und Entwicklung von Informationssystemen, Informatik aktuell, pp. 3–18. Springer, Heidelberg (1993)
29. Mayr, H.C.: Towards business process modeling in KCPM. In: [11], p. 27 (1998)
30. Mayr, H.C., Kop, C.: A user centered approach to requirements modeling. In: Modellierung 2002, Gesellschaft für Informatik. LNI, vol. P-12, pp. 75–86 (2002)
31. Mayr, H.C., Kop, C., Esberger, D.: Business process modeling and requirements modeling. In: First International Conference on the Digital Society (ICDS 2007), Los Alamitos, CA, pp. 8–11 (2007)
32. Process Mining Group, Eindhoven University of Technology: `www.processmining.org`
33. Reisig, W.: A Primer in Petri Net Design. Springer, Heidelberg (1992)
34. Weske, M.: Business Process Management – Concepts, Languages and Architectures. Springer, Heidelberg (2007)

Temporal Consistency of View Based Interorganizational Workflows

Johann Eder[1,2] and Amirreza Tahamtan[2]

[1] Dept. of Informatics Systems, Alpen-Adria University of Klagenfurt, A-9020 Klagenfurt, Austria
eder@isys.uni-klu.ac.at

[2] Dept. of Knowledge and Business Engineering, University of Vienna, Rathausstrasse 19/9, A-1010 Vienna, Austria
amirreza.tahamtan@univie.ac.at

Abstract. Interorganizational workflows are a major step in automating B2B electronic commerce and to support collaborations of organizations on the technical level. Process views are an important conceptual modelling approach for interorganizational workflows as they allow interaction and communication while internal and private parts of the process can be hidden. However, it is essential to guarantee that an interorganizational workflow is free of conflicts and the overall quality assurances of the whole workflow can be achieved. This paper proposes an approach for checking temporal consistency of interorganizational workflows crossing boundaries of organizations.

Keywords: Interorganizational Workflow, Workflow View, Temporal Constraints, Conformance, Consistency.

1 Introduction

Automation of tasks and processes plays an essential role in order to provide cheaper services with a better quality. In recent years, workflow management systems (WfMS) have provided an effective and powerful tool for this aim. It is necessary for organizations to build short or long term cooperations with other organizations to reach the overall goal of a business process. Spread of the internet, as a means of communication , provides a powerful infrastructure for interorganizational workflows. Such workflows enable autonomous organizations, which may be geographically distant, to cooperate with each other. A challenging point when constructing such workflows is the balance between autonomy and cooperation. On one hand, organizations must reveal some information to the business partners necessary for communication, monitoring, tracking purposes, etc. and on the other hand organizations want to hide their internal process logic to protect their know-how and improve their business secrecy, or simply keep the possibility for quickly improving their business processes or adapting their business processes to changing environments without having to go through a process of changing collaboration agreements. For external partners, as well, it is neither necessary nor desirable to have access to all parts of the provider's internal workflow as they do not want to be overloaded by unnecessary data and messages and are interested in those parts of the workflow which address them. Additionally, it is a well established principle in software engineering to keep the coupling between parts as loose as possible.

Workflow views provide a mean for this aim. Views define the accessible and visible parts of a process for external partners. A view can be a subset of the (activities) of the original workflow or represent the original workflow in an abstracted or aggregated fashion. External users communicate and interact with the view and not with the private executable workflow. Views are in charge of redirecting the data and information to the executable workflow and also forwarding them to the external user. In case of Business-to-Customer (B2C) the external user is normally a human user or an agent and in case of Business-to-Business (B2B) application the external user is another organization which in turn may interact and take part in the interorganizational workflow through its view. By using views, organizations are able to expose as little information as possible but sufficient for the communication and interaction to reach the goals of the business process. In this work we focus on Business-to-Business applications and assume that each organization takes part in the interorganizational workflow through its workflow view i.e. the shared business process is an integration of the partners' views. Partners communicate by their views and not with their private workflows. Fig. 1 depicts this scenario. Three partners take part in the shared business process, each with their own private workflow. Partners provide a view on their private workflow and the interaction with other partners in the shared business process proceeds through views of the partners. In such a case,

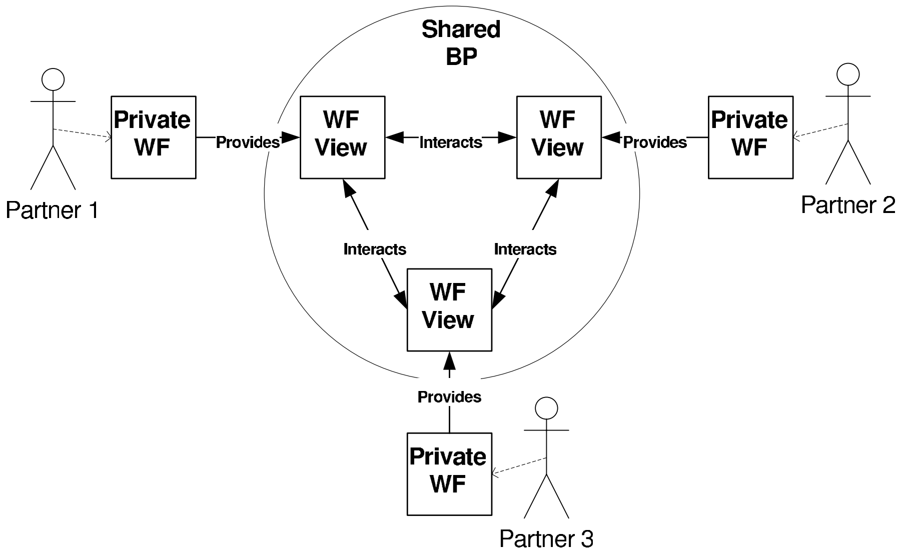


Fig. 1. An interorganizational workflow composed of views

it must be ensured that views are conformant with each other and can interact without conflict. Several issues such as structural conformance, data flow conformance, messaging conformance and temporal conformance must be checked in order to ensure the consistency and conformance of the overall interorganizational workflow. In this work we study the temporal conformance of workflow views in an interorganizational setting to ensure that interactions are done in a timely manner with respect to local and global constraints. A temporally conformant interorganizational workflow enables partners to execute without violating temporal constraints such as explicitly assigned deadlines and also reduces the cost of process because fewer temporal exception must be raised and handled accordingly. Such a mechanism is a helpful tool in hand of process designers and managers to foresee the possible upcoming temporal violations and react to them in a predictive manner. In this paper only the case with two partners is considered but in a straightforward manner the concepts can be extended and applied when more than two partners are involved.

2 Related Work

[1] applies views to enable interaction between autonomous workflows in an interorganizational setting. Partners in a business process own their private workflows and take part in a shared business process, here called coalition workflow, through views. A view is an abstraction of the corresponding private workflow and reflects the communication requirements of the coalition workflow. The views outsource the execution to the according private workflow. A tightly coupled approach between private workflows and its view(s) based on state dependencies and a loosely coupled between views in the coalition workflow based on control flow dependencies is applied. This work mainly discusses the views from an state perspective and is silent on correctness issues of views or how views may be built correctly. [2] proposes an order preserving approach for constructing views. This work also applies an aggregation approach for constructing virtual activities out of base activities of the private workflow. The authors present three rules for construction of process views and an algorithm for automatic generation of possible process views with respect to these rules. [3] can be seen as a combination of [1,2]. The construction of process views originate from the authors' previous work [2] enriched with the state mapping used in [1]. However the authors use slightly different terminology for naming the states. The state of the virtual activity is a function of the states of its member base activities which can be used for monitoring the state of the virtual activities by other partners. [4] proposes an approach based on software oriented architecture (SOA) for interconnection of workflows. A semantic registry for publishing and discovery of services is proposed, enabling other organizations to build a cross-organizational cooperation by finding and binding to other partners. Views contain only cooperative tasks. i.e. tasks that either send or receive data to/from external workflows. The interorganizational workflow consists of virtual activities which are in charge of transferring data to/from executable activities. A virtual activity is a subset of cooperative

activities. A trusted third party is responsible for monitoring the interaction policies. This work can be seen as an effort for enabling interorganizational workflows for a dynamic setting where partners and their interconnections are not predefined. However, the need for negotiation and a contract for setting up the interaction policy and identification of public virtual activities still remains and is only shifted after finding the partners from the registry. Besides, the assumption that a view consists of only cooperative tasks, may not be sufficient in all scenarios. [5] propose to use views for interoperability in cross-organizational workflows. The balance between security and trust is considered to be achieved through views. In other words, views restrict the access on the workflow and conceal the internal, private or unnecessary information. A view is defined as a structurally correct subset of workflow definition. The authors consider mainly access rights on objects associated with a view. Moreover, they do not describe how views can be constructed from a workflow and an integrated view in an interorganizational interoperation is missing. [5] presents a technique for identification of relevant tasks included in a view. With respect to a given role, some metrics for identification of most relevant tasks are defined . Workflow operations are used as criteria to evaluate role-task relevance. The metrics are extracted and calculated by analyzing the workflow logs. The proposed technique can be used for an optimized construction of views. However, a history of interaction between external user and the private process should be at hand. [7] introduces a methodology for decomposition of complex processes in scientific domain and building view based on flows. The interactions among flows are triggered by external messages. Each flow can have multiple views. The views of a flow are built based on the analysis of the required incoming messages, its immediate responses and the dependency between data and messages. The overall process is an integration of views of all partners. [8] introduces an object oriented approach for workflow views called the object deputy model. The definition and concept of views are taken from [5]. A deputy object has its own persistent identifier and is a subclass of its source object. A bilateral link between objects and one of its deputies allows for inheritance and update propagation. In this work views are used for two main purposes: restriction of access and composition. Authors do not mention how views can be constructed for mentioned applications. A workflow restriction view is defined the same as a workflow view, a structurally correct subset of a workflow definition. A workflow composition view is a virtual workflow composed of components from different workflows, which may span across organizational boundaries.

The time management concepts come from a related field, namely workflow management research. One of the eraliest works on time properties is [9]. Allen describes a temporal representation using the concept of temporal intervals and introduces a hierarchical representation of relationships between temporal intervals applying constraints propagation techniques. This work describes thirteen ways in which an ordered pair of interval can be related. [10] provides a methodology for calculating temporal plans of workflows at design time, instantiation time and run time. It considers several temporal constraints like lower bound, upper bound and fixed-date constraints and explains how these constraints can be incorporated. Moreover, a model for monitoring the satisfaction of temporal constraints during run time is provided. Eder et al. in [11] present a model for calculation of temporal plans and propose some algorithms for calculation and incorporation of time constraints. [12] provides a technique for modeling and checking time constraints whilst conditional and parallel branches are discriminated. In addition, an unfolding-method for detection of scheduling conflicts is provided. Marjanovic in [13] represents the notions of duration space and instantiation space and describes a technique for verification of temporal constraints in production worklfows. Our approach is complementary to that introduced in [13] in the way that a temporal plan for execution of all activities is calculated.Temporal aspects of web services, a very realted field to workflow research, have been studied in [14,15,16]. [14] uses temporal abstractions of business protocols for their compatibility and replaceability analysis based on a finite state machine formalism. [15,16] exploit an extension of timed automata formalism called Web Service Time Transition System (WSTTS) for modeling time properties of web services. [17] provides an architecture for interorganizational workflows and [18] provides an algorithm for checking the structural conformance of the participating partners' workflows.

3 Workflow Views

We assume that each partner has a private workflow which is only visible to the owner of the workflow and external parties have no knowledge about the process structure and internal logic of the private workflow. In order to provide the necessary information for communication and interaction with other partners and/or service requester, the owner of the workflow provides views for each partner. In other word, a workflow can have many workflow views each for one partner or for a group of partners. In this way the workflow is able to interact with external parties whilst protecting private aspects of the process. In this paper we assume two ways for constructing views: abstraction and aggregation.

By application of abstraction operator (called τ-operator in process algebra [19]) it is possible to make parts of the process invisible to other external observer. This operator provides a mean for construction of views. By application of this operator, parts of the process that do not contribute to the interaction with another partner, are not interesting for external partner or are intended to be hidden because of privacy issues, can made unobservable and the rest of the process can be exposed as the view on the process. Left part of fig. 2 depicts an example. The internal workflow (left most part) consists of four activities. After receipt of the request, the request is processed, the history of the buyer profile is updated and the results are sent back to the requester. After making two activities invisible, the requester can interact with the view of the workflow containing two tasks *Receive Request* and *Send Results* which are sufficient for interaction between the requester and the provider.

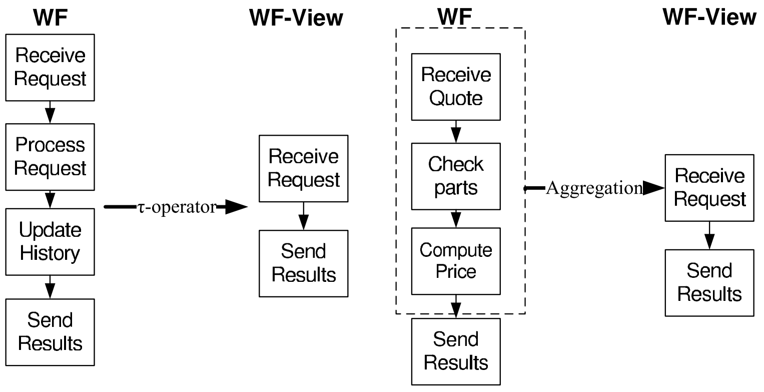


Fig. 2. Application of abstraction operator(left part), Aggregation (right part)

Another approach is using virtual activities for constructing views. In this approach some executable activities of the private workflow can be grouped into a so-called virtual activity. Virtual activities are in charge of forwarding and receiving data to and from executable activities of the private workflow. The right part of fig. 2 illustrates an exemplary application of aggregation. In this simple example, at first the quote is received by the workflow. It is checked which parts are needed for the production, the price is computed and finally the result is sent back to the external partner. To construct a view, the first three activities are grouped into the virtual activity "Receive Request". The service requester interacts with the view and the view sends the results to the executable activities and also to the external partner. Note that there is not a unique way of constructing views. Executable activities can be grouped in many different ways into virtual activities. It may depend on the interacting roles, constraints of the workflow owner and the context of the interorganizational workflow.

4 Temporal Plans of Workflows and Their Views

Assume a situation when a view of a workflow is defined and this view takes part in an interorganizational workflow. It is necessary to calculate the temporal execution plan of the views. In order to avoid temporal conflicts and exceptions it must be clear for the other partners in which temporal interval they can send messages and in which temporal interval they can expect messages. We assume that each workflow has a deadline and based on the calculation of the temporal plan of the private workflow, we calculate the temporal plan of its views. In other words we calculate a valid interval in which an activity, executable or virtual can execute.

4.1 Model Description

Time is considered as discrete values expressed in some basic units like Minutes (M), Seconds (S) etc. The basic concepts used for calculation of the temporal plans come originally from the field of project management and operations research [20] like CPM and PERT. There are two kinds of temporal constraints:

– **Implicit constraints** are derived implicitly from the structure of the process e.g. an activity can start execution if and only if all its predecessors have finished execution.
– **Explicit Constraints** e.g. assigned deadlines, can be set explicitly by the process designer or enforced by law, regulations or business rules

As the basic modeling language we used timed activity graphs or timed graphs [10,11]. They are familiar workflow graphs where nodes correspond to activities and edges the dependencies between activities, enriched with temporal information. Fig. 3 shows an example of a node.

Activity Name	Activity Duration
Best Case Earlies Possible Start	Best Case Latest Allowed End
Worst Case Earlies Possible Start	Worst Case Latest Allowed End

Fig. 3. A node in the timed graph

All activities have durations. $a.d$ denotes the duration of an activity a. At the first use of the model an estimation of the activity durations, e.g. expert opinion, may be used. Later, workflow logs can be mined for actual activity durations. In this work deterministic values for activity durations are used. We are aware that activity durations may vary in real life applications. However, we use fixed values for clarification of the concepts and avoid the math involved in probability distributions. We calculate an interval in which an activity may execute. This interval is delimited by *earliest possible start* (eps-value) and *latest allowed end* (lae-value). $a.eps$ denotes the *eps*-value of an activity a and is the earliest point in time in which an activity a can start execution. $a.lae$ represents the latest point in time in which an activity a can finish execution in order to hold the assigned deadline. Both *eps* and *lae* values are calculated for *best case* and *worst case*. If there is an XOR-split in the workflow, there are multiple paths to be chosen and based on some evaluated conditions at run-time one of the available paths is executed. The best case is given, if the shortest path is executed and we have the worst case when the longest path is executed. It is possible that XOR-split has other branches whose length lies between best and worst cases. In this paper only best and worst cases are considered. If all branches of an XOR-split have the same length, both cases have the same *eps* and $lae - values$. *eps*-values are calculated in a forward pass by adding the *eps*-value of the predecessor to its duration. For example $b.eps = a.eps + a.d$ if an activity a is a predecessor of an activity b. If an activity a has multiple predecessors, e.g. if an activity a is an immediate successor of an AND-join, the maximum of *eps*-value of predecessors of a is taken into account. The $eps - value$ of the first activity or the set of first activities are set to 0.

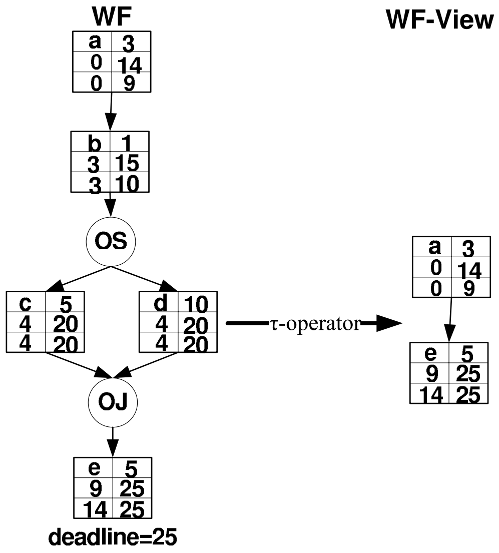


Fig. 4. Temporal plan of a WF and its view by application of τ-operator

In contrast to the *eps*-values, *lae*-values are calculated in a backward pass by subtracting the *lae*-value of the successor from its duration, e.g. $a.lae = b.lae - b.d$ if an activity b is a successor of an activity a. If an activity a has multiple successors, e.g. if activity a is an immediate successor of an AND-split, the minimum of *eps*-value of predecessors of a is taken into account.The *lae*-values of the last activity or the set of last activities are set to the assigned deadline. If no deadline is assigned, length of the longest path may be used as deadline. In this case the critical path has no buffer time available.

In addition to *eps* and *lae*-values we can define *earliest possible end* (epe-value) and *latest allowed start* (las-value) for the activities. However, given activity durations they may be calculated interchangeably applying the following formulas: $a.epe = a.eps + a.d$ and $a.lae = a.las + a.d$. When parallel structures are present in the model, the longest path (worst case calculation) between the split and join nodes is always considered. Hence, best case has the same temporal values as the worst case. The reason is that the parallel join must wait in any case for the longest branch of the parallel structure to commit. For a more detailed discussion refer to [12]. Given the deadline of 25, left Part of Fig. 4 shows the calculated temporal plan of the depicted workflow.

4.2 Calculation of Timed Graphs of Views

As noted, we assume two ways for constructing views: application of τ-operator and aggregation. If τ-operator is applied on the executable private workflow process, there is no need to recalculate the timed execution plan of the view. In this case, after calculation of the temporal plan of the private process, the calculated values can be directly taken over for the activities contained in the

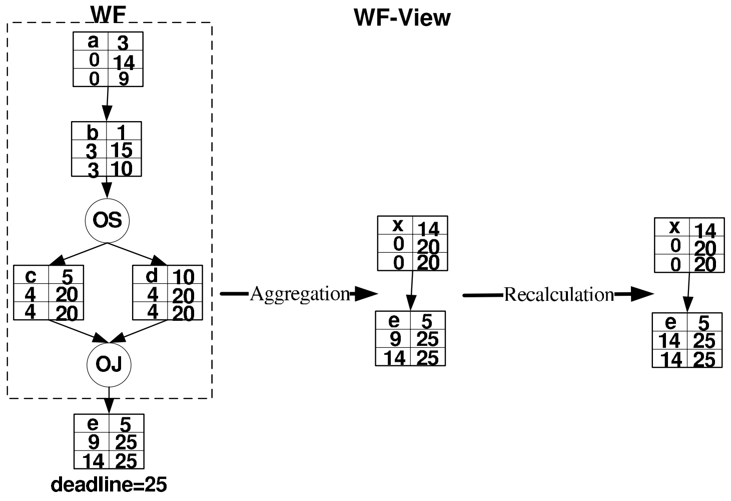


Fig. 5. Temporal plan of a view by application of aggregation

view. See Fig. 4 for an example. If aggregation is applied, the temporal values of the virtual activities in views can be calculated in a straightforward manner. The duration of the virtual activity is sum of the durations of the executable activities contained in it. Note that the parallel or join structure can be considered as one activity whilst the length of its longest path is taken as its duration. This yields to a worst case estimation of the duration of the virtual activity. But for other partners it is more important to know in which time interval they can interact with the workflow with a valid temporal window. Hence, worst case duration for the virtual activities is a reasonable estimation. This yields always to a temporally valid interaction with the view and therefore guarantees the correct temporal behavior of the flow. The eps-value of the virtual activity is the eps-value of its first executable activity or the minimum value of the set of its first executable activities. The eps-value of the virtual activity is the eps-value of its last executable activity or the maximum value of the set of its last executable activities. The temporal values of other activities that are not contained in another virtual activity can be taken over directly from temporal plan of the private workflow. Figure 5 shows an example for this case. As can be seen in Fig. 5, the virtual activity x has a duration of 14 time units, where as the activity e in the view has an eps-value of 9 time units. It implies that the virtual activity e in the view of Fig. 5 can start execution before the virtual activity x (its direct predecessor) has finished execution, which is a violation of the structural constraints. The reason is that the virtual activity x reflects its worst case values whilst eps-value of activity e is its best case value. To resolve this issue, it is necessary to recalculate the *eps*-values of the view. Note that the *lae*-values remain unaffected. This can be done in the same way as explained before (the forward pass calculation). The end results are depicted in the right most part of the Fig. 5.

5 Temporal Consistency of Views

In an interorganization workflow several partners take part in the public shared process. We assume that each partner participates in the interorganizational workflow with its view. Fig. 6 illustrates such a scenario. This workflow consists of there partners, a buyer, a seller and a supplier. The dashed lines between virtual activities demonstrate a message exchange and the number above them shows the ordering of the message sequence. This example contains interactions between a buyer and a seller (right part) and seller and a supplier (left part). For sake of brevity the private workflows of the participating partners are omitted and only the views are illustrated.

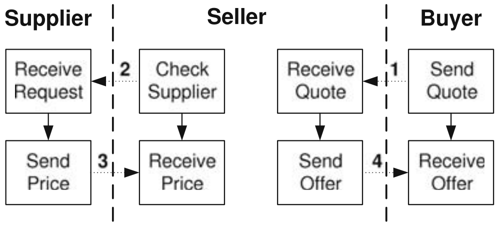


Fig. 6. An interorganizational workflow

The buyer sends a request for quote to the seller (message 1), the seller checks if the needed parts for the production of the item is deliverable by its supplier (message 2), if the parts required for manufacturing are deliverable by the supplier, the supplier sends the price to the seller (message 3). After receipt of the price from the supplier, the seller calculates the whole price and sends it back to the buyer (message 4). In order to check if two activities of different views are temporally consistent, it must be checked if the execution intervals of both corresponding activities overlap. Corresponding activities are in this sense those activities that communicate with each other. In other words, activities that are sender or receiver of the same message e.g. activities "Send Quote" of buyer and "receive Quote" of seller or activities "Check Supplier" of seller and "Receive Request" of supplier. Put it another way, in order to check the temporal consistency of two corresponding activities it must be checked if there is any temporal interval in which both activities can execute. Let $x.[eps, lae]$ denote the valid execution interval of an activity x contained in a view. This interval can be calculated based on the temporal execution plan of the private executable workflow as explained earlier. Let $wc.x.[eps, lae]$ and $bc.x.[eps, lae]$ denote the execution interval of an activity x for worst case and best case respectively. Let x be an activity and y its corresponding activity in another view. Five cases can be identified:

– $x.[eps, lae] \bigcap y.[eps, lae] = \varnothing$, these activities can not interact.
– $wc.x.[eps, lae] \bigcap wc.y.[eps, lae] \neq \varnothing$, in any case both activities can interact

- $bc.x.[eps, lae] \bigcap wc.y.[eps, lae] \neq \varnothing$, activity y can interact in any case but activity x only at its best case
- $wc.x.[eps, lae] \bigcap bc.y.[eps, lae] \neq \varnothing$, activity x can interact in any case but activity y only at its best case
- $bc.x.[eps, lae] \bigcap bc.y.[eps, lae] \neq \varnothing$, in this case, both activities can interact only at their best cases

If there is no overlap of the execution interval of corresponding activities, those activities are temporally not consistent. Note that there are some intermediate or mixed forms between best case and worst case that we do not consider. The proposed approach for checking temporal consistency between two activities can be extended in a straightforward manner to other forms of communications e.g. if an activity has multiple receivers. In such a case it must be checked if the intersection of more than two temporal intervals is not empty and in which case they can communicate, i.e. worst case or best case. In order to decide if two views are temporally consistent, all pairs of corresponding activities must be temporally consistent. Finally an interorganizational workflow is temporally consistent if all of its participating views are temporally consistent. Other issues for full consistency like messaging conformance, data flow conformance and structural conformance are out of scope of this paper.

6 Conclusion

Consistency of interorganizational workflows are important issues to be taken into account when designing, negotiating and executing such a coalition among different organizations. In this paper we have presented a technique for temporal consistency checking of an interorganizational workflow whilst each partner takes part with its view. This technique helps process designers to check the consistency before executing and hence reducing the cost of process as fewer exceptions must be raised and handled because of temporal violations.

References

1. Schulz, K.A., Orlowska, M.E.: Facilitating cross-organisational workflows with a workflow view approach. Data and Knowledge Engineering 51(1), 109–147 (2004)
2. Liu, D.R., Shen, M.: Workflow modeling for virtual processes: An order-preserving process-view approach. Information Systems 28(6), 505–532 (2003)
3. Liu, D.R., Shen, M.: Business-to-business workflow interoperation based on process-views. Decision Support Systems 38(3), 399–419 (2004)
4. Chebbi, I., Dustdar, S., Tata, S.: The view-based approach to dynamic inter-organizational workflow cooperation. Data and Knowledge Engineering 56(2), 139–173 (2006)
5. Chiu, D.K.W., Cheung, S.C., Karlapalem, K., Li, Q., Till, S.: Workflow View Driven Cross-Organizational Interoperability in a Web-Service Environment. In: Bussler, C.J., McIlraith, S.A., Orlowska, M.E., Pernici, B., Yang, J. (eds.) CAiSE 2002 and WES 2002. LNCS, vol. 2512, pp. 41–56. Springer, Heidelberg (2002)

6. Liu, D.R., Shen, M.: Discovering role-relevant process-views for disseminating process knowledge. Expert Systems with Applications 26(3), 301–310 (2004)
7. Li, Q., Shan, Z., Hung, P.C.K., Chiu, D.K.W., Cheung, S.C.: Flows and Views for Scalable Scientific Integration. In: Procs. of InfoScale 2006, Hong Kong. ACM International Conference Proceeding Series, vol. 152 (2006)
8. Shan, Z., Long, Z., Luo, Y., Peng, Z.: Object-Oriented Realization of Workflow Views for Web Services- An Object Deputy Model Based Approach. In: Li, Q., Wang, G., Feng, L. (eds.) WAIM 2004. LNCS, vol. 3129, pp. 468–477. Springer, Heidelberg (2004)
9. Allen, J.F.: Maintaining knowledge about temporal intervals. Communications of the ACM 26(11), 832–843 (1983)
10. Eder, J., Panagos, E.: Managing Time in Workflow Systems. In: Lawrence, P. (ed.) WfMC WorkFlow Handbook 2001, J. Wiley & Sons, Chichester (2001)
11. Eder, J., Panagos, E., Rabinovich, M.: Time Constraints in Workow Systems. In: Jarke, M., Oberweis, A. (eds.) CAiSE 1999. LNCS, vol. 1626, pp. 286–300. Springer, Heidelberg (1999)
12. Eder, J., Gruber, W., Panagos, E.: Temporal modeling of workflows with conditional execution paths. In: Ibrahim, M., Küng, J., Revell, N. (eds.) DEXA 2000. LNCS, vol. 1873, Springer, Heidelberg (2000)
13. Marjanovic, O.: Dynamic Verification of Temporal Constraints in Production Workflows. In: Proc. of the Australasian Database Conference (2000)
14. Benatallah, B., Casati, F., Ponge, J., Toumani, F.: On Temporal Abstractions of Web Service Protocols. In: Proc. of CAiSE Forum (2005)
15. Kazhamiakin, R., Pandya, P., Pistore, M.: Timed Modelling and Analysis in Web Service Compositions. In: Proc. of ARES 2006 (2006)
16. Kazhamiakin, R., Pandya, P., Pistore, M.: Representation, Verification, and Computation of Timed Properties in Web Service Compositions. In: Proc. of ICWS 2006 (2006)
17. Eder, J., Lehmann, M., Tahamtan, A.: Choreographies as Federations of Choreographies and Orchestrations. In: Proc. of International Workshop on Conceptual Modeling of Service-Oriented Software Systems, Tuscon, AZ, USA (2006)
18. Eder, J., Lehmann, M., Tahamtan, A.: Conformance Test of Federated Choreographies. In: Proc. of the 3rd International Conference on Interoperability for Enterprise Software and Applications (I-ESA 2007), Madeira Islands, Portugal (2007)
19. Bergstra, J.A., Klop, J.W.: Algebra of communicating processes with abstraction. Comput. Sci. 37, 77–121 (1985)
20. Philipose, S.: Operations Research A Practical Approach. Tata McGrawHill, New Delhi, New York (1986)

Engineering Design Performance (Extended Abstract)

Vadim Ermolayev[1], Wolf-Ekkehard Matzke[2], and Richard Sohnius[2]

[1] Zaporozhye National Univ., 66, Zhukovskogo st., 69063 Zaporozhye, Ukraine
vadim@ermolayev.com
[2] Cadence Design Systems, GmbH, Mozartstr. 2, D.85622 Feldkirchen, Germany
{wolf,sohnius}@cadence.com

A reader may be slightly puzzled by the title. Indeed, it might say either what performance in engineering design is, or how to engineer design performance. We mean both. Research we are going to report about aims at developing a rigorous engineering methodology for reaching optimal performance in the processes of engineering design. The methodology and the software tool are based on a fine-grained model of performance and its environment. The domain we specifically focus on for validation, evaluation and further exploitation is engineering design in microelectronics and integrated circuits. However, the models presented in this talk are more generic and may be applied in various industrial sectors or academic disciplines.

Our research in formalizing design performance has solid industrial motivation. In the sector of microelectronic and integrated circuits design this motivation can be demonstrated using a system law formulated by Gordon Moore [1]. One of its important corollaries says that the number of components on a chip doubles roughly every 10 to 24 months. It is known in the sector that doubling the number of components crammed onto a silicon chip causes approximately the order of magnitude increase in engineering design effort [2]. In contrast, the productivity of engineering design environments (electronic design automation tools, technologies, methodologies, skills), although increasing, grows substantially slower. Since time to market frame remains roughly constant, the only way to fit this window is hiring more designers and blowing up the budgets. Given a fixed design environment, ensuring that an engineering design process is performed in an optimal way may bring serious benefits, lowering the mentioned gap between design productivity growth and the increase in chip complexity. Engineering design performance should be measured in a way keeping the process on the most optimal possible trajectory.

Today's performance measurement and management practices are based mostly on strategic level benchmarking – finding a place of a company among the others on the industry sector bench. The prevailing methodology is the use of balanced scorecards [3]. Such a benchmarking provides reasonably sound indications of what is good or bad in terms of performance at a strategic management level. However, it does not fully satisfy industrial demand. Krause [4] has analyzed the following weaknesses of the contemporary business performance management (PM) approaches pointing to them as to the reasons of dissatisfaction in industry: (i) strategic PM approaches are driven by "lagging"[1] but not "leading" metrics; (ii) resulting PM methodologies and

[1] The quotes are retained from the source [4].

systems tend to be static[2]; (iii) there is a significant "abstraction" gap between strategic PM approaches and available knowledge acquisition and representation methodologies; (iv) PM is traditionally based on rigid organizational structuring but not on desired properties of the required business processes; (v) the "metrics" are not transparent, are vague and do not clearly reveal the method of measurement and the sources of data. Our experience backed up by the opinion of [5] suggests adding: (vi) the role of a human designer and his pro-active collaboration in a design team is almost neglected; (vii) existing frameworks do not allow revealing the reasons of the weaknesses of a design system.

Answering "why" questions requires a sort of a paradigm shift in modeling and assessing a design system and performed design processes. In addition to building and measuring high-level heuristic performance characteristics we need to acquire and use a deeper knowledge about the processes, their environment, and their performers to make the assessments more justified. These bits of knowledge should cover engineering design processes and their support by technical, human, and organizational components comprising the aspects of design complexity, designers' competencies and abilities, concurrency and iteration of design tasks, dependencies, interfaces and collaboration effects at required level of detail. A negative consequence of taking this complex way in performance assessment and management is that the volume and the complexity of data to be processed are far too high to perform such an analysis by hand before the changes pass the point of no return. Therefore, a methodology and an intelligent software tool capable to partially automate such analyses are required. Once they are available, they become important factors ensuring better, closer to optimal, performance demonstrated by a design system. The objective of PSI[3] project is to develop such a methodology and a tool capable to discover the hidden reasons for the weaknesses of a design system and accounting for the pro-activity of human designers and the stochastic character of the external and internal influences.

In this talk we focus on the presentation of the theoretical framework of PSI. We first give a high-level sketch of the main ideas, concepts, and approaches forming our framework. These are: an Engineering Design Process, a Design System as the environment of an Engineering Design Process, a Performance Management Process and its environment, external and internal events, and Time. As far as process model is central to PSI, we continue with presenting our state-based approach to model an Engineering Design Process as a transformation process of developing a Design Artifact in its representations and transiting from an initial state, through the sequence of intermediate states, to its target state. We assume that such a process is dynamic because a decision to choose the next step on the path is taken at each state by assigning an action to a designer and providing the required resources and tools. We continue with highlighting the complexities induced by the changing character of design specifications and requirements by introducing the notion of a process state which is sensitive to the Design Artifact requirements. Accounting for this sensitivity allows us to propose a rich typology of design actions which properly reflect the specificities of the domain of microelectronic engineering design. Some examples of different types of actions are: creation actions, further elaboration actions, verification actions,

[2] … and linear [6].

[3] Performance Simulation Initiative (PSI) is the R&D project of Cadence Design Systems, GmbH.

de-bugging actions, refinement actions, cessation actions. Next, we proceed with explaining and specifying the model of an Actor – a designer, a manager, who may play different roles in the process, and the model of a design team which is a part of a design system. The model of an actor and a design team, as a self-regulating pro-active social system, is based on intelligent software agents. Like people in a human design team, these agents may have different beliefs about the peers and about their environments as well as different individual behaviors. However, individual agents, as the members of a design team collaborate in an Engineering Design Process and seek for the most possible achievable coherence in their goals by reaching agreements. We conclude by briefly presenting the concept of PSI Performance Measurement and Management Methodology and by outlining the issues which are still in our research agenda.

References

1. Moore, G.: Cramming more components onto integrated circuits. Electronics 38(8) (1965)
2. van Staa, P., Sebeke, C.: Can Multi-Agents Wake Us from IC Design Productivity Nightmare? In: Mařík, V., Vyatkin, V., Colombo, A.W. (eds.) HoloMAS 2007. LNCS (LNAI), vol. 4659, pp. 383–386. Springer, Heidelberg (2007)
3. Kaplan, R.S., Norton, D.P.: The Balanced Scorecard: translating strategy into action. Harvard Business School Press, Boston, Massachusetts (1990)
4. Krause, O.: Beyond BSC: a process based approach to performance management. Measuring Business Excellence 7(3), 4–14 (2003)
5. O'Donnel, F.J., Duffy, A.H.B.: Design Performance. Springer, London (2005)
6. Ermolayev, V., Matzke, W.-E.: Towards Industrial Strength Business Performance Management. In: Mařík, V., Vyatkin, V., Colombo, A.W. (eds.) HoloMAS 2007. LNCS (LNAI), vol. 4659, pp. 387–400. Springer, Heidelberg (2007)

Integration – Reflections on a Pivotal Concept for Designing and Evaluating Information Systems

Ulrich Frank

University of Duisburg-Essen, 45141 Essen, Germany
ulrich.frank@uni-duisburg-essen.de
http://www.wi-inf.uni-duisburg-essen.de/FGFrank/

Abstract. Integration is one of the pivotal concepts with respect to analyzing, designing and evaluating information systems. Integrating software components as well as integrating information systems with the surrounding action system is a core activity of most software development projects. Despite the outstanding importance of integration in Information Systems, only little work has been done on developing a conception of integration. This paper presents a conception of integration that can be applied to information systems as well as to their integration with the action systems they are supposed to support. It allows for differentiating degrees of integration. It also accounts for additional levels of integration, such as meta level or instance level integration.

Keywords: Coupling, integration, IT business alignment, redundancy, reusability, semantics.

1 Introduction

The benefits of integration are probably part of every curriculum in Information Systems and Computer Science. Integration helps to avoid redundancy, hence, contributes to integrity. It is also regarded as an effective means to accelerate business processes and to support decision making. Catering for integration is a pivotal objective during the development of information systems. On the one hand, it suggests designing a common conceptual foundation, e.g. an object model, for the applications or components to be integrated. On the other hand, it recommends the use of system components that are dedicated to integration, such as Database Management Systems, so called Middleware, Workflow Management Systems etc. In addition to the build-time stage, integration is a major issue during system maintenance. A plethora of projects is aimed at healing the problems caused by isolated, heterogeneous software systems. Since these systems are often still vital for a company's performance, IT managers tend to avoid their removal. However, ex post integration faces severe challenges. Numerous consulting companies and tool vendors have responded to this demand by offering methods and systems, referred to as Enterprise Application Integration, Data Warehouse Systems, Middleware etc.

While suprisingly little publications are available that target integration explicitly – two of the rare examples are (cf. [3], [1]) – numerous research activities are aimed at fostering both, ex ante and ex post integration of information systems. Research on conceptual modeling targets the development of modeling languages and methods that allow for developing a common conceptual foundation for integrating the components of an information system. Research on ontologies is intended to support the development of integrated information systems, too ("ontology-driven development", (cf. [4]). Ontologies are also suggested for re-engineering purposes in order to integrate heterogeneous applications (cf. [2]; [4]). Federated databases are a further research field that is focused on ex post integration.

In contrast to the vast amount of research that is focused on the integration of information systems, it is surprising that there is hardly any publication available that is aimed at developing a conception of integration. This may be contributed to the fact that integration is regarded as a core term, which does not need further explanation. However, such an implicit notion of integrity is not sufficient for designing or evaluating information systems. Therefore, I will attempt to develop a conception of integration that should serve this purpose. It is rather a reconstruction of the actual use of the term than a proposal for an entirely new concept. In the first part of the study, a preliminary concept serves to analyze various aspects of integration and related benefits. Subsequently, a refined concept will be presented, which allows differentiating degrees and levels of integration.

2 Integration: Aspects and Benefits

The term integration represents both, a process and its result, i. e. a feature of a system or a set of systems. The focus in this paper is on integration as a system feature. According to the colloquial meaning of the term, integration results from constructing a whole from previously isolated parts. However, such a general conception is not sufficient to account for the peculiarities of information systems. Also, the image of hardware parts that can be composed only, if they have fitting physical shapes, is not appropriate. Information systems are not physical, but linguistic artifacts. Integrating two linguistic artifacts requires that they are able to communicate – either directly or through some kind of mediator. In other words: If two components of a system are not able to communicate, they are not integrated. Note, that I use the term 'component' as an abstraction of software artifacts such as applications, modules, or components in a more specific sense. Communicating means exchanging data, e.g. by sending messages or by using shared memory. However, catering for data exchange only is not sufficient: Communication can work only, if both components share a common interpretation of the data. Otherwise, communication would be a threat to efficiency and to system integrity. Against this background, we can outline a first draft of what we mean by integration: Two components of an information system are integrated, if they can communicate. If they have the ability to communicate but do not make use of it, they are *potentially integrated.*

2.1 Dimensions of Integration

To cope with the complexity of large systems during systems analysis and design, it is common to make use of abstractions that focus on particular aspects only. Static or data abstractions, captured typically through data models, reduce a system to a description of its data structures. Functional abstractions focus on the functions, a system is supposed to perform. Data flow diagrams would be an example of a functional abstraction. Finally, dynamic abstractions represent the control flow within the processes of a system. They could be realized through state charts or – on a higher level of abstraction – through business process models. Object-oriented abstractions can be regarded as a composition of static and functional abstractions. These three essential abstractions of information systems can be used for illustrating different dimensions of integration.

Static Integration
Our preliminary concept of integration can be applied directly to this dimension. Two components of an information system are statically integrated, if they can exchange data – either directly or through a mediator. For this purpose, statically integrated components need to share common concepts that specify the semantics of the exchanged data, e.g. common data types, classes or other data structures. Apparently, static integration allows for sharing data and – as a consequence – for avoiding data redundancy. However, static integration does not imply data sharing. It only requires common concepts that define the semantics of the exchanged data. If, for instance, two components exchange data, they may do this only once in a while and still manage their own – redundant – data in the meantime. In this case, however, there would be a better chance to protect data integrity, e.g. through some kind of synchronisation protocol, because both components would share the same meaning of duplicate data. If two applications have access to common data but do not make use of it, they would be regarded as potentially integrated. The benefits of static integration are obvious. Firstly, there is no need for hazardous and expensive interpretations of data that origin from other components. Secondly, if data are shared by both components, there is no need for updating different copies. Apparently, both aspects contribute to the integrity and efficiency of an information system. Note that the proposed concept of integration requires at least one integrated component to write data that is being used by the other components. Hence, applications that have read-access only to some common resource files provided by the operating system are integrated with the operating system, but not between themselves. Also, a so called data warehouse does not contribute to the integration of legacy applications, since the database that is aggregated from operational level data, is usually not being used by operational level systems. The operational level systems do not share the schema of the data warehouse. Nevertheless, a data warehouse creates the impression of integrated data for those tools that use it. We could call it virtual integration.

Functional Integration
Communication that is restricted to somehow exchanging shared data does not allow for directly addressing another component of the system. This is different with functional integration. Two components of an information system are functionally integrated if they communicate through calling functions. This requires them sharing the meaning of those functions. At the same time, it requires them to be statically integrated. Only then can they use the data that is required or delivered by the functions that were called. Functional integration allows the participating components to share the functions they have in common. Therefore it contributes to avoiding redundant functions, which in turn facilitates system maintenance. In order to reduce the number of communication channels, a central mediator can be deployed. It can either dispatch function calls to a prospective receiver or to a central function library. Note, however, that functional integration is not accomplished, if two components use a common library without exchanging data. If the functions allow for write-access, it will at least be a case of potential integration. While functional integration implies static integration, data integration will often depend on functional integration as well: If, e.g. two components communicate through a common file system or a common DBMS, they both need to share the required access functions. A specific asset of functional integration is the ability to establish data exchange between components that do not share the same data representation – but only the same interfaces. This allows for leaving existing data structures unchanged, which is a specific benefit for achieving ex-post integration. There is, however, a severe downside to this approach: integration of this kind does not cater for avoiding data redundancy – a problem that is often ignored by proponents of service-oriented architectures. Hence, functional integration can contribute to "covering the mess" instead of removing it.

Dynamic Integration
Functional integration allows two components to cooperate in the sense that they mutually use/provide services. However, it is not sufficient for the integrated components to jointly execute a certain process. This is subject of dynamic integration. It requires a common collaboration or process schema, which includes common event types. Furthermore, it implies functional integration. A process schema defines a process e.g. through event-action-rules: If a certain event is generated by one of the integrated components, it will trigger the appropriate component to execute a certain function. Such a decentralized execution plan requires all the integrated components to have access to the common process schema – and to identify the components that are in charge of executing a particular function. Taking into account that a component may participate in more than one process schema, this approach will often imply a remarkable effort for the specification and implementation of the components. It corresponds to the coordination of autonomous software artifacts ("agents") according to a common plan. To avoid this effort, integration of the components can be relaxed by deploying a central coordinator. Relevant events generated within a

component would then have to correspond to common event types defined in a central schema. In addition to that, the components would have to be functionally integrated.

Organizational Integration
Orthogonal to the dimensions discussed above, there is a further dimension of integration that is of particular relevance for Information Systems: The efficient use of information systems requires them to be integrated with the organizational action systems they are supposed to support. This dimension has gained remarkable attention both in business practice and academia, often referred to as "IT-Business-Alignment". To define a conception of organizational integration, we build on the previous definitions: An information system and the actions system it is supposed to support are integrated, if they are able to communicate. This requires common or better: corresponding concepts, since the specification of concepts in information systems is different from natural language terms. If, for instance, a database schema includes the concept of a "PreliminaryInvoice", integration suggests that this is an established term of the action system as well. Otherwise users cannot use this component of the system properly. Organizational integration can be differentiated into static, functional and dynamic. Static refers to communication that is based on common static concepts, such as the above example. Functional integration refers to the concepts that specify functions which correspond to functions or tasks within the action system. Finally, dynamic integration is aimed at coordinating functions provided by the information system and human actions according to a process schema. It requires events generated within an information system that correspond to concepts, users are familiar with. On the other hand, events within the action system that are relevant for the information system should also correspond to a common concepts. There are different approaches to accomplish organizational integration. Firstly, the concepts of an information system can be adapted to the concepts used in the action system. In this case, requirements analysis is aimed at identifying the terminology of the intended application domain in order to map it to corresponding concepts for designing the information system. Note that 'terminology' refers mainly to concepts, not to their designation. Secondly, the prospective users can adapt to the information system's concepts. This does not only require learning the concepts but also mapping them to existing terminology and – eventually – adapt the existing terminology and corresponding patterns of action. Thirdly, there is the option of mutual adaptation, which will often be a convincing choice: With respect to user acceptance, it is not a good idea to ignore existing terminology and established patterns of action. At the same time, both existing terminology and established patterns of action are often insufficient with respect to actual and future business requirements. Hence, organizational integration does not only require analyzing existing domains and their prevalent language. Instead, it suggests (re-) designing new action systems and reconstructing existing terminologies that are in line with corresponding information systems – thus contributing to the evolution of new language games.

3 A Refined Conception of Integration

While our first conception of integration includes the distinction of various dimensions, it does not allow for differentiating different qualities or intensities of integration. Therefore, I will suggest a refinement of the concept that accounts for different degrees and levels of integration.

3.1 Degrees of Integration

In general, integration requires common concepts that provide a foundation for communication. However, with respect to facilitating communication not any common concept is of equal value. For communication to be efficient and consistent, common concepts should allow for specifying the exchanged data in a way that minimizes information loss. In other words: The amount of semantics, a common concept includes, should not be lower than that of any of the corresponding concepts in the components to be integrated. Note that this use of the term "semantics" corresponds to information content: The more possible interpretations are excluded by the specification of a concept, the higher its semantics. This allows for refining the concept of integration to express different degrees of integration: The higher the amount of semantics incorporated in common concepts used by a set of integrated components, the higher the *degree of integration* of these components. If, for instance, two components exchange bytes, the extent of possible interpretations is apparently larger than for the exchange of data, which are defined as floating point numbers. Referring to a concept such as "Customer" or "Invoice" would promote an even higher degree of (static) integration. This concept can be applied to all dimensions of integration. The semantics of a function is more difficult to describe than that of data. On the one hand, it depends on the semantics of its interface: A function that is passed a number is less specific – hence allows for more interpretations – than a function that is passed a more complex structure. Also, additional constraints, e.g. expressed through pre- or postconditions, increase a function's semantics. The degree of dynamic integration depends on the semantics of event action rules that define coordination. The semantics of event-action-rules depends on the semantics of the involved event types and the semantics of the functions that represent the actions. An event type such as "file was modified" allows for more interpretations than "product database was updated" or even "price of product x was changed". Fig. 1 illustrates various degrees of integration in all three dimensions.

The refined concept of integration can also be applied to organizational integration. If, for instance, customer information is stored in text files using a text processor, the level of organizational integration is low: the common semantics of "Customer" is reduced to a string. If an accounting system features a concept of "Account" that directly corresponds to the use of the term in the intended application domain, the degree of integration would be higher – in other words: the chance for communication failures would be less. If an ERP system includes an elaborate concept of "OrderProcessing" that is not known in its application

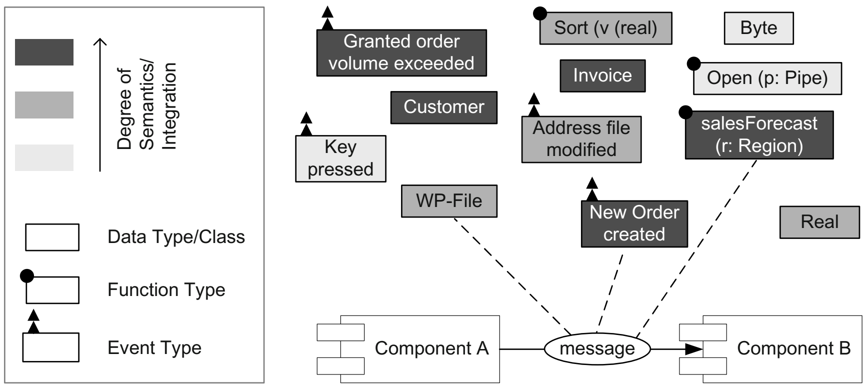


Fig. 1. Degrees of Integration

domain, the level of integration would – at first – be low, too. Only, if system introduction and training resulted in successful adoption of the concepts suggested by the ERP system, integration could be accomplished. The higher the degree of organizational integration, the better is the chance to avoid friction between human action systems and the supporting information system, in other words: the better is the alignment of business and IT. Apparently, there is a difference between the dimensions of system integration and organizational integration: Due to the interpretative flexibility and the learning capabilities of human users, a perfect fit is not necessary. Due to the interpersonal diversity of meaning, it would not be detectable either. While the refined concept of integration allows for gradually differentiating degrees of integration with respect to certain concepts, it does not allow for specifying the degree of an entire information system. If two information systems have only one highly specialized concept in common, would they feature a higher degree of integration than two other system that share a number of concepts with less semantics? Against this background, it might be tempting to define a formal concept of integration that would allow for calculating corresponding metrics, e.g. the average or total degree of integration of an entire information system. However, while such an approach might be regarded as satisfactory by some, it is likely to produce distortion. The notion of semantics we use is not a pure formal one. For instance: With respect to formal semantics, there is no difference between typing an attribute "name" as String and typing an attribute "revenue" as String. However, accounting for the meaning, we contribute to the term "revenue", it would clearly indicate a poor degree of integration, if two components used String as a common concept.

3.2 Levels of Integration

So far, our focus was on concepts that are required to facilitate integration. Integration, however, can also be related to the instance level – and to meta levels as well.

Instance Level Integration
Communication requires common concepts. The concepts can be refered to using a common namespace, e.g. the types of an IDL. This allows for assigning semantics to instance level data that is exchanged between the components of an information system. However, for communication on the instance level to be efficient, refering to corresponding concepts alone is not sufficient. A unified identification, hence a common namespace, would not only help reducing communication load – it would be sufficient to exchange identifiers only – it would also allow for differentiating between instances. For this reason, it makes sense to expand the previous notion of integration by accounting explicitly for the instance level: Two components are integrated on the instance level, if they share a common namespace to identify instances. Integrating instances implies conceptual integration, i. e. common concepts to specify the semantics of the instances. Instance level integration through common namespaces is an important feature of so called middleware or DBMS. These technologies allow for identifying data objects accross the boundaries of specific namespaces maintained by separate components. Apparently, this corresponds to static integration. Integrating function instances is a different issue. At first, it is not as obvious what a function instance is: Is it a particular implementation, a particular implementation within a particular context or a particular execution? There may be many instantiations of a particular implementation. Therefore, regarding a particular implementation as an instance, makes sense only for functions that provide stateless services. A service offered by an object is a prototypical example of a particular implementation within a particular context. The identification of such a function requires the identification of the corresponding object. Hence, instance level integration of functions is an issue only in those cases, where the static context is relevant; in other words: instance level integration of functions implies static integration of those elements that constitute the relevant context. Core concepts required for dynamic integration are events, functions and corresponding execution rules. Particular events are generated e.g. through the instantiation of data or objects, their deletion or state changes. If, in a cooperation scenario, e.g. a workflow, the coordination of two components demands for taking into account particular events, there is need for dynamic integration on the instance level. If, for instance, in a workflow management scenario, an incoming order is represented through an object, changes that apply to this particular object – e.g. representing the approval of the order – are relevant particular events. In order to accomplish this kind of integration – either in a decentralized cooperation scenario or through some kind of mediator – it is required to define a namespace for event instances that allow for idendifying events throughout the entire workflow (and maybe within an even wider context). The benefits of instance integration are obvious: It is the precondition for avoiding redundancy and for different components working jointly on common concepts. Achieving instance level integration can require a remarkable effort. Not only that it requires to maintain a common namespace, which covers distributed components. It also requires appropriate instruments for instance lifecycle management. In the case

of static integration, this includes assigning a common name (identifier), when an object is instantiated and ensuring referential integrity when it is removed. This is similar for function instances: new instances need to be registered and assigned an identifier. In case, a function instance is removed, referential integrity is an issue, too. In the case of dynamic integration, instance life cycle management is even more demanding. There is, e.g. need for a protocol that defines, when an event is regarded as consumed, hence, needs to be removed from the common event space. To summarize, instance level integration faces two main problems: consistently identifying and managing instances accross the set of components that are to be integrated. The safest approach to consistent identification and naming is the use of unambiguous identifiers for the real world entities to be represented in a system. If every relevant object carried its identification – like the identification number in a passport or a number used to identify a specific instance of a product type – there would be no longer the need to care for consistently assigning identifiers to instances in information systems. If these objects would also carry the definition of their type, assigning real world objects to representations within information systems would be even more efficient and safe. In the long run, this may even result in overcoming the distinction between a real world object and its representation within an information system: Every object could carry a representation of itself, together with a unique identifier. Given the standards, bandwiths and affordability required to realize such a vision, a tremendous impact on the economics of logistics processes can be expected.

Meta-level Integration
If – in human communication – we use terms other participants do not know, we can still explain these terms using other, common terms. The effort such an explanation takes depends on the available common language. If, for instance, the concept "Interface" as it it used in the UML needs to be described, it will be much easier, if one can refer to the technical terminology of software engineering than to colloquial language only. A similar approach to communication is possible in information systems, too. For example, the concept of an "Interface" is specified through a meta model. Components that are able to interpret the concepts of the corresponding meta model, could infer the semantics of the concept, even if it not part of a schema they have access to. However, the UML meta model is hardly sufficient to specify the semantics of particular classes, e.g. of a class "Product". The meta level concepts would only specify that it is a class with certain features. A further example would be an XML document representing an invoice that is transmitted together with its DTD. While the receiving component – if it includes an adequate interpretation of XML documents – would be able to identify that the document is a valid XML document, it could not infer that the DTD describes the concept "Product". If one used a more specific language on the meta level, there would be the chance to describe concepts more specifically, leaving less room for interpretations. For instance: A formal "Business Language" could include a meta concept such as "Product", which would capture the essence of all product types within a certain range.

As a consequence, every component that has a reference to this meta concept could infer that all instances of this meta concept represent product types. For a comprehensive example and its application to prevalent architectures see ([5]). Note that in addition to common meta level concepts there would be need for unified designators – which might be multilingual.

4 Challenges

The obvious advantages of integration are constrasted by reservation. On the one hand, warning voices can be heard from those who are in charge of managing IT. They doubt the economic benefits of integration. On the other hand, there are software engineers who advocate a more relaxed concept of integration in order to better cope with the demand for flexibility.

4.1 Complexity and Risk

Designing and implementing information systems that feature a high degree of integration is certainly more demanding than building isolated systems with a more modest degree of integration: It is not sufficient to strive for abstractions that fit a particular application. Instead, abstractions should result in concepts that fit a whole range of applications. In order not to jeopardize the integrity of the system during its lifecycle, abstractions should be resistant against changing requirements. With the number of components to be integrated the likelihood that abstractions fail in this respect is growing. Ex post integration is even more challenging, since it requires touching legacy systems. Demanding a high degree of integration in this case requires modifying existing, often insufficiently documented code. From the perspective of those who are responsible for integration projects, this is a rather threatening scenario, since numerous sources of hidden risks can be expected. While there is not doubt that building integrated systems is associated with remarkable complexity and risk, that does not necessarily discredit the striving for a high degree of integration. One should take into account that reducing complexity usually requires increasing it first. Building, e.g., a DBMS is certainly a complex endeavour. If it was successful, however, it contributes to a remarkable decrease of complexity – and effort – for those who build applications. Therefore, from an economic point of view, the additional effort for realizing a higher degree of integration should be related to the corresponding benefits. This will certainly require a thorough evaluation, e.g. by analyzing the effect of integration on current and future business processes. There is, however, no doubt that especially cross-organizational integration is facing a paradox situation: On the one hand, it offers exciting economic perspectives. On the other hand, the incentives for pioneering actors to invest into cross-organizational integration are often not sufficient.

4.2 Integration Versus Loose Coupling

In recent years, loosely coupled systems seem to be the name of the game in software development. However, coupling is not a well defined term. On the one hand, it refers somehow to the principle of information hiding: Two components are loosely coupled, if their mutual dependance is restricted to a well-defined interface. Hence, the internal representations of the components are not affected. On the other hand, coupling refers to the specifity – or semantics – of these interfaces. If a component's interface is specified on a low level of semantics, it is more likely to find further components that can be coupled to it. Hence, loose coupling suggests to use interfaces only to integrate systems and to avoid a high degree of integration. At first sight, such a recommendation may seem useful. However, it has clear shortcomings. Firstly, integrating two components through concepts with little semantics is a clear threat to system integrity. Secondly, reducing integration to interfaces, i. e. abstracting from the internal representations of the involved components, leaves an extremely relevant issue open: data redundancy. Note that this is different with information hiding in object-oriented software development: Each object is thought to represent a unique real world entity. Therefore, the data represented in objects should not be redundant.

Apparently, there is a tradeoff between the benefits of integration and the threats of dependance. Deciding for the right balance in a particular case recommends taking into account two aspects. Firstly, one should analyse whether the essence of the concepts used to accomplish a higher degree of integration is the same for all participating components. If this is the case, the changing requirements related to some components would not affect the common concepts. In other words: One should put emphasis on the quality of the abstractions used to accomplish integration. Secondly, one should not neglect the concepts used in other systems that one might want to integrate in future times. While both aspects create a remarkable challenge, it is not a good idea to sacrifice the benefits of integration for the supposed sake of loose coupling without further consideration.

5 Concluding Remarks

When it comes to designing and evaluating information systems, integration is a pivotal term. This recommends a thorough conceptualisation and assessment. The ideas presented in this paper are preliminary only. They leave many questions open. Also, a concept that is based on "semantics" adopts some of the unsolved mystery that is still encompassing this term. Currently, integration is an ambivalent concept. Its clear benefits are contrasted by threats. This is similar to reusability, which marks the other side of the same coin: The more semantics a reusable artifact incorporates, the larger its benefit in a certain reuse case, the higher, however, the likelihood that is does not fit a particular context. Unfortunately, it is not possible to overcome this conflict through invidual action. There is, however, a promising approach to developing a solution to this unsatisfactory situation: The creation of *reference models* (cf. [7]; [6]) that serve as blueprints for an entire range of information systems. Unfortunately, the development and establishment of reference models exceeds the resources available to single academic institutions by far. Also, it is required to involve prospective users and vendors in time. Traditional approaches to organizing research on information systems are not appropriate for that. Instead, there is need for large communities that bundle their forces not only to develop and evaluate reference models, but also to promote them. A recent initiative, started by the SIG MobIS within the German Informatics Society (GI), is aimed at applying the idea of Open Source Software to the development of open reference models ([8]), (www.openmodels.org). While it is a demanding process to establish such initiatives, it seems the only chance to exploit the potentials offered by integration and reusability.

References

1. Anderson, N.H.: Foundation of information integration theory. Academic Press, New York (1981)
2. Bouras, A., Gouvas, P., Kourtesis, D., Mentzas, G.: Semantic Integration Of Business Applications Across Collaborative Value Networks. Springer, Boston (2007)
3. Wiederhold, G. (ed.): Intelligent Integration of Information. Kluwer Academic Publishers, Boston (1996)
4. Guarino, N.: Formal Ontology and Information Systems. In: Guarino, N. (ed.) Formal Ontology in Information Systems. Proceedings of FOIS 1998, Trento, Italy, June 6-8, 1998, pp. 3–15. IOS Press, Amsterdam (1998)
5. Frank, U.: Modeling Products for Versatile E-Commerce Platforms – Essential Requirements and Generic Design Alternatives. In: Arisawa, H., Kambayashi, Y., Kumar, V., Mayr, H.C., Hunt, I. (eds.) ER Workshops 2001. LNCS, vol. 2465, pp. 444–456. Springer, Heidelberg (2002)
6. Frank, U.: Evaluation of Reference Models. In: Fettke, P., Loos, P. (eds.) Reference Modeling for Business Systems Analysis, pp. 118–140. Idea Group, Hershey (2006)
7. Fettke, P., Loos, P.: Classification of reference models - a methodology and its application. Information Systems and e-Business Management 1(1), 35–53 (2003)
8. Frank, U., Strecker, S.: Open Reference Models – Community-driven Collaboration to Promote Development and Dissemination of Reference Models. Enterprise Modelling and Information Systems Architectures 2(2), 32–41 (2007)

Aspect Based Conceptual Modelling of Web Applications

Athula Ginige

AeIMS Research Group,
University of Western Sydney, Australia
a.ginige@uws.edu.au

Abstract. Though there are many Web application modelling approches these are aimed at assiting the implmentaion process. There are many benefits in creating a conceptual model of the Web application to verify whether the application domian requrements have been propery understood. We have developed a way of modelling a web application at a conceptual level based on set of aspects; Overall Presentation, Primary Navigation, Hypertext Model, Process Model and Personalisation Model. We have found that this set of conceptually non-overlapping aspects greatly simplifies the modelling task. We also observed that application domain users found it easy to relate to these aspects and provide feedback on the accuracy of the model.

Keywords: Conceptual modelling, aspect based modelling, process model, hypertext model.

1 Introduction

The process of developing a Web application, like many other development activities, can be viewed as the conversion of a concept into a physical system. Until we create the physical system, the web application that we create needs to be represented as models in different domains and at different levels of abstraction.

The original concept often originates in a domain other than computing, such as science, business, education, health, government etc. These concepts originate as a solution to a problem related to providing convenient access to information, such as places of interest in the context of tourism; assisting in conducting a task such as bill payment; automating a process such as applying for leave, which consists of a sequence of tasks; or a calculation such as creating a production schedule.

People are thinking of increasingly more complex web-based applications to assist in conducting activities in various domains. Obtaining necessary requirements for such applications has been a problem, not only in relation to developing web applications but also when developing software applications. Thus when people conceptualise increasingly more complex applications, capturing these requirements becomes more difficult.

It has been observed that the introduction of an information system to an organisation results in changes to their business processes, which in turn causes changes to the implemented information system. Users also discover better and more efficient ways to use the system, and request these changes. These changes lead to a cyclical evolution of the business processes, users and the information system. This is called co-evolution [1, 2], and is a very common occurrence in Web-based applications, making it necessary to design the system such that requirements can evolve while the system is in use [3].

Modelling is an inborn ability possessed by humans to reduce the complex universe to a world that is relatively easy to manage. In modelling, important phenomena are mapped to notions and represented by signs. For example, the notion of a website can be represented as a hierarchical diagram or a graph. Using an agreed method of representing concepts associated with web applications will assist the thinking process of developers, and also facilitate the communication among people involved in developing the web application.

There are other benefits of modelling. We can use the fact that computers are good at manipulating signs to create tools to assist in the Web development process. For example, if we can represent the structure of a Web site as a graph, we can develop a software module to read the graph and generate the physical menu to be displayed on the website.

This paper presents a set of aspects that we discovered by modelling a large number of Web applications, that can be used to create a comprehensive conceptual model of a Web application that most application domain users can relate to. These aspects have minimal coupling, and can be easily mapped onto an extended MVC architecture. Thus each aspect can be changed with negligible impact on other implemented aspects. Such an approach can support co-evolution.

2 Web Application Modelling Approaches

The concept for a web application originates in an application domain. Firstly, we need to create a conceptual model of the web application using concepts belonging to the application domain. Such a model can be used to verify whether the proposed application meets the requirements of the application as originally conceptualised.

Once we establish the conceptual model, we then need to develop a logical architecture model using concepts in an implementation domain. In the computer domain (the web domain is a subset of the computer domain, based on thin client-server architecture) these concepts could be an access control module, view generator, workflow engine, object or data model etc. which can be implemented using technology available in the computer domain.

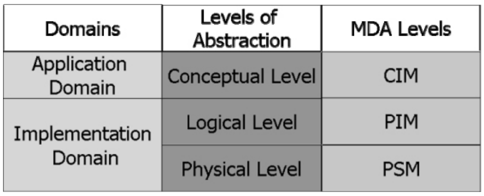
Once a suitable implementation platform and technology is selected, we need to develop a physical architecture model using concepts such as algorithms, flow charts and database schemas to realise the concepts used for the logical architecture model. From the physical architecture model the specific artefacts can be created as programme files and databases.

The Object Management Group (OMG) has proposed an approach known as Model Driven Architecture [4] or Model Driven Development (MDD), where the above-described process is viewed as a series of model transformations. In MDD, the application is first modelled based on business requirements, independent of the implementation technology at the computation-independent level (CIM). These models are then transformed into platform-independent analysis and design models (PIMs). Implementation platform-specific models (PSMs) are then created and the application code is generated for the targeted implementation platform using model compilers [5].

The main aim of the MDA and MDD approaches is to make it easy to implement the application in different technologies by developing automated techniques for model transformation and compilation to generate the application for various implementation platforms using these same high-level models [5]. Research emphasis has focused on developing various model transformation and compilation techniques, rather than how to create conceptual level models that can be used to verify whether application domain requirements have been properly understood, or what changes need to be done to the implemented application due to co-evolution.

The relationship among different levels of abstraction in the generic modelling approach and the MDA approach is shown in Table 1.

Table 1. Relationship among Domains and Abstraction Levels in different development approaches



Domains	Levels of Abstraction	MDA Levels
Application Domain	Conceptual Level	CIM
Implementation Domain	Logical Level	PIM
	Physical Level	PSM

Many researchers have proposed other ways to model web applications or some aspects of web applications [6-11]. A comparison of these methods can be found in [12] which has grouped these Web application modelling methods into four categories, detailed below:

- Data driven, based on entity relationship models such as Relationship Management Methodology –RMM, Hera, Web Modelling Language (WebML) [9].
- Hypertext-based, such as Hypertext Design Models (HDM), HDM-lite, Web site design method (WSDM)
- Object-oriented methods such as Object Oriented Hypermedia design methods (OOHDM), UML based Web Engineering (UWE) [10], Objected Oriented web solutions (OOWS), Object oriented Hypermedia (OO-H)
- Software-oriented methods such as Web Application Extension (WAE)

All these approaches aim to accurately capture requirements from the application domain and design modules or sub-systems at the logical level of the implementation domain using generic computer and web concepts. Once an implementation platform and technology is decided upon, further details are added to theses models so that the required artifacts can be created.

These approaches have provided valuable insights into how a web application can be modelled using implementation domain concepts such that it is easy to implement the model.

As can be seen from above broad categorisations, different web modelling approaches have mainly looked at the data or object model and hypertext model aspects of a web application. For example, people have attempted to model workflow as hypertext models [10] which overly complicates the model. It is true that non-linear navigation within a document (navigation links) as well as accessing different actions in a workflow implementation (process links) are both implemented using hyperlink tags (ie. <a href= >). At a conceptual level however, these are two different concepts. In the first instance the same person clicks on a link to access a different part of the information or to retrieve some additional information; in the workflow instance different people use hyperlinks to access a form to perform a task. Trying to model these two different concepts in the same way at a conceptual level overly complicates the conceptual model.

As can be seen from the above discussion, existing modelling methods are mainly focused on creating a logical model or a logical architecture based on some conceptual model of an application or set of requirements. These are not aimed at creating a conceptual model that application domain users can easily relate to in terms of the application domain concepts that have been used to create the model and thereby provide feedback to eliminate any errors in the model. Thus it is useful to identify a set of aspects that will assist us to model a Web application at a conceptual level based on application domain concepts.

For many years researchers have studied issues related to the conceptual modelling of data, as the majority of early, large and complex applications such as payroll systems, accounting packages etc. were predominantly data driven.

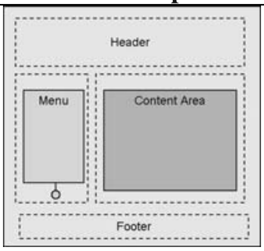
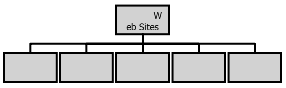
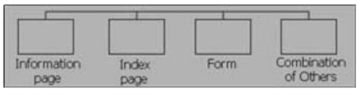
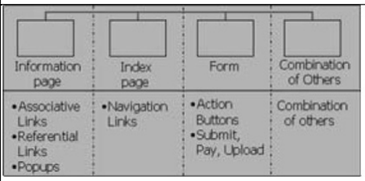
Unlike data or object models, web applications incorporate many other aspects such as hypertext, navigation, access control, personalisation, and process automation. Thus a conceptual model of a web application needs to incorporate all these aspects in addition to the data model. It should also be possible to map selected aspects to represent concepts associated with a Web application on to a logical architecture to create the physical application.

By integrating different aspects to form the overall application, and by keeping the aspects sufficiently independent or by reducing coupling wherever possible, each aspect can be changed in line with evolving organisational needs without affecting other aspects of the implemented application.

3 Aspect Based Modelling

Aspect-based modelling refers to selecting a set of aspects that are rich enough and with minimum coupling to fully capture the requirements using application domain concepts, which can then be mapped onto implementation domain concepts such as an authentication subsystem, view generator, data model, hypertext models etc.

By analysing many different types of websites we have identified following aspects as the starting point as these are visible on a website to the application domain users.

Aspect	Description	Examples
Overall Presentation	Each web site normally has a distinct layout and colour scheme. The overall layout will typically consist of a header, main menu, an area to display specific contents and a footer.	
Primary Navigation	This provides a primary means of accessing various sections of the Website, typically using a menu displayed vertically on the left-hand side of the website or horizontally at the top. An unstructured or structured search mechanism is another way to provide this primary navigation.	
Page Composition	On a website one can identify four types of pages: ● Information pages consisting of hypertext and hypermedia ● Forms for inputting information to the Web application to perform a task ● Index pages to accessing other types of pages ● Pages which are a combination of information, forms and indexes	 Information page Index page Form Combination of Others
Secondary Navigation	Each type of page can have different types of links to access related information or to perform a task such as submitting a form or uploading a file.	 Information page Index page Form Combination of Others •Associative Links •Referential Links •Popups •Navigation Links •Action Buttons •Submit, Pay, Upload Combination of others
Access Control	Access control specifies the relationships between users and pages, such as what information users can view or what actions they can perform.	All users can view home page. Only administrator can create new users.

Personalisation	For a given URL the information presented to the user and how this information is presented can be customised so that each user receives a personalised page based on the context.	Presentation = f1(user, context)
		
		Information = f2(user, context)

Though it is not difficult for users to specify the overall presentation and primary navigation structure for a web application, it is a tedious task to specify the composition of each page as well as secondary navigation links or buttons required for each of the pages, especially for a web application with a large number of pages. Thus we need a higher-level model from which we can drive the composition and secondary navigation links and buttons for each page automatically.

To arrive at a suitable high-level model we first need to identify the different types of web applications, as each type of application needs to be modelled differently.

4 Models for Different Types of Web Applications

Today we tend to use web applications to provide information to the user, or support the information flow associated with performing a business process. Web applications that provide information to the user are called information-centric web applications and web applications that support business processes are called process-centric web applications. Often a complex Web application will have both these subtypes.

4.1 Information-Centric Web Applications

The focus of this type of application is to present large amounts of information to the user in an effective way. Websites that provide news, travel information or product information are examples of Information-centric web applications. An important feature of these applications is that at no time can users modify the information stored in the data repositories.

The information can come from one or more data repositories in one or many computers stored as HTML or XML files or in databases. The emphasis is on how to compose the web pages and assist users in navigating from page to page in an effective manner. At the basic level these types of web applications can be modelled as page views composed of information from different information sources as shown in Figure 1.

Thus by specifying a set of information sources and how information from these sources is combined to create different pages we can model basic information-centric web applications.

Often there are secondary navigation links on these pages. If there are only a few information pages then the secondary navigation links can be explicitly specified. If there are a large number of pages however, we need to specify some rules to automatically derive these links. For example, if the page is a product catalogue then we can have a rule specifying a hyperlink from the product name to an image of the product. Another example of specifying secondary navigation link is to have a glossary and a rule which states that if the user on any page clicks on a word described in the glossary, this description should appear in a pop-up window. During the implementation phase developers need to create a mechanism to insert the appropriate hyperlinks when rendering a page during runtime. In such an implementation, administrators of the system can keep adding new terms to the glossary without worrying about how to create the necessary secondary navigation links.

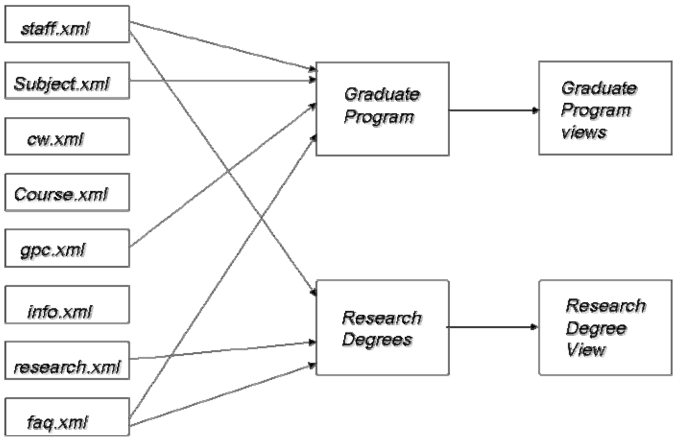


Fig. 1. Basic Conceptual model of an Information Centric Web Application

Thus we can model an information-centric web application at a conceptual level by specifying the various data sources, how information from these sources is combined to compose different pages and a set of rules to derive the secondary navigation links. This is known as a hypertext model. Table 2 shows essential and optional aspects required to model an Information-Centric Web application.

Table 2. Aspects of Hypertext Model

Aspect	Modelling Method
Data or Information sources	Object diagrams, database schema or Smart Business Object Modelling Language (SBOML)[11].
Page Composition model	Set of rules specifying information sources and the object instances to be shown on each page or page composition diagrams as used in WebML[9]
Secondary Navigation Model	Set of rules specifying how secondary navigation links should be created or using WebML[9] notations.
Optional Access Control Model	Set of rules specifying web pages that each user can access. If there are many users it is better to assign users to groups or roles and specify what web pages each group or role can access.

4.2 Process-Centric Web Applications

The aim of process-centric web applications is to assist in managing the flow of information required to support a business process. In the field of software engineering it is well understood how to model such applications as a set of use cases [13] or tasks. A conceptual model for such an application is a business object being modified by different users who perform a task or an action through a form interface as shown in Figure 2. Thus a Process-Centric web application model should contain an object or data model, an action model to specify a sequence of interfaces required to perform an action (or a use case), an interface model to specify details of each interface, and an access control model to specify the actor who can perform the action.

An action model specifies the interaction between a user and the system required to complete an action. This could be a user selecting a set of books on a screen that they want to purchase. The system then provides a second screen showing the total amount of the order and payment options. When the user selects a payment option the system provides a third screen to enter the relevant payment details, such as their credit card number. The system then provides the penultimate screen for the user to verify that all the details are correct before processing the order and at the end provides a confirmation screen.

The interface model specifies what attributes of the object are to be shown on each of the screens and whether each of the displayed attributes are read-only, or whether the user can edit or enter new values.

When a user logs in, we can determine what actions the user can perform from the access control model. From this information we can generate a personalised primary navigation model for the logged-in user to access the functions to perform these actions.

A variation on the above model is where it is necessary to specify the order in which the business object needs to be populated, such as an employee in an organisation applying for leave using a web-based application. This process can be modelled as an employee through a form interface partially populating an instance of a leave object. A link is then provided to employee's manager to view the information provided by the employee and decide whether to approve or reject the leave request. This decision (and any explanatory comments) can be recorded in the leave object. A link is then provided to an officer in the Human Resource division to process the leave application. This person now can look at the information stored in the leave object and process the information accordingly. Thus to support a process of this nature we need a sequence of actions (rather than a set of actions) also known as a workflow. Therefore, in addition to the above-mentioned models we also need a workflow model.

From time to time various managers and senior executives may like to receive summary reports detailing who is on leave at a specific time or how many employees have taken leave in the last six months etc. Thus these applications, in addition to form interfaces to perform actions, should also provide various summary reports to meet the organisation's needs. These reports can be modelled as forms with read-only access to data. There is no need for a separate model for reports, as the interface model used for forms can also be used for reports.

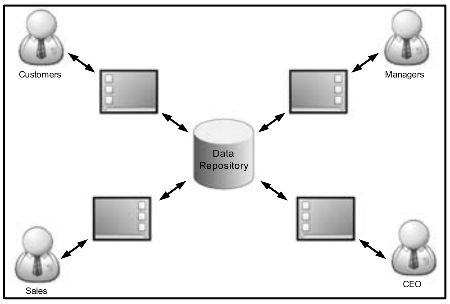


Fig. 2. Model of a Process-Centric Web Application

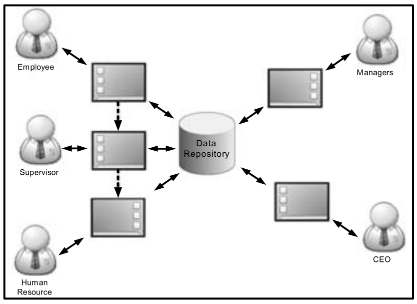


Fig. 3. Process-Centric Web Application with a workflow

The same pattern of a form being routed based on business rules of the leave processing application can be applied for many other business applications such as purchase order processing, travel applications etc.

Thus the conceptual model for such applications is a business object being populated by a sequence of actions via form interfaces as shown in Figure 3. Using this conceptual model we can generate a range of process-centric web applications by specifying the object attributes, actions and related interfaces to perform the actions, people who can perform the specific actions, and the sequence in which actions should take place.

Table 3 shows the essential and optional aspects required to model a Process-Centric Web application.

Table 3. Aspects of Process Model

Aspect	Modelling Method
Data or Information Object	Object diagrams, database schema or Smart Business Object Modelling Language (SBOML)[11].
Action Model	Sequence of interactions between a user and the system required to perform an Action. This can be modelled using UML interaction diagram.
Interface Model	This can be modelled as a table. Each row represents an attribute of the information object. Each column represents an interface. Each cell specifies on the corresponding interface whether the attribute is displayed or not and if displayed what CRUD (Create, Read, Update, Delete) operations can be performed on the attribute.
Access Control Model	Set of rules specifying actions each user can perform. If there are many users it is better to assign users to groups or roles and specify what actions each group or role can perform.
Optional Workflow Model	This can be specified as a state table. From this information an index page can be created to provide access to pending workflow tasks.

We can make use of the above-described hypertext model and process model to derive page composition, secondary navigation and access control aspects of the conceptual model as shown in Figure 4.

For a specific URL a user can get a web page that is personalised in terms of its presentation and information content. For the same URL a different user will get different information. This can be specified as a set of rules. Some rules for Personalisation can be derived from access control specification. For example, based on information pages and actions the user has access to, we can create a personalised primary navigation menu for each user when they login.

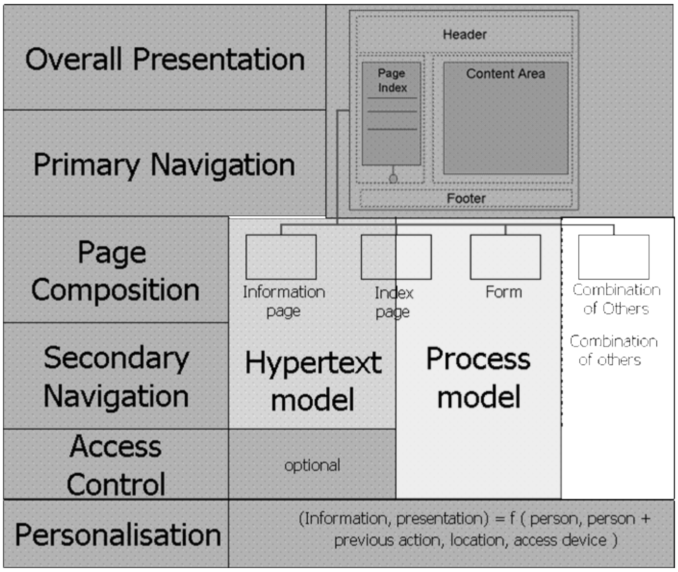


Fig. 4. Aspects to be modelled to create a conceptual model for a Web Application

5 Mapping Aspects to MVC Architecture

The Model View Controller architecture pattern was first described in 1979 by Trygve Reenskaug [14]. The essential purpose of MVC is to bridge the gap between the human user's mental model and the digital model that exists in the computer.

We extended the Model to include the access control model, process model and the hypertext model. The Controller module implements the access control rules specified in the access control model. The rules in the access control model are also used to generate a personalised primary navigation menu for logged in users. The generation of views was done using a set of templates. Figure 5 shows the extended MVC architecture that we used to map the aspects in the conceptual model of the web application. The numbers in this figure show the sequence in which a user request is processed.

We have successfully implemented this architecture and a set of tools to create some of the models using the CBEADS framework [2, 3, 11, 15, 16].

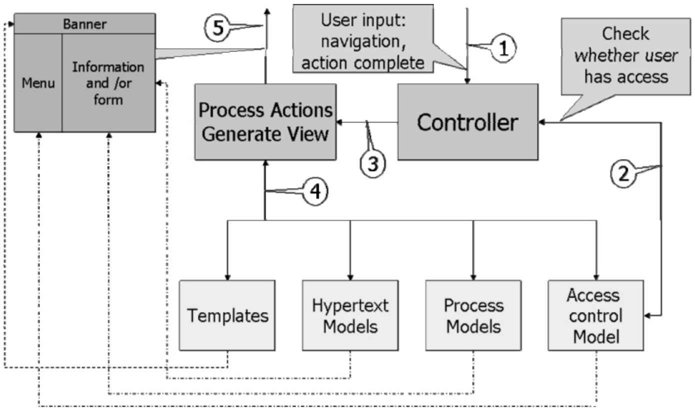


Fig. 5. Extended MVC architecture used map aspects in the Conceptual Model

6 Conclusions

We have presented an approach to create a conceptual model of a Web application using a set of aspects that are well understood by people at the application domain who will be using the application. The aspects that we identified are Overall Presentation, Primary Navigation, Hypertext Models, Process Models and Personalisation Models.

We have used this aspect-based conceptual modelling approach to develop some large, complex web applications [17].We have found that this set of conceptually non-overlapping aspects greatly simplify the modelling task. We also observed that end users found it easy to relate to these aspects.

We also showed that these models can be mapped to an extended MVC architecture pattern to create the logical architecture of the system. As each aspect is mapped to a specific logical architectural unit, any changes to the models can easily be incorporated into the implemented system thus supporting co-evolution.

Thus the development of an aspect-based conceptual model of a web application that users can easily relate to and provide feedback on can greatly assist the development of web applications. At present we are investigating ways of automatically generating the required artifacts to implement a web application from an aspect-based conceptual model.

References

1. Costabile, M.F., Fogli, D., Marcante, A.: Supporting Interaction and Co-evolution of Users and Systems. In: Advanced Visual Interfaces -AVI (2006)
2. Ginige, A.: Designing Web Applications to Support Co-Evolution. In: 2nd International Conference on Web Engineering and Applications 2007, Narosa Publishing House, Bhubaneshwar, India (2007)

3. Ginige, A., De Silva, B.: CBEADS©: A framework to support Meta-Design Paradigm. In: Stephanidis, C. (ed.) HCI 2007. LNCS, vol. 4554, pp. 107–116. Springer, Heidelberg (2007)
4. Soley, R.: Model Driven Architecture. Object Management Group (2000)
5. Koch, N., Zhang, G., Escalona, M.J.: Model Transformation from Requirements to Web System Design. In: International Conference on Web Engineering (ICWE 2006), ACM Press, Palo Alto, California (2006)
6. Fratenali, P., Paolini, P.: A conceptual model and a tool environment for developing more scalable and dynamic Web applications. In: Schek, H.-J., Saltor, F., Ramos, I., Alonso, G. (eds.) EDBT 1998. LNCS, vol. 1377, Springer, Heidelberg (1998)
7. Schewe, K.-D., Thalheim, B., Zlatkin, S.: Modelling and Stories in Web Information System. In: Information Systems Technology and its Applications (ISTA), Salt Lake Ciy, Utah, USA (2004)
8. Schwabe, D., Rossi, G., Barbosa, S.D.J.: Systematic hypermedia application design with OOHDM. In: Seventh ACM conference on Hypertext, ACM Press, Bethesda, Maryland, United States (1996)
9. Ceri, S., Fratenali, P., Bongio, A.: Web Modelling Language (WebML): A Modelling language for designing Web sites. In: WWW9 Conference, Amsterdam (2000)
10. Koch, N., Kraus, A.: Power of UML-based Web Engineering. In: Second International Workshop on Web-oriented Software Technology (IWWOST 2002), Málaga, Spain (2002)
11. Liang, X., Ginige, A.: Smart Business Objects: A new Approach to Model Business Objects for Web Applications. In: 1st International Conference on Software and Data Technologies, Setubal, Portugal (2006)
12. Kappel, G., et al.: Web Engineering - Systematic Development of Web Applications. Wiley, Chichester (2006)
13. Rumbaugh, J., Jacobson, I., Booch, G.: The Unified Modelling Language Reference Manual. Addison-Wesley, Reading (1999)
14. Reenskaug, T.M.H.: Model, View, Controller Pattern (1979), `http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html`
15. Ginige, A., et al.: Smart Tools to support Meta-Design Paradigm for Developing Web based Business Applications. In: Baresi, L., Fraternali, P., Houben, G.-J. (eds.) ICWE 2007. LNCS, vol. 4607, Springer, Heidelberg (2007)
16. Liang, X., Ginige, A.: Enabling an End-User driven Approach for Managing Evolving User Interfaces in Web Business Applications. In: 2nd International Conference on Software and Data Technologies - ICSOFT 2007, Barcelona, Spain (2007)
17. Ginige, J.A., Ginige, A.: Challenges and Solutions in Process Automation in Tertiary Education Institutes: An Australian Case Study. In: 6th IEEE International Conference on Computer and Information Science (ICIS 2007), Melbourne, Australia (2007)

The Influence of Mobile Computing to Information Systems Technology

Otthein Herzog, Michael Lawo, Hendrik Witt, and Michael Boronowsky

TZI and MRC, Universität Bremen

Abstract. The vision of Mobile and Wearable Computing is motivated by the observation that the simple transfer of the desktop paradigm is not sufficient in situations where the user needs to focus on a task related to the real world.

Thus a new Mobile Computing paradigm is required. A functional definition is a system that can be used at any time and anywhere and does not in any way disturb the user's interaction with the real world.

Key properties required to achieve this are

(1) non- disruptive user interfaces characterised by a low cognitive load,
(2) hands-free operation,
(3) an unobtrusive form factor,
(4) the ability to model, recognise and act upon events in the environment (context sensitivity), and
(5) seamless, ubiquitous connectivity.

On a certain high level the above requirements are fairly obvious. From the application point of view it is equally obvious that today no system really fulfils them. What is less obvious is how the above functional definition translates into a technical specification and how the development towards fulfilling such specification should best proceed in the near future.

In the technical and scientific community there is a heated debate about what constitutes such systems with visions ranging from building upon commercially available 'PC on a belt' and PDA solutions to integrate concepts of transistor level integration of electronics into textiles.

The talk will describe in detail the new requirements for user interfaces and the state of context detection. The architecture of Mobile and Wearable Computing systems, the middleware and applications in domains like healthcare, production, maintenance will be outlined in some detail based on the experience of the authors from different research projects on the national and European level.

Engineers Discovering the "Real World" – From Model-Driven to Ontology-Based Software Engineering

Wolfgang Hesse

Fachbereich Mathematik und Informatik, Univ. Marburg,
Hans Meerwein-Str., D-35032 Marburg
hesse@informatik.uni-marburg.de

Abstract. Going along with the successful dissemination of the Unified Modelling Language (UML) and the growing popularity of the Model driven Development (MDD) initiative models are occupying the world of Software Engineering. As *prescriptive* means they are not only changing – and maybe soon dominating - our approaches to software construction but in their *descriptive* function they have also considerably influenced our views on software application domains. Since software systems and their application domains are almost ubiquitous we can say: models reflect and represent our view on the real world. In philosophical terms, models have no longer just a technical end and purpose but they reflect our *Weltanschauung*, i.e. are part and centre of modern ontology.

In this contribution I shall focus on the ontological dimension of software modelling. Models are like windows between the outer, "real" world and our inner world of understanding – described and reflected by software. Software development is *reality construction* and with our application models we re-construct our surroundding world. By this re-construction it is converted (and reduced) to a *world of objects* modelled as data capsules and maybe soon controlled by RFID chips. So called "ontologies" are the dictionaries of this construction process.

1 Introduction: The Prescriptive and Descriptive Role of Models

There is no doubt that models have become the central element of software development. The successful dissemination of the Unified Modelling Language (UML) and the growing popularity of the Model Driven Architecture/Development (MDA/MDD) initiative [M-M 03] are clear indicators of this methodological shift. Models have overtaken the role of specifications: Where twenty years ago methodologists debated about algebraic, operational, formal or semi-formal specifications now UML and MDA/MDD are the undisputed leaders in the software methodology scene.

Is this all just a renaming – old wine in new skins? Apparently not since models are at the same time more and less powerful than specifications. Less powerful since a specification is expected to be unambiguous and complete: it has to serve as a counterpart for the verification of any computer program built to fulfil the specification. It is an exact *pre-image* of what is to be built and an unmistakable measure of the correctness of the result. Models do often miss such strict quality criteria. On the other hand, models can be more powerful than specifications insofar as they do not only appear as pre-images but also as *after-images* and, in particular, and play both roles at the same time.

Stachowiak has emphasised the relationship of each model to another object - its *original* - to which it is in some respect similar or equal (cf. [Sta 73, Lud 02]). Now it is important whether the model comes into existence *before* or *after* the original. In the first case we call it *prescriptive* and a *pre-image*, otherwise *descriptive* and an *after-image*. We can use a simple and intuitive notation (a modified form of that in [Küh 06]) to denote both situations:

(1.1) $\mathcal{O} \blacktriangleright \mathcal{M}$ ("$\mathcal{O}$ is modelled by $\mathcal{M}$") and $\mathcal{M} \blacktriangleleft \mathcal{O}$ ("$\mathcal{M}$ is concretised to $\mathcal{O}$")

Examples of prescriptive models are the paper model house or a 3D visualisation produced by an architect or the above mentioned specification of a computer program. Examples of descriptive models are geographical maps or toy puppets, buildings or trains.

In modern Software Engineering, models often play a double role: They are *pre- and descriptive* at the same time. A UML model of a certain application domain or *Universe of Discourse (UoD)* - say a library - may represent this domain including books, journals, authors, borrowers and their relationships and it can serve as some sort of specification for the library administration software system to be developed. In general terms, a model $\mathcal{M}$ stands for an original domain $\mathcal{OD}$ and is at the same time a blueprint for a "system original" $\mathcal{SO}$ to be built:

(1.2) $\mathcal{OD} \blacktriangleright \mathcal{M} \blacktriangleleft \mathcal{SO}$

In a previous paper I have compared this kind of models with *Janus* – the double-faced Roman God who guarded the entrance of almost every Roman patrician house (cf. [Hes 06]). Like Janus, UML models look at the same time *outside* (towards the application domain) and *inside* (to the software to be developed).

Fig. 1. The double-faced God Janus on a Roman coin

Model-driven Development and *Model-driven Architecture (MDD/MDA, cf.* [M-M 03]) are relatively new keywords which emphasize the prescriptive aspect of models. In MDA terms, a Platform-independent model (PIM) is taken as pre-image for building various

Platform-specific models (PSM's) which in turn specify implementations (system originals PSO) to be executed on those particular platforms:

(1.3) $\mathcal{PIM} \blacktriangleleft \mathcal{PSM} \blacktriangleleft \mathcal{PSO}$

The transition from $\mathcal{PIM}$ to $\mathcal{PSM}$ may be supported (and partially automated) by a sequence of transformation steps which enrich the $\mathcal{PIM}$ by platform-specific details and gradually transform it to the $\mathcal{PSM}$. The transition from $\mathcal{PSM}$ to $\mathcal{PSO}$ is supported by code-generating tools.

Here the question arises to which degree this approach reflects the double-face character of models or in other words: how the *descriptive* character of models may be emphasised and supported. In the MDA/MDD literature, the *Computation-independent model* ($\mathcal{CIM}$) is mentioned as a model prior to the $\mathcal{PIM}$. However, little is said about the linguistic form and level of the $\mathcal{CIM}$ and its relationship to the $\mathcal{PIM}$, to its original domain and the requirements which give reason to these models. In my view, the $\mathcal{CIM}$ and $\mathcal{PIM}$ are not only *driving* the following development but they are themselves *driven* by the original domain and its requirements. However, so far there are no MDA/MDD concepts or tools to take this aspect into account [Hes 06].

If we look at UML and its offerings to support the very early (*analysis* and *pre-design*) stages of software development we are confronted with a similar gap. The only UML concepts for this stage are *use cases* and *use case diagrams*. Though they have been advocated by their originator Ivar Jacobson as being "object-oriented" they represent mainly functional behaviour and are not "oriented" to any objects – apart from their relationship to the actors which, however, are not part of the system themselves. As a consequence, we can state a gap on the structural (class/object) side of UML concepts: There are no UML elements that support the gathering and management of knowledge about a given original domain from an *objects analysis* perspective. As we will see below, the existing UML concepts ("class" and "object") are not appropriate for the purposes of this early development phase.

To sum up: If we agree on the double-face character of models and concentrate on their descriptive function which is particularly relevant in the early analysis and requirements elicitation phases we come upon some gaps in the MDA/MDD methodology as well as in the UML language. In the following chapters I am going to further discuss these gaps and possible ways to fill them. In particular, *glossaries* (as have e.g. been proposed within the Klagenfurt Conceptual Predesign (KCP) methodology, cf. [M-K 02]) seem to be appropriate to fill the UML gap.

Likewise, an extended MDA/MDD development & transformation approach starting at the original domain with ontologies in glossary form might fill the second gap. This approach has been defined and is presently being elaborated by a joint task group of the universities of Klagenfurt and Marburg. In the later sections (4 ff.) of this article, I will sketch this new approach as an extended MDD process which will be called *ontology-based* (OBSE, cf. [Hes 05], [BHR$^+$ 07]).

In order to better motivate this approach, two clarifications seem necessary. They concern a closer analysis (1) of the role of models throughout the software development process – cf. section 2 - and (2) of the term "object", what we might consider as “objects of the real world” and how these may be represented by model elements – in particular during the very early analysis phase – cf. section 3.

2 The Role of Models in the Software Development Process

Models are now present and intensively used in almost all parts of the software development process – sometimes in either and often in both their descriptive and prescriptive roles. The following table is intended to give an overview on important model elements, their primary characteristics and uses (cf. fig. 2).

Model elements, concepts, diagrams	**as after-image**	**as pre-image**
Use cases	• describe system behaviour from users perspective	• contain/imply requirements on system functionality
Use case diagrams	• describe interaction between users (*actors*) and system as well as selected use case relationships	• may be used as spec for system functions (e.g. shown as activity diagrams)
Classes and objects, structure diagrams (UML)	• represent static structure (Classes, objects, associations) of the application domain	• spec for data definitions, files, DB schemata and tables
States, state transitions, state charts	• reflect behaviour of application objects	• spec for state attributes and operations
Interactions, interaction diagrams	• illustrate message exchange between application objects	• spec for control flow between system objects and operations
Activities, activity diagrams (UML)	• describe processes of the application domain	• spec for system functions
Packages, components	• reflect higher structures of application domain	• spec for system design into modules and components
Code	• reflects all aspects of application domain to be implemented as software	• specifies and controls the execution of all system functions

Fig. 2. Models and their elements in the software development process

In this table, model elements and concepts have been listed in the order of their appearance during the system development process and we can first conclude that there is no *one single* intermediate model in this process as suggested by formula (1.2) but rather a *chain of models* ranging from its earlier to its later phases. Generally, we observe that the early-phase concepts (mainly use cases and corresponding diagrams) have their focus on the descriptive side while the later-phase concepts (packages, components, code) are primarily prescriptive and the concepts in between are equally used for both purposes – maybe with a certain priority on prescriptive use.

Another observation is concerned with the static and dynamic character of model elements and concepts. A closer analysis shows that static concepts (classes, objects, components, packages) are mostly used for system design and construction, i.e. for prescriptive purposes. As tools for domain analysis and description they are less used - and less appropriate as shall be illustrated by the following example.

Let us consider as an example a sales management system of a mail-order firm for clothing products specified in a requirements document. The essence of this document is a collection of use cases where terms like *customer, family name, given name, age, sex, size, clothing, clothing piece, dress size, price, …* are occurring. Specific properties, characteristics, relationships and constraints of these items – which in common represent essential knowledge on the UoD – are scattered somewhere in the document – if at all – or have to be inquired from the future system owners or users.

UML offers just two mechanisms to deal with this kind of knowledge: (1) leave it in (and search it from) the use case descriptions or (2) build a class model like that of fig. 2.

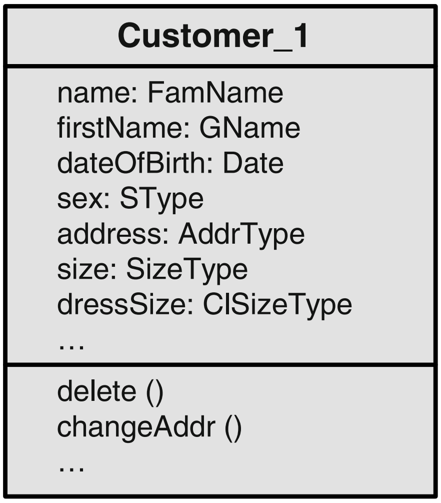


Fig. 3. A model for the class *Customer*

Both options are rather unsatisfactory: In the use case descriptions the knowledge on objects is hidden behind functional aspects, it is fragmented and scattered between various use cases and possibly incomplete. Detailed descriptions of objects or classes (as e.g. delivered by users through personal communication) have to be "wrapped" into use cases and thus may be obscured.

On the other hand, knowledge on objects of the UoD can well be gathered and structured in an UML class model – but this kind of modelling requires a lot of design decisions which might be premature and compromising at this stage of the development process. For example, it is by no means clear whether "address" is to be modelled by a "class" or by an attribute, whether the attributes concerned with clothing are to be modelled by an own class and how they are associated with the classes, which connections within the application domain are to be modelled by associations or by class-attribute connections and so on.

All this kind of problems may be solved and the UML gap may be filled by glossaries – as have been proposed by Mayr et al. and are employed in the KCP methodology (cf. [M-K 02] and various other papers). Fig. 4 shows a small piece of a possible glossary for the *Customer* UoD introduced above.

Project: Cloth sales management			AoD: Customer adminstration		IG: Organisational Dpmt			
id#	name	classification	quantity-description	examples	value domain	syno-nyms	textual description	requirement sources
D001	Customer	thing-type		<Cu_Schmidt>		D032		UC1, UC2
D002	FName	thing-type		Schmidt	string		FamName	UC1, UC5
D003	GName	thing-type		Robert	string		GivenName	UC1, UC5
D004	DateOfBirth	thing-type		12.12.1974	Date			UC1, UC5
D005	Sex	thing-type		m	{m, f}			UC5
D006	Address	thing-type						UC2
D007	Street	thing-type		Müllerstr. 54	string			UC2
D008	PostalCode	thing-type		80799	Integer			UC2
D009	City	thing-type		München	string			UC2
...								

Fig. 4. A KCP glossary for *Customer* and related terms

Like all other kinds of models, glossaries are used both as after-images (of the things, relationships and other phenomena of the UoD) and as pre-images – e.g. for UML classes which can be generated from glossaries by automated transformation tools [SMK 04]. Moreover, glossaries are a natural and practical starting point for building ontologies [KMZ 04]. This has made them the obvious candidates for our joint approach to Ontology-based Software Engineering (OBSE, cf. [BHR$^+$ 07] and below).

3 On Real World Objects, Conceptions and Ontologies – A Semiotic Approach

There is no doubt that the early phases of Software Engineering (of requirements elicitation, analysis of the UoD etc.) deal to a larger degree with objects of the "real world". We call our main-stream tools and techniques *object-oriented*, take it for granted that everybody knows what an *object* is and thus suggest a natural connection between what we call the "real world", its objects and their "mapping" by models and pieces of software.

But the quotation marks already indicate a certain doubtfulness or at least haziness of this superficial view. In this article, I cannot summarise (not to mention redraw) a millennial philosophical discussion about these issues – which would start with Parmenides' study of a boat consisting of numerous parts where maybe none of them was part of the original vessel, which might continue with endless discussions about the *essence of things* or the Kantian *Ding an sich* (thing per se) and run into modern disputes on realism or constructivism (cf. [H-B 01]). But whatever philosophical position we might adopt, it is quite obvious that the mental images of the world "out there" we are dealing with are not identical with that world. Often we cannot precisely delimit and describe – much less completely grasp the material objects around us – not to speak about all the immaterial ones like *friendship, work, peace, war,* etc. Even if we observe that large groups of humans can arrive at a broad consensus about certain objects and domains und therefore might be inclined to assume the existence of *objective knowledge* about such things, we never can be sure about it as well-known examples of scientific turns (e.g. the turn from a geocentric to a heliocentric world view) have shown. Thus in their total essence all the material and most virtual objects we deal with in our life stay *inaccessible* to us.

Of course, we are also unable to store objects (as such) in our computers but all we can achieve is to store facts, features, attributes etc. *about* such objects. In order to visualise this situation, I want to recall *semiotics*, the *science of signs and their meanings* and, in particular, the well known semiotic triangle in its tetrahedron version as introduced and used in the FRISCO report (*FRamework of Information System Concepts,* cf. [FHL$^+$ 98]). Following its line of reasoning, in our information systems we are always using *representations* which *refer to* objects (*referents*) of some *domain* or UoD (cf. fig. 5).

However, the connection between a representation and its referent is by no means always clear, objective and unambiguous (and therefore is shown as a dashed line in fig. 5) – as can e.g. easily be deducted from the well-known homonym/synonym problems. The connection is not "just there" but is established (constructed) as a (primarily subjective) *conception* of an actor who uses a representation for dealing with some conceived referent. Groups of actors can make themselves understood by negotiating and joint action and thus establish a *quasi-objective conception* of certain common domains. This is the way how traffic signs, international codes, icons and – to a certain degree – human languages work.

Here is the point where *ontologies* come in (cf. [Gua 98], [Hes 02]). As we have just seen, *total ontology* – in the sense of *complete* access to, penetration and understanding of *all* aspects of a given domain – is impossible. Therefore ontology (as a human activity) can either try to find and understand prerequisites, conditions and constraints for *our being* in the world as it is known to us or try to reduce the infinity of aspects and features of real world domains and their objects to negotiated and/or prefabricated conceptions – i.e. to *conceptual models*. The first approach is that of traditional philosophy, the second is that of Artificial Intelligence – nowadays shared and adopted by many other branches of Computer Science – including Software Engineering.

If we want to depict this situation we should rather replace the little circle in the left hand side corner of fig. 5 by a cloud which would symbolise our (permanently and irredeemably) impeded access to the real world domains and objects. On the other side, it is the declared and indispensable goal of (western-oriented) scientific activity to lift that cloud or at least to build proxies for what might be behind it.

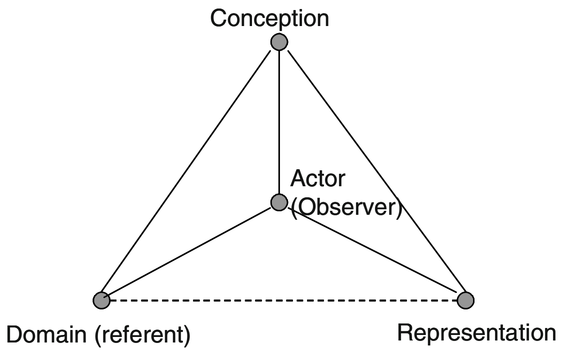


Fig. 5. The semiotic tetrahedron of FRISCO

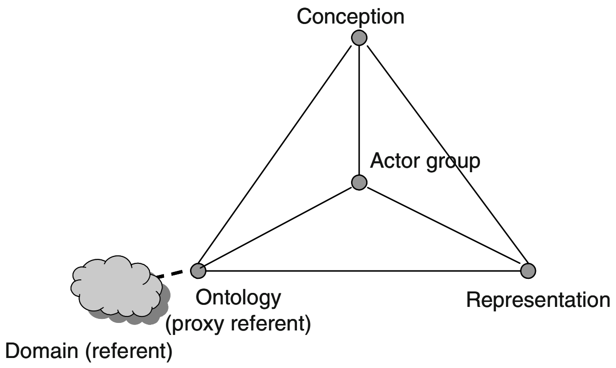


Fig. 6. Modified semiotic tetrahedron

Turning back to the principal problem of software domain analysis, i.e. to determine and abundantly describe a domain, its objects and relationships, we have to acknowledge the limitedness of such activity. Since we are unable to completely take possession of our *customer*, to record all of his or her infinitely many properties, aspects and features we build models (= conceptions) and use them as proxies for the inaccessible original. Thus every conceptual model is a necessary *reduction* of its original to a finite, closed set of features which are selected according to the goals and purposes of a particular project or application.

Ontologies are conceptual models with a more encompassing and ambitious pretension: They should collect all available or at least all useful knowledge about a certain domain – independent of particular projects or applications. As conceptual models, they are as well reductions of their originals but on the other hand they should be *open* and *possibly infinite* to gather all present and future knowledge about the domain in question. In our view, glossaries are the natural and most useful form for this kind of ontology.

In fig. 6, this modified situation is summarised and depicted: *Representations* are human sign tokens (words, icons, pictures, pieces of software, etc.) which denote objects of the real or an imagined world. *Conceptions* are closed, reduced maps of features of such objects – produced for some particular goal or purpose as is common practice with models in Software Engineering. *Ontologies* are proxies for *domains* of the real world (inaccessible, behind the cloud). As such they are open and possible infinite collections of knowledge about their domains. The dashed line between *representation* and *domain* has been replaced by a solid one to the new *ontology* corner. This reflects the situation that an *actor group* has achieved agreement on a certain domain – an agreement which in turn is often mistaken as *objective* and *definite knowledge* about it.

4 An Extended Metamodel for Ontology-Based Software Development

Metamodels are a well-suited means for summarising and exposing the concepts of a modelling language or methodology. As a work summary of our joint task group on Object-oriented Analysis ("OOAK"), we have published the SAMMOA metamodel in 1998 [H-M 98] – some years before UML and its metamodels became popular. If we look at these in their present form, we still find use cases as the only construct supporting the early phases of Software Engineering. However, as we have seen in the previous sections, they are rather inappropriate for gathering and structuring knowledge about *objects* of a certain UoD, and glossaries turn out to be much better suited for this purpose. Therefore I have proposed in an earlier article to extend the UML metamodel by a new category of classifiers: the glossaries. Since glossaries act as proxies for domain objects I have also called them *A-objects* (for *application objects*, cf. [Hes 06]). An example of an A-class *Customer* (= schematic description of a set of uniform customer A-objects) is the glossary depicted in fig. 4.

With this extension, we now have description means for all kinds of objects occurring during the various stages of software development (cf. formula 1.2 in section 1): A-classes can be used as proxies for objects of the source domains, UML classes

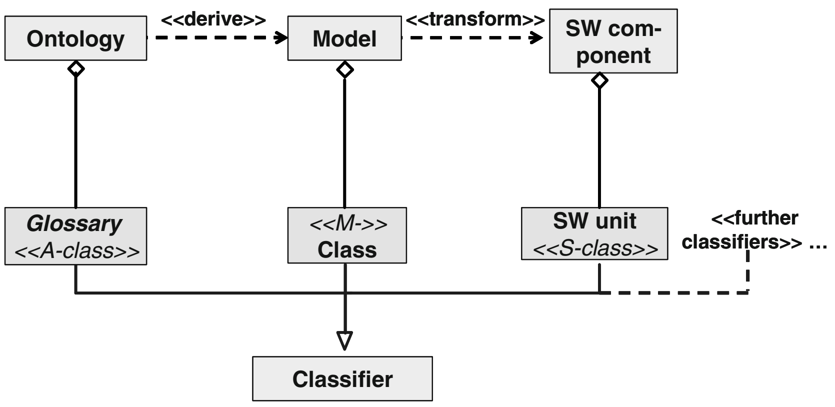
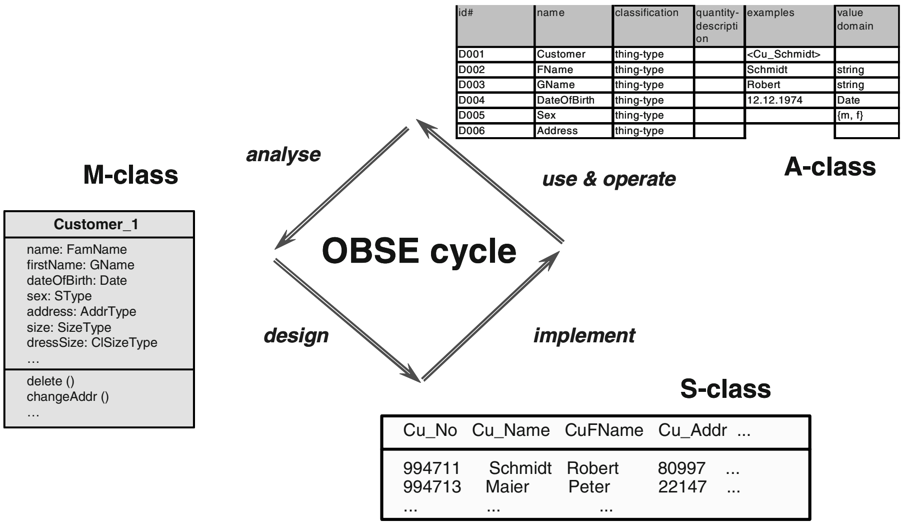


Fig. 7. Skeleton metamodel for objects of a throughout SW development process

(here called M-classes) for the well known static model elements and classes of some OO programming language (here called S-classes) for the target software domain.

The skeleton of an extended metamodel comprising these old and new categories of classifiers is depicted in fig. 7. Compared to the well-known UML metamodel with its classifiers *class, component, use case, interface* etc, it has been augmented by the new categories of *(A- and S-) classes* and *ontologies*. An ontology consists of one or several glossaries (= A-classes), a model of one or several (M-) classes, and a software component of several programming units (S-classes).



id#	name	classification	quantity-description	examples	value domain
D001	Customer	thing-type		<Cu_Schmidt>	
D002	FName	thing-type		Schmidt	string
D003	GName	thing-type		Robert	string
D004	DateOfBirth	thing-type		12.12.1974	Date
D005	Sex	thing-type			{m, f}
D006	Address	thing-type			

Cu_No	Cu_Name	CuFName	Cu_Addr	...
994711	Schmidt	Robert	80997	...
994713	Maier	Peter	22147	...
...	...	...		

Fig. 8. OBSE development cycle with object descriptions on various levels

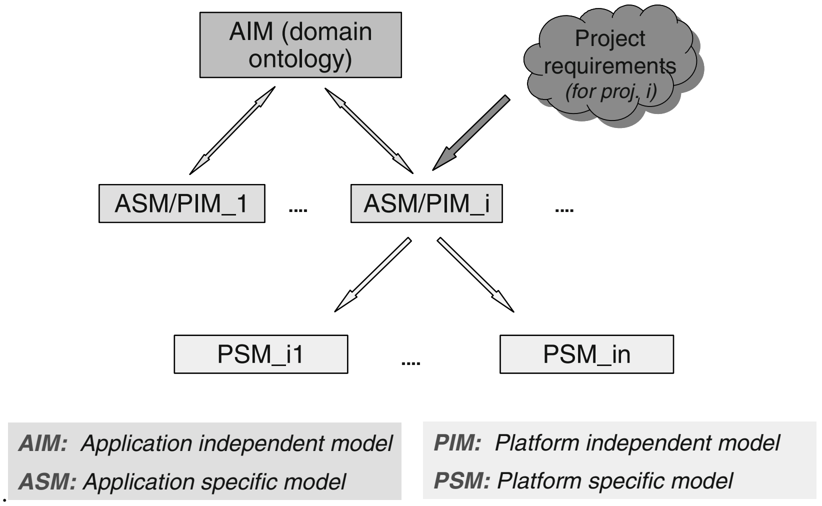

AIM (domain ontology)
Project requirements (for proj. i)
ASM/PIM_1
....
ASM/PIM_i
....
PSM_i1
....
PSM_in
AIM: Application independent model
ASM: Application specific model
PIM: Platform independent model
PSM: Platform specific model

Fig. 9. OBSE - an extended MDA approach to application modeling

This metamodel also reflects our extended understanding of Model-driven development in the OBSE framework (cf. [BHR⁺ 07]). In fig. 8, a paradigmatic OBSE development cycle is depicted. It starts at the top with A-classes and A-objects described by glossaries and it runs through M-classes (UML models) to S-classes of some programming language or relational database.

We agree with the fundamental MDA idea of viewing the software development process as a *chain of model transformations*, e.g. leading from platform-independent to platform-specific models (*PIM* to *PSM*, cf. fig. 9, lower part). But, unlike MDA, we suggest to start that chain much earlier than at the *PIM* (or UML class diagram) level. Application objects (= instances of A-classes) are the natural starting point of a truly object-oriented analysis and are effectively represented by glossaries. Two types of sources for such objects are disposable: Requirements descriptions (e.g. in the form of use cases) for particular projects or ontologies. Extending the MDA terminology we might call the latter application-independent models (*AIM*) – from which application-specific parts (*ASM's*) may be extracted to build the *PIM*. This vision of OBSE has been sketched in fig. 9.

5 Conquering the Real World – By RFID and Internet of Things?

In the previous sections, I have briefly summarised the changing role of modelling in Software technology. It started with low-level computer programs which are mainly prescriptive models (of actions to be performed when they are executed) and was continued (among others) with high-level programming languages, Entity-Relationship Models and UML diagrams which allow much closer and better understandable descriptions of real world domains and problems. Now, glossaries and ontologies pave the way to open, potentially infinite descriptions of software application domains. At least from the realist philosopher's point of view, it seems that real world objects can be "approached" arbitrarily close.

This raises the question whether real world objects (or at least material ones like articles in a supermarket or animals in a farm) can *directly* be accessed – e.g. by implanted RFID chips. One might assume that the philosophical questions raised in section 3 and illustrated by the cloud of fig. 6 can be solved by a simple technical trick – the "Internet of Things" as the ultimate step of computer scientists' penetration of the real world!

There is no doubt that this step is technically feasible, that parts of this vision might come true sooner or later and would substantially affect our daily life. However, the philosophical core question searching for the "essence" of things will definitely *not* be "solved" by technical means. RFID chips can of course facilitate the search for and identification of particular material things but they never can precisely delimit their boundaries – not to speak of revealing the inherent infinity of properties, capabilities, aspects, inner values etc. of the animate and inanimate things populating our home on Earth.

References

[BHR+ 07] Bachmann, A., Hesse, W., Russ, A., Kop, Ch., Mayr, H.C., Vöhringer, J.: OBSE – an Approach to Ontology-based Software Engineering in the Practice. In: Proc. EMISA conference, St. Goar. LNI, pp. 129–142. Koellen-Verlag (2007)

[FHL+ 98] Falkenberg, E., Hesse, W., Lindgreen, P., Nilsson, B.E., Oei, J.L.H., Rolland, C., Stamper, R.K., Van Assche, F.J.M., Verrijn-Stuart, A.A., Voss, K.: FRISCO - A Frame-work of Information System Concepts - The FRISCO Report. IFIP WG 8.1 Task Group FRISCO (1998), Web version: http://www.mathematik.uni-marburg.de/~hesse/papers/fri-full.pdf

[Gua 98] Guarino, N.: Formal Ontology and Information Systems. In: Proc. FOIS 1998, Trento (Italy), pp. 3–15. IOS Press, Amsterdam (1998)

[H-B 01] Hesse, W., van Braun, H.: Wo kommen die Objekte her? Ontologisch-erkenntnistheoretische Zugänge zum Objektbegriff. In: Bauknecht, K., et al. (eds.) Informatik 2001 - Tagungsband der GI/OCG-Jahrestagung, Computer-Gesellschaft, Bd. II, pp. 776–781, books_372ocg.at; Bd. 157, Österr (2001)

[H-M 98] Hesse, W., Mayr, H.C.: Highlights of the SAMMOA framework for object oriented application modelling. In: Quirchmayr, G., Bench-Capon, T.J.M., Schweighofer, E. (eds.) DEXA 1998. LNCS, vol. 1460, Springer, Heidelberg (invited talk, 1998)

[Hes 02] Hesse, W.: Das aktuelle Schlagwort: Ontologie(n). In: Informatik Spektrum, Band 25.6 (2002)

[Hes 05] Hesse, W.: Ontologies in the Software Engineering process. In: Lenz, R., et al. (eds.) EAI 2005 - Tagungsband Workshop on Enterprise Application Integration, GITO-Verlag, Berlin (2005), http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-141/

[Hes 06] Hesse, W.: Modelle - Janusköpfe der Software-Entwicklung - oder: Mit Janus von der A- zur S-Klasse. In: Proc. Modellierung 2006. LNI, vol. P-82, pp. 99–113. Springer, Heidelberg (2006)

[KMZ 04] Kop, Ch., Mayr, H.C., Zavinska, T.: Using KCPM for Defining and Integrating Domain Ontologies. In: Bussler, C.J., Hong, S.-k., Jun, W., Kaschek, R., Kinshuk, Krishnaswamy, S., Loke, S.W., Oberle, D., Richards, D., Sharma, A., Sure, Y., Thalheim, B. (eds.) WISE Workshops 2004. LNCS, vol. 3307, pp. 190–200. Springer, Heidelberg (2004)

[Küh 06] Kühne, Th.: Matters on (Meta-) Modeling. Journal on Software and Systems Modeling (SoSym) 5(4), 369–385 (2006)

[Lud 02] Ludewig, J.: Modelle im Software Engineering - eine Einführung und Kritik. In: Glinz, M., et al. (eds.) Proc. Modellierung 2002. LNI, vol. P-12, Springer, Heidelberg (2003)

[M-K 02] Mayr, H.C., Kop, Ch.: A User Centered Approach to Requirements Modeling. In: Glinz, M., Müller-Luschnat, G. (eds.) Modellierung 2002 - Modellierung in der Praxis - Modellierung für die Praxis. LNI, vol. P-12, pp. 75–86. Springer, Heidelberg (2003)

[M-M 03] Miller, J., Mukerji, J.: MDA Guide. Version 1.1.1, Object Management Group (2003)

[SMK 04] Salbrechter, A., Mayr, H.C., Kop, C.: Mapping Pre-designed Business Process Models to UML. In: Hamza, M.H. (ed.) Proc. of the 8th IASTED International Conference on Software Engineering and Applications, pp. 400–405. ACTA Press, Cambridge (2004)

[Sta 73] Stachowiak, H.: Allgemeine Modelltheorie. Springer, Wien (1973)

Holistic Engineering of Ultra-Highspeed Mobile Information and Communication Systems

Matthias Jarke

Information Systems Group, RWTH Aachen University & Fraunhofer FIT
Ahornstr. 55, D-72074 Aachen, Germany
jarke@dbis.rwth-aachen.de

Abstract. The convergence of wireless communications with internet protocols, semi-structured data models, distributed computing and multimedia processing has led to a revolution in products, services, and infrastructures for mobile information and communication management. For the next steps of this evolution, we are hitting limits of physical, computational, and organizational complexity simultaneously. Research questions span disciplinary boundaries from application engineering and usability all the way down to electronics and physics, as the traditional boundaries between protocol stack layers prevent innovative holistic solutions. Some of these challenges are addressed in UMIC, an excellence cluster on Ultra-highspeed Mobile Information and Communication funded at RWTH Aachen University as part of the German National Excellence Initiative since late 2006. This paper presents a brief summary of UMIC challenges and initial results mostly from an application perspective.

1 Introduction

In the late 1990s, two major trends took center stage in the information and communication sector: the wide availability of the World Wide Web, which opened the Internet to millions of users, and the explosive growth of mobile telephony which has now reached about 2.000 million phones worldwide. In the new millenium, after a short break due to overheated expectations, these trends continued in slightly surprising directions. Instead of the predicted development of the Internet towards a *"semantic web"* [BHL01] in which mostly subsystems talk to each other, easy-to-use *"social software"* tools and significantly broader media usage have turned the Internet into a Social Web, often called Web 2.0 [Re05]. Semantic technologies are being postponed in a new business web vision of a "Web 3.0". Communities and semi-formal folksonomies have taken precedence over ontologies for the moment, taking advantage of novel data models and programming approaches such as XML-integrated web service programming environments. Examples such as Internet Telephony (e.g. Skype), photo and video collections (flickr, Youtube, …), social networking communities (Xing, Facebook, StudiVZ, …), and collaborative virtual reality (WOW. Second Life) attract tens to hundreds of millions of users and generate significant businesses.

Simultaneously, the much touted evolution from the GSM standard (which still underlies 80–90% of mobile telephony) towards so-called 3G (and beyond) systems based on the UMTS standard has faced many difficulties and has been undercut by GSM extensions as well as by fast and cheap Internet access by fixed-line telephony such as DSL or local wirelass local area network (WLAN). Nevertheless, growth of mobile communities both in terms of users and in terms of functionality continues, especially in areas with little wired infrastructure.

Today's business visions focus on the *convergence* of these technological and user-driven trends. Wireless communication should be merged with faster Internet protocols, distributed computing, and multimedia processing, in new applications characterized by easy usability and social attractivity for millions to billions of users. For example, the Germany ICT industry association BITKOM, together with the CeBIT tradefair and major players in journalism and consulting, have established an annual "Convergators Award" (cf. www.convergence.de) for the most innovative applications that highlight the business and society value of convergence.

While entries in such competitions illustrate the potential, it cannot be denied that the worldwide availability of such converged applications -- let alone mobile information and communication systems that would give us anyplace-anytime mobile usage with the cost and quality we now take for granted in stationary Internet – is still far away. Seemingly simple basics such as power supply for mobile devices, or reliable and fast protocols in mobile multi-user settings pose difficult challenges as we reach limits of present technology. Higher-level questions concern security and privacy, socially acceptable and cost-effective placement of base stations for mobile communications in densely populated areas, usability of ever more and smaller devices and "intelligent" ambience, or social acceptance of mobile grid computing.

Many of these challenges require interdisciplinary cooperation over a broad range of areas from the physics foundation of electrical engineering via computer science and information systems engineering to the social sciences. The Excellence Cluster UMIC (Ultra-Highspeed Mobile Information and Communication Systems) at RWTH Aachen is a 35 mio. € research center funded within the national Research Excellence Initiative of the German government since late 2006 to address some of these issues (cf. www.umic.rwth-aachen.de). Research in UMIC is organized according to three major layers investigating

- future mobile applications and services from a system engineering and application engineering perspective,
- innovative concepts for future wireless transport platforms from a communications engineering as well as information theory perspective, and
- technological concepts concerning radio subsystems and systems-on-a-chip at the sub-micron level from electrical engineering and applied physics perspectives.

However, these perspectives are not separated from each other as strictly as in the layered communication protocol architectures of the past such as ISO-OSI or TCP/IP. Efficient solutions for "non-functional" goals such as performance and cost, system reliability, security and privacy, energy efficiency and environmental impact require novel cross-disciplinary methods, analysis and design tools, even system architectures which are bundled in a fourth area of the Excellence Cluster (cf. fig. 1).

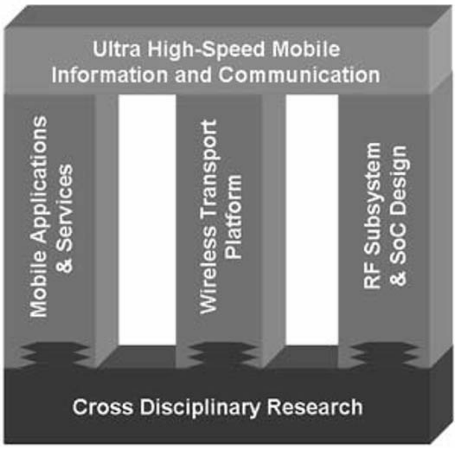

Ultra High-Speed Mobile
Information and Communication
Mobile Applications & Services
Wireless Transport Platform
RF Subsystem & SoC Design
Cross Disciplinary Research

Fig. 1. Structure of the UMIC Research Cluster

In the remainder of this paper, we shall first illustrate the present state of "convergence" by an example. We then outline where we see some of the main upcoming challenges by analyzing two UMIC lead scenarios for mobile applications and services. In sec. 3, we summarize some of the cross-disciplinary challenges and give examples of methods and initial results how to address them. We end with some conclusions and an outline of future work.

2 Convergence and the UMIC Lead Scenarios

In this section, we first present an example of what is presently considered under the label of "convergence" in mobile settings, and then look at some of the future challenges and scenarios studied in UMIC.

2.1 Convergence Example: Mobota

One of the winners of the 2008 Convergators Award is an outdoor system for mobile training and competition called mobota (WPH06], developed at Fraunhofer FIT's Center for Software Concepts (www.zfs.rwth-aachen.de). Using a GPS-enabled PDA or similar device, amateur sports enthusiasts can compete in movement sports such as running, biking, or triathlon independent of time, because their movements are tracked and recorded in a community environment based on MS Sharepoint and can be visualized in different manners on platforms such as Google Earth or MS Virtual Earth. For example, the map on the left of fig. 2 shows some biking paths taken by two bikers, whereas the graphs on the right demonstrate the height profile of their routes and their speeds in each part of the way. The PDA shows current user guidance but could also include actual competitive performance, body sensor data, and the like. The mobota example thus addresses map technology, location-tracking and location aware services, some simple sensor integration, visualizations, and community sharing technologies, including e.g. some security and privacy facilities. Beyond the sports application domain, many other geo-based community services can be assisted by such a system; sharing of experience among wheelchair drivers how best to enter subway stations is an experiment currently under investigation.

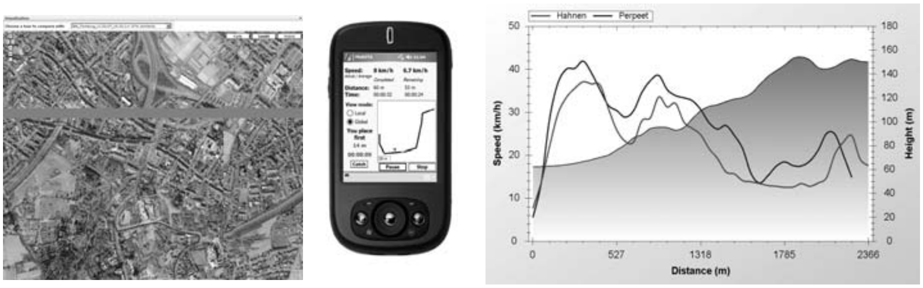


Fig. 2. Mobota Mobile Training Assistant

Mobota operates successfully in areas where a good wireless infrastructure with not too much interference is available, as long as users are not moving too fast. Poor base station infrastructure, large user numbers or high user speed, complex sensors or multimedia transfers still pose major problems in terms of availability, performance, and quality-of-service, as do long competition times where batteries quickly run out. Moreover, mobota faces a relatively stable scenario with not too much system evolution needs at runtime, and web servers are still placed on non-mobile devices.

2.2 The UMIC Lead Scenarios

If we have learned one thing from the past experiences in the Internet and mobile telephony, it is that it is extremely hard to predict what users will do with radically new technology. In UMIC, we therefore follow a research strategy that interleaves technological advances at all layers of the UMIC structure (see fig. 1) with user envisionments and experiments within a participative design strategy. Both technological directions and usage scenarios will be adapted over time by mutually learning from each other.

UMIC research in mobile applications and services pursues two lead scenarios, dubbed “Virtual Campfire” and “HealthNet”.

The Virtual Campfire lead scenario is based on the observation that the impact of mobile wireless technologies is probably much higher in areas of the world where no traditional wired infrastructure exists. Mobile devices are likely not just to remain web service clients but will also take the role of web service providers, or operate in a mobile P2P setting. Fixed network topologies will have to be augmented with ad-hoc networking where mobile devices serve as transmission stations to compensate for a weak terrestrial infrastructure. Energy efficiency is an even bigger problem than in developed areas, and security solutions can not be just concentrated on a few servers but must be supported by the mobile devices as well.

Our specific research scenario builds on prior work on an Afghan Community Information System (ACIS) for the rebuilding of cultural heritage research and culture-preserving city reconstruction after over 20 years of interruption and destruction due to external interventions and civil wars [KSJ*05]. In ACIS, a stationary geographic database server and various media servers were integrated into a reflective community architecture via a light-weight application server [Sp07] which offers three important features:

- cross-media information management based on the MPEG-7 metadata standard and related multimedia annotation tools,
- object-based and role-based security, and
- the "reflective" possibility to define pattern-based social network analysis tools over the community and to add /change services at runtime [JK08].

Mobile devices which are essential in the field in Afghanistan could only be used as ACIS clients with very limited capability. In the Virtual Campfire scenario, several extensions are being pursued to empower mobile researchers and city planners more effectively. Firstly, multimedia extraction and adaptation mechanisms have been developed to assist in the acquisition and presentation of multimedia information through small multimedia devices. Indeed, there is a major stream of UMIC research investigating the mobile acquisition, transfer, and presentation of high-quality graphics, video, and audio streams far beyond present capabilities, augmented by CHI research on the question how to interact effectively with such capabilities in future miniature mobile devices or with ambient environments that need to coordinate multiple such devices and capabilities.

A special problem for small mobile end user devices is the establishment of coherence among multiple media-based components, especially the question how multiple media-based informations and evidences can be presented in coherent theories. We have therefore developed a non-linear *multimedia story-telling* service as a mechanism for coherent information integration and information delivery which has even given the overall demonstrator the name: Virtual Campfire is the "place" where communities swap multimedia stories about their results and plans. The overall architecture of the current Virtual Campfire demonstrator is summarized in fig. 3.

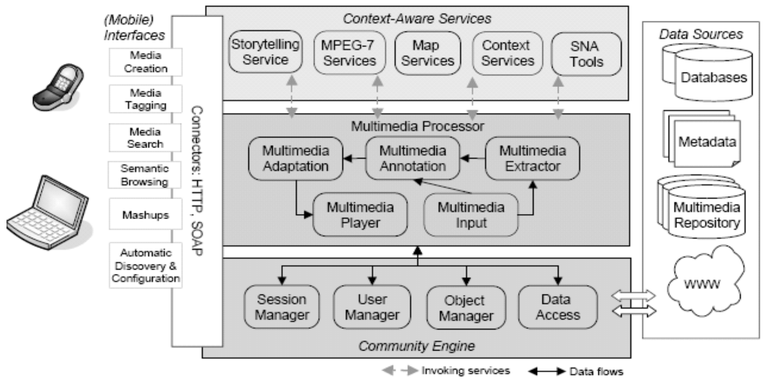


Fig. 3. Virtual Campfire Architecture using the ATLAS Community Engine

A key technological advance that distinguishes Virtual Campfire from the initial ACIS system, however, is the use of *mobile web servers*. As fig. 4 shows, mobile web service client usage is quickly growing in practice but there, the mobile user always has to take the initiative to upload media information (e.g. photos, GPS measurements) to (stationary) web servers. Since 2003, groups from communications engineering and information systems at RWTH Aachen, together with Ericsson Eurolabs, have successfully developed full-fledged standard conformant web server software on small mobile devices such as smart phones, and have made studies of their performance as well as scalability and security aspects [AHW07, Ge07, SJP07]. In this manner, mobile or stationary clients can download research data gained in the field in a synchronous or asynchronous manner, without interrupting the mobile researcher's work. In a peer-to-peer setting, information can even be transferred across several mobile stations, bundled in story-telling structures, and made available in a configured structure. We are just beginning to explore the practical potential of these prototype solutions in the field, but initial experiences are encouraging, despite the clear limitations of present wireless network technology.

To define these limitations and challenges is of course the research strategy goal of the Virtual Campfire scenario within UMIC, while at the same time providing better practical solutions to the ACIS community.

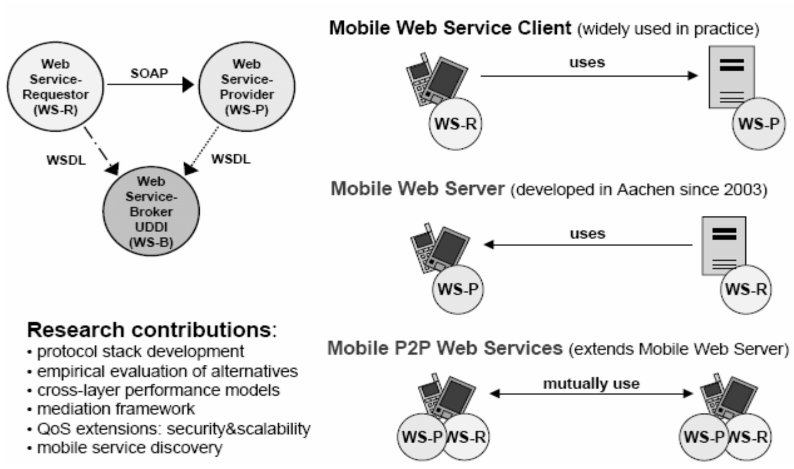

Web Service-Requestor (WS-R)
SOAP
Web Service-Provider (WS-P)
WSDL
WSDL
Web Service-Broker UDDI (WS-B)
Research contributions:
• protocol stack development
• empirical evaluation of alternatives
• cross-layer performance models
• mediation framework
• QoS extensions: security&scalability
• mobile service discovery
Mobile Web Service Client (widely used in practice)
uses
WS-R
WS-P
Mobile Web Server (developed in Aachen since 2003)
uses
WS-P
WS-R
Mobile P2P Web Services (extends Mobile Web Server)
mutually use
WS-P WS-R
WS-P WS-R

Fig. 4. Mobile Web Service Provisioning

The second UMIC lead scenario, *HealthNet*, is set in industrialized countries with solid wired as well as wireless infrastructure. Correspondingly, HealthNet addresses a different set of challenges related to very high and heterogeneous bandwidth requirements that would challenge future ultra-highspeed transport platform infrastructures. The HealthNet scenario comprises two sub-scenarios, one focussed on the "normal case" of mobile patients, the other on the "emergency case" of mobile doctors. Some of the characteristics of HealthNet are discussed below and summarized in fig. 5.

The first part of HealthNet addresses a consequence of demographic change. In addition to enhanced quality of life, 30 billion € per year could be saved in Germany alone if elderly people could postpone their move to special-care homes by one year on average. Many researchers have therefore proposed "ambient assisted living" scenarios in which

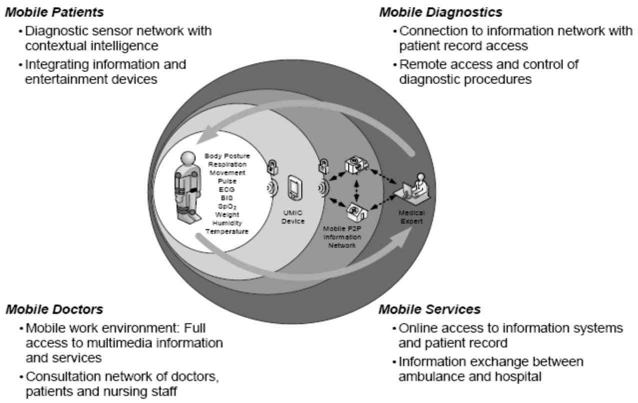

Mobile Patients
• Diagnostic sensor network with contextual intelligence
• Integrating information and entertainment devices
Mobile Diagnostics
• Connection to information network with patient record access
• Remote access and control of diagnostic procedures
Body Posture
Respiration
Movement
Pulse
ECG
BIS
SpO$_2$
Weight
Humidity
Temperature
UMIC Device
Mobile P2P Information Network
Medical Expert
Mobile Doctors
• Mobile work environment: Full access to multimedia information and services
• Consultation network of doctors, patients and nursing staff
Mobile Services
• Online access to information systems and patient record
• Information exchange between ambulance and hospital

Fig. 5. Overview of HealthNet Scenario in UMIC

body sensors, home sensors, and possibly robots would assist the elderly and mobile nursing personnel in health monitoring and assistance. Such a setting requires research in wearable and eHome design and computing aspects (integrating the sensors in clothes in a non-intrusive and easy-to-maintain manner) as well as a connection to information networks with advanced artefact detection, filtering, and compensation, and with context changes due to mobility (e.g. from home to transport to other buildings). In UMIC, we focus on the heterogeneous information integration and scalability implications of such a setting. While the amount of sensor data about individuals is quite modest, data about a very large numbers of patients have to be collected, sorted apart, and analyzed in an overall remote health monitoring setting. This creates complex demands e.g. on mobile data stream classification and aggregation, and on the fast detection of emergency events among large numbers of mobile clients. Subspace clustering methods from data stream mining are being experimented with to identify the relevant patterns in such a highly heterogeneous and noisy setting [AKMS07].

In an emergency situation where mobile doctors or even lay personnel need to assist a patient very quickly, the communication pattern changes significantly. In this sub-scenario, we encounter tele-consulting by video and audio conferencing, access to multimedia patient records, upload of mobile multimedia patient data, and many other ingredients of very high performance information transfer needs which need to be provided efficiently and reliably in different contexts. While many of these applications can be demonstrated today under controlled conditions already, their broad availability at affordable costs is the research goal of UMIC.

3 Cross-Layer Challenges and Approaches

The UMIC scenarios illustrate a variety of challenges to technology, architecture, and management structures which need to be addressed in an interdisciplinary nature across all the layers from technology to application. As mentioned before, UMIC is organizing this kind of cross-layer challenges according to specific non-functional requirements which have been identified as especially relevant: performance, availability and reliability, security and privacy, and energy efficiency. We can only outline these topics here, without going into details of the transport platform, base technology, or mathematical modeling aspects.

Traditional *performance* modeling approaches of communication systems are typically focussed on a single level of abstraction within the protocol stacks. However, more and more approaches try to optimize performance across layers and need corresponding cross-layer performance models. Examples of such questions investigated in UMIC include, e.g.,

- Is it more efficient to base a mobile web service provider on the TCP or the UDP transport protocol?
- Can we efficiently emulate and integrate different kinds of peer-to-peer protocols using overlay networks [SW05]?
- How can we identify good self-organizing strategies for so-called cognitive radios which can dynamically adapt to local load changes?
- Can Shannon's traditional information theory which is based on one-to-one communication be extended to deal much more effectively with multi-user communication settings?

Hardware development is approaching the nano-scale and thus the atomic level. This may yield more performance, lower costs, and maybe lower energy needs but has fundamental implications for quantum-scale uncertainty that cannot be captured at the level of base technology alone as in current systems. *Reliability* questions will have to be addressed at all levels of the system, even up to the user level. One of the directions of research pursued in UMIC is game-theoretic approaches that would mediate among conflicting results obtained from technology-level uncertainty compensated by redundancy.

Security and privacy are becoming even more difficult than now if more and more users and servers get mobile. Simply adding security features as an afterthought is not just a significant performance burden, as our experiments with mobile web servers show, but advanced cryptographic computations also eat up a lot of battery life. Studies in UMIC have shown that this problem is hardly solvable with standard processors. Therefore, the design of security algorithms and policies is deeply intertwined with the design of heterogeneous multi-processor systems on a chip that employ simple special-purpose processors, possibly running in parallel, for tasks such as cryptography, thus gaining performance and saving energy at the same time.

Energy efficiency in general is another critical factor of mobile information and communication systems which can only be reached through a concerted effort at all layers and scales. As fig. 6 indicates, the demands from generation to generation of communication technology grow dramatically. While the speed of memory (following Moore's law) has been able to follow this demand, the figure shows the beginning of limits in microprocessor technologies which leads to the need for multi-processor technologies as mentioned above, and the slow development of battery power. The latter is not a question of engineering competence; indeed, the energy density of the batteries in mobile phones, for example, has reached a level comparable to TNT or, to use another analogy, the energy level a few centimeters away from the exhaust of a starting space rocket. Therefore, other solutions have to be found.

Besides the already mentioned reduction of energy consumption by resorting to heterogeneous multiprocessors, the reduction of unnecessary listening and sending in noisy or irrelevant environments is another strategy UMIC is pursuing. For example, instead of continuously listening for information on the air, multi-dimensional temporal and spatial indexing techniques from the database area can be employed at the application level to focus on relevant information, and only seriously beginning to listen if something interesting is found on the index. Related techniques also exist at the transport platform level, e.g. scalable distributed methods for load balancing different available channels in order to minimize average noise for the end user devices and thus saving their energy.

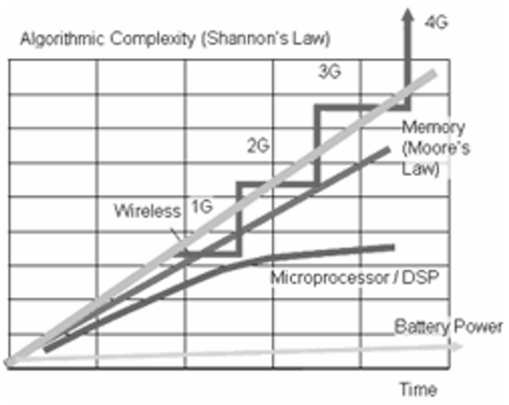


Fig. 6. Growing discrepancy between demands of generations of communication technology vs. developments in component technology and battery power

A holistic energy optimization strategy under investigation in UMIC will try to combine all these ingredients in scalable and adaptable overall architectures. Extensions will combine such approaches with mathematical, information theory research which is optimizing the distribution of base stations for wireless communications [MMV07]. Models of cost and quality of service under consideration of geographic distribution as well as e.g. three-dimensional aspects (buildings, mountains …) have already found usage in practice with several network providers. But they need to be augmented with social acceptance criteria in the non-user population (e.g. no base station very close to a kindergarden due to health impact fears) and in the user population. For example, in densely populated areas with many impediments, multi-hop networks where end user devices would co-serve as transmission stations into areas shadowed by buildings would significantly reduce the need for base stations, but it is completely unclear if this would be accepted by the users who may fear not just unexpected loss of power but also misuse e.g. for security breaks.

4 Conclusions

In this paper, we have outlined some of the user scenarios and engineering challenges the next generations of mobile information and communication systems will be faced with. We have also demonstrated with many examples that most of the challenges are interdisciplinary, and solutions require analysis and strategies from many different perspectives. As information systems designers, we shall need to interact much more than in the past with some of the technological foundations in order to create cost-effective and highly usable innovations, but we equally have to avoid the trap of getting caught by uncritical running after base technology steps which forget the user side. In the UMIC cluster, some foundational questions in this setting are being investigated while a continuous stream of demonstrators and experiments in cooperation with other researchers, users, and industry should maintain the needed reality check.

Acknowledgments. This work was supported in part by the Deutsche Forschungsgemeinschaft in Excellence Cluster UMIC and in SFB/FK 427 "Media and Cultural Communications" as well as by Microsoft Deutschland GmbH through the Center for Software Concepts at RWTH Aachen University and Fraunhofer FIT. Thanks are due to the UMIC coordinators Gerd Ascheid and Otto Spaniol, my co-area coordinators Petri Mähönen, Tobias Noll, and Stefan Kowalewski, and especially the involved members of the Information Systems group, Yiaweh Cao, Ralf Kensche, Ralf Klamma, Christoph Quix, and Satish Srirama.

References

[AHW07] Aijaz, F., Hameed, B., Walke, B.: Towards peer-to-peer long-lived mobile web services. In: 4th IEEE Intl. Conf. Innovations in Information Technology, Dubai (2007)

[AKMS07] Assent, I., Krieger, R., Müller, E., Seidl, T.: DUSC: dimensionality unbiased subspace clustering. In: Perner, P. (ed.) ICDM 2007. LNCS (LNAI), vol. 4597, pp. 409–414. Springer, Heidelberg (2007)

[BHL01] Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web, May 17, 2001. Scientific American (2001)

[Ge07] Gehlen, G.: Mobile Web Services – Concepts, Prototype, and Traffic Performance Analysis. Diss. RWTH Aachen (2007)

[JK08] Jarke, M., Klamma, R.: Reflective community information systems. In: Manolopoulos, Y., et al. (eds.) ICEIS 2006. LNBIP, vol. 3, pp. 17–28. Springer, Heidelberg (2008)

[KSJ*05] Klamma, R., Spaniol, M., Jarke, M., Cao, Y., Jansen, M., Toubekis, G.: ACIS – Intergenerational community learning supported by a hypermedia Afghan sites and monuments database. In: Proc. 5th IEEE Intl. Conf. Advanced Learning Technologies (ICALT 2005), Kaohsiung, Tw, pp. 108–112 (2005)

[MMV07] Mähönen, P., Mathar, R., Vary, P.: Kommunikationstechnik für das mobile Internet von morgen. RWTH Themen 1/2007, pp. 26–31 (2007)

[Re05] O'Reilly, T.: What isWeb 2.0? (2005), http://www.oreilly.com/, 30.9.2005

[Sp07] Spaniol, M.: Informatische Methoden zur Unterstützung von Transkription, Lokalisierung und Adressierung in kulturwissenschaftlichen Communities. Diss. RWTH Aachen University (2007)

[SJP07] Srirama, S., Jarke, M., Prinz, W.: Mobile web service mediation framework. In: Proc. MW4SOC Workshop, 8th Intl. ACM/IFIP Middleware Conf., Newport Beach, Ca (2007)

[SW05] Steinmetz, R., Wehrle, K.: Peer-to-Peer Systems and Applications. Springer, Heidelberg (2005)

[WPH06] Wirsam, W., Prinz, W., Hahnen, J.: Social aspects in communities of sports people equipped with location-aware mobile devices. In: Proc. CHI 2006 Workshop on Mobile Social Software, Montreal, Canada, pp. 22–27 (2006)

A Framework for Building Mapping Operators Resolving Structural Heterogeneities*

Gerti Kappel[1], Horst Kargl[1], Thomas Reiter[2], Werner Retschitzegger[2], Wieland Schwinger[3], Michael Strommer[1], and Manuel Wimmer[1]

[1] Institute of Software Technology and Interactive Systems
Vienna University of Technology, Austria
{kappel,wimmer,kargl,strommer}@big.tuwien.ac.at
[2] Information Systems Group
Johannes Kepler University Linz, Austria
{reiter,retschitzegger}@ifs.uni-linz.ac.at
[3] Department of Telecooperation
Johannes Kepler University Linz, Austria
wieland.schwinger@jku.ac.at

Abstract. Seamless exchange of models among different modeling tools increasingly becomes a crucial prerequisite for the success of model-driven engineering. Current best practices use model transformation languages to realize necessary mappings between concepts of the metamodels defining the modeling languages supported by different tools. Existing model transformation languages, however, lack appropriate abstraction mechanisms for resolving recurring kinds of structural heterogeneities one has to primarily cope with when creating such mappings.

We propose a framework for building reusable mapping operators which allow the automatic transformation of models. For each mapping operator, the operational semantics is specified on basis of Colored Petri Nets, providing a uniform formalism not only for representing the transformation logic together with the metamodels and the models themselves, but also for executing the transformations, thus facilitating understanding and debugging. To demonstrate the applicability of our approach, we apply the proposed framework for defining a set of mapping operators which are intended to resolve typical structural heterogeneities occurring between the core concepts usually used to define metamodels.

1 Introduction

Interoperability Between Modeling Tools. With the rise of Model-Driven Engineering (MDE) [20] models become the main artifacts of the software development process. Hence, a multitude of modeling tools is available supporting different tasks, such as model creation, model simulation, model checking, model

* This work has been partly funded by the Austrian Federal Ministry of Transport, Innovation and Technology (BMVIT) and FFG under grant FIT-IT-810806.0.

transformation, and code generation. Seamless exchange of models among different modeling tools increasingly becomes a crucial prerequisite for effective model-driven engineering. Due to lack of interoperability, however, it is often difficult to use tools in combination, thus the potential of MDE cannot be fully utilized. For achieving interoperability in terms of transparent model exchange, current best practices (cf., e.g. [21]) comprise creating model transformations based on mappings between concepts of different tool metamodels, i.e., the metamodels describing the modeling languages supported by the tools.

Problem Statement. We have followed the aforementioned approach in various projects such as the ModelCVS project [12] focusing on the interoperability between legacy case tools (in particular CA's AllFusion Gen) with UML tools and the MDWEnet project [1] trying to achieve interoperability between different tools and languages for web application modeling. The prevalent form of heterogeneity one has to cope with when creating such mappings between different metamodels is *structural heterogeneity*, a form of heterogeneity well-known in the area of database systems [3,14]. In the realm of metamodeling structural heterogeneity means that semantically similar modeling concepts are defined with different metamodeling concepts leading to differently structured metamodels. Current model transformation languages, e.g., the OMG standard QVT [17], provide no appropriate abstraction mechanisms or libraries for resolving recurring kinds of structural heterogeneities. Thus, resolving structural heterogeneities requires to manually specify partly tricky model transformations again and again which simply will not scale up having also negative influence on understanding the transformation's execution and on debugging.

Contribution. The contribution of this paper is twofold. First, a framework is proposed for building reusable mapping operators which are used to define so-called metamodel bridges. Such a metamodel bridge allows the automatic transformation of models since for each mapping operator the operational semantics is specified on basis of Colored Petri Nets. Colored Petri Nets provide a uniform formalism not only for representing the transformation logic together with the metamodels and the models themselves, but also for executing the transformations, thus facilitating understanding and debugging. Second, to demonstrate the applicability of our approach we apply the proposed framework for defining a set of mapping operators subsumed in our mapping language called CAR. This mapping language is intended to resolve typical structural heterogeneities occurring between the core concepts usually used to define metamodels, i.e., class, attribute, and reference, as provided by the OMG standard MOF [16].

Structure. The rest of the paper is structured as follows. In Section 2 we introduce our framework for defining mapping operators in order to establish metamodel bridges. In Section 3 the mapping language CAR is presented. Section 4 discusses related work, and finally, in Section 5 a conclusion and a discussion of future work is given.

2 Metamodel Bridging at a Glance

In this section, we describe the conceptual architecture of the proposed *Metamodel Bridging Framework* in a by-example manner. The proposed framework provides two views on the metamodel bridge, namely a *mapping view* and a *transformation view* as illustrated in Figure 1.

At the mapping view level, the user defines mappings between elements of two metamodels (M2). Thereby a mapping expresses also a relationship between model elements, i.e., the instances of the metamodels [4]. In our approach, we define these mappings between metamodel elements with mapping operators standing for a processing entity encapsulating a certain kind of transformation logic. A mapping operator takes as input elements of the source model and produces as output semantically equivalent elements of the target model. Thus, it declaratively describes the semantic correspondences on a high-level of abstraction. A set of applied mapping operators defines the mapping from a left hand side (LHS) metamodel to a right hand side (RHS) metamodel further on subsumed as *mapping model*.

For actually exchanging models between different tools, the mapping models have to be executed. Therefore, we propose, in addition to the mapping view, a transformation view which is capable of transforming models (M1) from the LHS to the RHS on basis of Colored Petri Nets [6].

2.1 The Mapping View

For defining mapping operators and consequently also for building mapping models, we are using a subset of the UML 2 component diagram concepts. With this formalism, each mapping operator can be defined as a dedicated component, representing a modular part of the mapping model which encapsulates an arbitrary complex structure and behavior, providing well-defined interfaces to the environment. The resulting components are collected in a mapping operator library which can be seen as a domain-specific language for bridging metamodels. The user can apply the mapping operators expressed as components in a plug&play manner, i.e., only the connections to the provided and required interfaces have to be established manually.

Our motivation for using UML 2 component diagrams for the mapping view is the following. First, many software engineers are likely to be familiar with the UML component diagram notation. Second, the provided and required interfaces which can be typed, enable the composition of mapping operators to resolve more complex structural heterogeneities. Third, the clear separation between *black-box* view and *white-box* view of components allows switching between a high-level mapping view and a detailed transformation view, covering the operational semantics, i.e., the transformation logic, of an operator.

Anatomy of a Mapping Operator. Each mapping operator (as for example shown in the mapping model of Figure 1) has *input ports* with required interfaces (left side of the component) as well as *output ports* with provided interfaces (right side of the component). Because each mapping operator has its own *trace model*,

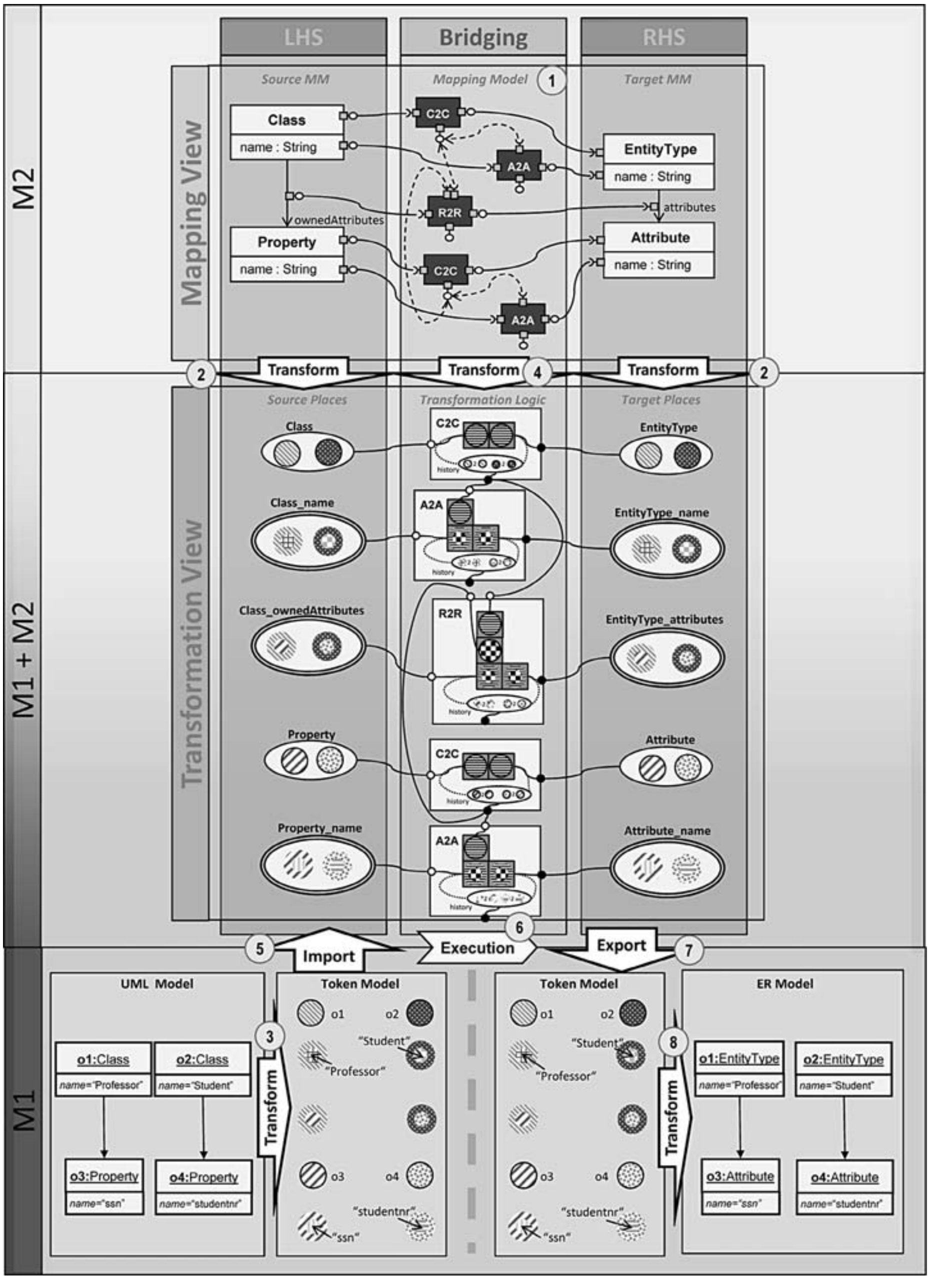


Fig. 1. Metamodel Bridging Framework by-example

i.e., providing a log about which output elements have been produced from which input elements, an additional *providedContext port* with a corresponding interface is available on the bottom of each mapping operator. This port can be used by other operators to access the trace information for a specific element via *requiredContext ports* with corresponding interfaces on top of the operator.

In the mapping view of Figure 1 (cf. step 1), an example is illustrated where a small part of the metamodel of the UML class diagram (cf. source metamodel) is mapped to a part of the metamodel of the Entity Relationship diagram (cf. target metamodel). In the mapping view, source metamodel elements have provided interfaces and target metamodel elements have required interfaces. This is due to the fact that in our scenario, models of the LHS are already available whereas models of the RHS must be created by the transformation, i.e., the elements of the LHS must be streamed to the RHS according to the mapping operators. Consequently, *Class* and *Property* of the source metamodel are mapped to *EntityType* and *Attribute* of the target metamodel with *Class2Class* (*C2C*) operators, respectively. In addition, the C2C operator owns a providedContext port on the bottom of the component which shall be used by the requiredContext ports of the appropriate *Attribute2Attribute* (*A2A*) and *Reference2Reference* (*R2R*) operators to preserve validity of target models. In particular, with this mechanism it can be ensured that values of attributes are not transformed before their owning objects has been transformed and links as instances of references are not transformed before the corresponding source and target objects have been transformed.

2.2 The Transformation View

The transformation view is capable of executing the defined mapping models. For this, so called *transformation nets* [18] are used which are a special kind of Colored Petri Nets consisting of source places at the LHS and target places at the RHS. Transitions between the source and target places describe the transformation logic located in the bridging part of the *transformation net* as shown in Figure 1.

Transformation nets provide a suitable formalism to represent the operational semantics of the mapping operators, i.e., the transformation logic defined in the white-box view of the component due to several reasons. First, they enable the execution of the transformation thereby generating the target model out of the source model, which favors also debugging of a mapping model. Second, it allows a homogeneous representation of all artefacts involved in a model transformation (i.e., models, metamodels, and transformation logic) by means of a simple formalism, thus being especially suited for gaining an understanding of the intricacies of a specific metamodel bridge.

In the next paragraphs, we discuss rules for assembling metamodels, models, and mapping models into a single transformation net and how the transformation can actually be executed.

Places Represent Metamodels. First of all, places of a transformation net are used to represent the elements of the source and target metamodels (cf. step 2 in

Figure 1). In this respect, we currently focus on the three major building blocks of metamodels (provided, e.g. by meta-metamodels such as MOF), namely *class*, *attribute*, and *reference*. In particular, classes are mapped onto one-colored places whereby the name of the class becomes the name of the place. The notation used to visually represent one-colored places is a circle or oval as traditionally used in Petri Nets. Attributes and references are represented by two-colored places, whereby the name of the containing class plus the name of the reference or of the attribute separated by an underline becomes the name of the place (cf. e.g. *Class_name* and *Class_ownedAttributes* in Figure 1). To indicate that these places contain two-colored tokens, the border of two-colored places is double-lined.

Tokens Represent Models. The tokens of the transformation net are used to represent the source model which should be transformed according to the mapping model. Each element of the source model is expressed by a certain token, using its color as a means to represent the model element's identity in terms of a String (cf. step 3 in Figure 1). In particular, for every object, a one-colored token is produced, whereby for every link as an instance of a reference, as well as for every value of an attribute, a two-colored token is produced. The *fromColor* for both tokens refers to the color of the token that corresponds with the containing object. The *toColor* is given by the color of the token that corresponds with the referenced target object or the primitive value, respectively. Notationally, a two-colored token consist of a ring (carrying the fromColor) surrounding an inner circle (depicting the toColor).

Considering our example shown in Figure 1, the objects *o1* to *o4* of the UML model shown in the M1-layer are transformed into one-colored tokens. Each one-colored token represents an object identity, pointed out by the object name beneath the token. E.g., the tokens with the inner-color *"Student"* and *"Professor"* have the same outer-color as their containing objects and the token which represents the link between object *o1* and *o3* has the same outer-color as the token representing object *o1* and the inner-color corresponds to the one-colored token representing object *o3*.

Transitions Represent Mapping Models. The mapping model is expressed by the transformation logic of the transformation net connecting the source and the target places (cf. Step 4 in Figure 1). In particular, the operational semantics of the mapping operators are described with transitions, whereby the behavior of a transition is described with the help of preconditions called *query-tokens* (LHS of a transition) and postconditions called *generator-tokens* (RHS of a transition). Query-tokens and generator-tokens can be seen as templates, simply visualized as color patterns, describing a certain configuration of tokens. The pre-condition is fulfilled and the transitions fires, if the specified color pattern described by the query-tokens matches a configuration of available input tokens. In this case, the postcondition in terms of the generator-tokens produces the required output tokens representing in fact the necessary target model concepts.

In the following, the most simple mapping operators used in our example are described, namely C2C, A2A, and R2R.

C2C. The white-box view of the C2C operators as shown in the transformation view of Figure 1 ensures that each object instantiated from the class connected to the input port is streamed into the mapping operator, the transition matches a single token from the input port, and streams the exact token to the output port. This is expressed in the transition by using the most basic query-token and generator-token combination, both having the same color pattern. In addition, every input and output token combination is saved in a history place representing the trace model which is connected to the *providedContext* port and can be used as trace information by other operators.

A2A. The white-box view of the A2A operator is also illustrated in the bridging part of the transformation view in Figure 1. Two-colored tokens representing attribute values are streamed via the input port into the mapping operator. However, a two-colored token is only streamed to the output port if the owning object of the value has been already transformed by a C2C operator. This is ensured in that the transition uses the same color pattern for the one-colored query-token representing the owning object streamed from the *requiredContext* port and for the outer color of the two-valued query-token representing the containing object of the attribute value. Only, if a token configuration matches this pre-condition, the two-colored token is streamed via the generator-token to the output port. Again, the input tokens and the corresponding output tokens are stored in a history place which is connected to the *providedContext* port.

R2R. The white-box view of the R2R operator shown in the transformation view of Figure 1 consists of three query-tokens, one two-colored query-token representing the link and two one-colored query-tokens for accessing trace information from C2C operators. The two-colored query-token must have the same inner and outer colors as provided by the C2C trace information, i.e., the source and target objects must be already transformed. When this precondition is satisfied by a token configuration, the two-colored token representing the link is streamed via the generator-token to the output port.

Execution of the Transformation Logic. As soon as the metamodels are represented as places, which are furthermore marked with the respective colored tokens depicting the concepts of the source model (cf. step 5 in Figure 1), the transformation net can be started. Now, tokens are streamed from the source places over the transitions into the target places (cf. step 6 in Figure 1).

Considering our running example, in a first step only the transitions of the C2C operators are able to fire due to the dependencies of the A2A and R2R operators. Hence, tokens from the places *Class* and *Property* are streamed to the appropriate places of the RHS and all combinations of the queried input and generated output tokens are stored in the trace model of the C2C operator. As soon as all necessary tokens are available in the trace model, depending operators, i.e., the A2A and R2R operators, are also able to fire.

Generation of the Target Model. After finishing the transformation, the tokens from the target places can be exported (cf. step 7 in Figure 1) and transformed back into instances of the RHS metamodel (cf. step 8 in Figure 1).

In our example, the one-colored tokens *o1* to *o4* contained in the target places are transformed back into objects of type *EntityType* and *Attribute*. The two-colored tokens which represent attribute values, e.g., *"Professor"* and *"Student"*, are assigned to their containing objects, e.g., *o1* and *o2* whereas *"ssn"* and *"studentnr"* are assigned to *o3* and *o4*. Finally, the two-colored tokens which represent links between objects are transformed back into links between *o1* and *o3*, as well as between *o2* and *o4*.

3 Mapping Operators of the CAR Mapping Language

3.1 Motivating Example

Based on experiences gained in various interoperability projects [9,10,23,24] it has been shown that although most meta-metamodels such as MOF offer only a core set of language concepts for defining metamodels, numerous structural heterogeneities occur when defining modeling languages.

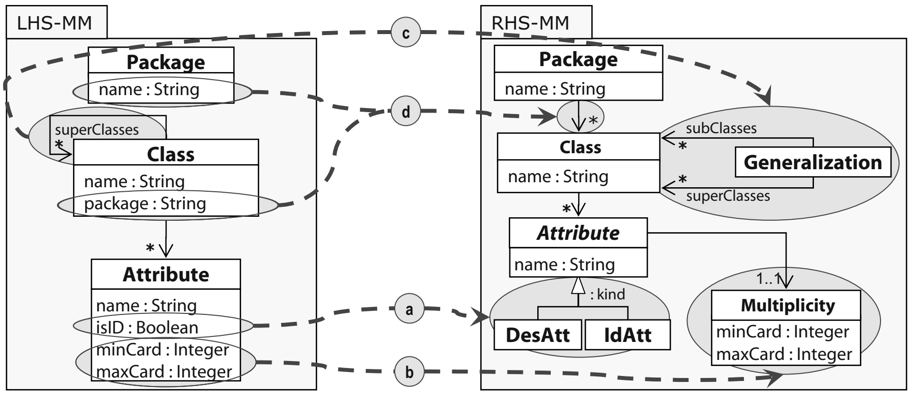
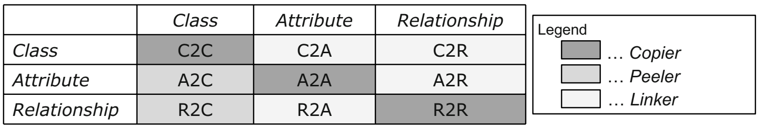


Fig. 2. Structural Heterogeneities Between Metamodels - Example

As an example for structural metamodel heterogeneity consider the example shown in Figure 2. Two MOF-based metamodels represent semantically equivalent core concepts of the UML class diagram in different ways. Whereas the LHS metamodel uses only a small set of classes, the RHS metamodel employs a much larger set of classes thereby representing most of the UML concepts which are in the LHS metamodel implicitly defined as attributes or references explicitly as first class citizens. More specifically, four structural metamodel heterogeneities can be found which require mapping operators going beyond the simple *one-to-one* mappings provided by the mapping operators in Section 2.

3.2 CAR Mapping Language at a Glance

At this time, we provide nine different core mapping operators for resolving structural metamodel heterogeneities as depicted in Figure 2. These nine mapping operators result from the possible combinations between the core concepts of meta-metamodels, namely *class*, *attribute*, and *reference*, which also led to the name of the CAR mapping language. These mapping operators are designed to be declarative and bi-directional and it is possible to derive executable transformations based on transformation nets. One important requirement for the CAR mapping language is that it should be possible to reconstruct the source models from the generated target models, i.e., any loss of information during transformation should be prevented. In Figure 3, the mapping operators are divided according to their functionality into the categories *Copier*, *Peeler*, and *Linker* which are explained in the following.



	Class	*Attribute*	*Relationship*
Class	C2C	C2A	C2R
Attribute	A2C	A2A	A2R
Relationship	R2C	R2A	R2R

Legend
- ■ ... *Copier*
- ▨ ... *Peeler*
- □ ... *Linker*

Fig. 3. CAR Mapping Operators

Copier. The diagonal of the matrix in Figure 3 depicts the symmetric mapping operators of the CAR mapping language which have been already discussed in Section 2. The term symmetric means that the input and outport ports of the left side and the right side of the mapping operators are of the same type. This category is called *Copier*, because these mapping operators copy one element of the LHS model into the RHS model without any further manipulations.

Peeler. This category consists of mapping operators which create new objects by "peeling"[1] them out of values or links. The A2C operator bridges heterogeneities which are resulting from the fact that a concept is expressed as an attribute in one metamodel and as a class in another metamodel. Analogously, a concept can be expressed on the LHS as a reference and on the RHS as a class which can be bridged by a R2C operator.

Linker. The last category consists of mapping operators which either link two objects to each other out of value-based relationships (cf. A2R and R2A operator) or assign values or links to objects for providing the inverse variants of the A2C and R2C operators (cf. C2A and C2R operator).

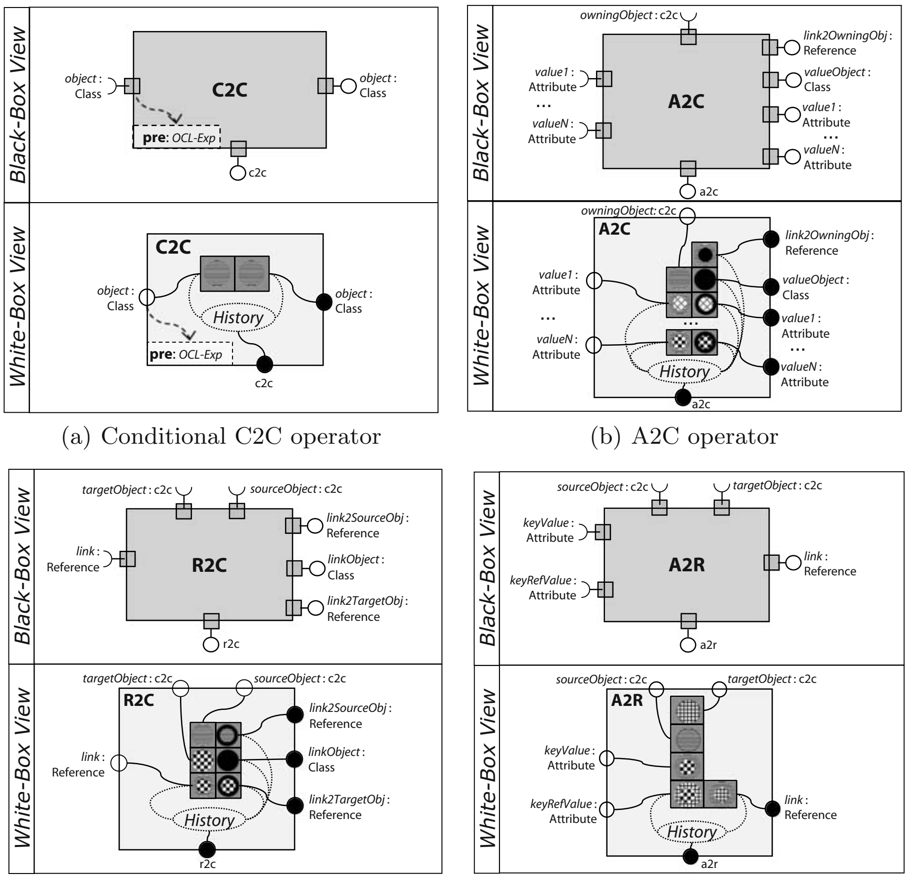
To resolve the structural heterogeneities depicted in Figure 2, in the following subsections the necessary mapping operators are discussed in detail, comprising

[1] Note that the term "peeling" is used since when looking at the white-box view the transformation of an attribute value into an object requires in fact to generate a one-colored token out of a two-colored token.

besides a variation of the C2C operator mainly mapping operators falling into the above mentioned peeler and linker category.

3.3 Conditional C2C Mapping Operator

Problem. In MOF-based metamodels, a property of a modeling concept can be expressed via a discriminator of an inheritance branch or with an additional attribute. An example for this kind of heterogeneity can be found in Figure 2(a), namely between *Attribute.isID* on the LHS and the subclasses of the class *Attribute* on the RHS. This heterogeneity is not resolvable with a primitive C2C operator per se, because one class on the LHS corresponds to several classes on the RHS whereby each mapping is only valid under a certain condition. On



(a) Conditional C2C operator

(b) A2C operator

(c) R2C operator

(d) A2R operator

Fig. 4. Black-box and white-box views of CAR mapping operators

the model level, this means that a set of objects has to be splitted into several subsets based on the object's attribute values.

Solution. To cope with this kind of heterogeneity, the C2C operator has to be extended with the capability of splitting a set of objects into several subsets. For this we are annotating the C2C operator with OCL-based preconditions assigned to ports as depicted in Figure 4(a). These preconditions supplement the query-tokens of the transitions by additionally allowing to specify constraints on the source model elements. The reason for introducing this additional mechanism is that the user should be able to configure the C2C operator without having to looking into the white-box view of the operator, realizing its basic functionality.

Example Application. In the example shown in Figure 5, we can apply two C2C mapping operators with OCL conditions, one for mapping *Attribute* to *DesAtt* with the precondition *Attribute.isID = false*, and one for mapping *Attribute* to *IdAtt* with the precondition *Attribute.isID = true*. In addition, this example shows a way how mappings can be reused within a mapping model by allowing inheritance between mappings. This mechanism allows to define certain mappings directly between *superClasses* and not for each *subClass* combination again and again (cf., e.g., the A2A mapping between the *Attribute.name* attributes). For more details about the inheritance mechanism, the interested reader is kindly referred to [11].

3.4 A2C Mapping Operator

Problem. In Figure 5(b), the attributes *minCard* and *maxCard*, which are part of the class *Attribute* at the LHS, are at the RHS part of a dedicated class *Multiplicity*. Therefore, on the instance level, a mechanism is needed to "peel" objects out of attribute values and to additionally take into account the structure of the LHS model in terms of the attribute's owning class when building the RHS model, i.e., instances of the class *Multiplicity* must be connected the corresponding instances of class *Attribute*.

Solution. The black-box view of the A2C mapping operator as illustrated in Figure 4(b) consists of one or more required interfaces for attributes on the LHS depending on how many attributes are contained by the additional class, and has in minimum three provided interfaces on the RHS. The first of these interfaces is used to mark the reference which is responsible to link the two target classes, the second is used to mark the class that should be instantiated, and the rest is used to link the attributes of the LHS to the RHS. Additionally, an A2C operator has a required interface to a C2C, because the source object is splitted into two target objects, thereby only one object is created by the A2C, the other has to be generated by an C2C operator which maps the LHS class to its corresponding target RHS class.

The white-box view of the A2C operator shown in Figure 4(b) comprises a transition consisting of at least two query-tokens. The first query-token guarantees that the *owningObject* has been already transformed by a C2C operator. The other query-tokens are two-colored tokens representing the attribute values which have as fromColor the same color as the first query-token. The post-condition of the transition consists of at least three generator-tokens. The second generator-token introduces a new color, i.e., this color is not used in the pre-condition part of the transition, and therefore, the generator-token produces a new object with an unique identity. The first generator-token is used for linking the newly created object appropriately into the target model and the other two-colored generator tokens are used to stream the values into the newly generated object by changing the fromColor of the input values.

Example Application. In Figure 5, the attributes *minCard* and *maxCard* are mapped to attributes of the class *Multiplicity*. Furthermore, the reference between the classes *Attribute* and *Multiplicity* is marked by the A2C mapping as well as the class *Multiplicity*. To assure that the generated *Multiplicity* objects can be properly linked to *Attribute* objects, the A2C mapping is in the context of the C2C mapping between the *Attribute* classes.

3.5 R2C Mapping Operator

Problem. In Figure 5(c), the reference *superClasses* of the LHS metamodel corresponds to the class *Generalization* of the RHS metamodel. This kind of heterogeneity requires an operator which is capable of "peeling" an object out of a link and to additionally preserve the structure of the LHS in terms of the classes connected by the relationships at the RHS.

Solution. The black-box view of the R2C mapping operator, as depicted in Figure 4(c), has one required interface on the left side for pointing to a reference. On the right side it has three provided interfaces, one for the class which stands for the concept expressed as reference on the LHS and two for selecting the references which are responsible to connect the object which has been peeled out of the link of the LHS into the RHS model. To determine the objects to which the peeled object should be linked, two additional required interfaces on the top of the R2C operator are needed for determining the corresponding objects of the source and target objects of the LHS.

The white-box view of the R2C mapping operator, as illustrated in Figure 4(c), consists of a pre-condition comprising three query-tokens. The input link is connected to a two-colored query-token, the fromColor corresponds to the query-token standing for the source object and the toColor corresponds to a query-token standing for the target object. The post-condition of the transition introduces a new color and is therefore responsible to generate a new object. Furthermore, two links are produced by the other generator-tokens for linking the newly generated object with the corresponding source and target the objects of the LHS.

Example Application. In Figure 5, the reference *superClasses* in the LHS metamodel is mapped to the class *Generalization* by an R2C operator. In addition, the references *subClasses* and *superClasses* are selected for establishing an equivalent structure on the RHS as existing on the LHS. For actually determining the *Class* objects which should be connected via *Generalization* objects, the R2C operator has two dependencies to C2C mappings. This example can be seen as a special case, because the reference *superClasses* is a reflexive reference, therefore both requiredContext ports of the R2C operator point to the same C2C operator.

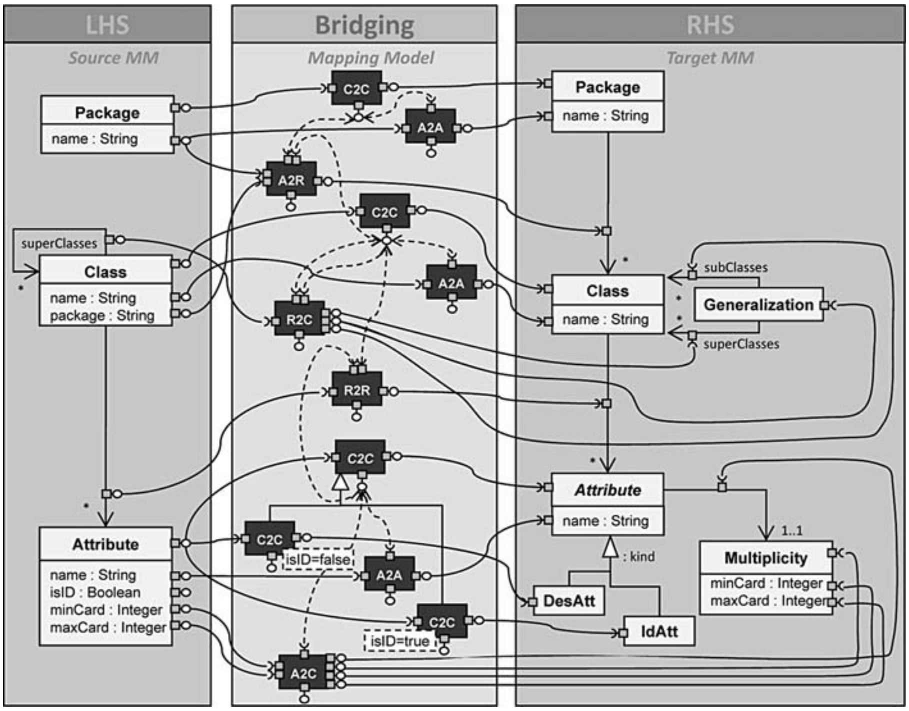


Fig. 5. Example resolved with CAR (Mapping View)

3.6 A2R Mapping Operator

Problem. The attribute vs. reference heterogeneity shown in Figure 2(d) resembles the well-known difference between value-based and reference-based relationships, i.e, corresponding attribute values in two objects can be used to "simulate" links between two objects. Hence, if the attribute values in two objects are equal, a link ought to be established between them.

Solution. For bridging the value-based vs. reference-based relationship heterogeneity, the A2R mapping operator as shown in Figure 4(d) provides on the LHS two interfaces, one for marking the *keyValue* attribute and another for marking the *keyRefValue* attribute. On the RHS, the operator provides only one interface for marking the reference which corresponds to the *keyValue/keyRefValue* attribute combination.

The white-box view of the operator comprises a transition which has four query-tokens. The first two ensure that the objects which are referencing each other on the LHS have been already transformed. The last two are the *keyValue* and *keyRefValue* query-tokens whereby the inner-color (representing the attribute values) is the same for both tokens. The generator-token of the transition produces one two-colored token by using the outer-color of the *keyRefValue* query-token as the outer-color and the outer-color of the *keyValue* query-token as the inner-color.

Example Application. In Figure 5, the A2R operator is used to map the *Package.name* attribute as the key attribute and the *Class.package* attribute as the keyRef attribute of the LHS metamodel to the reference between *Package* and *Class* on the RHS metamodel.

4 Related Work

With respect to our approach we can distinguish between two kinds of related work: first, related work in the field of model-driven engineering concerning the design of reusable model transformations, and second, related work in the field of ontology engineering concerning the usage of dedicated mapping languages for bridging structural heterogeneities.

4.1 Reusable Model Transformations

Generic Model Transformations. Typically model transformation languages, e.g., ATL [7] and QVT [17], allow to define transformation rules based on types defined as classes in the corresponding metamodels. Consequently, model transformations are not reusable and must be defined from scratch again and again with each transformation specification. One exception thereof is the approach of Varró et al. [22] who define a notion of defining generic transformations within their VIATRA2 framework, which in fact resembles the concept of templates in C++ or generics in Java. VIATRA2 also provides a way to implement reusable model transformations, although it does not foster an easy to debug execution model as is the case with our proposed transformation nets. In addition, there exists no explicit mapping model between source and target metamodel which makes it cumbersome to reconstruct the correspondences between the metamodel elements based on the graph transformation rules, only.

Transformation Patterns. Very similar to the idea of generic transformations is the definition of reusable idioms and design patterns for transformation rules described by Karsai et al. [2]. Instead of claiming to have generic model transformations, the authors propose the documentation and description of recurring problems in a general way. Thus, this approach solely targets the documentation of transformation patterns. Implementation issues how these patterns could be implemented in a generic way remain open.

Mappings for Bridging Metamodels. Another way of reuse can be achieved by the abstraction from model transformations to mappings as is done in our approach or by the ATLAS Model Weaver (AMW) [5]. AMW lets the user extend a generic so-called weaving metamodel, which allows the definition of simple correspondences between two metamodels. Through the extension of the weaving metamodel, one can define the abstract syntax of new weaving operators which roughly correspond to our mapping operators. The semantics of weaving operators are determined by a higher-order transformation that take a weaving model as input and generates model transformation code. Compared to our approach, the weaving models are compiled into low-level transformation code in terms of ATL which is in fact a mixture of declarative and imperative language constructs. Thus, it is difficult to debug a weaving model in terms of weaving operators, because they do not explicitly remain in the model transformation code. Furthermore, the abstraction of mapping operators from model transformations expressed in ATL seems more challenging compared to the abstraction from our proposed transformation net components.

4.2 Ontology Mapping for Bridging Structural Heterogeneities

In field of ontology engineering, several approaches exist which make use of high-level languages for defining mappings between ontologies (cf. [8] for an overview). For example, in Maedche et al. [15], a framework called *MAFRA* for mapping two heterogeneous ontologies is proposed. Within this framework, the mapping ontology called *Semantic Bridge Ontology* usually provides different ways of linking concepts from the source ontology to the target ontology. In addition to the Semantic Bridge Ontology, MAFRA provides an execution platform for the defined mappings based on services whereby for each semantic bridge type a specific service is available for executing the applied bridges. In [19], Scharffe et al. describe a library of so called *Ontology Mediation Patterns* which can be seen as a library of mapping patterns for integrating ontologies. Furthermore, the authors provide a mapping language which incorporates the established mapping patterns and they discuss useful tool support around the pattern library, e.g., for transforming ontology instances between different ontology schemas.

The main difference to our approach is that ontology mapping approaches are based on Semantic Web standards, such as *OWL* and *RDFS*, and therefore contain mapping operators for typical description logic related mapping problems, e.g., *union* or *intersection* of classes. We are bridging metamodels expressed in MOF, a language which has only a partially overlap with OWL or RDFS, leading to different mapping problems. Furthermore, in contrast to the ontology mapping frameworks, we provide a framework allowing to build new mapping operators by using well-known modeling techniques not only for defining the syntax but also for the operational semantics of the operators.

5 Conclusion and Future Work

In this paper we have introduced a framework allowing the definition of mapping operators and their application for building metamodel bridges. Metamodel bridges are defined by the user on a high-level mapping view which represents the semantic correspondences between metamodel elements and are tested and executed on a more detailed transformation view which also comprises the transformation logic of the mapping operators. The close integration of these two views and the usage of models during the whole bridging process further enhances understandability and the debugging of the defined mappings in terms of the mapping operators. The applicability of the framework has been demonstrated by implementing mapping operators for resolving common structural metamodel heterogeneities.

Future work will focus on making current matching engines for automatically finding correspondences between metamodels aware of the CAR mapping operators in combination with a supervised machine learning approach [13].

References

1. Vallecillo, A., et al.: MDWEnet: A Practical Approach to Achieving Interoperability of Model-Driven Web Engineering Methods. In: Baresi, L., Fraternali, P., Houben, G.-J. (eds.) ICWE 2007. LNCS, vol. 4607, Springer, Heidelberg (2007)
2. Agrawal, A., Vizhanyo, A., Kalmar, Z., Shi, F., Narayanan, A., Karsai, G.: Reusable Idioms and Patterns in Graph Transformation Languages. In: Proc. of the Int. Workshop on Graph-Based Tools (GraBaTs 2004), Italy (2004)
3. Batini, C., Lenzerini, M., Navathe, S.B.: A Comparative Analysis of Methodologies for Database Schema Integration. ACM Comput. Surv. 18(4), 323–364 (1986)
4. Bernstein, P.A., Melnik, S.: Model management 2.0: manipulating richer mappings. In: Proc. of the ACM SIGMOD Int. Conf. on Management of Data, China (2007)
5. Fabro, M.D.D., Bézivin, J., Jouault, F., Breton, E., Gueltas, G.: AMW: a generic model weaver. In: Proc. of the 1ère Journée sur l'Ingénierie Dirigée par les Modèles (IDM 2005), France (2005)
6. Jensen, K.: Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use. Springer, Heidelberg (1992)
7. Jouault, F., Kurtev, I.: Transforming Models with ATL. In: Bruel, J.-M. (ed.) MoDELS 2005. LNCS, vol. 3844, Springer, Heidelberg (2006)
8. Kalfoglou, Y., Schorlemmer, W.M.: Ontology Mapping: The State of the Art. In: Dagstuhl Seminar Proceedings: Semantic Interoperability and Integration (2005)
9. Kappel, G., Kapsammer, E., Kargl, H., Kramler, G., Reiter, T., Retschitzegger, W., Schwinger, W., Wimmer, M.: Lifting Metamodels to Ontologies - A Step to the Semantic Integration of Modeling Languages. In: Nierstrasz, O., Whittle, J., Harel, D., Reggio, G. (eds.) MoDELS 2006. LNCS, vol. 4199, Springer, Heidelberg (2006)
10. Kappel, G., Kargl, H., Kramler, G., Schauerhuber, A., Seidl, M., Strommer, M., Wimmer, M.: Matching Metamodels with Semantic Systems - An Experience Report. In: Workshop Proc. of Datenbanksysteme in Business, Technologie und Web (BTW 2007), Germany (2007)
11. Kappel, G., Kargl, H., Reiter, T., Retschitzegger, W., Schwinger, W., Strommer, M., Wimmer, M.: A Framework for Buidling Mapping Operators Resolving Structural Heterogeneities - Extended Version. Technical report, Vienna University of Technology (2008)

12. Kapsammer, E., Kargl, H., Kramler, G., Kappel, G., Reiter, T., Retschitzegger, W., Schwinger, W., Wimmer, M.: On Models and Ontologies - A Semantic Infrastructure Supporting Model Integration. In: Proc. of Modellierung 2006, Austria (2006)
13. Kargl, H., Wimmer, M.: SmartMatcher - How Examples and a Dedicated Mapping Language can Improve the Quality of Automatic Matching Approaches. In: 1st. Int. Workshop on Ontology Alignment and Visualization (OnAV 2008), Spain (2008)
14. Kashyap, V., Sheth, A.P.: Semantic and Schematic Similarities Between Database Objects: A Context-Based Approach. VLDB J. 5(4), 276–304 (1996)
15. Maedche, A., Motik, B., Silva, N., Volz, R.: MAFRA - A MApping FRAmework for Distributed Ontologies. In: Gómez-Pérez, A., Benjamins, V.R. (eds.) EKAW 2002. LNCS (LNAI), vol. 2473, Springer, Heidelberg (2002)
16. OMG. Meta Object Facility (MOF) 2.0 Core Specification (2004), `http://www.omg.org/docs/ptc/03-10-04.pdf`
17. OMG. MOF 2.0 Query/View/Transformation Specification (2007), `http://www.omg.org/docs/ptc/07-07-07.pdf`
18. Reiter, T., Wimmer, M., Kargl, H.: Towards a runtime model based on colored Petri-nets for the execution of model transformations. In: 3rd Workshop on Models and Aspects, Germany (2007)
19. Scharffe, F., de Bruijn, J.: A language to specify mappings between ontologies. In: Proc. of the 1st Int. Conf. on Signal-Image Technology & Internet-Based Systems (SITIS 2005) (2005)
20. Schmidt, D.C.: Guest editor's introduction: Model-driven engineering. IEEE Computer 39(2), 25–31 (2006)
21. Tratt, L.: Model transformations and tool integration. Software and System Modeling 4(2), 112–122 (2005)
22. Varró, D., Pataricza, A.: Generic and Meta-transformations for Model Transformation Engineering. In: Baar, T., Strohmeier, A., Moreira, A., Mellor, S.J. (eds.) UML 2004. LNCS, vol. 3273, Springer, Heidelberg (2004)
23. Wimmer, M., Schauerhuber, A., Schwinger, W., Kargl, H.: On the Integration of Web Modeling Languages: Preliminary Results and Future Challenges. In: Baresi, L., Fraternali, P., Houben, G.-J. (eds.) ICWE 2007. LNCS, vol. 4607, Springer, Heidelberg (2007)
24. Wimmer, M., Schauerhuber, A., Strommer, M., Schwinger, W., Kappel, G.: A Semi-automatic Approach for Bridging DSLs with UML. In: Workshop Proc. of 7th OOPSLA Workshop on Domain-Specific Modeling (DSM 2007), Canada (2007)

Metamodeling: Some Application Areas in Information Systems

Dimitris Karagiannis, Hans-Georg Fill, Peter Höfferer, and Martin Nemetz

University of Vienna, Department of Knowledge Engineering, Brünner Straße 72, A-1210 Vienna, Austria
{dk,hg,ph,mn}@dke.univie.ac.at
http://www.dke.univie.ac.at

Abstract. Metamodeling is a powerful concept in the area of information systems that can be applied to solve a variety of tasks. The goal of the paper at hand is to provide an insight into these application areas. In order to do so first the basic thoughts behind metamodeling are introduced. Then it will be shown that there are applications in the context of the *creation* and *composition* of metamodels that have to be supported by "cross-sectional" aspects like visualization and simulation. Finally, we will describe some of these applications in more detail.

Keywords: Metamodeling, intellectual capital management, semantic interoperability, visualization.

1 Introduction

This paper is going to provide insight into some application areas of metamodeling. The best way to realize such an intention is to start with a discussion of the meaning of the main term. When considering the term "metamodeling" we see that it consists of two parts: "meta" and "modeling". Let's begin with a deeper look into the second one.

"Modeling" basically denotes the process of creating models. Models in computer science, in turn, are seen as *"a representation of either reality or vision".* ([30], p. 187). Therefore, they describe certain subjects under consideration either as they are or as they should be. Of course, this representation is not able to include all aspects of the original but can only focus on some of them (property of reduction) and a model is always intended for a specific purpose (property of pragmatics) [26].

Models can be classified according to the language that is used for their creation. Non-linguistic or iconic models use signs and symbols that have an apparent similarity to the concepts of the real world that are being modeled. Linguistic models on the other hand use basic primitives (i.e. signs, characters, numbers, ...) that do not have any apparent relationship with the part of reality being modeled except the one that is defined in an explicit way [29]. Nearly all models

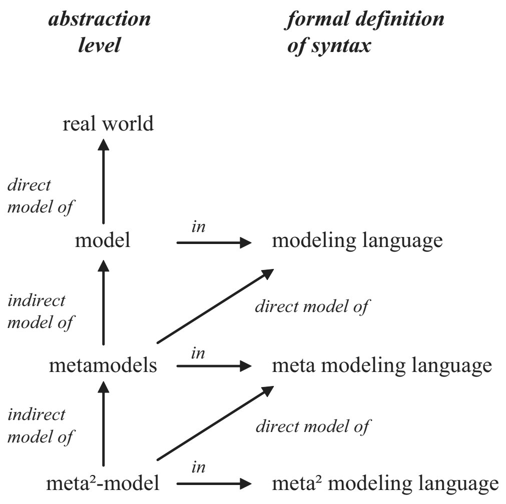


Fig. 1. Metamodeling hierarchy (adapted from [29])

used in computer science are of the latter linguistic type[1] on which we restrain ourselves hereafter.

Now, we can turn to the second part of "metamodeling" which is the "meta"-prefix that comes from Greek and according to Webster's Dictionary literally means either "between", "with", "after", or "akin to". One of the actual usages of this prefix defines a kind of "beyond"-relationship. This also fits the term "metamodel" as a metamodel can be seen beyond other models in that it is a *"a model of models"* [20]. It is important to understand this statement in a correct way where the following example might help. Consider the famous Mona Lisa who shall be our subject under consideration now. The painting by Leonardo Da Vinci that can be found in the Louvre in Paris is an iconic model of Mona Lisa. Now consider a poster reproduction of this painting that can be bought in the museum shop. This poster again is a model, or to be more precise a model of a model (the painting). But it is not a model of model*s* (please notice the plural). For something to be a metamodel it has to describe a set of other models which has to be understood as providing the means for the creation of other models. Put in other words this implies that *"A metamodel is a model of a modelling language"* [4].

A graphical representation similar to figure 1 is typically used to explain this in more detail while introducing the so-called *metamodeling hierarchy*. On the top layer there is the subject under consideration that shall be modeled. This is done with the help of a modeling language. For instance, when creating a database for let's say the management of student data we can use the Entity-Relationship modeling technique (ERM, [2]) in order to abstract reality. The available modeling primitives of the modeling language (in the case of ERM:

[1] Linguistic models can be further distinguished in being realized with textual or graphical/diagrammatic languages [12].

entities, weak entities, relationships with different cardinalities) are described in the metamodel. The metamodel is also generated using another language, the meta modeling language which primitives in turn can be defined by another meta layer which is called meta-meta-layer or meta^2-layer containing a meta^2-model using a meta^2 modeling language. Theoretically, this process can be carried on to the n-th level but for practical reasons it is stopped most often at the meta^2-layer.

Research work concerning metamodeling can focus on different aspects. First *"conceptual" formalizations* of metamodels concerning their syntax and semantics – i.e. their actual representation using logical rules, graphical representations, or the like – can be explored. Next the role of metamodels in terms of theoretical applications can be elaborated whereas we distinguish the two basic options of *"create"* and *"compose"*. The two aspects of formalization and possible application areas allow for solving concrete tasks in certain domains which we refer to as "metamodel domain solutions" that finally can be instantiated in order to deal with real world problems.

In the following we are not going into any more detail concerning the "conceptual" formalizations of metamodeling. We are rather focusing on the application areas that we are discussing in section 2. Subsequently, some of these areas are described in more detail in section 3.

2 Application Areas of Metamodeling

In Summer 2006 we did a literature survey concerning existing work in the area of metamodeling. In order to do so we examined 77 papers from 18 well known journals in different communities of computer science that is to say software engineering, databases, knowledge engineering, and information systems. For a complete list of the reviewed journals and more information on the search process please refer to [14].

When we started our literature survey we wondered whether we could find different typical metamodel applications for different areas of computer science. That would be to assume that, for instance, the knowledge engineering community is using metamodels in one special way whereas let's say the software engineering community is making a completely different use. In fact, we realized that metamodels are utilized to solve two fundamental types of tasks that we would like to denote as *"create"* and *"compose"*.

"Create" involves the generation of new metamodels (i.e. modeling languages) from scratch. This is a highly domain-specific task as the created metamodel has to provide language constructs for all specific aspects that have to be grasped by the models that are then generated using this metamodel. The advantage of using metamodels is quite apparent: a commonly accepted understanding of relevant "real-life"-concepts is guaranteed and new model instances can be created in a structured way whereas it is ensured that all relevant aspects are taken into account and nothing of importance is forgotten. Rosemann and Green [23] emphasize this when speaking of metamodels as clarifying the understanding,

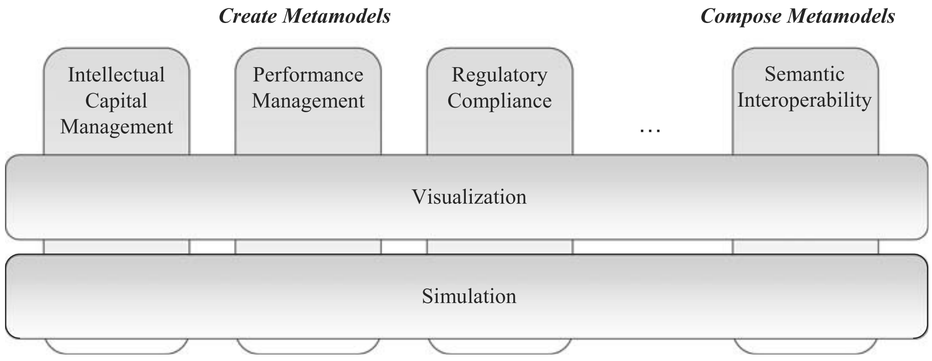


Fig. 2. Some application areas of metamodeling

simplifying the communication, being a means for structuring and analyzing, and finally being the enabler for the derivation of new modeling techniques.

"Compose" on the other hand denotes the combination of already existing metamodels in order to create one single new metamodel for a specific purpose. This purpose may, for instance, be to achieve interoperability of the resources that are modeled using the source metamodels that are composed [13] or to create "combined" models that are able to depict all aspects in one single model that had to be grasped in various separate models before (see section 3.2). Composition is done in a mapping process that relates various elements of different source metamodels with one another[2].

Now if we think of actual application areas for metamodeling we can distinguish between "create"- and "compose"-applications. New metamodels can be created in a variety of domains including intellectual capital management, performance management, regulatory compliance, knowledge management, business process management, and so on. The composition of metamodels is a means to achieve semantic interoperability, for instance. Finally, there are also some "cross-sectional" functions that go together with the concept of metamodeling like visualization and simulation. These findings are summarized in figure 2. In the next section we are going to describe some of these application areas in more detail.

3 Metamodeling: Some Applications

In the following we want to provide some insight into actual applications of metamodeling. First we are introducing an approach that shows the creation of a new metamodel in the domain of intellectual capital management. Then the composition of metamodels for achieving semantic interoperability is described. Finally, we discuss the cross-sectional role of visualization in the context of metamodeling.

[2] This process is described in some more detail in section 3.2 on page 182.

3.1 A Metamodel for Intellectual Capital Management

The reporting of intellectual capital is gaining steadily more on importance as the overall discrepancy between an enterprise's market value compared to its book value is getting bigger [27]. When looking at the S&P 500 Companies and their ratio of market-to-book value, it can be determined that the ratio has reached in March 2001 more than five times the value of December 1977 [16]. This huge portion of value of an enterprise that cannot be seen on the traditional financial balance sheet implies a lot of risks [15]. Due to the variety of intellectual capital reports that are available, a comparison of values stated in these reports can most likely not be executed.

The here presented Intellectual Capital Report Benchmarking (ICRB) Framework consists of a set of four areas. As indicated in figure 3, these four areas interact with one another in the following way: By focusing on step (1) of the framework, the organizational structure of an organization under study (OSUS), i.e. the main processes as well as its organizational charts will be elicited as a basis for the application of the ICRB framework as both the processes and organizational charts will serve as the starting point for the identification, utilization, management, and reporting of an organization's intellectual capital. However, as soon as an organization aims to manage and/or reports its intellectual capital, it can choose from a variety of available intellectual capital methods. These methods may differ completely and thus, divergent results may be produced when relying on diverse intellectual capital methods ([18], but cf. e.g. also [1], pp. 283-374, who indicates exemplarily the differences of 25 methods for intellectual capital management and reporting). Thus, the selection of a method for

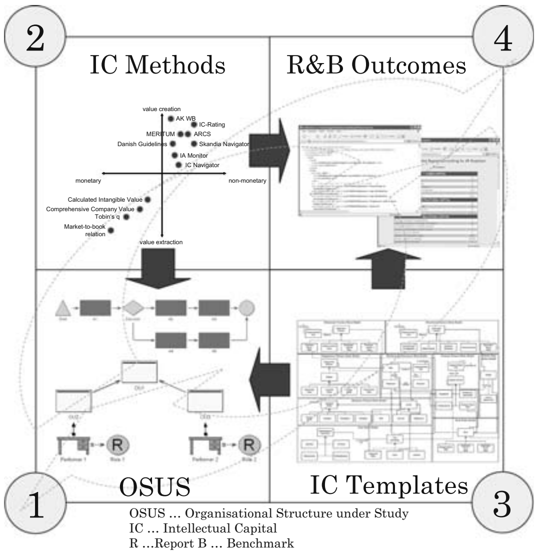


Fig. 3. Intellectual Capital Report Benchmarking Framework

intellectual capital management and reporting, as it is indicated in step (2) in figure 3 on the preceding page, influences the outcomes in the long run due to diverging data, but even more important the application of available intellectual capital templates in the short run. These templates (marked with step (3) in figure 3 on the previous page) are seen as constructs that organize both data material describing an organization's intellectual capital and linkages of these data with the organizations processes and organizational charts. Eventually, these templates have been realized by establishing metamodels for intellectual capital reporting, as it is indicated in figure 4 on the facing page. By this step many of the existing intellectual capital methods can be applied on an organization by keeping the required intellectual capital data in certain areas of the instantiated metamodels. Thus when applying the ICRB framework, in step (4) outcomes in form of reports can be created (periodically) by relying on diverse intellectual capital methods, which allows in turn a comparison of values as it anticipated above.

Finally, the ICRB framework is accompanied by a step model (indicated in figure 3 on the previous page by the dashed cyclic arrows), which shall act as a guideline for the introduction of the ICRB frameworks in organizations. The ICRB Step Model starts with the *elaboration of the strategy and tactical targets*, wherein the firm's global strategy is defined as an intellectual capital report and depends also on the company's strategic and tactical objectives. Further, step 2, the *definition of the business processes and working environments* has to be executed as the firm's business processes have to be modeled for being able to step forward to the *extraction of intellectual capital-relevant business processes and working environments*. Here, those business processes that contain intellectual capital-intensive activities have to be identified, before in step 4, the *intellectual capital processes are defined* and subsequently in step 5, the *selection of applicable intellectual capital reporting concepts* can be chosen. Finally, step 6 contains the *evaluation of reported values*, where based on automatically created intellectual capital reports, it can be checked whether the company's performance is satisfactory according to its prior-set strategy [18].

In the following, the afore-referred to metamodels for intellectual capital reporting shall be outlined briefly:

As the firm's business processes play a major role when considering an organization's intellectual capital, they have to be included in the considerations concerning the *creation* of a metamodel for intellectual capital reporting. In the following, figure 4 on the facing page [17] gives an overall picture of the whole modeling method containing seven metamodels for intellectual capital reporting, one metamodel for business processes and one metamodel for working environments. The seven intellectual capital reporting metamodels are again classified into two categories: Primary model types and secondary model types, whereas the first consists of the human factor metamodel (HFM), the structural factor metamodel (SFM), and the relational factor metamodel (RFM). The secondary model types are composed of the risk metamodel (RM), the cost metamodel (CM), the competence pattern metamodel (CPM), and the patent map metamodel (PM). The primary model types contain information that is directly related to and can

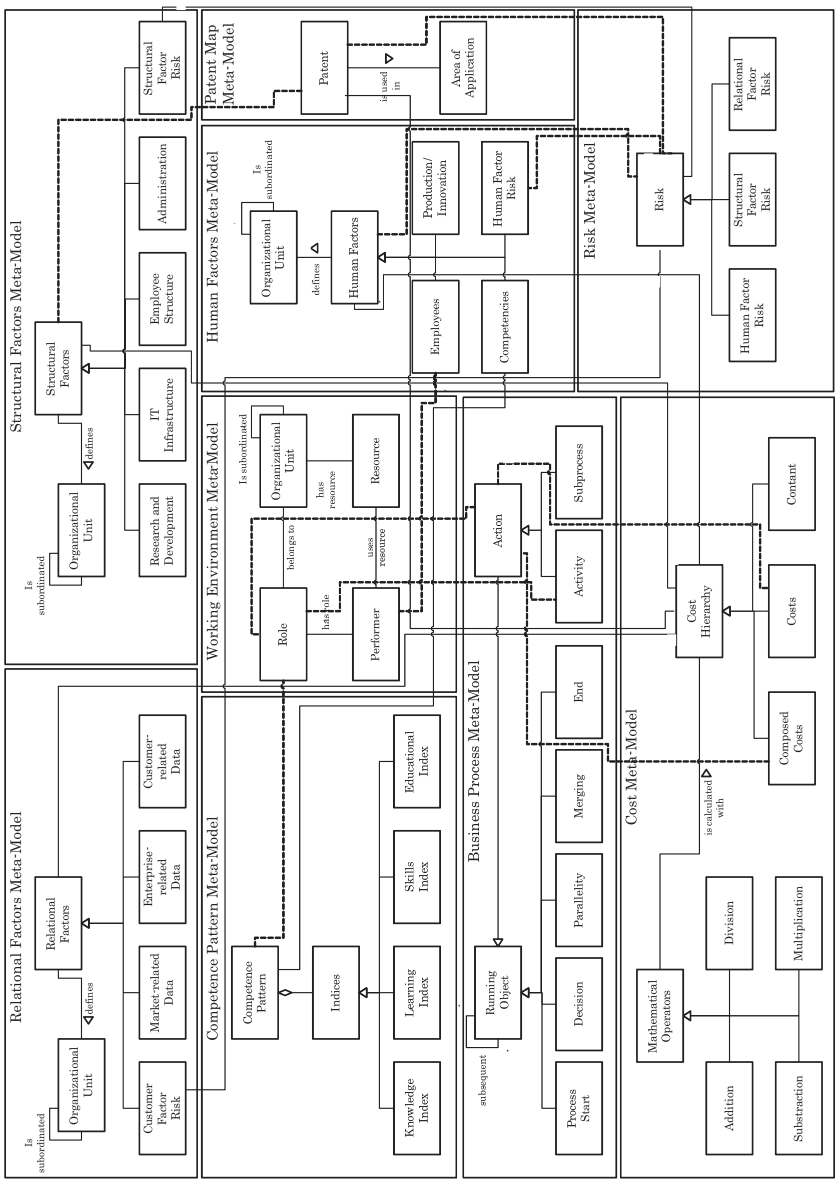


Fig. 4. Metamodels for Intellectual Capital Reporting [17]

be found in intellectual capital reports, whereas the secondary model types are composed of data that is relevant for the linking and/or calculation of values relevant for intellectual capital reports. Hence, they represent the supporting means for the values to be included in the primary model types [17].

The final outcome of the modeling of an organization's intellectual capital are reports that indicate the development of an organization's intangibles. Hereby, the meta-data for the reports is visualized with the aid of XML and XSLT, which allows for the automatic transformation of these meta-data into various formats, as e.g. HTML, RTF but also the visualization of data in a spreadsheet program like MS Excel. These by one click automatically created and selectable reports should serve as key performance indicators, which do both measuring intellectual capital factors and also support the management in the course of either the establishment or the maintenance of environments that allow employees to think freely, to take over responsibility, and to share knowledge, which in turn helps the firm to create higher profits [19].

3.2 Combining Metamodels and Ontologies to Achieve Semantic Interoperability

The topic of integrating data and ensuring the interoperability of information systems and business processes is of great practical importance which can already be seen by the fact that according to Gartner up to 40% of the companies' information technology budgets are spent on integration issues [10]. The heterogeneities that have to be dealt with in this context are usually classified to be of syntactical, structural or semantic nature [22] whereas resolving the latter seems to be most laborious as 60–80% of the resources of integration projects are spent on reconciling semantic heterogeneities [3].

In this section we want to demonstrate that the *composition* of metamodels is a powerful means in order to achieve semantic interoperability of modeled resources. A composition of metamodels can be realized in a process of comparing two metamodels with one another in order to identify *semantic equivalences* between metamodel elements. These equivalence relationships can then be used either as loosely defined mappings as well as a basis for the creation of one single new integrated metamodel. Related work concerning these issues can, for instance, be found in [31] where an abstract syntax of a mapping definition language is introduced as well as a language for integration rules that allows for the combination of two source metamodels in one integrated metamodel. The shortcoming of this and other similar approaches is that the semantics of the metamodel elements is not taken into account in an explicit way. The mapping process is mostly done by domain and metamodeling experts that have to put forth a lot of knowledge concerning the meaning of metamodel elements in order to be able to identify equivalent parts.

In [13] a generic architecture is introduced that allows for the (semi-)automatic identification of semantic equivalences which is achieved through the establishment of *semantic mappings* between metamodel elements and ontology concepts that provide explicit and machine-processable representations of meanings. Figure 5 on the next page depicts this architecture showing that the abstract metamodels 1 and 2 are to be mapped with one another whereas for demonstration purposes the relationship between A and B shall be examined in more detail. Basically, there are two aspects concerning semantics that have to be

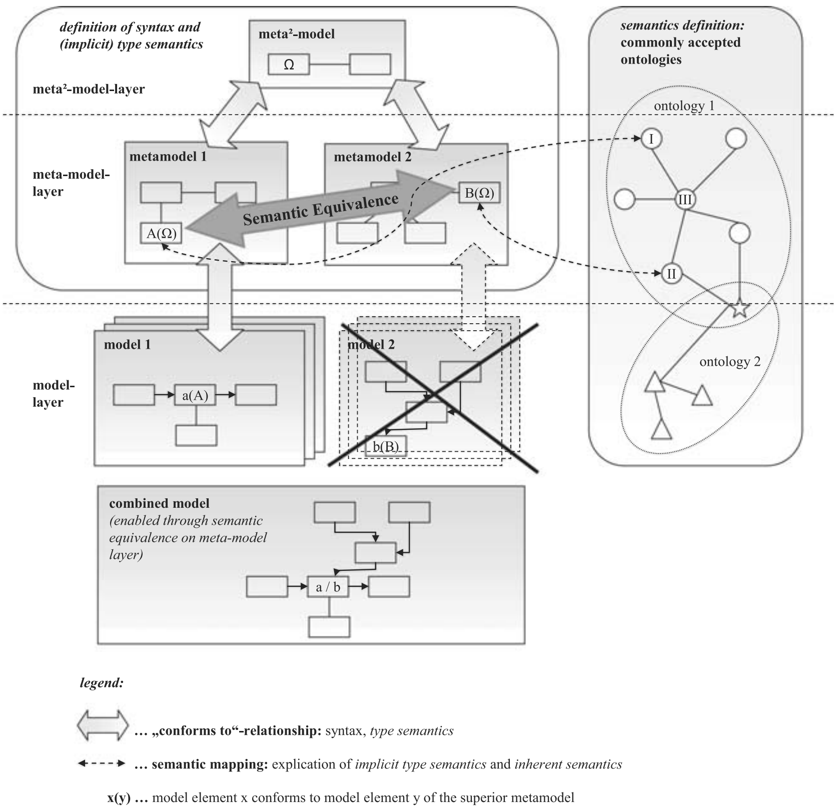

definition of syntax and (implicit) type semantics
meta²-model
Ω
meta²-model-layer
semantics definition: commonly accepted ontologies
ontology 1
I
III
II
meta-model-layer
metamodel 1
metamodel 2
A(Ω)
B(Ω)
Semantic Equivalence
model-layer
model 1
a(A)
model 2
b(B)
ontology 2
combined model
(enabled through semantic equivalence on meta-model layer)
a / b
legend:
… „conforms to"-relationship: syntax, type semantics
… semantic mapping: explication of implicit type semantics and inherent semantics
x(y) … model element x conforms to model element y of the superior metamodel

Fig. 5. Architecture for the composition of metamodels using ontologies (adapted from [13])

taken into account when comparing (meta-)model elements: first is *type semantics* and second *inherent semantics*. The difference between these two aspects can be explained best using an example like, for instance, from the area of business process modeling. Consider an "activity" of a business process model that has been generated using the Business Process Modeling Notation [21]. The fact that something is an "activity" means that something is done by some performer using some resource. We denote this as type semantics. In the process model this activity is now given a name like "deny proposal" which is some more meaning as we exactly know what is done ("deny") on which resource ("proposal"). This additional meaning is called inherent semantics.

When comparing two (meta-)model elements for semantic equivalence both semantic aspects have to be considered. Type semantics is provided by the metamodeling hierarchy whereas inherent semantics is provided when mapping (meta-)model elements with ontology concepts.[3] In our example in figure 5 we

[3] These interrelationships are described in some more detail in [13].

see that A and B are carrying the same type semantics as they are both derived from Ω. Additionally, they have the same inherent semantics which can be inferred through the fact that ontology concept I (which is connected to A) and ontology concept II (connected to B) are semantically equivalent through an intermediate concept *III*, which is taken as valid assumption in this example.

Now, one benefit that arises from revealing semantic equivalence between different elements of metamodels (and therefore language descriptions) is that model instances that have been created using these modeling languages can be made interoperable or can be integrated as well. This is also indicated in figure 5 on the preceding page where two models 1 and 2 can be combined in one single new model due to the knowledge that A and B are equivalent. This induces that a and b are equivalent concerning their type semantics. The inherent semantics of a and b can be explicated the same way as described above for the metamodel elements and if this semantics type turns out to be equivalent as well (which is taken as granted in this example) then a and b can be unified in one single element in a new model as indicated in figure 5.

Finally, the link from metamodel mapping and integration to semantic interoperability has to be elaborated as it may not be obvious. This link exists as metamodels provide the means to create models which in turn represent reality. Now, when semantic equivalences between (meta-)model elements are revealed the underlying models or represented aspects of reality, respectively, are found to be equivalent as well. In the context of business process models this can now for instance be used in order to establish interoperability of real processes that are carried out in different departments or organizations (cf. [13]).

3.3 Metamodels and Visualization

In business informatics the visual representation of models, data and their relationships plays an important role. Examples can be found in several areas that are either core parts of business informatics or contribute important aspects to the field such as enterprise modeling, descriptive statistics, graph theory or knowledge management ([7]). In the following the specific relation between metamodeling and visualization shall be highlighted.

Whereas metamodeling can be seen as a method to identify the concepts and relationships of a certain domain for the purpose of model creation by human users, the role of visualization is often seen from a different point of view. In most international publications in the area of computer science visualization is mainly regarded from a technical, data-oriented perspective (cf. e.g. [25]; [24]): There, it is used as a technique to discover new facts in large data sets through mapping data to graphical representations. By applying methods such as color coding or specialized types such as parallel coordinates ([9]) previously unknown relationships that are contained in the data may be found. Examples include the discovery of relationships in business data ([11]) or applications in the natural sciences such as the mining of the human genome ([28]). However, visualization is not only a technical science but has a long tradition in other fields as well. These include the different domains of art and design and the foundations of visualization in the area of geometry and mathematics as well as in the field of visual language theory and semiotics [7]. It thus seems obvious to take into account not only the technical aspects of visualizations focusing on methods for representing data but also its conceptual foundations. In this sense visualization fulfills a function that is similar to that of metamodeling: Based on a number of graphical primitives compositions of these primitives can be created that may then be used for representing human knowledge as well as information. In addition, these compositions can not only be created by human users but may also be assembled based on computer algorithms, e.g. for creating visualizations for analyzing data. This additional aspect closes the loop of the concept of visualization to the technical, data-oriented view as described above. One way to formalize the conceptual aspects of visualization has been described by the approach of Semantic Visualization ([5], [6]). The central idea of this approach is to define visual objects that contain not only graphical representations but provide mechanisms to influence the representations by inherent control structures and variables. To integrate visualizations that have been specified elsewhere it is also

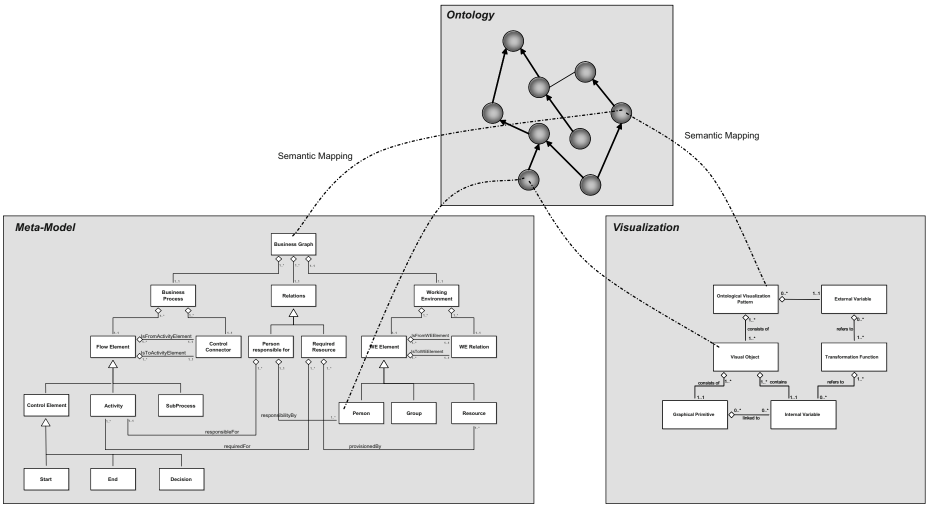


Fig. 6. Application of semantic visualizations to metamodels [8]

Functional Implementation	*Automatic*	*Semi automatic*	*Manual*
Standardization	*Fully Standards based*	*Partly Standards based*	*Proprietary*
Deployment	*Static*		*Dynamic*
Time Reference	*On demand*		*Pre-configured*

Fig. 7. Dimensions of the mapping between metamodels and visualizations [8]

possible to recursively insert visual objects into each other. Thereby the syntactic basis for a conceptual view on visualizations is established. To relate these visual objects to other conceptualizations such as metamodels the approach of Semantic Visualization defines linkages between the visual objects and semantic schemata. The visual objects together with these semantic annotations are termed Ontological Visualization Patterns. In detail it is then possible to annotate an arbitrary visual object (e.g. one that represents a gauge) together with its variables (e.g. for modifying the position of the needle in the gauge) by the corresponding semantic concepts (e.g. thermometer and temperature). Thereby a semantic description of the visualizations is determined.

To link the Ontological Visualization Patterns to metamodels it is equally necessary to semantically annotate the metamodels. By traversing from metamodels to the corresponding semantic schema over to the semantically annotated visual objects a direct mapping of visual representations and metamodel classes and relations can be established (see figure 6).

The mapping of Ontological Visualization Patterns to metamodel elements can be done in a variety of ways ([7]): The functional implementation can either be accomplished automatically through the application of logical reasoning via the ontology, semi automatically with additional input from a user or manually. For the definition of metamodels and ontologies it can be reverted to standards or proprietary technologies. The definition of Ontological Visualization Patterns may also be realized with proprietary techniques or by the extension of existing standards such as Scalable Vector Graphics ([7]). The mapping can either be static or dynamic - in the sense that the patterns are tightly coupled to the metamodel elements or may be dynamically allocated. Similarly, the assignment may be performed on demand or based on pre-configured mappings. A summary of these mapping dimensions is given in figure 7.

4 Conclusion

This paper provided an introduction into possible application areas of metamodeling in the context information systems. After a short description of the term metamodel we provided a classification of these applications according to the basic metamodel operations “create” and “compose” that have to be supported by “cross-sectional” aspects like visualization and simulation. We described the creation of a domain-specific metamodel for intellectual capital management and the composition of metamodels using ontologies in order to achieve semantic interoperability. Finally, we took a glimpse on how metamodel elements can be mapped to visualization patterns.

References

1. Andriessen, D.: Making Sense of Intellectual Capital. Elsevier Butterworth-Heinemann, Oxford (2004)
2. Chen, P.P.S.: The entity-relationship model - toward a unified view of data. ACM Transactions on Database Systems 1(1), 9–36 (1976)

3. Doan, A., Noy, N.F., Halevy, A.Y.: Introduction to the Special Issue on Semantic Integration. SIGMOD Record 33(4), 11–13 (2004)
4. Favre, J.M.: Foundations of Meta-Pyramids: Languages vs. Metamodels - Episode II: Story of Thotus the Baboon. In: Bézivin, J., Heckel, R. (eds.) Language Engineering for Model-Driven Software Development, Dagstuhl, Germany (2005), vol. Language Engineering for Model-Driven Software Development, Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI)
5. Fill, H.G.: Basic Conceptions for Semantic Visualization. In: Schweighofer, E., et al. (eds.) e-Staat und e-Wirtschaft aus rechtlicher Sicht - Aktuelle Fragen der Rechtsinformatik - Tagungsband des 9. Internationalen Rechtsinformatik Symposions IRIS 2006, pp. 482–485. Boorberg Verlag (2006)
6. Fill, H.-G.: Semantic Visualisation of Heterogeneous Knowledge Sources. In: Hinkelmann, K., Reimer, U. (eds.) Modellierung für Wissensmanagement - Workshop im Rahmen der Modellierung 2006, Sonderdrucke der Fachhochschule Nordwestschweiz, pp. 17–27 (2006)
7. Fill, H.-G.: Visualisation for Semantic Information Systems. PhD thesis, Universität Wien (2006)
8. Fill, H.G., Karagiannis, D.: Semantic Visualization for Business Process Models. In: Proceedings of the Twelth International Conference on Distributed Multimedia Systems - International Workshop on Visual Languages and Computing 2006, Knowledge Systems Institute, Grand Canyon, USA, pp. 168–173 (2006)
9. Fua, A.-H., Ward, M.O., Rundensteiner, E.A.: Hierarchical Parallel Coordinates for Exploration of Large Datasets. In: 10th IEEE Visualization 1999 (VIS 1999), IEEE, Los Alamitos (1999)
10. Haller, A., Cimpian, E., Mocan, A., Oren, E., Bussler, C.: WSMX - A Semantic Service-Oriented Architecture. In: IEEE International Conference on Web Services (ICWS 2005), pp. 321–328. IEEE Computer Society, Los Alamitos (2005)
11. Hao, M., Dayal, U., Hsu, M.: Visual Data Mining for Business Intelligence Applications. In: Lu, H., Zhou, A. (eds.) WAIM 2000. LNCS, vol. 1846, Springer, Heidelberg (2000)
12. Harel, D., Rumpe, B.: Modeling Languages: Syntax, Semantics and All That Stuff - Part I: The Basic Stuff. 2006-05-26 (2000)
13. Höfferer, P.: Achieving Business Process Model Interoperability Using Metamodels and Ontologies. In: Österle, H., Schelp, J., Winter, R. (eds.) Proceedings of the 15th European Conference on Information Systems (ECIS 2007), pp. 1620–1631 (2007)
14. Karagiannis, D., Höfferer, P.: Metamodels in Action: An Overview. In: Filipe, J., Shishkov, B., Helfert, M. (eds.) ICSOFT 2006 - First International Conference on Software and Data Technologies, vol. 1, pp. IS–27–IS–36. Insticc Press (Setúbal, 2006)
15. Karagiannis, D., Nemetz, M., Schwab, M.: Dashboards for Monitoring Compliance to Regulations - A SOX-based Scenario. In: Electronic Proceedings of IGO 2006 - International Conference on Integrating Global Organizations (2006)
16. Lev, B.: Intangibles: Management, Measurement, and Reporting. Brookings Inst. Press, Washington (2001)
17. Nemetz, M.: A Meta-Model for Intellectual Capital Reporting. In: Reimer, U., Karagiannis, D. (eds.) PAKM 2006. LNCS (LNAI), vol. 4333, pp. 213–223. Springer, Heidelberg (2006)
18. Nemetz, M.: Towards a Model for Creating Comparable Intellectual Capital Reports. In: Maurer, H., Tochtermann, K. (eds.) Proceedings of I-Know 2006 - International Conference on Knowledge Management, JUCS (Graz, 2006)

19. Nemetz, M., Karagiannis, D.: Unification of Intellectual Capital Reports with IT-Support. In: Electronic Proceedings of the World Congress on Intellectual Capital, Hamilton, Ontario, Canada (2007)
20. Object Management Group. MDA Guide Version 1.0.1. 2007-01-22 (2003)
21. Object Management Group. Business Process Modeling Notation Specification. 2006-10-18 (2006)
22. Obrst, L.: Ontologies for Semantically Interoperable Systems. In: CIKM 2003: Proceedings of the Twelfth International Conference on Information and Knowledge Management, pp. 366–369. ACM Press, New York (2003)
23. Rosemann, M., Green, P.: Developing a meta model for the Bunge-Wand-Weber ontological constructs. Information Systems 27(2), 75–91 (2002)
24. Rushmeier, H., Turk, G., Van Wijk, J.J.: In: Proceedings of the 2004 IEEE Visualisation Conference, IEEE Computer Society, Austin, TX, USA (2004)
25. Silva, C.T., Gröller, E., Rushmeier, H.: In: Proceedings of the 2005 IEEE Visualisation Conference, IEEE Computer Society, Minneapolis, MN, USA (2005)
26. Stachowiak, H.: Allgemeine Modelltheorie. Springer, Wien (1973)
27. Stewart, T.A.: The wealth of knowledge: Intellectual Capital and the Twenty-First Century Organization. Currency Book, New York, London (2001)
28. Stolk, B., Abdoelrahman, F., Koning, A., Wielinga, A.: Mining the Human Genome using Virtual Reality. In: Bartz, D., Pueyo, X., Reinhard, E. (eds.) Fourth Eurographics Workshop on Parallel Graphics and Visualization, The Eurographics Association (2002)
29. Strahringer, S.: Metamodellierung als Instrument des Methodenvergleichs: eine Evaluierung am Beispiel objektorientierter Analysemethoden. In: Berichte aus der Betriebswirtschaft, Shaker, Aachen (1996)
30. Whitten, J.L., Bentley, L.D., Dittman, K.C.: Systems analysis and design methods, 6th edn. McGraw-Hill Irwin, Boston (2004)
31. Zivkovic, S., Kühn, H., Karagiannis, D.: Facilitate Modelling Using Method Integration: An Approach Using Mappings and Integration Rules. In: Österle, H., Schelp, J., Winter, R. (eds.) Proceedings of the 15th European Conference on Information Systems (ECIS 2007), St. Gallen, Switzerland, University of St. Gallen, June 7-9, 2007, pp. 2038–2049 (2007)

Collaborative Model Driven Software Development for SOA-Based Systems

Thomas Karle[1] and Andreas Oberweis[2]

[1] Research and Development
PROMATIS software GmbH, Germany
thomas.karle@promatis.de
[2] Institut für Angewandte Informatik und Formale Beschreibungsverfahren (AIFB)
Universität Karlsruhe (TH), Germany
oberweis@aifb.uni-karlsruhe.de

Abstract. One of the main factors of success for the implementation of SOA-based software systems in a globally networked environment is efficient and effective collaboration. To support globally distributed collaborative software engineering, model driven development processes are indispensable. In this paper we present a new framework based on XML nets, a variant of high-level Petri nets, and XML standards to manage the development of complex SOA-based systems. The framework provides a model-based analysis and implementation of prototypes and complete application systems based on generators. The model covers all relevant aspects from the underlying web services up to the front-end of a respective web application.

1 Introduction

Many software developing organizations decide to implement their products or projects offshore or in globally networked environments. In parallel new working methods such as teleworking and telecooperation based on the internet provide completely new organization models which are independent of place and time. During the implementation of SOA-based software systems there are team members with completely different knowledge and skills involved. They implement components on different layers of this architecture [Do05]. The size of such systems, the complexity of service-oriented architectures, and the use of modern development paradigms like extreme programming requires a high degree of collaboration. Worldwide acting teams pose another challenge for software development in such projects [CC05].

To handle these requirements adequately, a higher abstraction level in the form of a specific description model is necessary. Based on description models, complex systems like SOA-based application systems may be managed from the analysis and design up to the operation. They allow the separation of business aspects and technical aspects and support the communication between the different team members. Software development for service oriented architectures requires clear responsibilities due to the complexity of the architectures which should also be considered in the description model. A formal and technologically independent description provides manageability of the technology changes that are to be expected particularly in the area of web applications. Models force the developers to take the business requirements into consideration. Models may be used for the generation of prototypes, applications and test cases. Finally they ease the maintenance of the software system due to the implicit graphical documentation of the models and the generated code. This allows getting new team members very fast into the details of the project so that productivity is considerably improved.

2 Existing Approaches

For modeling and generating application systems many different methods and languages have been proposed. In this section four existing approaches will be briefly discussed:

Model Driven Architecture (MDA) provides different models, different platforms and different abstraction levels [MM01]. MDA supports the automatic or semi-automatic transformation of the different model types and finally the generation of application systems. One important characteristic of MDA is the compact description of MDA models for the handling of complex systems. Furthermore MDA allows the use of the UML standard integration with tools from different software providers. A problem of MDA is to keep manual changes after code generation consistent with the model.

UML-based Web Engineering (UWE) [HK01, KK02] is an approach which aims at modeling and implementing web applications. UWE provides the semi-automatic generation of JEE-based web applications. The transformation of the logical objects in the models to their physical representation is carried out with XSLT based on the framework Cocoon. The UML models are transformed in XMI and used to generate the application. The result of the generation is a set of Java Servlets and Java Server Pages. The application logic has to be implemented manually using Enterprise Java Beans. The application is based on persistent objects in a database layer. UWE provides the same advantages as MDA. A disadvantage is that the approach is currently bound to JEE technology and that it does not provide a personalization concept.

The Web Modeling Language (WebML) [CFB00] aims to create an intelligent information structure for data-intensive web applications. WebML should close the gap between requirements specification and the subsequent phases of a project. The specification of the application is structured in four different orthogonal perspectives: The structure model which is compatible to entity relationship models or UML class diagrams, the hypertext model which defines the navigation, the presentation model which describes the layout in an abstract XML syntax in a front-end independent way and the personalization model which defines personalized content. WebML provides a comprehensive modeling environment for web applications that considers navigation and personalization. The disadvantage is the complexity that implies a long training period for the team members.

The Navigational Development Technique (NDT) [Es04] is a process-oriented method for the development of web applications. The method covers the whole implementation process but it is currently focused on the requirements specification. NDT forces to document each step of the approach by a specific template. The templates are structured like tables and can partially be filled by a textual description. The main advantage of NDT is the early consideration of the navigational aspects in the analysis phase. This provides a detailed specification. A disadvantage is the missing of generation features for executable code and the informal definition based on natural language.

3 Requirements for Collaboration in SOA Projects

To implement business processes within a SOA-based architecture different people with different skills work on the several layers of such a system shown in Fig. 1. The business people define and specify their needs using business process management tools. Developers can be divided in specific developers for the front-end of the system and developers that implement the business logic by programming web services and BPEL processes. Collaboration in case of a SOA project should support and improve the teamwork of those different team members. Due to the fact that most of these projects are implemented in a globally distributed environment efficient and effective collaboration is one of the main factors of success [Ga05, LLH06, SDK99].

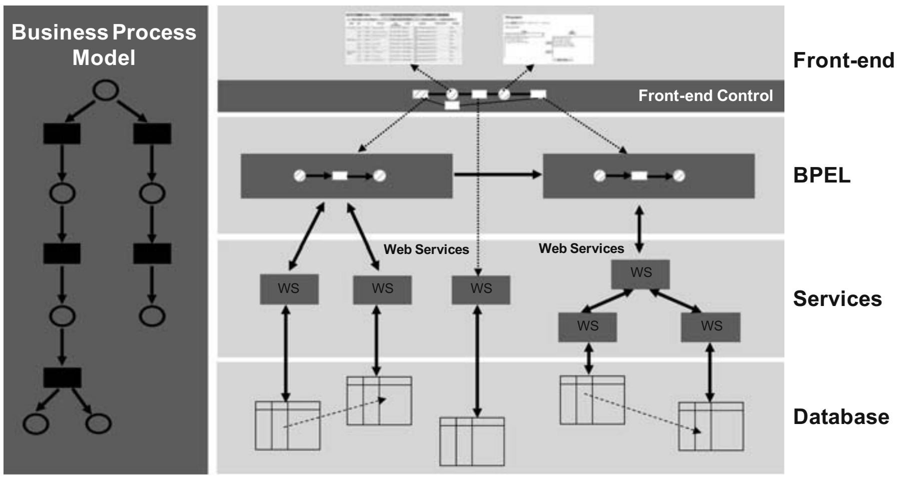


Fig. 1. Implementation of Business Processes based on a SOA

The collaboration should provide a common repository that contains all produced artifacts of the SOA project. Business requirements and their implementation in the form of front-ends, web services etc. have to be linked with each other. Furthermore generators based on models should be provided to avoid the "creeping requirements" problem. As a result of the generators' application a main part of the system may be presented already in the earlier phases of a project. The aim of the approach described in this document is to define a comprehensive model that includes all specific definitions for the different layers of a SOA-based system including the dependencies between business process models, front-ends, front-end controls, BPEL processes and web services.

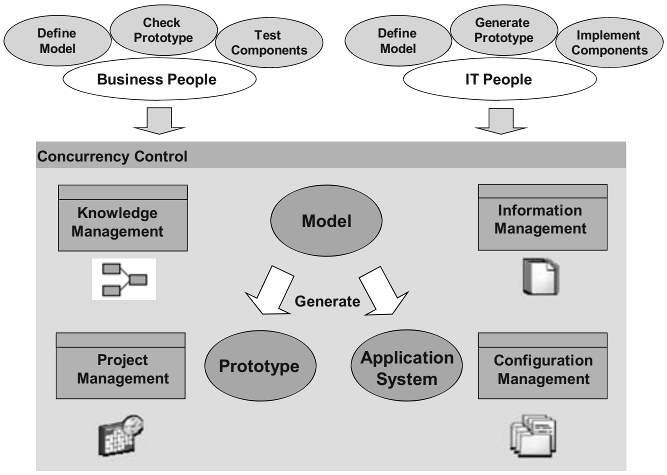


Fig. 2. Collaborative Software Engineering

Fig. 2 shows the requirements for such a collaborative software engineering environment where business people and IT people work together [GHS98]. Business people and software system architects collaboratively define the requirements in the description model. Based on models, prototypes are generated by the developers. The prototypes are checked by the business people whether they fulfill the specified requirements. IT people implement the software components of the application system also based on generation. Additional functionality might be provided. Besides the models and the generators the collaborative software development environment should provide the following collaboration functions [KN05]:

Concurrency Control: The basic concurrency control for collaboration, which includes the access to tasks, information and applications, should be provided by a portal. Usually portals contain user administration, personalization and access control functionality on the level of the portal components, the so-called portlets.

Information Management: In a software development project information should be provided in the specific context. Components like e-mail, instant messaging, forums and web conferencing may be used in a collaboration portal for the basic communication. Furthermore content management components provide news and documents over the portal. The different roles and responsibilities of the team members should be used for the information distribution. This information flow should be controlled by workflow technology.

Project Management: For coordination functions a collaborative environment should provide task management or date definition functionality. For monitoring and controlling also workflow technology can be used.

Knowledge Management: Knowledge Management within the scope of collaborative software development is provided by the assignment of tasks to required information and to appropriate experts. Required information includes models of the specification like business process models or specific documentations of the used tools and methods.

Configuration Management: To ensure that all team members work on a consistent state of their artifacts configuration management is necessary. The configuration management should not be restricted to the components that have to be implemented. It should also consider models, documents and further information. The access to a specific version of an artifact should provide all related information in the corresponding version, e.g. the current documentation of the required development tool.

4 Model for SOA-Based Web Applications

In this section an XML-net-based approach for the model driven implementation of SOA-based web applications is introduced. The model is based on the assumption that the business logic is implemented with web services. Existing databases are encapsulated by this web service layer. The human user's input is processed from the front-end and in an SOA-based environment usually a service from the business logic layer is called. The front-end receives a result from the service that contains data. The front-end processes this data and presents it to the user. This standard interaction process (1,2,3,4) is shown in Fig. 3. However, it is also possible that the service does not response, or that the front-end answers directly without calling a service. An example for this case (1,4) is a situation where the input of a user is identified as being wrong and an error message is displayed.

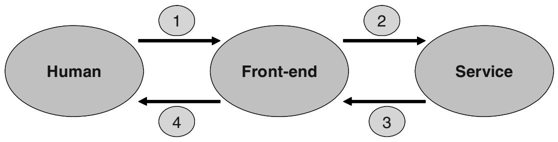


Fig. 3. Human Interaction in SOA-based Systems

4.1 Data Structure Model

The data structure model defines the data structures of the data objects that can be used by the web application. For the modeling of the application functionality it is necessary to have access to the attributes of data objects and the relationships to other data objects. The data structure model is based on XML schema. The data types of the attributes correspond to the XML schema data types. Furthermore, similar to an entity-relationship model, the relationships between the data objects are shown with the corresponding cardinality.

Fig. 4 shows two data objects and their relationship. In a front-end of an application this data objects may be managed in a master-detail form. Master-detail forms display a master row and multiple detail rows within a single page. The data structure model provides the basis for the modeling of the functional requirements of the application. This contains all structuring methods that are available using XML schema.

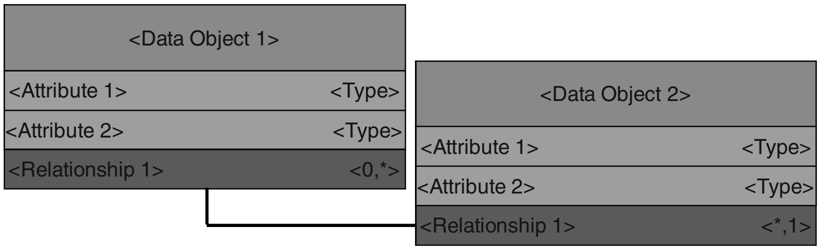


Fig. 4. Data Structure Model

4.2 Service Model

An important precondition for defining models for web service based applications is the availability of a set of services that should be used in the context of the application system. In the model this set of services is provided by a defined structure. This structure defines the calling of a web service and contains the following information:

Unique Name, Description, Class name (Call of the web service), Method name (Call of the web service), Data objects (Parameters), Data objects (Results) and Exceptions (Exception classes). Besides a unique name and a description of the service the class and the method name of the technical call of the web service is defined. Furthermore the service model contains the data objects of the data structure model that are used to set the parameters of the web service. In the same way the data objects that describe the results of the call are assigned. Finally the possible exceptions that can occur after the call of the service should be defined in the service definition. For each exception the modeler has to assign the corresponding exception class.

4.3 Interaction Model

The interaction model defines the interaction between user and front-end. It is the base model to define the front-end of a SOA-based web application. Two types of interaction can be distinguished: active interactions and passive interactions. Active interactions correspond to step 1 in Figure 3. These are user actions like input or the pressing of a button. Passive actions correspond to step 4 in Fig. 3 and are for example displayed results of an active interaction or error messages. The interaction model covers all events that might occur in the context of a web application. An important detail of the interaction model is that given interactions are structured in so-called areas. An area is a closed functional part of a web application which provides the context for a defined interaction behavior of the user. The interaction model has to consider decisions of the users. An example is the selection of navigation buttons in a menu. The sequence of interaction steps in a graphical user interface is not fixed. All enabled navigationable elements of a web page can be used in arbitrary sequence. Also the navigation between different areas is defined in the interaction model.

The interaction model is based on XML nets [LO03], a kind of high-level Petri nets [Pe81]. The places in a so-called interaction net of the model represent the current area of the user focus and the status of the area as a XML structure. This structure is described in section 4.5. There are two different types of places in interaction nets

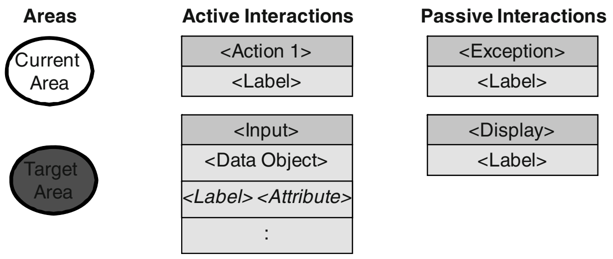


Fig. 5. Elements of an Interaction Model

shown in Fig. 5. Current area places represent states of the current area, for example the specific assignment of fields in this area. Target area places mark final places which represent the result of a decision of the user to navigate to another area. A transition represents an active or a passive interaction. Fig. 5 shows also different transition types. One type of an active interaction is the so-called action. Actions define activities that are performed by the user in a specific area. For example the click on a button may be an action. Each action has a specific result and provides a specific functional call. An action provides the navigation to another area or it can call a web service. It can also provide both, the call of a web service and the navigation in one step. Furthermore an active interaction may be the input in the fields of a specific area. For this case a specific active element for the input of data is provided, the so-called input. The input is assigned to a data object to define the possible set of attributes for the input. In an interaction net the input elements must be assigned to data objects of the data structure model. This takes into consideration that the data processing elements in a web application access the web service layer. In this way dynamic input fields can be connected with the data objects provided by a web service. There are two types of passive actions in the model: exception and display. Exceptions are started when an error or a specific event occurs and the user has to be informed about the exception. The display interaction shows information or results. Interactions are labeled by a unique name within the area. The attributes from the data objects can

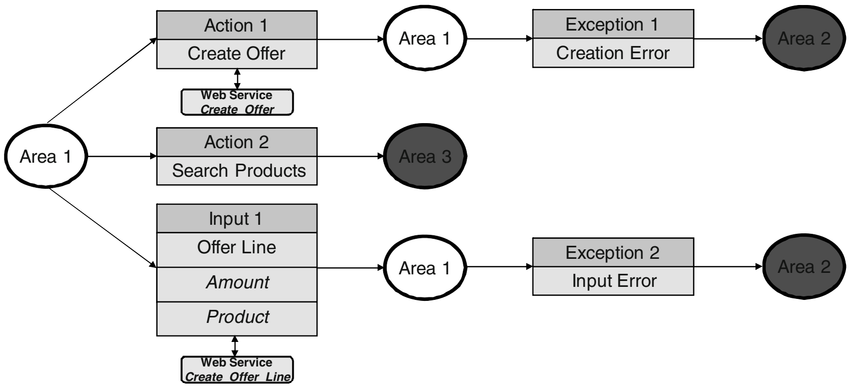


Fig. 6. Example of an Interaction Net

also be labeled within the input elements. All labels are used from the framework for generating application code. Based on the structure of the flows of an interaction net and the labels the generating engine creates buttons, links or input fields for the screens of the application.

Fig. 6 shows an example of an interaction net for area 1 with simple active and passive elements. The user can call the function *Create_Offer* (action 1) that is implemented as a web service by a click on the button *Create Offer*. After action 1 an exception might occur. The exception automatically navigates to area 3 for showing and handling the exception. The user can navigate to the separate area 3 by a click on the button *Search Products* for searching available products. Furthermore, the interaction net contains an input element that is assigned to the web service *Create_Offer_Line* and an exception element that might occur after the input of data and that is handled on area 2.

4.4 Front-End Model

In each interaction net the navigation from one area to the next is defined. An interaction net is independent from pages or from the different front-ends on which an application can be executed. There may be different front-ends for an application on a mobile phone, a handheld device or a personal computer. Depending on the different types of front-end the structure of interactions may be different. In front-end models the areas of the interaction models are assigned to pages in so-called front-end nets. So it is possible to assign for example two areas to one page for the front-end net of the personal computer and to assign the same two areas to two pages for the front-end net of the handheld device. A front-end net only shows the transitions between different areas and pages. Detailed functional calls like inputs are shown in the more detailed interaction nets. Developers and business people can discuss an overview of the whole application structure provided by front-end nets at a very early stage of a project. Combined with a generated prototype modelers can define a clear specification of the application system. The front-end model is the basis for generating a prototype or a complete application. Fig. 7 shows a front-end net where different areas with their internal interaction net are combined to model a comprehensive front-end behavior.

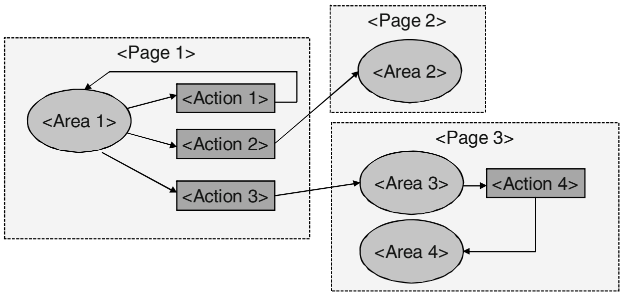


Fig. 7. Front-end Net

The front-end net in Fig. 7 may define the front-end behavior of a specific application on a mobile phone where the available space for pages is limited to a small display. Each area is placed on a separate page. If this functionality also should be provided by a browser on a personal computer the modeler can define a separate front-end net where area 1 and area 2 may be placed both on one single page.

4.5 Area Structure and Page Structure Model

The approach contains additional models to define the static structure and layout of areas. This information is stored in XML files. The separated area structure definition supports an organizational separation of layout and function. The structure definition of the area is based on a table. This structure table has user-defined rows and columns. Every element has its own cell in such a table. The used tables in the area structure may be nested. A specific case in the area structure definition is the so-called multi record table. Multi record tables are required to arrange the input element with its attributes of the interaction model on an area. A further important requirement of area structuring is the use of static elements. There are three types of static elements that can be used in the area structure definition: Static text, Static images and other static content.

In the same way elements are arranged in areas, where the areas by themselves must be arranged on pages when they are assigned to pages in a front-end model. The simplest case is a page which contains exactly one area. In this case there remains nothing to do for the modeler. If there are more areas on one page structure tables can be used like they are described in the area structure definition. The modeler has similar features like defining an area.

4.6 Personalization Model

In web applications the users usually act in different roles with different grants. In the specified model different aspects of personalization are defined. Roles for personalization are defined by a unique name and an additional description. The roles may be assigned with different grants to pages, areas and single elements to control the access and the navigation possibilities for each role. The different grants that can be assigned by the modeler are: view, execute and administrate. The following personalization levels exist: page, area and element.

4.7 Process Model

In the process model the executable business processes have to be defined. The so-called web service nets provide the orchestration of web services. Web service nets are based on XML nets as well as the interaction nets or the front-end nets. Web service nets can be directly transferred in executable BPEL Code. A detailed description of web service nets can be found in [KM05]. The creation of BPEL code based on the web service nets is provided by a specific generator of the framework.

4.8 Technology Dependent Model

To provide a specific code generation it is necessary to define technology dependent aspects, e.g. for Java Enterprise Edition, .NET or other technologies [CST05]. These aspects of the specific technology have to be defined in a separate model and to be assigned to the corresponding aspects in the technology independent parts of the model.

5 Model Based Collaboration

The different models are connected with each other. Interaction nets are refinements of regions of front-end nets. Interaction nets and web service nets are related by the web services that are used by a specific BPEL process and a specific front-end. Based on these relationships a spanning graph completely based on XML nets can be created. This spanning model can be used as a base structure for a repository that provides configuration management and concurrency control. The artifacts to manage in the repository are web services, the different model types to define the components of the SOA-based system. Additional information about the team and the responsible persons may be added to the artifacts in the repository. For example the process owners i.e. the business people that are responsible for a specific business process are assigned to the corresponding process model. Developers are assigned to their generated components or their technology dependent part of the model. Based on the model dependencies locks can be set on models and components for concurrency control. Both, optimistic and pessimistic methods can be used. A pessimistic method locks in case of an update of an artifact all dependent artifacts. For example if a web service is locked for an update then all front-ends and BPEL processes that access to this web service are locked as well. An optimistic method allows concurrent changes on dependent artifacts and checks for conflicts during the unlocking.

Dependencies of changes can be identified and team members can be informed automatically. For example if a front-end modeler adds an attribute to an input element in an interaction model, the developer of the corresponding web service may be informed about this change. Changes on the models can directly activate the generation of the corresponding software components. The tight integration of models and application code supports collaborative working project teams in different phases of a project.

Corresponding team members can be found for discussions based on the dependencies. Information may be added and shown directly at specific points of the model. The spanning model may be used for navigation through the project from the business side to the corresponding technical artifacts to provide transparency, i.e. it may be used to visualize the structure of complex projects. Information to be shown in this graphical navigation includes locks, change histories, links to further models or components.

For project management tasks can be automatically generated to coordinate the team. For example if a web service has changed, tasks for responsible developers of front-ends or BPEL processes can be created automatically. Furthermore the degree of completion of the implementation of business processes including all required artifacts may be calculated automatically using the dependencies.

The generators of the framework are designed for supporting collaborative working project teams in different phases of a project and may be used already in the early phases of a project to generate a prototype based on models. This helps to discuss possible solutions with business people and to avoid misunderstandings in the specifications. In the final phases it helps to implement the solution based on the specified requirements defined in the models.

6 Conclusion

The presented approach provides a comprehensive method and framework based on XML nets and XML standards to manage complex SOA-based systems in a collaborative way. The front-end generator of the framework supports code generation for different technologies that may be used in every phase of a software project. The specification of an application's behavior is completely based on XML nets and XML standards. Modelers have to learn only few basic concepts to define the system. The business processes can be modeled with XML nets too. The definition of the information is completely stored in XML format. The separation between technology independent aspects and technology dependent aspects provides a clear separation of responsibilities on the one hand. So in the analysis phase it is easy for business people to understand the defined models since the technical definitions are separated. Also the structure definition and layout design can be managed separately. On the other hand all associations between the different models are stored in a common repository and provide a consistent view on the solution. The approach and the framework consider different front-ends and different target technologies.

The framework is implemented in the project INCOME2010 at the Institute of Applied Informatics and Formal Description Methods (AIFB) at the Universität Karlsruhe and is currently evaluated in several software development projects. It is an open framework that can be adopted for different projects with different technologies. Future plans include the extension of the generators of the framework for other technologies and the development of further collaborative features on the underlying model.

References

[CC05] Cook, C., Churcher, N.: Modelling and Measuring Collaborative Software Engineering, University of Canterbury, Christchurch, New Zealand (2005)

[CFB00] Ceri, S., Fraternali, P., Bongio, A.: Web Modeling Language (WebML): A modeling language for designing Web sites. In: Proc. of the 9th WWW Conference (May 2000)

[CST05] Cardone, R., Soroker, D., Tiwari, A.: Using XForms to Simplify Web Programming. IBM Watson Research Center, New York (2005)

[Do05] Doubrovski, V., Grundler, J., Hogg, K., Zimmermann, O.: Service-Oriented Architecture and Business Process Choreography in an Order Management Scenario: Rationale, Concepts, Lessons Learned, OOPSLA 2005, San Diego (2005)

[Es04] Escalona, M., Reina, A., Torres, J., Mejias, M.N.: A methodology to deal with the navigation aspect at the requirements phase. In: Workshop on Early Aspects: Aspect-Oriented Requirements Engineering and Architecture Design, OOPSLA Conference (2004)

[Ga05] Gasson, S.: Boundary Knowledge-Sharing in E-Collaboration. In: Proc. 38th Hawaii Intl. Conf. on System Sciences, Hawaii (2005)

[GHS98] Grundy, J., Hosking, J.: Serendipity: Integrated Environment Support for Process Modeling, Enactment and Work Coordination. In: Automated Software Engineering, Kluwer Academic Publishers, Netherlands, pp. 27– 60 (1998)

[HK01] Hennicker, R., Koch, N.: Modeling the User Interface of Web Applications with UML. Practical UML-Based Rigorous Development Methods - Countering or Integrating the eXtremists, Toronto (2001)

[KK02] Kraus, A., Koch, N.: Generation of Web Applications from UML Models Using an XML Publishing Framework. Ludwig-Maximilians University of Munich, Integrated Design and Process Technology, IDPT-2002 (2002)

[KM05] Koschmider, A., Mevius, M.: A Petri Net based Approach for Process Model Driven Deduction of BPEL Code, OTM Confederated International Conferences, Agia Napa, Cyprus (2005)

[KN05] Kock, N., Nosek, J.: Expanding the Boundaries of E-Collaboration, IEEE Transactions on Professional Communication, Vol. 48, No. 1 (March 2005)

[LLH06] Lefebvre, E., Lefebvre, L., Hen, G.: Cross-Border E-Collaboration for New Product Development in the Automotive Industry. In: Proc. 39th HICSS, Hawaii (2006)

[LO03] Lenz, K., Oberweis, A.: Inter organizational Business Process Management with XML Nets. In: Ehrig, H., Reisig, W., Rozenberg, G., Weber, H. (eds.) Petri Net Technology for Communication-Based Systems. LNCS, vol. 2472, Springer, Heidelberg (2003)

[MM01] Miller, J., Mukerji, J.: Model-Driven Architecture (MDA), Object Management Group, Document Number ormsc/01-07-01 (2001), http://www.omg.org

[Pe81] Peterson, J.: Petri Net Theory and the Modeling of Systems. Prentice-Hall, Englewood Cliffs (1981)

[SDK99] Swaby, M., Dew, P., Kearney, P.: Model-based Construction of collaborative systems. BI Technology Journal 17(4), 78–90 (1999)

A Meta-model-Driven Tool Integration Development Process

Felix Klar, Sebastian Rose, and Andy Schürr

Technische Universität Darmstadt, Real-Time Systems Lab,
D-64283 Darmstadt, Germany
{klar,rose,schuerr}@es.tu-darmstadt.de

Abstract. The integration of languages and tools that are used for model-driven IT system development purposes is a hot research topic. Quite a number of model transformation and weaving approaches have been published rather recently that simplify the implementation of the needed integration components. Surprisingly, we are not aware of any publications that present comprehensive engineering processes for the systematic analysis, design, validation, and verification of the needed model coupling and translation services. This paper is a first contribution for the development of a standard modeling tool integration process with a main focus on the early scenario-driven requirements elicitation phases.

Keywords: Model coupling and transformation, tool integration development process.

1 Introduction

Integration of IT system development tools has been and still is an active research area for more than two decades [2,11,13]. In the early days the main focus has been laid on building Eclipse-like integrated development environments (IDEs) with homogeneous user interfaces, integrated data repositories, and middleware services for the propagation of events and messages between different components. With the increasing popularity of the *model-driven development (MDD)* of software the focus has been changed from studying low-level tool integration middleware to designing more abstract model coupling, composition, and translation techniques. As a consequence dozens of quite different model transformation and integration languages and tools have been proposed rather recently for these purposes. The reader is referred to [10] for a first survey and classification of these approaches.

The construction of an integrated IT system engineering environment that supports all activities of an MDD process and hence combines quite a number of different "commercial of the shelf (COTS)" tools is nevertheless still quite a complex and error-prone task. First of all tool adapters (wrappers) have to be built that abstract from the details of a specific COTS tool's API and introduce a standardized programming interface. Furthermore, very often a separate database (warehouse, repository) is built that stores and integrates the data of all involved tools. Finally, model analyzers, transformers, and weavers are created that offer the required services for an integrated model-driven IT system development process.[1] These model inspecting and manipulating components either directly operate on the data stored inside the regarded COTS tools via their adapters or make use of a copy of the required tool data that is exported to a separate repository.

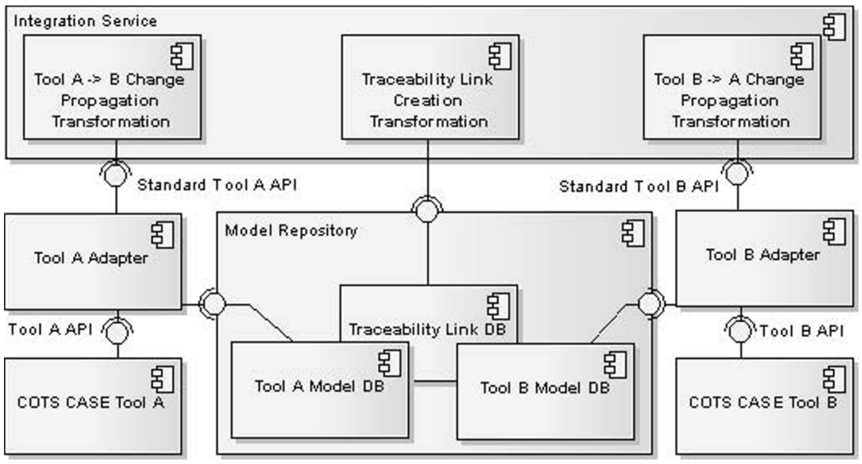


Fig. 1. Simplified Architecture of a Tool Integration Framework

All these components—shown in Figure 1—are usually developed following a rather straight-forward *meta-model-driven development (MMDD)* process: (1) meta-models of the involved modeling languages are used to generate standard tool adapter interfaces and model repositories. (2) Tool adapter implementations for the generated interfaces including required data export/import functions are then written by hand. (3) Finally, ordinary programming or more abstract model transformation languages are selected to realize the required model integration services.

Thus, the development of integrated modeling environments has reached a state-of-the-art comparable to IT system application development processes in the 60ies of the last century. A first generation of domain-specific programming (model transformation) languages and their compilation and debugging tools are available, but we do not yet know how to

– collect and organize the requirements for a model integration service
– parametrize, modularize, and refine huge libraries of meta-models and model manipulation services (sometimes called "megamodels"),
– and validate model analysis and manipulation services systematically.

[1] Note that we use the term "model" in a broad sense including e.g., informal requirements documents and pieces of source code.

To summarize, a typical MMDD process today just consists of the low-level implementation activities of a standard (e.g., V-model-based) IT system development process. Joint efforts are needed to develop missing requirements engineering, high-level design, and quality assurance methods. This paper is a first step towards that goal with a main focus on requirements engineering activities. For a discussion of the state-of-the-art of model-transformation-in-the-large concepts the reader is referred to [7], for first ideas concerning the systematic testing and certification of model transformations cf. [15].

In the following sections of this paper we will first introduce our running example and present our vision for the early phases of a new MMDD process. Afterwards, each phase of the proposed process will be explained in more detail in a separate section. The conclusions summarize the contributions of this paper and list open problems as well as planned future research activities.

2 Integration Development Process

As a running example, we will use the development of a bidirectional, incrementally working model translation service throughout this paper. This service translates between development artifacts stored inside a typical requirements engineering and a UML modeling tool. In order to keep the example as simple as possible we have selected a scenario, where the UML tool (Enterprise Architect (EA) from Sparx Systems), henceforth called *source tool*, supports the creation of high-level use case diagrams as shown in Figure 2. A standard requirements engineering tool (DOORS from Telelogic), termed *target tool*, is employed to create more detailed textual use case descriptions and to record other sorts of (non-functional) requirements. An envisioned e-commerce system developer may, e.g., (1) create a first set of use case diagrams and translate them into skeletons of more detailed textual descriptions, (2) complete and modify the generated text skeletons afterwards, and (3) synchronize the resulting text descriptions with the use case diagrams of step 1 using our integration services.

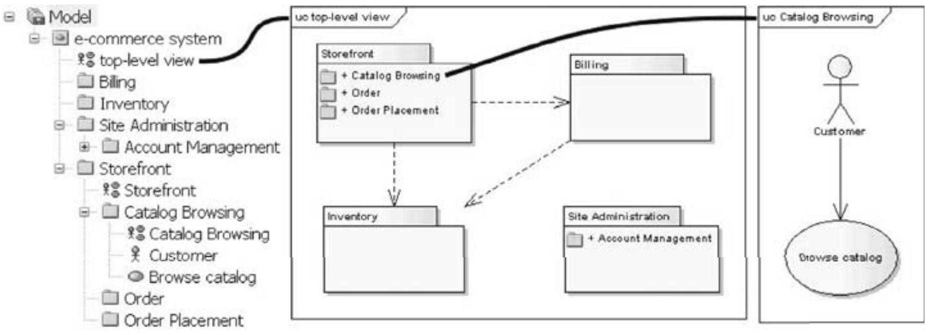


Fig. 2. Running example: Use cases of an e-commerce system

Figure 2 shows an Enterprise Architect snapshot with some use cases of the e-commerce system copied from [3]. This simple example contains some packages as well as a use case diagram with one actor, a single use case, and an association between the actor and the use case. In the sequel we will demonstrate how to develop model tool integration services that translate such a UML model into requirements stored in DOORS (and the other way round) and create appropriate traceability links between related artifacts in both tools. We will use the meta-modeling and model transformation tool MOFLON for that purpose; more specifically, so-called *triple graph grammars (TGGs)* [12] are used to specify the required bidirectional model translation and to generate an appropriate service implementation. The reader is referred to [1] for further details concerning the realization of model translations and tool adapters with MOFLON TGGs.

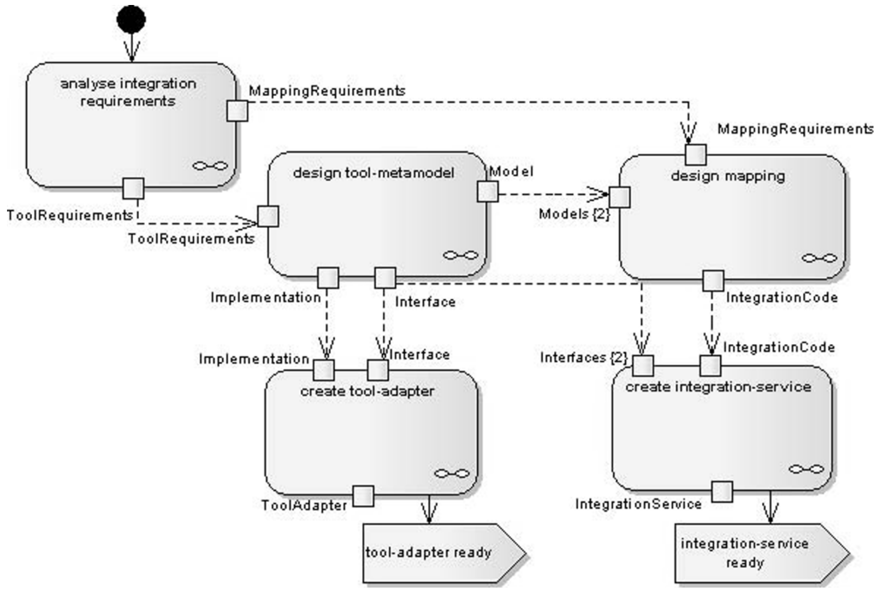


Fig. 3. Overview of MMDD Tool Integration Process

The requirements engineering, design, and implementation phases of the corresponding MMDD process are sketched in Figure 3 as activity diagram. The presented MMDD process consists of a series of fundamental activities of common software engineering processes. According to [14] and [5] these activities are software specification (i.e. requirements analysis), software design, and implementation. Our process starts with the analysis of the integration requirements. The source and target modeling elements to be integrated will be identified, as well as the representation of these elements in their tools. This activity produces two artifacts: (1) requirements for the definition of the elements that should be contained in the meta-models of the regarded modeling languages and (2) requirements for the definition of mappings between these meta-models. After the requirements have been identified, the tool meta-models are created. They must contain the elements that are necessary for the definition of mappings between source and target modeling elements. The outputs produced by this activity are a meta-model which is required for designing the mapping, a set of interfaces, i.e. the API for the manipulation of meta-model instances and an implementation of these interfaces. Having the meta-models of both tools and the mapping requirements as input, the TGG mapping can be designed. Afterwards the integration services can be created which provide functionality for realizing the mapping between the elements of both tools. Parallel to these tasks the tool adapters can be developed, as they are independent of the mapping design and the creation of integration services.

Our process allows activities to trigger changes, which cause a new iteration step that may start at any preceding activity. The next sections will go into more detail concerning the specific activities of our process.

3 Integration Requirements Analysis

The tool integration development process starts with the analysis of the requirements for the tool integration project. As described in Section 2, our project integrates UML use case artifacts that are created with Enterprise Architect and requirements artifacts that are created in DOORS. The requirements artifacts will be used in such a way, that they form a textual representation of use cases.

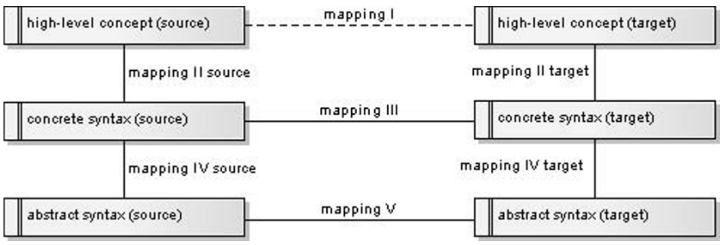


Fig. 4. Mapping definitions between artifacts

Figure 4 shows the artifacts of the integration requirements analysis and the connections between them in the form of mappings on different levels of detail. The studied integration starts with the identification of pairs of (source, target) modeling concepts. We call these concepts *high-level concepts*, as we are not yet interested in any tool-specific details concerning, e.g., the notation (*concrete syntax*) or the internal representation (*abstract syntax*) of the regarded modeling artifacts. In our running example the high-level source and target concepts are rather similar, but in general the integrated concepts may differ considerably. On the following level of detail we record the integration requirements from an end-users point of view and specify the concrete syntax representations of the related modeling concepts. On the third level of detail the integration requirements are specified from a tool builder's point of view, who is interested in the abstract syntax representations of related modeling concepts.

We will now discuss the mappings shown in Figure 4 in more detail. The requirements for the definition of these mappings are identified following an example-driven approach. The construction of pairs of snapshots of related modeling artifacts—on the concrete as well as on the abstract syntax level—plays an important role in this process[2]. Natural language sentences summarize and generalize the results of the example-driven requirements elicitation approach as shown below.

For the identification of the high-level concepts that drive the integration process of our running example, a requirements engineering expert first selects the required subset of concepts on source and target side. As a best practice, he then creates concrete syntax examples that cover all these concepts. These examples will help throughout the requirements analysis. The customers who will use the integration software later on should also be involved in this phase to guarantee that their favorite modeling elements and idioms are supported. The elements needed for our running example (cf. Figure 2) on the source side are nested packages, package dependencies, actors, use cases, and associations. In the following we will limit the model integration scenario to these high-level concepts. On the target side we will use the terminology of the tool-independent requirements specification interchange format RIF [4] that offers concepts like specification group, specification object, specification (object) type, and so forth. Based on this input we can specify *mapping I* as follows:

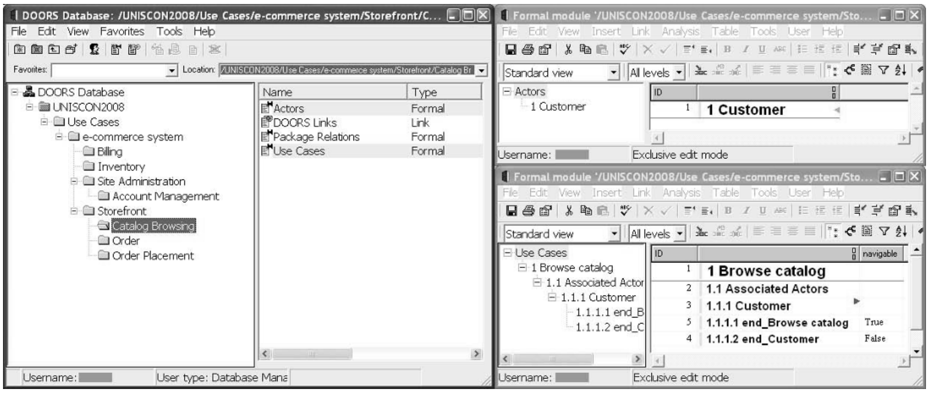
Mapping I:
- A source use case package is mapped to a target specification group.
- The name of a source use case package is equal to the name of a target specification group.
- Use cases and actors inside use case packages are mapped onto different types of specification objects inside specification groups.
- ...

Right now, we know what the integration looks like on a conceptual level. In the next step we have to analyze how the identified elements are represented by the tools, i.e. how elements look like in concrete syntax. Therefore, we need experts of both tools: an experienced tool user as well as a developer that is proficient in the tool APIs. Additionally it makes sense to involve a meta-modeling expert in the following steps that precede the definition of the meta-models of the regarded two modeling languages. Ideally the expert, who will create the tool adapter, should be involved, too; she will implement the generated standard API of the designed meta-models on top of the APIs of the two tools.

Enterprise Architect "natively" supports the definition of use cases, as it is a UML tool. The use cases shown in Figure 2 are screenshots of Enterprise Architect. They denote the concrete syntax of the use case elements. The representation of elements in Enterprise Architect, therefore, simply follows the conventions

[2] Due to lack of space we cannot present examples of these pairs in this paper.

of the UML. In contrast, DOORS is neither an UML nor a requirements engineering tool that directly implements the RIF standard. As a consequence, we do not only have to elaborate a mapping of UML use cases to RIF concepts (as done by the definition of mapping I), but we also have to set up conventions how RIF concepts are represented in DOORS. In DOORS, primitive specification objects are called formal objects; so-called formal modules may contain formal objects of one type only (all objects of a module have the same attributes) and folders may contain folders or formal modules in turn.


DOORS Database: /UNISCON2008/Use Cases/e-commerce system/Storefront/C...

Location: /UNISCON2008/Use Cases/e-commerce system/Storefront/Catalog Br

```
DOORS Database
  UNISCON2008
    Use Cases
      e-commerce system
        Billing
        Inventory
        Site Administration
          Account Management
        Storefront
          Catalog Browsing
          Order
          Order Placement
```

Name	Type
Actors	Formal
DOORS Links	Link
Package Relations	Formal
Use Cases	Formal

Username: User type: Database Mana

Formal module '/UNISCON2008/Use Cases/e-commerce system/Sto...

Standard view, All levels

```
Actors
  1 Customer
```

	ID
1	1 Customer

Username: Exclusive edit mode

Formal module '/UNISCON2008/Use Cases/e-commerce system/Sto...

Standard view, All levels

```
Use Cases
  1 Browse catalog
    1.1 Associated Actor
      1.1.1 Customer
        1.1.1.1 end_B
        1.1.1.2 end_C
```

ID		navigable
1	1 Browse catalog	
2	1.1 Associated Actors	
3	1.1.1 Customer	
5	1.1.1.1 end_Browse catalog	True
4	1.1.1.2 end_Customer	False

Username: Exclusive edit mode

Fig. 5. Representation of use cases in DOORS

Figure 5 shows how our e-commerce use case example is represented in DOORS. Packages are mapped to a similar folder hierarchy with formal modules as leaves of this hierarchy. Actors and use cases are inserted as formal objects into modules. As a consequence the source and target *mappings II* and *mapping III* are defined as follows:

Mapping II source:

– A use case package is mapped to an EA package.
– The name of a use case package is equal to the name of an EA package.
– ...

Mapping II target:

– A specification group is mapped to a DOORS folder.
– The name of a specification group is equal to the name of a DOORS folder.
– Every DOORS folder that represents a specification group contains three formal modules and one link module.
– Exactly one of these formal modules is named 'Actors'.
– Exactly one of these formal modules is named 'Use Cases'.
– ...

Mapping III:
- An EA package is mapped to a DOORS folder.
- The name of an EA package is equal to the name of a DOORS folder.
- Every DOORS folder that represents an EA package contains three formal modules and one link module.
- Exactly one of these formal modules is named 'Actors'.
- Exactly one of these formal modules is named 'Use Cases'.
- ...

As one can see, the combination of mapping I and both mapping II specifications implicitly specify a first version of mapping III. The implicit specification is the result of a composition of mapping I and both mapping II specifications. Often the thus derived specification of mapping III has to be refined and explained in more detail building pairs of related modeling fragment instances and producing pairs of snapshots on the concrete syntax level. Despite of the above mentioned redundancies we recommend to specify mapping II and mapping III explicitly in all cases for the following reasons: in requirements engineering it is good engineering practice to create a list of goals for a project first, which is then used to derive a more detailed list of requirements as the appropriate means to achieve the selected goals. Thus, high-level concepts play the role of goals for a model integration project, whereas mapping III acts as a more detailed list of requirements that are still formulated from an end-users point of view. The mapping II definitions serve as a kind of traceability links between high-level goals and more detailed requirements.

The specification of *mapping IV* is simply the translation of concrete to abstract syntax representations of regarded modeling elements. The abstract syntax representations of the previously produced concrete syntax snapshots are instances (object diagrams) of tool data meta-models that will be regarded later on. The creation of the meta-model requirements and mapping IV normally go hand in hand. Finally, a first version of *mapping V* can be produced by building the composition of source and target mapping IV with mapping III. Often this derived version of a horizontal mapping is refined taken more specific details of the implementations of the two regarded modeling tools on the abstract syntax level into account.

Mapping V is the direct input for the creation of the formal specification of the regarded modeling tool integration by means of TGGs in the following section. Nevertheless, all the other mappings that have been created during the presented requirements analysis process are very useful for a systematic derivation of mapping V. Furthermore, assume that we want to replace one of the integrated tools by another one that supports similar modeling concepts, but uses (slightly) different syntax for that purpose. In that case parts of the requirements specifications produced above can be reused and we do not have to start from scratch again.

4 Integration Design

The design phase of the tool integration process consists of the construction of both tool meta-models and the definition of the mappings between them, as shown in Figure 3. Our meta-case tool MOFLON can be used for that purpose. It supports the OMG meta-modeling standard MOF 2.0 as well as TGGs for the declarative rule-based definition of model mappings. At first glance both the definition of a meta-model and the definition of a TGG specification look like low-level design or even coding activities. Therefore, we have to stress the fact that the handling of real-world examples also requires high-level design activities like decomposition of complex meta-models and model transformations into reusable, refinable, and parametrizable fragments. For further details concerning these meta-modeling-in-the-large activities the reader is referred to [7].

If not already provided by the developers of the regarded tools, tool meta-models are designed by *meta-model designers* on the basis of the requirements produced in the preceding requirements analysis phase. In our case neither DOORS nor Enterprise Architect come with their own meta-model despite of the fact that Enterprise Architect is a tool for the language UML with a rather sophisticated meta-model. Unfortunately, Enterprise Architect's API is not based on the UML meta-model, but uses its own small set of modeling language concepts. A meta-model specifies a more or less tool-specific abstract syntax of the regarded modeling language. Nevertheless, a tool meta-model may deviate from the implementation of the internal data structure and the API of the tool. It does provide a view onto the API of the tool that can be used by all subsequent activities throughout the tool integration scenario and has to reflect all required concepts. Within the tool meta-model there might exist additional elements that do not exist in the tool's implementation or it might have a more well-defined structure than the tool itself. It is for instance a matter of debate whether the constructed tool meta-model of Enterprise Architect either directly reflects its internal data structures with a small number of (meta-)classes such as "Element" or "Connection" or whether it relies on the UML meta-model and provides classes like "UseCase", "Association" etc. as we did assume in the preceding section.

The developed meta-models will be used by different activities of our MMDD process on different levels of abstraction. Integration rules will use the meta-model directly as model, whereas *adapter implementers* will use interface and implementation code generated from the meta-model. As a negative effect, the meta-model is a critical artifact. Changes will directly affect all subsequent activities.

Mappings are designed by *mapping designers* using the abstract syntax mapping requirements produced in the requirements analysis. Additionally both tool meta-models are required, which must, therefore, be in appropriate state before the mapping can be designed. All activities performed by mapping designers consist of modeling and generation steps only. If all inputs are present, the mappings can now be specified using TGGs. Every mapping will be represented by a TGG correspondence link type between a source element (e.g., package in Enterprise Architect) and a target element (e.g., folder in DOORS). For each correspondence link type, rules must be specified that describe the simultaneous evolution of both source and target models and the correspondence link model.

As it would go beyond the scope of this paper to completely introduce TGGs and how these mappings are specified, we refer the interested reader to [8].

After the mapping specification is complete, two more generation steps are required. Firstly, another model instance is generated that contains a more explicit representation of the mappings. During another generation step, this model is used to generate source code that realizes the specified mappings.

The more complex the mapping requirements are, the more time-consuming mapping definitions will be. To achieve a constant project progress we suggest an iterative approach. Mapping definitions should be incrementally completed step by step, defining reasonable subsets of mappings. On each intermediate result, there might be a generated code artifact. Consequential, the input for the following activity (see Section 5) is available earlier. This is advantageous, compared to producing all the mapping definitions (and the corresponding meta-models) in one single step.

5 Integration Implementation

The following activities include implementing software for direct usage inside the tool integration framework. Two roles are involved: (1) *tool adapter implementers* that work on tool adapters and (2) *integration service implementers* that work on integration services. The output of both activities are either a tool adapter or an integration service, which is a special kind of adapter.

Implementing a tool adapter highly depends on the definition of the associated meta-model (see Section 4). As input from the *meta-model design* activity, two software artifacts are processed during tool adapter implementation. The first artifact is a generated meta-model (adapter) interface compliant with the JMI[3] standard, which must be implemented by the regarded tool adapter. A generated standard interface implementation together with XMI[4] import and export functions is delivered as the second input artifact. It must be completed to achieve a tool adapter specific behavior. This can be difficult and depends on the grade of documentation, complexity, and maturity of the specific tool and its API.

Finally, the integration service implementation that depends on both source and target meta-model interfaces must be produced. It consists of two parts: (1) a repository for all required types of traceability links and (2) the consistency checking and translation code, which manipulates the integrated two models and the traceability links between them. Just a few pieces of source code have to be implemented manually during this activity (mapping, e.g., the traceability link repository interface onto the services of a DBMS); the largest amount of code is generated at the end of the mapping design activity as already explained in the preceding section.

The final step of the tool integration process is the instantiation of our integration framework. This instantiated framework is called *integrator*. The framework

[3] `http://java.sun.com/products/jmi/`
[4] `http://www.omg.org/spec/XMI/`

provides predefined extension points that are bound to previously developed tool adapters and integration services at run-time. The required adapters and integration services can be loaded via a graphical user interface of the *integrator* and are combined to scenarios. Each scenario thus realizes a tool integration process with a different set of properties including the set of integrated tools and the required set of services (forward and backward transformation, etc.). Furthermore, a scenario determines whether the model integration services work on a separate model repository or directly on the provided tool adapter interfaces. In the first case, the tool adapters are only used to export and import models to a repository with an identical interface, in the second case they are used to directly manipulate the models inside their tools.

6 Conclusion

In this paper we have presented a first version of a meta-model-driven tool integration development (MMDD) process with a main focus on the requirements engineering phase. The description of this process clearly identifies types of development artifacts, developer roles, and related categories of activities for requirement elicitation, design, and implementation phases of a V-model-like engineering process. Variations of this proposal can be developed easily that allow for the enactment of an iterative and incremental development process.

To the best of our knowledge our MMDD process is the first proposal of this kind for the following reasons: In the past the model transformation community laid the main focus on the design and implementation of new languages and tools that support efficient coding of model transformations. First proposals concerning high-level design (modeling-in-the-large, megamodeling) and systematic testing activities have been published rather recently, but we are not aware of any proposals how to organize the requirements engineering process of a model integration/transformation project—with one exception: [16] presents a scenario- and use-case-driven development process for graph transformations, which manipulate a single graph instance (model) only. We have adopted the idea of an example-driven approach from this publication and adapted it to the more general context of a model integration project.

Furthermore, it is worth-while to mention that the DBMS community has developed a plethora of schema mapping and integration techniques that are needed for the construction of federated database systems and for enterprise application integration purposes [9]. It is our impression that these techniques complement to a certain extent the approach presented here. The database system integration approach based on the definition of a reference schema may be adopted to refine the first high-level concept definition step of the presented requirements analysis procedure. Afterwards, we have to follow a different route for the following two reasons: the abstraction hierarchies that are useful for model integration purposes do not coincide with the n-layer abstraction hierarchies of the DBMS community. Furthermore, bidirectional rule-oriented model transformation specifications are rather different from high-level query definitions or low-level database change propagation implementations. It is up to future work to clarify this hypothesis.

Finally, we have to stress the fact that the presented MMDD process does not yet address quite a number of identified problems. Further research activities are, e.g., urgently required to develop more efficient tool adapter and model view creation procedures. Furthermore, we are just starting to learn how to validate model transformation and integration implementations systematically by combining appropriate formal verification [6] and dynamic testing techniques [15] developed by the graph transformation community.

References

1. Amelunxen, C., Königs, A., Rötschke, T., Schürr, A.: MOFLON: A standard-compliant metamodeling framework with graph transformations. In: Rensink, A., Warmer, J. (eds.) ECMDA-FA 2006. LNCS, vol. 4066, pp. 361–375. Springer, Heidelberg (2006)
2. Brown, A.W., Carney, D.J., Morris, E.J., Smith, D.B., Zarrella, P.F.: Principles of CASE tool integration. Oxford University Press Inc., New York (1994)
3. Conallen, J.: Building Web Applications With UML, 2nd edn. Addison-Wesley, Reading (2003)
4. HIS. Specification Requirements Interchange Format (RIF) (May 2007)
5. Jacobsen, I., Booch, G., Rumbaugh, J.: The Unified Software Development Process. Addison-Wesley, Reading (1999)
6. Kastenberg, H., Rensink, A.: Model checking dynamic states in GROOVE. In: Valmari, A. (ed.) SPIN 2006. LNCS, vol. 3925, pp. 299–305. Springer, Heidelberg (2006)
7. Klar, F., Königs, A., Schürr, A.: Model transformation in the large. In: The 6th Joint Meeting of the ESEC/FSE, USA,, pp. 285–294. ACM, New York (2007)
8. Königs, A.: Model Transformation with Triple Graph Grammars. In: Proc. Workshop on Model Transformations in Practice (2005)
9. March, S.T. (ed.): Heterogeneous databases. In: ACM Computing Surveys (CSUR) (Special issue), vol. 22. ACM, New York (September 1990)
10. Mens, T., Czarnecki, K., Gorp, P.V.: A taxonomy of model transformations. In: Dagstuhl Seminar 04101 on Language Engineering for Model-Driven Software Development (2005)
11. Nagl, M. (ed.): IPSEN 1996. LNCS, vol. 1170. Springer, Heidelberg (1996)
12. Schürr: Specification of Graph Translators with Triple Graph Grammars. In: Mayr, E.W., Schmidt, G., Tinhofer, G. (eds.) WG 1994. LNCS, vol. 903, pp. 151–163. Springer, Heidelberg (1995)
13. Schürr, Dörr,: Special Section on Model-based Tool Integration. Journal of Software & Systems Modeling 4(2), 109–170 (2004)
14. Sommerville, I.: Software Engineering, 8th edn. Addison Wesley, Reading (2007)
15. Stürmer, I.: Systematic Testing of Code Generation Tools. PhD thesis, Pro BUSINESS, Berlin (2006)
16. Zündorf, A., Leohold, J., Müller, D., Gemmerich, R., Reckord, C., Schneider, C., Semmelrodt, S.: Using object scenarios for requirements analysis - an experience report. In: Modellierung, Insbruck (2006)

Information and Communication Systems for Mobile Emergency Response

Janine Lachner and Hermann Hellwagner

Klagenfurt University, Department of Information Technology
Universitätsstraße 65-67, 9020 Klagenfurt, Austria
{janine.lachner,hermann.hellwagner}@itec.uni-klu.ac.at

Abstract. This discussion paper attempts to propose emergency response and disaster management as worthwhile areas of applied research for the information system community. The typical requirements, entities and activities involved in specifically mobile emergency response operations are summarized. Recent research contributions in this area are exemplarily reviewed in order to give a deeper insight into the role and use of mobile information and communication systems. Finally, the major challenges and research needs regarding information systems are summarized, with a view to draw the attention of information systems researchers to this interesting and important field.

Keywords: Emergency response, disaster management, information systems, mobile information and communication technology.

1 Introduction

Recently, there has been an increased research interest in deploying information technology (IT) in the area of disaster management and emergency response. Until a few years ago, the use of IT in this field was quite limited, but its potential for increasing the effectiveness of disaster response and recovery was well acknowledged, and technology and design proposals for integrated communication and information systems for disaster management came up [1].

In 2003, an international community was established with the goal of sharing knowledge and views regarding design and use of information systems in this field of research. This community was named ISCRAM – International Community on Information Systems for Crisis Response And Management [2]. Additionally, several research projects as well as conference and workshop series have been launched to address different aspects in this area.

Today, the IT branch considered most promising for the emergency response (ER) area is clearly *mobile* IT, simply because "an important part of this work is done in the field", yet "with little or no infrastructure to rely on" (cited from the preface of [3], the initial workshop of a dedicated series for mobile ER). Since "a primary challenge in responding to […] disasters is communication" [4], the authors of this paper, researchers in multimedia communication, decided to extend their work to this worthwhile area. After all, multimedia data such as images and video clips of a disaster site, or GIS information or presentations augmenting the reality of ER personnel, for instance, are gaining importance for ER operations. However, the underlying on-site communication infrastructure for these types of data is typically wireless, mobile, changing dynamically, in an ad hoc manner. These properties make research on multimedia communication in these environments an interesting endeavor.

Yet, in this discussion paper we intend to point out the role of, and the challenges for, information systems (IS) in mobile ER, in order to make the IS research community aware of this important application area and the research questions involved. Even as far as IS are concerned, the adoption of technology seems to be slow: "the use of IT is oftentimes still limited to the stationary headquarter dispatch system" [5], where "dispatch" denotes the activity of planning and sending resources (ER personnel, vehicles and material) to a disaster site and monitoring the response operation.

To show how information and communication systems may contribute to efficient and effective ER, we first review the typical entities, organization and operation of emergency services in Section 2. In Section 3, we exemplarily describe specific aspects and recent research contributions regarding mobile IT for ER (mainly from [3]), structured along the typical information flow in response and rescue operations. Section 4 attempts to work out the major research questions in the IS field that need to be addressed, and Section 5 provides concluding remarks.

2 Mobile Emergency Response Operations

A coarse classification for different types of disasters is given in [6] as follows:

- *Man-made* disasters, such as terrorist attacks, plane crashes, fire blasts
- *Natural* disasters, such as earthquakes, flood waves, tsunamis, avalanches
- *Epidemic or pandemic* disasters, such as widespread viruses and diseases.

Since in this approach some emergency situations may be classified in two or even all classes, this differentiation is not enough to derive special characteristics. A more specific distinction concerns the step-wise description of the process of ER [6]:

- *Detection* of an incident by searching the environment
- *Assessment* and classification of an observed incident
- *Alerting* responsible entities with the aim to respond
- *Mitigation* and planning of response activities
- Actual *response* and engagement of necessary resources
- *Recovery* and rehabilitation.

Depending on the particular characteristics and requirements of a disaster type or certain step in the process chain, different strategies and procedures regarding mobile emergency response have to be developed. In an attempt, though, to work out the typical entities and organization of disaster and emergency response operations and of corresponding communication, information and management systems, Figure 1 depicts the most common three-tier arrangement [1].

On the *strategic* level, the headquarters of public emergency agencies like police, fire departments, medical services, civil defense and other organizations, are alerted of a disaster situation, plan, allocate and dispatch the resources to respond to the incident, monitor their operations, provide information (e.g., maps) and instructions to the emergency site and re-allocate resources, if needed. Ideally, the decisions and activities among the different public emergency services are coordinated, also by means of and in terms of the involved IS. In large-scale disasters, consultation and coordination with government authorities is required as well.

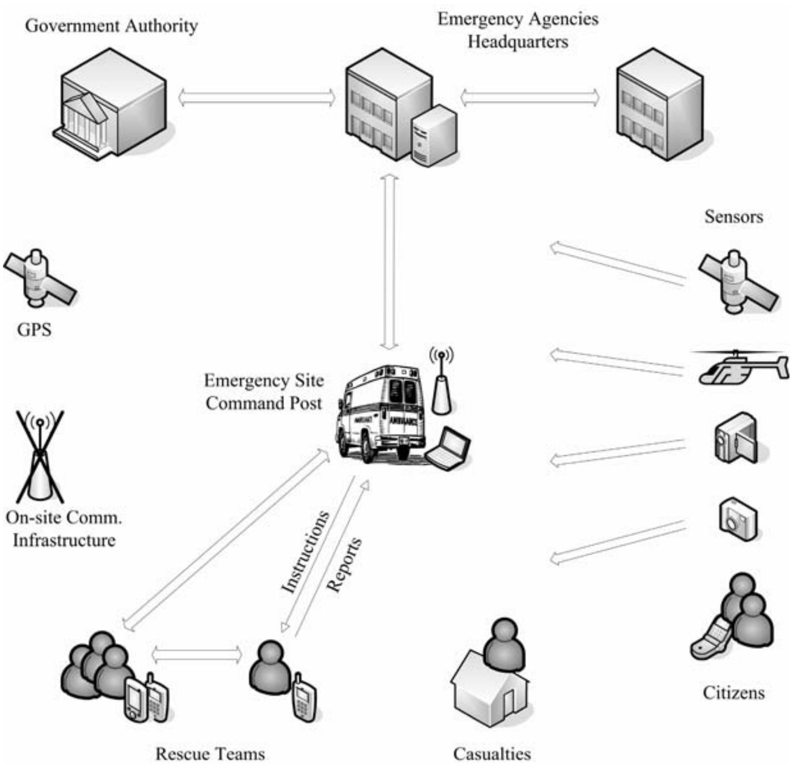

Government Authority
Emergency Agencies Headquarters
Sensors
GPS
Emergency Site Command Post
On-site Comm. Infrastructure
Instructions
Reports
Rescue Teams
Casualties
Citizens

Fig. 1. Three-tier organization of entities and operations in an emergency response scenario

On the *tactical* level, officers in the so-called (mobile) command post on the emergency site are responsible for on-site command and control of, and communication with, the rescue teams and/or individuals. Here, the on-site, dynamically changing operational picture (situational awareness), derived from the reports and data delivered by the responders and, increasingly, sensors, will lead to informed decisions and instructions on targeted actions to be performed by the response personnel. In other words, the task to be coped with on this level is basically "coordination of resources in space and time", yet in a hostile and risky space and mostly under significant time pressure [7]. The timeliness, precision and completeness of the information, decisions and instructions are crucial for the effectiveness of the operations and the safety of the response teams. The problem on this level is that an intact communication infrastructure cannot be assumed, yet reliable communication has highest priority. For voice communication, specific radio networks like TETRA are usually being employed in order to avoid "chaos in the air" resulting from traditional broadcast-type radio channels [1]. However, other data communication may have to take place over ad hoc, unreliable networks. A further issue is that some information needed for the response operation may not be available on site, but must be requested from the strategic level, e.g., a specific map, picture or stock list of hazardous goods.

On the *operational* level, the primary concern of the teams of responders is to provide immediate relief, medical care or other assistance to survivors, to fight the disastrous conditions (e.g., fire) or to find victims. Depending on the current situational view and their experience, the responders may adapt the decisions and instructions of the commanding officer or even the team leader in order to deliver the most effective help or not to risk their own lives. For their operation, it is of minor importance and mostly distractive to acquire data or to send reports about their situation to the commanders, thus data are increasingly being read from sensors and communicated to the tactical level automatically, e.g., vital parameters and amount of pressurized air of firefighters. The responders are now data sources, but clearly also data sinks, receiving, e.g., instruction messages, maps or data on missing persons [1]. Furthermore, they need to communicate and exchange data among themselves.

As already indicated above and illustrated in Figure 1, there are other data sources that may provide data (push mode) or may be requested data from (pull mode). On the one hand, civilians may offer or provide assistance and, e.g., be able to contribute pictures or video material to the response operations. On the other hand, several external sensor systems or sensor networks may be queried to deliver important data, e.g., GPS coordinates (most frequently used), video surveillance footage, aerial or satellite imagery, weather parameters, or measurements on toxic gases. This data may be requested by or supplied to any level in the hierarchy of ER entities.

Table 1 summarizes the major responsibilities, activities and characteristics pertaining to the entities involved in disaster response operations.

Table 1. Major activities and characteristics of entities involved in emergency response

Entities *Characteristics*	*Govmt. authorities* *Emergency agency*	*Emergency site command officer*	*Individual or team of responders*
Level of decisions and activities	Strategic	Tactical	Operational
Major responsibilities	Planning, resource allocation, dispatch, decisions, coordination, predictions	Dynamic on-site coordination of emergency forces	Immediate aid, relief, medical care, direct assistance
Data intensity of activities	High: exclusively based on data	Medium: dependent on precise and timely data	Low: minimum use of data desired
Data abstraction, aggregation level	High: mostly in aggregated form	Medium: aggregated situational view	Low: specific situational data
Spatial range of activities	Global	Emergency site	Local
Temporal scale of activities	Long	Medium	Short
Mobility	None or low	Low or medium	High

3 Information Flow in Mobile Emergency Response Situations

The flow of information between the involved entities as introduced above and, thus, the role and contributions of information systems can be considered as the most important factor regarding a successful mobile response to a disaster or emergency situation. In this context, different questions have to be asked:

- What information is necessary?
- How can the information be communicated (re. interfaces, devices, and networks)?
- Who is involved in the information flow?
- Where is the information required?
- What is the information used for and how long is it needed (persistency issues)?

The following sections attempt to summarize different recent approaches regarding the above identified questions. This is intended to give a more detailed picture of the requirements and procedures of mobile emergency response operations.

3.1 What Information Is Necessary?

It is crucial that certain information about the incident and conditions of involved people is communicated to the response teams. Several report categories including different information are required to handle a mobile response situation [8]:

- Mission related data such as administrative or insurance data
- Information about the incident itself (time and date of call, location of the incident, situational reports, etc.), which may differ widely among different ER agencies
- Clinical information (observation of vital signals, pupils, skin, etc.)
- Medical treatment/measurements such as defibrillation and medication
- Results and transfer data (assessment of injuries on handover).

Besides this basically patient-based and incident-based information, other contextual information such as the current location of a user, current time or the time of past or present events, user activities, user preferences and profiles of users should be considered. Information collected from disaster monitoring systems may also be needed to respond accurately.

Additionally, information regarding devices and networks, i.e., the information and communication system itself, is required. In order to achieve an improvement of the communication infrastructure of a mobile ER information system, information from two different domains is used: semantic information about the communication system itself (e.g., the purpose of deployment of the system) and information that can be extracted directly from the infrastructure (by evaluating network structure and activities, e.g., performance parameters). These two domains are not necessarily unrelated, as knowledge about semantics can be extracted from particular network activity.

3.2 How Can the Information Be Communicated?

In disaster response scenarios, the necessary information has to be communicated as fast and accurately as possible to the responsible entities. Failures or shortcomings regarding communication can sometimes make the difference between life and death [9]. Therefore, the used interfaces and devices as well as the underlying communication networks have to suit certain demands.

Visualization and Interfaces

An important aspect in a mobile response scenario is the evolution from paper-based emergency report forms to electronic solutions to document medical necessities and procedures at emergency scenes [8]. One simple but important advantage of electronic solutions is that immediate electronic transmission of patient data and diagnoses is provided, allowing an early preparation of involved entities, e.g., headquarters or hospitals. The paper-based information used to arrive comparatively late, e.g., together with the patient at the targeted hospital where both were handed over to the emergency department simultaneously. Other objectives that can be achieved by electronic solutions include simple, fast, and mobile data acquisition and collection, as well as automatic localization possibilities (via GPS) [10].

Recently, there have been nation-wide efforts of countries for logging and recording data in medical emergency situations. However, a cross-national (e.g., European) documentation to support an emergency situation affecting several countries – such as a natural disaster – is not yet available. In order to increase the collaboration and common quality level of medical procedures in emergency situations, the standardization and cooperation of medical emergency systems is desirable [8]. Research efforts in this direction are for example the *European Emergency Data* project (EED) [11] or the *Hesculaep* project [12], both supported by the European Union.

In order to provide documentation of the needed information in a clear and easy way for all involved entities, text and graphical elements are used to illustrate the information. Examples of the graphical representation are icons indicating a certain characteristic such as a fracture or burns, or a drawing of a human body illustrating different kinds of injuries at different body locations with different severity by highlighting the anatomical portions in different colors [8]. An automatic color coding of the fields based on the entered values can be helpful to give a quick idea of how close or far the clinical values are from normality. It has to be guaranteed that the illustrations are both, big enough to be readable and selectable on a touch-screen by a finger or a stylus, and anatomically detailed enough to provide for a detailed, meaningful analysis of the injuries [8]. Additionally, this approach can be used to provide quick feedback by highlighting possible input errors [9]. Regarding text input, an interesting way to support the ER teams is handwriting recognition and generally recognition of pen-input [13], or to provide an on-screen virtual keyboard with automatic word completion [9]. In general, a system that is too complex will most likely not be accepted by the mobile response team [8].

Multi-modal interfaces proved to be especially important for disaster response scenarios. The goal of these approaches is to provide a connection to an interactive system with different kinds of hands-free input possibilities, such as speech input or pointing to objects. E.g., project *AMIRA* [14] addresses several issues regarding multimodal intelligence, with a focus on speech interaction. However, the utilization of only speech recognition is sometimes limited or not appropriate, due to environmental noise in ambulance trucks or helicopters [9].

Mobile Response Devices

A straightforward approach is to apply Tablet PCs or PDAs as communication devices in mobile ER situations. In [10], an overview of special requirements regarding the hardware is presented. The size of the display has to be small to be wearable, but on the other hand, the information needed must be visible and interaction with fingertips or a stylus on a touch screen has to be provided. Most display sizes of PDAs are comparably small, whereas the bigger sizes of Tablet PCs might result in unhandiness in a response situation. Another important aspect regards the weight of the handhelds, which should be minimal in emergency situations in order not to interfere with the rescue efforts. Energy-aware devices and a long battery lifetime are necessary to support the working hours of the emergency personnel. Features such as charging the device inside a vehicle should be considered to improve the availability of the devices. In order to operate in a disaster or emergency situation, special coatings and sealants for the hardware are needed. The protection is necessary if the devices are exposed to extreme temperatures, humidity, water sprays, dust and vibration.

In order to evolve from simple handheld devices to wearable "hands-free" devices, a first approach is the integration of handhelds into ER vehicles or into the clothing. In [8], the integration of a suitable PDA into a special environment suit for the medical personnel is presented. The display content of the PDA is visualized by a head-up display. Additionally, speech recognition is included to navigate and control the program in order to provide a hands-free approach that could be an advantage for physicians and other responsible persons.

A desirable approach is to include smart sensor systems, i.e., micro- and nanotechnology-based wearable equipment that can be directly integrated into garments and textiles. These systems are furthermore able to record heart and respiratory rate, temperatures, as well as the absolute position via GPS. In [15], the design of garments and textile sensors is discussed in detail. Also IST projects are interested in the applicability of computer systems that are integrated in clothes. E.g., the objective of *wearIT@work* is to support users in unobtrusive ways by wearing computer systems including sensors as a computer-belt [16].

Networking Issues

In emergency situations, an efficient communication system is crucial to be able to exchange the information collected in different areas. Disasters such as earthquakes or hurricanes can cause an existing communication infrastructure (e.g., telecommunication) to get damaged [17]. Therefore, a number of research activities deal with the establishment and operation of mobile ad-hoc networks, spontaneous networks composed of single nodes communicating with each other, without having a fixed infrastructure as in common IEEE 802.11 wireless networks.

3.3 Who Is Involved in the Information Flow?

The question of which entities are involved in an emergency response scenario can – at a first glance – be answered intuitively, as done in Section 2: the headquarters of the response activity, the on-site command and control post, the mobile rescue teams, as well as probably hospitals or other reception camps. However, if we consider that the information is communicated over a physical network, the flow of information can affect several other entities as well. Assuming an ad-hoc network infrastructure as discussed in Section 2, all nearby communication nodes can work together to allow for a stable communication. Thus, patients and passers-by can just as well contribute to the information flow as a helicopter surveying the surroundings, a wireless access point in a nearby building, or a robot participating in the response mission.

3.4 Where Is the Information Required?

Before this question can be answered, a more important aspect has to be addressed. It is crucial to identify where "where" is, i.e., spatial considerations regarding the participants in an information flow have to be made. In the emergency domain, much relevant information is inherently attached to places – and therefore "spatial" [18].

With the currently available GPS technology, it is possible to locate the (almost) exact position of a person. Location-based services exploit this technology to offer the user services that are of special interest for users in this area, without the need to enter a ZIP code or any other form of spatial information.

An enhancement of location-based services is mobile spatial interaction. This approach allows combining digital information with the user's direct surroundings [18]. Mobile spatial interaction has been discussed for several years, in [19] a next generation GIS is presented, featuring Smart Compasses, Smart Horizons, Smart Glasses, and Geo-Wands. These appliances combine a hand-held computer with a GPS receiver, a cellular phone, and a digital camera and enable the user to integrate spatial analysis into daily life activities. For ER situations, these spatially aware mobile phones could provide meaningful interaction with real-world objects, e.g., by attaching meaningful information (such as a label) to 3D objects or segments of the user's surroundings. Continuous interplay of virtual and spatial information by overlaying the user's field of view could be realized by devices like Smart Horizons or Smart Glasses. Another advantage of mobile spatial interaction over location-based systems is that a more accurate selection of, e.g., parts of a building, or a street is possible. This more fine-grained approach allows adding tags, e.g., only to a specific part of the roof that is of interest for the mission. Project *Point-to-Discover* by the Telecommunications Research Center Vienna tries to develop technical foundations for mobile spatial interaction [18].

The *SHARE* project [20] uses intelligent visualizations to support the operation management by giving "everybody the right information at the right time in the right way" [21]. The data regarding spatial information (such as streets, buildings, and infrastructure) is stored in spatial database management system (PostGIS, Oracle10g) that provides secure and fast transaction mechanisms. The information can be accessed through a map server (Geo Server) and interoperable Map Web Services to support different output devices (PCs, PDAs, laptops). Domain knowledge is used to represent the data and processes in a semantic structure, in order to allow queries for different purposes. E.g., a request regarding available water resources in a special area results in a list including the existing hydrants in a 500 m circle around the user's position (using a spatial filter) [22].

If the location information is available, it is important to decide where which information is needed, as well as when it is needed. The management of resources and data in search and rescue operations, most times supervised by the commander at the headquarters, is crucial. For example, spatial information can help by coordinating search activities or illustrating the spread debris at a crash site. However, all types of contextual information as discussed above can support to organize and manage the rescue mission and related processes.

3.5 What Is the Information Used For and How Long Is It Needed?

The first part of the question pertains to the major ER entities, responsibilities and activities as pointed out in Section 2, specifically in Table 1, and is thus not further detailed here.

The information obtained regarding the environment of an ER situation has different importance at different times. Some data is needed only for a very short moment, such as the current pulse rate of a patient or a currently available communication path in ad-hoc networks. Information about the current location of a user is only valid and interesting as long as the user remains at this position. Other data, however, can be important for the whole response action. Examples for such information are the location of the incident or the number of affected people and rescue teams. And then there is information that might be of concern for a long time, beyond one rescue mission. For use in post-disaster analyses as well as for simulation and training purposes, it is important to have a detailed log of the situational analyses, communication processes and information exchange during the response operations; future missions and the skills of the response teams (at all levels) can heavily benefit from such data. Other persistent data can be related to the diagnosis and medical treatment of casualties.

The comprehensive knowledge base that evolves if all possibly related information is merged, analyzed, and evaluated, can support decisions. An accurate representation of the knowledge, e.g., in ontologies, can help to make decisions by using reasoning systems, and thus, save lives. The project *AMIRA* [14] combines various components to enable the creation of a diagnostic and decision support applications for the management of critical incidents such as emergencies or disasters. Example components in AMIRA are a knowledge engine and a case-based reasoning approach [14]. Systems for collecting and handling emergency medical care data can as well significantly improve the effectiveness of rescue operations. Relevant information are efficiently recorded and communicated between on-site teams and headquarters. Additionally, an on-site patient classification is possible, by using information from medical databases to decide a proper course of action in the field [9].

4 Research Needs

In this section, we attempt to summarize the major research needs for (mobile) emergency response, mainly pertaining to the deployment and further development of information systems. This summary is inspired by our study as partially laid out before, as well as by [1], [6], [7], [14], and [23] (cited and excerpted in [7]). Communication issues are not covered in the following since they are of minor importance to the IS community.

- *Interoperability of information systems* (and of other technologies and of working practices). Large-scale disaster response is a multi-agency, multi-entity, complex operation that requires close coordination among the agencies and responders. While information systems should support these coordination actions and provide a consistent "joint operational picture" [7], the current status is that the jurisdictions, terminologies, deployed systems, data models, views, etc. do not interoperate across agencies, let alone across nations. Semantic Web technologies in general seem to provide appropriate means to address these obstacles. Standardization efforts like the ones of the OASIS Emergency Management Technical Committee [24] are underway and have produced standards like the Common Alerting Protocol (CAP) and the Emergency Data Exchange Language (EDXL) [25], but need to be extended for coordination on the strategic and tactical levels to become more efficient and effective.

- *Knowledge representation/management and decision support.* The activities involved in disaster response on the strategic and tactical levels, i.e., in general, resource management and coordination, are based on informed decisions and resulting targeted actions. The decisions rely on current situational awareness, the emergency agencies' rules, guidelines and procedures, and the expertise of the people involved, but should also make use of prior knowledge in the field, similar earlier cases and of experience of people not concerned with the mission at hand. Such rich, partially archival knowledge and its use calls for advanced techniques for knowledge representation and management as well as for systems that support the sense-making and decision-taking processes [26]. Again, techniques from Semantic Systems and Artificial Intelligence, like ontologies, case-based reasoning as described above and other conceptualizations as well as planning and reasoning techniques [27] have to be investigated and adopted here.
- *Flexible workflows and services.* Emergency response operations can be modeled as workflows, yet they must be extremely flexible since new events and information on the emergency situation can at any time significantly change the picture and modify the ER requirements and the processes taking place. The major issue is that the exchange of data and information can happen spontaneously and among any participating entities, thus comprehensive planning for workflow modifications in advance becomes impossible. A new IS architecture based on the Actor-Agent Communities (AAC) model was proposed here [6], where humans (actors) and software components (agents) form a complex network of autonomous information processing entities that collaborate. Other work [1] (as early as 2002) proposes a model of flexible and diverse information services where services configure themselves "dynamically and automatically" and "present the right data to the right actor at the right time with as little human intervention as possible" [1]. We are convinced that recent developments and achievements in the (Semantic) Web Services and Service Oriented Architectures research fields can bring most valuable contributions to IS for disaster response applications and need to be incorporated.
- *Rich data and information integration.* As indicated in Figure 1 and discussed in Section 2, disaster management involves a multitude of heterogeneous, distributed and dynamically changing data sources and data sets, ranging from maps and other GIS data to recorded voice messages, images and videos, and to governmental databases. Even "outsiders" like civilians not directly involved in the incident and several sensor data sources may provide valuable data and information to the response operations. It is a specific challenge for IS to be open for accepting and

integrating such rich data from diverse, probably unknown sources, to correlate (fuse) several data instances for a consistent situational picture, and to transfer the data and information to the right responsible entities. A prototype system was presented in [28]. In this context, security is of utmost concern since such openness can be abused for malicious attacks on the whole ER operation.

5 Conclusion

This discussion paper gave an overview of requirements, entities and activities involved in emergency response operations, mainly utilizing mobile information and communication technology (ICT). Selected recent research approaches and contributions were reviewed to give a more detailed picture of the problems involved in deploying ICT for disaster response activities. Subsequently, major research needs in the information systems area were worked out.

In conclusion, we are convinced that the major IS-related challenges of disaster response operations – namely, interoperability of information systems, knowledge representation and management, decision support, provisioning of flexible workflows and services, integration of rich data sets and information – can be successfully addressed using modern concepts and techniques from the research areas Service Oriented Architectures, Semantic Systems, Artificial Intelligence, Semantic Web, and Information Security. The IS community is invited to consider emergency response and disaster management as a challenging and worthwhile area for their research.

References

[1] Meissner, A., Luckenbach, T., Risse, T., Kirste, T., Kirchner, H.: Design Challenges for an Integrated Disaster Management Communication and Information System. In: First IEEE Workshop on Disaster Recovery Networks (DIREN 2002), New York, NY, USA (June 2002)

[2] International Community on Information Systems for Crisis Response And Management, http://www.iscram.org/

[3] Löffler, J., Klann, M.: Mobile Reponse 2007. In: Löffler, J., Klann, M. (eds.) Mobile Response 2007. LNCS, vol. 4458, Springer, Heidelberg (2007)

[4] Manoj, B.S., Hubenko Baker, A.: Communication Challenges in Emergency Response. Communications of the ACM 50(3), 51–53 (2007)

[5] Meissner, A., Eck, R.: Extending the Fire Dispatch System into the Mobile Domain. In: [3]

[6] Nieuwenhuis, K.: Information Systems for Crisis Response and Management. In: [3]

[7] Baber, C., Cross, J., Smith, P., Robinson, D.: Supporting Implicit Coordination Between Distributed Teams in Disaster Management. In: [3]

[8] Waldher, F., Thierry, J., Grasser, S.: Aspects of Anatomical and Chronological Sequence Diagrams in Software-Supported Emergency Care Patient Report Forms. In: [3]

[9] Chittaro, L., Zuliani, F., Carchietti, E.: Mobile Devices in Emergency Medical Services: User Evaluation of a PDA-Based Interface for Ambulance Run Reporting. In: [3]

[10] Hafner, C., Thierry, J.: Feasible Hardware Setups for Emergency Reporting Systems. In: [3]

[11] European Emergency Data (1997-2002), http://www.eedproject.de
[12] Hesculaep Project, http://hesculaep.andaluciasalud.com/webenglish/0101.asp
[13] Willems, D., Vuurpijl, L.: Designing interactive maps for crisis management. In: ISCRAM 2007 Conference, Delft, The Netherlands (May 2007)
[14] Auriol, E.: AMIRA: Advanced Multi-modal Intelligence for Remote Assistance. In: [3]
[15] Bonfiglio, A., Carbonaro, N., Chuzel, C., Curone, D., Dudnik, G., Germagnoli, F., Hatherall, D., Koller, J., Lanier, T., Loriga, G., Luprano, J., Magenes, G., Paradiso, R., Tognetti, A., Voirin, G., Waite, R.: Managing Catastophic Events by Wearable Mobile Systems. In: [3]
[16] WearIT@work Project, http://www.wearitatwork.com
[17] Nguyen, H., Gyoda, K., Okada, K., Takizawa, O.: On the Performance of a Hybrid Wireless Network for Emergency Communications in Disaster Areas. In: Third International Conference on Networking and Services, Greece (2007)
[18] Fröhlich, P., Simon, R., Kaufmann, C.: Adding Space to Location in Mobile Emergency Response Technologies. In: [3]
[19] Egenhofer, M.: Spatial Information Appliances: A Next Generation of Geographic Information Systems. In: 1st Brazilian Workshop on GeoInformatics, Campinas, Brazil (October 1999)
[20] SHARE Project, http://www.ist-share.org
[21] Andrienko, N., Andrienko, G., Gatalsky, P.: Exploratory Spatio-Temporal Visualization: an Analytical Review. Journal of Visual Languages and Computing, Special Issue on Visual Data Mining 14(6), 503–541 (2003)
[22] Ernst, V., Ostrovskii, M.: Intelligent Cartographic Presentations for Emergency Situations. In: [3]
[23] U.S. National Research Council: Summary of a Workshop on Using Information Technology to Enhance Disaster Management. The National Academies Press, Washington (2005)
[24] OASIS Emergency Management Technical Committee, http://www.oasis-open.org/committees/emergency/
[25] Iannella, R., Henricksen, K.: Managing Information in the Disaster Coordination Centre: Lessons and Opportunities. In: ISCRAM 2007 Conference, Delft, The Netherlands (May 2007)
[26] Potter, S., Kalfoglou, Y., Alani, H., Bachler, M., Shum, S.B., Carvalho, R., Chakravarthy, A., Chalmers, S., Chapman, S., Hu, B., Preece, A., Shadbolt, N., Tate, A., Tuffield, M.: The Application of Advanced Knowledge Technologies for Emergency Response. In: ISCRAM 2007 Conference, Delft, The Netherlands (May 2007)
[27] Pottebaum, J., Konstantopoulos, S., Koch, R., Paliouras, G.: SaR Resource Management Based on Description Logics. In: [3]
[28] Fahland, D., Gläßer, T.M., Quilitz, B., Weißleder, S., Leser, U.: HUODINI – Flexible Information Integration for Disaster Management. In: ISCRAM 2007 Conference, Delft, The Netherlands (May 2007)

Tacit Knowledge Management: Do We Need a Re-orientation of Traditional KM Approaches?

Franz Lehner

University of Passau
Innstr. 43
D-94032 Passau
lehner@uni-passau.de

Abstract. Knowledge Management is meanwhile generally recognized as a discipline established in many organisations and no longer needs a special justification. It is certainly not unrestrictedly positively seen and even some of its supporters saw a hype in the development of the last few years, which is fading and is being replaced by a realist view of what is feasible. In the attempt to gain a proof of success, one bumps into further difficulties since the effects of KM activities are often not easily or explicitly measurable. It is not clear if this would not have developed as positively when not measures would have been taken for the development of knowledge management. The goal of this paper is the attempt of a comprehensive explanation of a new theme which is incorporated in Tacit Knowledge Management (TKM), as well as a proposal for the integration of TKM in the traditional understanding of knowledge management.

Keywords: Knowledge management, tacit knowledge, organisational knowledge, KM paradigm, KM views, KM activities, TKM.

1 Background and Relevance of the Topic

Knowledge Management is meanwhile generally recognized as a discipline established in many organisations and no longer needs a special justification. It is certainly not unrestrictedly positively seen and even some of its supporters saw a hype in the development of the last few years, which is fading and is being replaced by a realist view of what is feasible. In the attempt to gain a proof of success, one bumps into further difficulties since the effects of KM activities are often not easily or explicitly measurable. It is not clear if this would not have developed as positively when not measures would have been taken for the development of knowledge management. The goal of this paper is the attempt of a comprehensive explanation of a new theme which is incorporated in Tacit Knowledge Management (TKM), as well as a proposal for the integration of TKM in the traditional understanding of knowledge management.

Based on the predominant positive experience with knowledge management, in spite of the criticism, and the fact that knowledge management is, in the meantime, for some tasks indispensable, the hitherto development shouldn't be basically questioned. The discipline of Knowledge Management seems the have come to a point, however, in what a methodical further development and a precise understanding of the addressed phenomena is imperative. At the same time, included is the analysis of so far neglected aspects of Knowledge Management. The feedback from management about KM in practice often brings up disappointment and reports about project failures. Beyond that poor results several question arise. Are we focusing the essential aspects? Are we taking the right measures to solve practical problems? How can we make sure that KM works at all?

Knowledge is the recognition of the relationship between cause and effect and is based on a systematic connection of information. The capability to use, adapt and further develop knowledge, to render it into a profitable, innovative product and service will mark the competition positions of the companies in the next years.

Polanyi originally divided knowledge into two forms, namely explicit and implicit knowledge. Although, one has a differentiated understanding of knowledge today, these differences still play an important role.

Explicit knowledge is knowledge, which can be verbalised and which can be stated. It can be described, questioned, discussed and transported through phrases in different forms, e.g. texts documents, data bases, mathematical formulas. Other people can work with this knowledge, it can be easily transmitted, and it can be understood through logical thinking

Implicit knowledge is however the part of knowledge, which cannot be totally expressed or understood in words. It includes skills, experiences, beliefs etc. and doesn't need extensive formal recording. But it is an important part of human behaviour. The existence and the importance of hidden knowledge (tacit knowledge) was first recognised and described by Polanyi (1966). Implicit knowledge is based on individual experience, personal ideas, beliefs, perspectives, philosophy of life, ideals, values and emotions. It includes skills, action routines, convictions, doctrine and mental schemas.

Knowledge should be treated as a production factor and be used as efficiently as possible. Knowledge management in connection with the modern management practice traditionally includes (among other activities)

- The "management" of externalised knowledge,
- supporting of the externalization of knowledge,
- the use of the externalised knowledge in connection with the operational tasks by the employees or software.

The externalised knowledge of an organisation is e.g. in its data bases, archives etc. and can in this way be optimised or examined. Implicit knowledge about methods of Knowledge Engineering are transformed through specific processes into explicit knowledge with varied success. Interesting is the not externalised knowledge (similar as with the human memory) whereby emergent structures have the greatest influence (e.g. memory, will of learning, absorption speed). Especially with these questions in the background, a concentration on externalised knowledge oversimplifies the real matters and blinds out many aspects. Knowledge itself is only found in humans resp. in the interaction of humans. Based on the foundation of this realization, a broader understanding of Knowledge Management has already been propagated in the last few years.

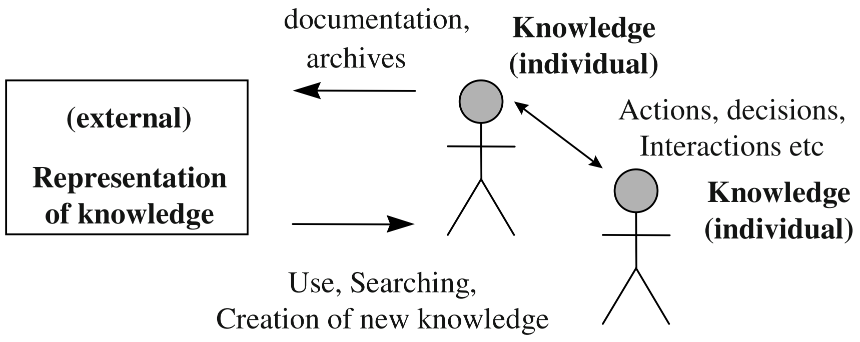

documentation, archives
Knowledge (individual)
(external) Representation of knowledge
Actions, decisions, Interactions etc
Knowledge (individual)
Use, Searching, Creation of new knowledge

Fig. 1. Context of knowledge management activities

2 Initial and Traditional Understanding of Tacit Knowledge Management

As a result of a growing interest in the existing knowledge of a company or an organisation as decisive success factor within economic competition, "tacit knowledge" is experiencing greater attention next to explicit knowledge. The goal of a European research projects about "Tacit Managerial Knowledge" (cf. http://www.tacit-knowledge.org) is to create a theoretical foundation for tacit knowledge in the area of management, which reflects the significance of the construct in professional success as well as containing the separation of different connected purposes and an explanation of terms.

The main interpretation of Tacit Knowledge Management aims so far at the management of hidden knowledge and the term TKM was used for the first time in this context, (cf. e.g. Kreiner 2002). Tacit or hidden knowledge, according to a study by Daniel Golem, makes up around 80% of the knowledge in organisations. By succeeding at least a part of this hidden knowledge to be made transparent, a considerable development potential would be assumed for the business.

A special significance is given in a business context next to the individual or personal knowledge in the collective and organisational knowledge. Schneider describes collective knowledge as a mixture of explicit and hidden or implicit knowledge in a relationship net. Collective knowledge is, in this process, an important source for competitive gain. Because of its nature, it is very difficult to dismantle, reproduce or transfer to other organisations. (cf. Schneider 1996, 21-22 as well as "Paradox of Replication" and the "Paradox of Intelligence"). Nonaka/Takeuchi define collective organisational knowledge to be understood as similar to " ... capability of company as a whole to create new knowledge, disseminate it throughout the organisation and embody it in products, services and systems" (Nonaka/Takeuchi 1995, 12). The process of development of collective knowledge requires the continuous cultivation of four forms of knowledge creation. In the shaping of each one of these four forms, it's a matter of a transition between explicit and hidden or implicit knowledge. Figure 2 gives an overview of these connections.

Transition	transition form	processes and examples
explicit -> explicit	***combination***	Traditional learning process, information Exchange, scientific research
explicit -> hidden	***Internalisation***	Automatisation, Transition to skills / expertise
hidden -> explicit	***Externalisation***	Reflexion, Reconfiguration
hidden -> hidden	***Socialisation***	Shared experience, imitation, exercises

Fig. 2. Typology of the knowledge formation (Schneider 1996, 22)

Figure 3 shows a further precision of explicit and implicit forms of knowledge. Wiegand points out specifically, in this connection, to the similarity with other diagrams (e.g. with Hedlund/Nonaka). In this case, though the ability to transfer knowledge is the main point of interest, based on Wiegand's assumption, that implicit knowledge generally cannot be transferred without intensive and personal contact. Furthermore important is the distinction of the different learning levels, which are also supported by different learning processes. Learning is then a further mechanism to collect knowledge.

learning level	explicit knowledge	Implicit knowledge
individual	Phone numbers	Biking, playing an instrument
group	Scope and goal of a project	Group norms
organization	Company goals	organizational norms, culture
knowledge community	Qualification for an occupational group	procedural routine in a specific situation.

Fig. 3. Examples of personalized knowledge forms (according to Wiegand 1996, 329)

In this context, a phenomenon should be pointed out, which one can observe in the area of speech and individual behaviour. Every individual speech is made up of passive and of an active vocabulary. The passive vocabulary is the total amount of all the words which a person knows und which help one to understand a text or a conversation, even if one never uses these words personally. We know, then, actually much more than what we can express (tacit knowledge). The active vocabulary is made up of a small amount of words and phrases which we often use and of which our thinking pattern is made up of. These words or phrases are then unconsciously used and reused. In this way, we move about in long practiced and learned thinking categories. The expansion or changing of these structures is connected with learning processes that can be laborious and rather time consuming. Very similar is the situation with individual behaviour. Also in this case, we have an active and a passive "behaviour vocabulary". The active behavioural vocabulary is defined through the total of habits, rules, norms, values etc. which make up our typical behaviour and actions.

This idea can be transferred with small modifications on organizations. Certain events and observations engrave themselves so deeply and lasting, that later they are very difficult to erase or overcome. This level of behaviour is directed to e.g. the bases of teaching of the organization and the organizational development. The whole behavioural disposition of an organization is established therefore in the organizational memory. In this connection, one must not forget that not all memory processes are really understood and that the analogy between human and organizational memory has its limits or can lead to misunderstanding. When putting too much emphasis on human memory, and here on defined trusted functions (as "forgetting"), the danger is that the development of knowledge management systems follows these patterns without critical reflections.

Coming back to the initial idea a conclusion is that the multiple aspects and phenomena around tacit knowledge so far are not addressed sufficiently. There is much evidence that a focus on tacit knowledge would be too narrow.

3 Proposal of an Extended View of Tacit Knowledge Management

The term TKM is not totally new but has not been used very often. Up to now it was used in the sense of managing tacit knowledge (especially in the meaning of making it explicit, or transferring between organisational units). Trying to summarize experiences from more than ten years of practical and theoretical work on KM we found that there are more tasks, activities etc. which are clearly related to KM but at the moment definitively not discussed within the field of KM. These activities are linked to KM but hidden, forgotten or unconscious.

Based on previous considerations, it is becoming clear that there are more tasks and activities in knowledge management than have been consciously perceived until now. A closer look at the postulated tasks and concepts of knowledge management brings us to a phenomenon that could be described in the analogy of "tacit knowledge" as Tacit Knowledge Management (T-KM). Compared to the previous usage of the term, a new and broader view of TKM will be suggested, whereby the following tasks and focus areas could be differentiated (including some overlaps). The topics identified so far which are not included by traditional understanding of KM or KM approaches shall serve as proposal for initiating a more substantial discussion and can be described as follows:

- We find a lot of companies with no or little conscious KM-activities – KM "happens" (nevertheless the question arises in which situations an active knowledge management is above simply letting things happen).
- The practically necessary activities do not refer to shared knowledge, resp. don't require the measures recommended in KM literature (theory – practice gap)
- KM-activities are intentionally introduced but are not known to all (resp. not to all that should know about them). Especially in bigger organisations uncoordinated KM-activities can be the consequence. TKM in this sense can mean a reduction of knowledge deficits about KM-activities.

- KM activities concentrate on information sharing, while KM processes and knowledge sharing are neglected (nevertheless they exist)
- Consequences of existing but not explicitly communicated goals of knowledge management (hidden agenda of KM resp. Management)
- Essential KM-processes understood as "autopoietical" (self-organising)
- Significance of hidden knowledge structures; i.e. informal structures and relationships, which have a specific meaning and which are actually more important than formal structures and tasks (under control of KM)
- Lack of consciousness about the knowledge with business relevance (as a consequence it is not clear what should be addressed by KM)
- Explicit KM activities are related to the business activities – and contrast to hidden and not communicated expectations (e.g. related to unexpected events)

If we admit that more KM activities than intentionally or consciously initiated take place the question is what are the implications and where is the border between explicit and implicit KM activities?

Tacit Knowledge Management has to do with the circumstance that in a company or an organisation more processes of knowledge management take place then are consciously initiated or known of (i.e. that the generally shared knowledge about KM or KM activities is too small). As there is tacit knowledge, there will also be TKM because of self organization. It will be important in the future to not let it have an accidental development. The setting up of Tacit Knowledge Management makes it possible for for an organisation to understand knowledge management in its entirety and to take definite measures to e.g. eliminate the deficit of meta knowledge about KM activities. The initial thoughts presented here could form a base for a more general concept and should demonstrate the need for more empirical evidence in the whole field of knowledge management. The success of knowledge management should not be left to an accidental development, but extended to an important and until now unnoticed field.

Immediate implications and conclusions are:

- The List of KM activities (within traditional approaches or models) should be extended.
- Discipline of KM should start to create a systematic list of generally accepted problems and phenomena related to KM. If we really want to help organizations we need to know what is "normal" and what is not. Therefore, empirical findings should be collected, integrated and systematically documented based upon a common understanding of KM.
- Empirical research should be intensified in order to clarify KM impacts as well as responsibilities (reducing uncertainty and vagueness).

We can finally use the metaphor of an iceberg to figure out possible consequences. The visible part above the water level would be formed by the traditional, explicit KM activities including KM projects, tools, staff etc. The rest is below the surface and nobody is aware of the real impacts until a "collision" happens. As this is just a metaphor or a concept from a scientific point of view it is quite unclear if we ever can draw a precise line between explicit and implicit parts of KM and how we can measure the real (overall) impact of KM.

References

1. Brauner, E., Becker, A.: Beyond Knowledge Sharing: The Management of Transactive Knowledge Systems. Knowledge and Process management 13(1), 62–71 (2006)
2. Goleman, D.: Emotional Intelligence. dtv Publ. Comp. (1977)
3. Hedlund, G., Nonaka, I.: Models of Knowledge Management in the West and Japan. In: Lorange, P., et al. (Hrsg) Implementing strategic processes, Oxford, pp. 117–144 (1993)
4. Kreiner, K.: Tacit knowledge management: the role of artifacts. Journal of Knowledge Management 6, 112–123 (2002)
5. Lehmann, H., Lehner, F.: Perspectives of knowledge sharing for the global firm: A transatlantic perspective Seventh Annual Global Information Technology Management (GITM) World Conference, Orlando, Florida, June 11, 12, & 13 (2006)
6. Lehner, F.: Wissensmanagement. München (2006)
7. Lehner, F., Maier, R., Klosa, O.: Organisational Memory Systems - Application of Advanced Database & Network Technologies in Organisations. In: Reimer, U. (Hrsg) Practical Aspects of Knowledge Management. Proceedings of the Second International Conference PAKM 1998, Basel, October 29-30, 1998, vol. 12, pp. 14-1–14-12 (1998)
8. Maier, R., Lehner, F.: Perspectives on Knowledge Management Systems – Theoretical Framework and Design of an Empirical Study. In: Hansen, H.R., Bichler, M., Mahrer, H. (Hrsg) Proceedings of the European Conference on Information Systems – ECIS 2000, Wien, Juli 3–5, 2000, vol. 1, pp. 685–693. Wirtschaftsuniversität, Wien (2000)
9. Nonaka, I., Takeuchi, H.: The Knowledge-Creating Company. How Japanes Companies Foster Crearivity and Innovation for Competitive Advantage. Oxford University Press, London (1995)
10. Nonaka, I.: The Knowledge-Creating Company. Harvard Business Review, 96–104 (November - December 1991)
11. Polanyi, M.: The Tacit Dimension, New York (1967)
12. Schneider, U. (Hrsg): Wissensmanagement, Die Aktivierung des intellektuellen Kapitals. Frankfurt (1996)
13. Wiegand, M.: Prozesse organisationalen Lernens, Wiesbaden (1996)

Collaborative Software Development: A Case Study of Model Interdependencies

Peter C. Lockemann

FZI Forschungszentrum Informatik an der Universität Karlsruhe,
Haid- und Neu-Str. 10-14, 76131 Karlsruhe

Abstract. Models are an abstraction of a physical or mental world or state of affairs. Since an abstraction concentrates on those aspects of the world deemed most salient or important, models always have some special objective in mind. Since different objectives may arise along a typical value chain, a corresponding model chain may arise. This paper uses a case study to examine the value chain along software development projects and discusses which models were encountered. Clearly, the choice of one model restricts the choice of models or the ensuing steps in the chain. The paper attempts to identify general relationships between models that reflect the restrictions.

Keywords: Meta-models, model interoperability, model-driven software development, model transformations, software supply chain.

1 Introduction

Models are an indispensable part of natural science, engineering or business. As abstractions of a physical or mental world or state of affairs they allow one to concentrate on the most salient or important aspects of the world. "Salient" or "important" must be seen in light of the objectives pursued, such as the phenomena one intends to observe, explain or design. Consequently, if several diverse objectives must be pursued one may have to employ more than one model.

A case in point is collaborative software development. Though still fairly uncommon, software development should profit from engaging core competencies wherever they find them, like many other industries do. In addition, development and production processes involve new competencies in communication, collaboration, integration, and technical and managerial control.

Clearly, there is no single model that would serve all these diverse competencies equally well. Instead there exist different models, each one tailored to specific objectives, and as one moves along the development value chain one changes from one model to the next. The changeover should be fairly seamless, hence successive models should somehow be compatible. Compatibility should be expressed by some sort of relationship.

In a three-year study, a team of researchers from Karlsruhe and Mannheim explored what organizational principles, tools and techniques would be needed to encourage collaborative software development. Models played a central role over the course of the study. These models and the kinds of relationships that determine the corresponding model chain are the subject of this paper.

2 Coordination Models

If a number of independent software companies are to collaborate on a case-by-case basis with the goal of collectively building a large software system, they must be willing to submit to some sort of coordination. Consequently, the organizational model for the coordination (the *coordination model*) is what should be established first. The model determines the interfaces across which to integrate the contributions, the process that determines the interaction and ultimately the technical support.

The prevailing model in industry is the "main contractor/subcontractor model" where one partner is the face to the customer, defines or negotiates for the product characteristics, then contracts with other partners for joint development, production and possibly assembly. It is thus the main contractor who takes overall responsibility, and hence determines the process and product models.

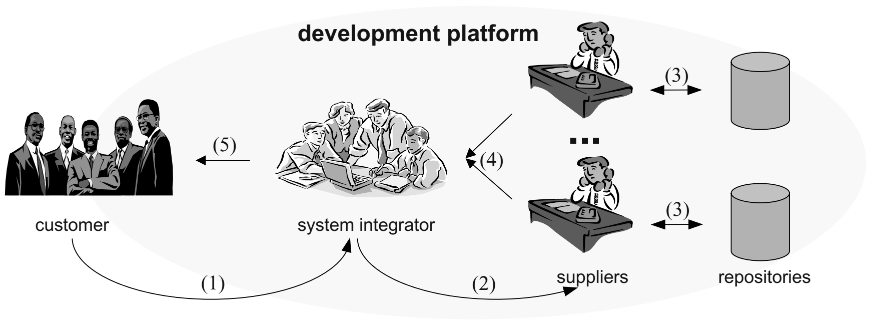


Fig. 1. Main contractor/subcontractor coordination model

Take Fig. 1. The customer takes the initiative and approaches a system vendor (1). The vendor will act as the system integrator of system components that it will procure from smaller software providers. Therefore, in a first step it solicits offers from potential suppliers, selects among these and negotiates final contracts (2). Each supplier will manufacture its system components according to specification and within agreed cost and deadlines. Each supplier should be able to draw on a wealth of earlier artifacts (specifications, designs, code) and adapt or augment them to reach its economical goals. Consequently, it maintains a repository, draws on its content and stores the new artifacts in it (3). Finally, the supplier delivers the components produced to the system integrator (4). The system integrator combines the various components into a complete system and ultimately delivers it to the customer.

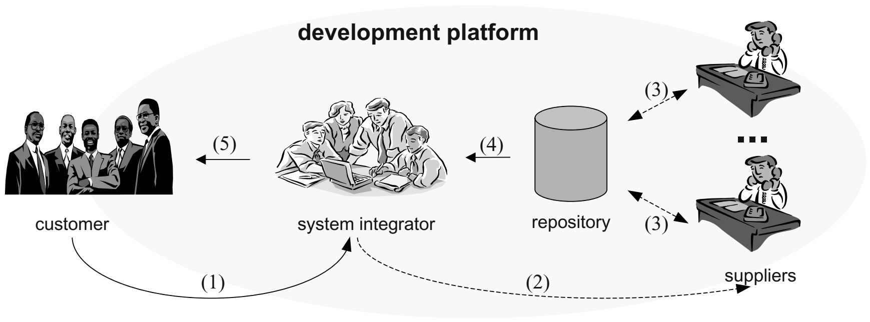


Fig. 2. Contractor/component coordination model

Often discussed but yet rarely practiced is the commercial off-the-shelf (COTS) approach. The corresponding coordination model foresees pro-active software suppliers that offer to the market specialized components as products, and advertise them through a common repository (as a kind of common product catalogue). Fig. 2 shows this "contractor/component market" model. As before, the customer approaches a system vendor who integrates the components into the total system and delivers it to the customer (1,5). However, the integrator now obtains the desired components from the common repository (4), and only if adaptations are necessary or some desired components do not exist must contracts be negotiated (2). The repository is filled "offline" (3).

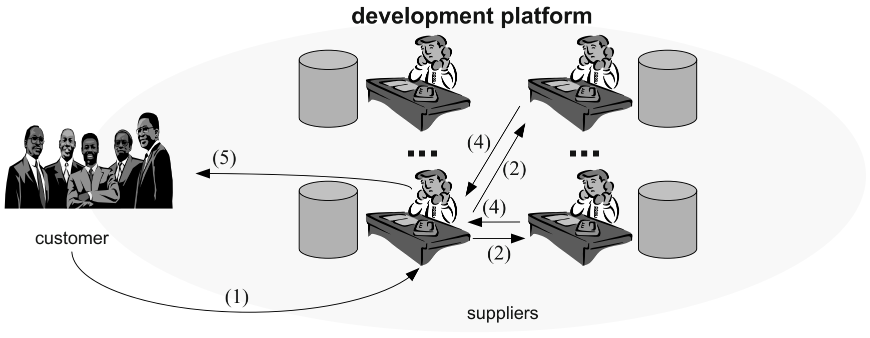


Fig. 3. Peer-to-peer coordination model

Discussions with software companies in our state of Baden-Württemberg suggest a third model without a dominant system integrator (a kind of "peer-to-peer" model, see Fig. 3). Small software companies operate independently on the market, soliciting, maintaining and protecting their own customer base, but also form networks for mutual support. Customers turn to the company they trust the most and contract for the desired order (1,5). The company will try to provide as much of the system as possible on its own, but will contact other companies within the network for support wherever this makes sense (2,4), and integrate their components into its own product.

Other models will have to accompany the coordination model. The business model identifies the financial benefits to each of the participants, and the legal contract model determines the mutual rights and obligations.

3 Analysis Models

3.1 Overall Requirements Analysis Process

All three coordination models offer "one face to the customer". This makes the system integrator the central coordinator and moderator, and the waterfall model the development model of choice. Consequently, there is a distinctive first phase of requirements analysis. Fig. 4 illustrates the analysis process, the position of the system integrator, and those who contribute to the analysis.

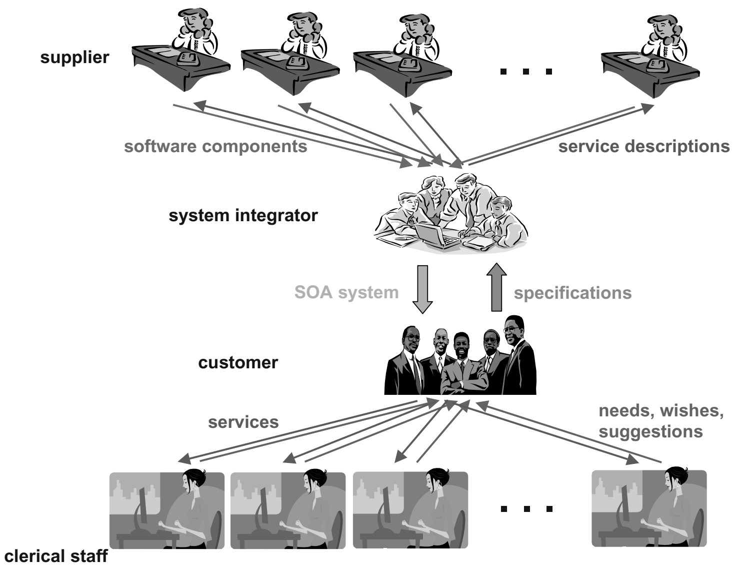


Fig. 4. Distributed requirements analysis process

It is best to think of the system delivered to the customer as a collection of services that are available to the clerical staff. In turn, then, the specifications should basically be expressed as a set of services to be implemented by the various suppliers. The implemented services are provided in the form of software components, and are integrated into a system that more or less follows a SOA philosophy. On the customer side a requirements specialist will submit proposals for services and in turn collect from the clerical staff comments, suggestions, wishes and further needs. System integrator and customer specialist will have to interact in order to determine which proposals are technically and economically feasible, and conversely which technical solutions are organizationally acceptable.

3.2 Distributed, Internet Based Requirements Engineering

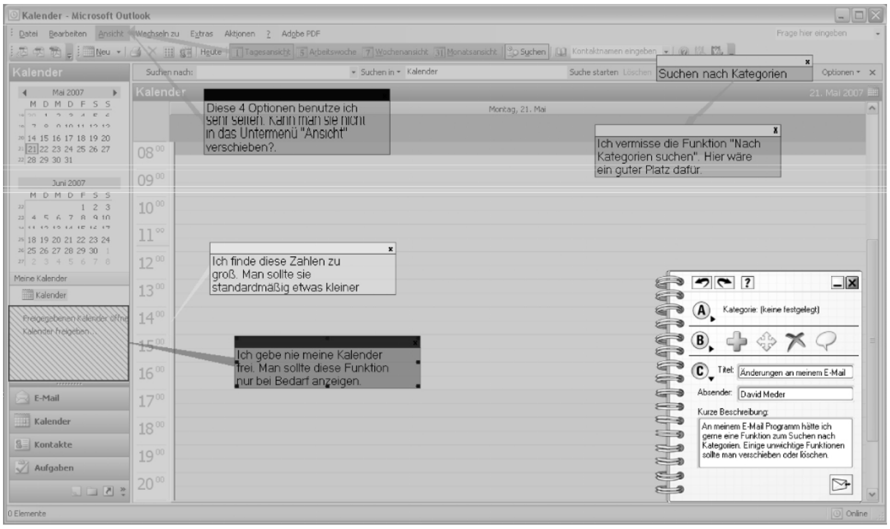
We will first examine in more detail the interaction between customer organization and system integrator. Our method DisIRE [1] consists of two iteratively performed phases. The first phase takes place purely within the customer organization under the coordination of a moderator and includes system users, system administrators and management (bottom third of Fig. 4). Its goal is to agree on an exhaustive list of requirements. An extension of the Easy WinWin method [2] is used that combines physical and Wiki-based virtual meetings and ensures ongoing trust between the stakeholders. The phase progresses in a number of steps, some of them iteratively: Brainstorming, exclusion of unrealistic proposals and categorization of the remaining ones, prioritization and discussion, detailing and reorganization of the categories. To avoid misunderstandings, a glossary of all technical terms encountered is filled. Versions are maintained in order to reconstruct earlier discussions and decisions.

The second phase performs a quantitative analysis of cost and benefits and involves on the customer side senior management and on the side of the system integrator senior technical staff (center third of Fig. 4). Senior management must decide which subset of the collected requirements offers the best return. It will, therefore, estimate for each requirement what the potential benefits are using a modification of the Cost-Value Approach [3] which in turn is based on the Analytic Hierarchy Process [4]. On the other hand, it is up to the senior technical staff to estimate the cost for providing a corresponding technical solution. The phase proceeds in four steps: Identifying requirements subsets (requirements that depend on one another, i.e., must be included in total), cost and benefit estimations, graphical representation of all requirements combinations with their aggregated cost and benefit, selection of the requirements to be implemented on the basis of the graphs, and further discussions.

3.3 Open Proposal

Clerical staff can be turned into stakeholders by involving them in the requirements analysis. Approaches of this kind have long been known under the heading of participatory design. We experimented with the participatory approach in the requirements collection phase for a banking application [5]. The basic idea is to introduce first proposals in the form of mockups (so there is an initial technical cost even during requirements analysis). As shown in Fig. 5 the staff may annotate the mockups with proposals that may concern functionality or the visual organization and appearance. The figure also demonstrates that the tool may assist the staff member in stating a proposal in a structured form (lower right). The annotated mockups become accessible by other colleagues, system administrators at the customer's and technical specialists at the system integrator's.

All may comment on the annotations, add their own suggestions or enter into a lively discussion with the originator or among themselves. In particular, interesting proposals may become the subject of an in-depth evaluation via DisIRE.


Kalender - Microsoft Outlook
Datei Bearbeiten Ansicht Wechseln zu Extras Aktionen ? Adobe PDF
Neu Heute Tagesansicht Arbeitswoche Wochenansicht Monatsansicht Suchen
Kalender
Suchen nach: Suchen in Kalender Suche starten Löschen Optionen
Mai 2007
Juni 2007
Montag, 21. Mai
21. Mai 2007
Meine Kalender
E-Mail
Kalender
Kontakte
Aufgaben
0 Elemente
Online

Suchen nach Kategorien

Diese 4 Optionen benutze ich sehr selten. Kann man sie nicht in das Untermenü "Ansicht" verschieben?.

Ich vermisse die Funktion "Nach Kategorien suchen". Hier wäre ein guter Platz dafür.

Ich finde diese Zahlen zu groß. Man sollte sie standardmäßig etwas kleiner

Ich gebe nie meine Kalender frei. Man sollte diese Funktion nur bei Bedarf anzeigen.

Kategorie: (keine festgelegt)

Titel: Änderungen an meinem E-Mail

Absender: David Meder

Kurze Beschreibung:

An meinem E-Mail Programm hätte ich gerne eine Funktion zum Suchen nach Kategorien. Einige unwichtige Funktionen sollte man verschieben oder löschen.

Fig. 5. Annotated screenshot in Open Proposal

4 Production Models

Small software companies tend to be experts in a relatively narrow field. That gives them a fair chance to reuse their software, or at least parts of it, across many different applications. We discuss the central role models play for reuse.

4.1 Model-Driven Software Development

Families of software systems are systems with a common abstract system model. If the model can constructively be used to produce a family member, the ensuing process is called model-driven software development (MDSD) [6] [7] [8], and the model a generative domain model, where domain simply refers to a particular application area, like customer relations, transport logistics, financial services.

The generative domain model consists of a problem space, a solution space, and transformation knowledge. The solution space includes the implementation components together with their permissible combinations. The problem space expresses the application-oriented concepts that application programmers would like to use to express their needs. The transformation knowledge specifies, among others default settings and dependencies, construction and optimization rules.

A domain architecture implements the generative domain model. The problem space is expressed by a meta-model that includes the concepts and rules for combining them, and in turn defines a domain specification language (DSL) for specifying a particular instantiation of the domain (the domain model). The components that form the solution space usually pre-exist as complete code, or in parameterized form (templates such as meta-classes) from which to generate the final code. The transformation knowledge is provided by tools for transforming the domain model to a collection of interacting software components supported by a specific platform. Consequently, a domain architecture is always platform-specific. There have been attempts at standardizing MDSD, such as OMG's MDA (model-driven architecture) [8].

In our case study most of the attention was directed towards the reuse of domain architectures. The argument was that both domain meta-models and platforms exhibit a number of commonalities that can be exploited by hierarchical arrangement similar to what has long been known for object class hierarchies [9] [10]. Instantiation semantics (essentially semantic enrichment or refinement) provide the means for moving from one layer to a more concrete layer until one reaches a particular domain. In parallel, code generation templates that transform models into source code can be structured in a corresponding fashion.

4.2 Concern-Driven Software Development

A weakness of MDSD is its concentration on functionality. Customers usually express extra-functional requirements as well, such as performance, persistence, robustness. Reuse under such constraints may take its cues from an approach called the software product line. Its objective is the repeated development of software that works from a common set of so-called concerns, each one reflecting a specific kind of requirement. The technique is to isolate the concerns such that each concern can be represented by a collection of code snippets (code fragment). The concern is maintained in a repository in parameterized form. Then, development of a software module proceeds in two stages. In the first, the concern is completed by instantiating the parameters according ot teh given specification. In the second stage, all needed concerns must be composed into a syntactically and semantically correct module according to a configuration plan, using predefined weaving points.

Fig. 6 shows the concern model for our case study [11]. The model assumes that each concern can be assigned a type (more accurately, a meta-type) that determines the syntax and semantics of its interface. Similar to standard programming, the type is embedded in a type hierarchy that completes syntax and semantics. Besides the type the composition interface includes a set of parameters that are to be instantiated, and a contract that expresses further constraints that are to be observed during the composition of concerns. On the lowest level a concern is simply a (generic) fragment. In general, though, a concern may itself have to be composed from smaller subconcerns. The subconcerns are again parameterized, and some of the instantiated parameters may be passed on to it. The composition plan determines how the concern will be composed from the subconcerns (including those provide by supertypes) using certain weaving points.

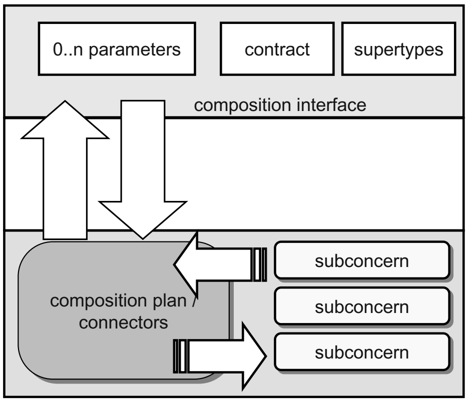


Fig. 6. Concern model

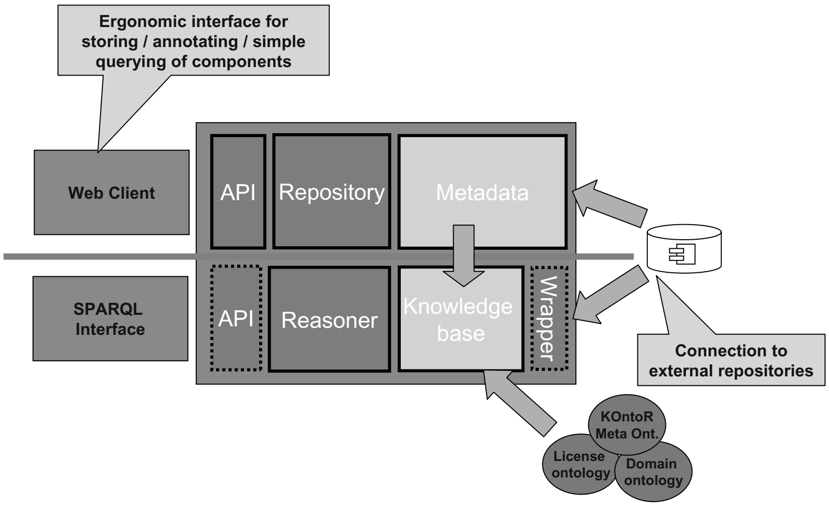


Fig. 7. System support for component-driven software development

4.3 Component-Driven Tracking

The system integrator will have to track the development status at each software provider, and each provider may wish to know as well. One approach to solve the problem is by annotating all development material with metadata when they go into a repository. The metadata may cover the domain semantics, domain-specific constraints, process models, and extra-functional properties. Standardization is achieved by imposing a metadata schema, and by adding machine processable background knowledge common to the community.

Fig. 7 illustrates the solution [12]. Components (coded modules, test cases, etc.) are maintained in the repository. Annotation may be in any specification language such as WSDL, UML, Corba IDL but follows a prescribed metadata schema as a common standard. The annotations ("instance knowledge") are kept in a separate metadata repository in a unified XML structure into which the individual annotations are translated before storage. The repository may be queried by a simple keyword search. The instance knowledge can be imported into the knowledge base using plug-ins in order to combine it with background knowledge that is provided in the form of one or more ontologies, e.g., a domain ontology for the intended subject area (e.g., financial services) and a license ontology [13]. A reasoner exploits the knowledge base to answer more complex queries than keyword search.

5 Traceability Model

In all coordination models, the partners hold collective responsibility. One major ingredient of shared responsibility are legally binding agreements among the partners. Technically , if modifications or corrections are needed the affected component as well as others depending on it must be identified together with the partners who have the needed expertise. Hence, it is imperative that each and everyone has a complete view not only of the state of all artifacts but also of their interdependencies.

Fig. 8 shows a centralized front-end solution for a "single point of information" to distributed repositories. All artifacts are collected into a knowledge base that

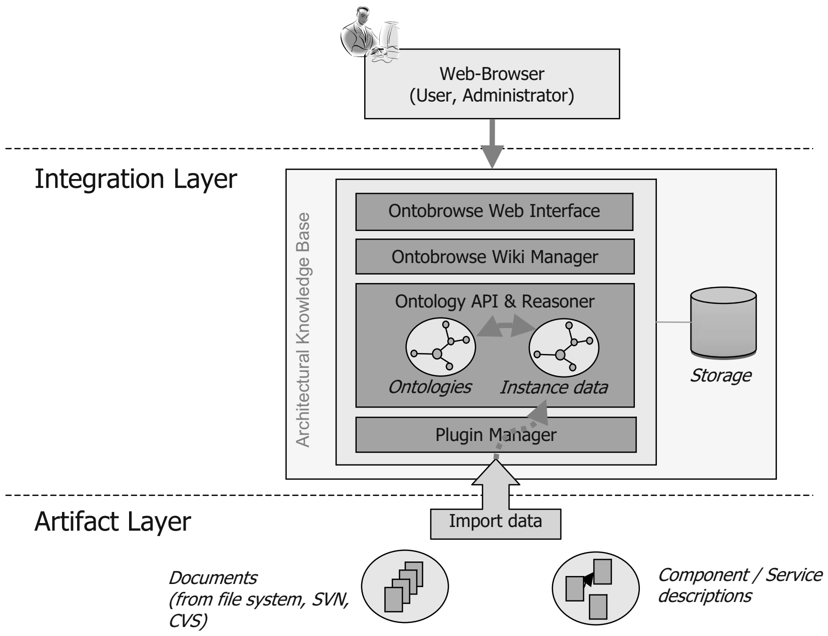


Fig. 8. Semantic Wiki-based support for tracking component interdependencies

is implemented on a Wiki basis [14]. They are subsequently augmented by semantic metadata based on standards like RDF ore OWL (hence the name "Semantic Wiki") [15]. For weakly structured documents the metadata are manually added by annotation, possibly after some automatic extraction of terms, and entered via specially written plug-ins, whereas the more formal descriptions can directly be translated and entered via standard plug-ins that provide a Web interface. Metadata generation is controlled by a software service ontology - essentially a model for the functional and extra-functional characteristics of software services. The ontology may be constructed and updated by means of one of the commercially available tools and are then imported via the Web interface. By providing a reasoner one can now deduce many of the interesting interdependencies.

6 Model Interdependencies

6.1 Model Layers

The models we discussed in Sections 2 through 5 are neither models nor meta-models in the traditional sense. They are rather prescriptions for organizing the work towards developing models. Consequently, we shall from now on refer to them as *model agendas*. Model agendas are chosen to serve a specific modeling purpose.

A *meta-model* is a collection of interrelated modeling constructs. A model agenda does not prescribe a specific meta-model, although it may limit the selection among meta-models. The choice of a specific meta-model depends on the particular domain that becomes the subject of executing an agenda (and often also of the designers' familiarity).

A *model* is an instantiation of a meta-model and is the ultimate objective of the development process.

As we demonstrated in the previous sections, collaborative software development employs a good number of diverse model agendas (some of them mentioned just in passing) and - one may suspect - of meta-models. Is one free to choose among them as one moves from one phase to the next? Or does the choice of one limit the available options during the succeeding phases? We examine two such restrictions, one on the agenda level and the other on the meta-model level.

6.2 Model Cohesion

The agendas in Sections 2 to 5, in this order, reflect a chronological order. First the participants must agree on the organizational and business environment, i.e., on the coordination model. Then the requirements analysis determines the specifications for the software system to be constructed according to one or more analysis models. This is followed by the design and development of the system components and their integration, making use of one or more production models. After delivery and over the remaining lifetime of the system, maintenance must obey an agreed-upon traceability model.

One may take an abstract view of the sequence of agendas as a production pipeline for software. Like in any physical production the production stations must conform to one another. Consequently, one can define a *cohesion* relationship on model agendas (cohesion: degree of strength of functional relatedness between operations [16]).

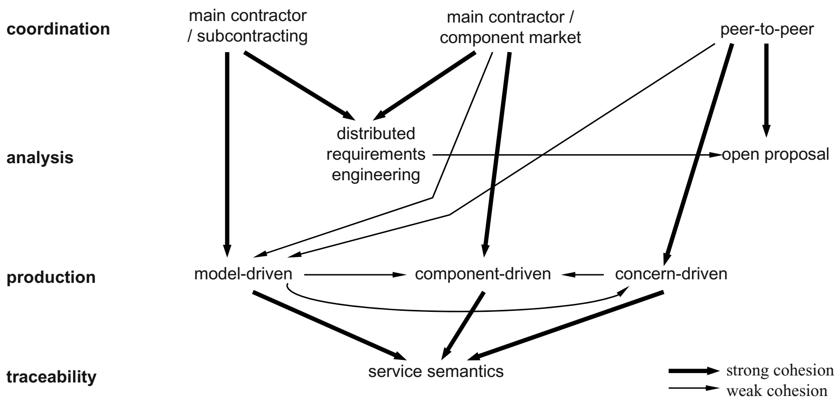


Fig. 9. Cohesion of model agendas

Fig. 9 summarizes the lessons that one may draw from the previous discussions. The figure differentiates between strong and weak cohesion. We give a few examples. Distributed requirements engineering relies on a strong system integrator, hence it seems natural to apply this process whenever there is a main contractor. In the peer-to-peer cooperation model the customer contact may lack the integration experience so a process more along the lines of open proposal with responsibilities spread across a larger community may be the better approach. But even in distributed requirements engineering the open proposal approach may be a good complement. The model-driven production model requires considerable initial investment, so again the main contractor model seems to offer the best preconditions. The concern-driven model may be a useful complement because the full benefit would only accrue if one can still influence the construction of the components. On the other hand, for a peer-to-peer cooperation a concern-driven production model seems a good direct choice. Component-driven tracking seems indispensable only to a component market. Traceability is a requisite no matter how one arrived at the components.

6.3 Model Interoperability

Once a pipeline has been established, each agenda must choose the meta-models it plans to employ. The choice is constrained if one wishes to automate the passage from one agenda to the next as much as possible: One should make sure that the concepts of the models can be related, e.g., are the same or can easily be mapped to one another. Consequently, the choice of meta-model will often determine the meta-model of the subsequently used model.

The ability for a system or a product to work with other systems or products without special effort on the part of the customer is often referred to as interoperability [17]. The definition comes close to our needs of easy passage between meta-models, hence we speak of *model interoperability* (more precisely: meta-model interoperability) [18] [19].

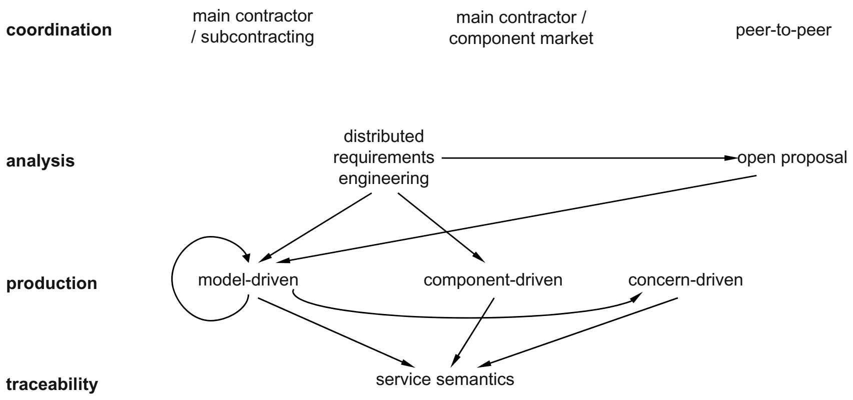


Fig. 10. Interoperability of meta-models

Model interoperability has extensively been studied in the realm of software engineering, though mostly in the context of MDA (for an overview see [20], for more recent examples [21] [22]). We have to take a wider view that spans all agendas. Fig. 10 indicates where one would have to enforce interoperability. First note that the coordination models are purely organizational whereas the three other layers carry strong technical aspects and, hence, have models that are to be transformed. For example, the model-driven and component-driven production models are essentially top-down so that they take the results of the requirements analysis as their input. Hence, mapping and refinement should be possible between their meta-models. Model-driven development internally uses several models that one should be able to map, and if used as part of model-driven development, concern-driven should use compatible meta-models. Traceability exploits the semantics, so the meta-models used for production and traceability should strongly be interrelated as well.

7 Conclusions

Software development in general, and collaborative software development in particular, is an undertaking that spans many processes with different objectives. Different objectives emphasize different aspects of the application and technical world. Models are an accepted means to separate the currently major aspects from the minor ones. As the objectives change across the various processes, so do the models that accompany them. Since new objectives emerge from earlier ones, the chosen models should remain compatible so that the results developed in one can easily be utilized in the next one.

This paper argues that – beyond the traditional emphasis on model properties and transformations – one should carefully study the interdependencies between models. Knowing these one could judge the resrictions of an earlier model on the choice of later models, or conversely how the choice of a later model would impact the choice of earlier models. We demonstrated for a particular project what models were encountered, and what kind of interdependencies between the models were observed. We gained some practical experience by building a demonstrator that integrated for the main contractor/subcontractor model the models discussed in the paper [23].

Acknowledgments. The contributions of the participants in the CollaBaWue case study are gratefully acknowledged: Jens Arndt, Astrid Behm, Michael Geisser, Hans-Jörg Happel, Armin Heinzl, Tobias Hildenbrand, Axel Korthaus, Volker Kuttruff, Asarnusch Rashid, Fabian Schfer, Stefan Seedorf, Michael Schwind, Peter Tomczyk, Jan Wiesenberger.

References

1. Geisser, M., Heinzl, A., Hildenbrand, T., Rothlauf, F.: Verteiltes, internetbasiertes Requirements-Engineering. Wirtschaftsinformatik 49(3), 199–207 (2007)
2. Gruenbacher, P.: Collaborative Requirements Negotiation with Easy WinWin. In: Ibrahim, M., Küng, J., Revell, N. (eds.) DEXA 2000. LNCS, vol. 1873, Springer, Heidelberg (2000)
3. Karlsson, J., Ryan, K.: A Cost-Value Approach for Prioritizing Requirements. IEEE Software 14(5), 67–74 (1997)
4. Saaty, T.L.: The Analytic Hierarchy Process. McGraw-Hill, New York (1980)
5. Rashid, A., Meder, D., Wiesenberger, J., Behm, A.: Visual Requirement Specification in End-User Participation. In: 1st Intnl. Workshop on Multimedia Requirements Engng (MeRE 2006) – Beyond Mere Descriptions, 14th IEEE Intnl. Requirements Engng. Conf (RE 2006) (2006)
6. Czarnecki, K., Eisenecker, U.W.: Generative Programming: Methods, Tools, an Applications. Addison-Wesley, Reading (2000)
7. Stahl, M., Völter, M.: Modellgetriebene Softwareentwicklung: Techniken, Engineering, Management. dpunkt.verlag (2005)
8. Beydeda, S., Book, M., Gruhn, V. (eds.): Model-Driven Software Development. Springer, Heidelberg (2005)
9. Aleksy, M., Schwind, M., Gitzel, R.: Generating Families of Bussiness Components from Metadata Hierarchies. In: Proc. 1st Intnl. Workshop on Engng. Complex Distributed Systems (ECDS) (2007)
10. Gitzel, R., Schwind, M.: Using Non-Linear Metamodel Hierarchies for the Rapid Development of Domain-Specific MDD Tools. Proc. 10th IASTED Intnl. Conf. on Software Engng. and Applications (SEA 2006) (2006)
11. Kutruff, V.: Realisierung von Softwareproduktlinien durch Komposition modularer Belangimplementierungen. Wirtschaftsinformatik 49(3), 171–179 (2007)

12. Happel, H.-J., Korthaus, A., Seedorf, S., Tomczyk, P.: KontoR: An Ontology-enabled Approach to Software Reuse. In: Proc. 18th Intnl. Conf. on Software Engng. and Knowledge Engng (SEKE), pp. 349–354 (2006)
13. Happel, H.-J., Seedorf, S.: Application of Ontologies in Software Engineering. In: Workshop on Semantic Web Enabled Software Engng., 5th Intnl. Semantic Web Conf. (2006)
14. Happel, H.-J.S., Seedorf, S.: Ontobrowse: A Semantic Wiki for Sharing Knowledge about Software Architectures. In: Proc. 19th Intnl. Conf. on Software Engng. and Knowledge Engng (SEKE), pp. 506–512 (2007)
15. Korthaus, A., Schwind, M., Seedorf, S.: Semantic Integration of Business Component Specifications with RDF Schema. In: Workshop on Semantic Web Enabled Software Engng., 4th Intnl. Semantic Web Conf. (2005)
16. Papazoglou, M.P., van den Heuvel, W.-S.: Business Process Development Life Cycle Methodology. Comm. ACM 50(10), 79–85 (2007)
17. Ralyte, J., Backlund, P., Kühn, H., Jeusfeld, M.A.: Method Chunks for Interoperability. In: Embley, D.W., Olivé, A., Ram, S. (eds.) ER 2006. LNCS, vol. 4215, pp. 339–353. Springer, Heidelberg (2006)
18. Kühn, H., Murzek, M.: Interoperability in Metamodelling Platforms. In: Konstantas, D., Bourrieres, J.-P., Leonard, M., Boudjila, N. (eds.) Interoperability of Enterprise Software and Applications, pp. 215–226. Springer, Heidelberg (2005)
19. Dominguez, E., Zapata, M.: Mapping and Interoperability: A Meta-Modelling Approach. In: Yakhno, T. (ed.) ADVIS 2000. LNCS, vol. 1909, pp. 352–362. Springer, Heidelberg (2000)
20. Czarnecki, K., Helsen, S.: Classification of Model Transformation Approaches. In: OOPSLA 2003 Workshop on Generative Techniques in the Context of Model-Driven Architecture (2003)
21. Atzeni, P., Cappellari, P., Bernstein, P.A.: A Multilevel Dictionary for Model Management. In: Delcambre, L.M.L., Kop, C., Mayr, H.C., Mylopoulos, J., Pastor, Ó. (eds.) ER 2005. LNCS, vol. 3716, pp. 160–175. Springer, Heidelberg (2005)
22. Valderas, P., Fons, J., Pelechano, V.: Transforming Web Requirements into Navigational Models: An MDA Based Approach. In: Delcambre, L.M.L., Kop, C., Mayr, H.C., Mylopoulos, J., Pastor, Ó. (eds.) ER 2005. LNCS, vol. 3716, pp. 320–335. Springer, Heidelberg (2005)
23. Arndt, J., et al.: Das Projekt CollaBaWue – praxisnahe Grundlagenforschung zur Industrialisierung der Softwareentwicklung. In: Haasis, K., Heinzl, A., Klumpp, D. (eds.) Aktuelle Trends in der Softwareforschung, MFG-Stiftung Baden-Württemberg, pp. 198–211 (2007)

Conceptual Modelling in an Evolving World

Roland T. Mittermeir

Institut für Informatik-Systeme,
Universität Klagenfurt
roland@isys.uni-klu.ac.at

Abstract. Conceptual models serve as models of indirection linking the information system and the section of the real world represented by this system. They are established during the analysis phase of system construction and might be changed in discrete steps if the application necessitates further (or fewer) aspects to be covered. This suffices for static domains.

In dynamic domains however, the quality of this link between an information system and reality is progressively deteriorating due to shifts in the semantics of the information captured in the system. The paper shows example domains where shifts of semantics would require gradual evolution of the conceptual model and proposes an approach for accommodating these requirements.

Keywords: Conceptual modelling, gradual schema evolution, knowledge representation, dynamic domains, fuzzy modelling.

1 Introduction

Nepejvoda claims that in programming "Theory and practice peacefully coexist like the two banks of a deep river. [..] Sometimes a challenge from one side is accepted by the other and a bridge arises" [14]. Reflecting on this quotation, we have to note that "sometimes" implies that quite often we are wandering on only one of the two banks, admiring, but not touching the other side. Only occasionally one arrives at a bridge where practice influences theory and vice versa. With "bridge" one associates not only the notion of linking the two banks, with bridge one also associates the concepts of solid banks and of pillars built on rock.

But many rivers have banks devoid of rock-like stability. Over time rivers may gradually move their banks. Bridge-builders have to be aware of this fact. Either, they find rocks upon which to ground their pillars or they have to build dams to regulate the river in such a way that the banks cannot move or will move only within the confines of the enclosing embankments.

Abstracting from the original quotation but remaining at the image of a river, we might conceive of a conceptual model as a bridge between the bank of reality and the bank of descriptive information about this reality. The analog seems appropriate in so far as conceptual models have to endure over time. While it might suffice for a program to have its interpretation as well as the interpretation of the data it processes defined only at the given point in time when it executes, complex and long living software systems evolve over time. The changes experienced by such a system are partly due to external change drivers resulting from an evolving application domain [11], partly they result from the system's internal evolutionary forces [9] and the interaction between these two mechanisms. Likewise, an information system has to cope with the changes in the domain it is modelling. The phases to be observed over the life-time of such long-living systems have been identified by Bennet and Rajlich [2]. They mention several causes why systems transit from the still active *evolution phase* to the defensive, if not to say regressive, *servicing phase*. It should be clear that in the servicing phase the system will eventually have problems to remain in sync with the external dynamics of its application domain.

A corner stone in the development of any large information system, irrespective of whether it is perceived from a software- or information system perspective, is its conceptual model. Classically conceptual models of information systems serve a dual role. They are built during the analysis of the section of the application (or domain) to be represented in the information system (IS). Thus, they serve to conquer the complexity of system construction, notably to give crisp semantics to the phenomena to be represented. As result of this *"crispification"* of real world phenomena, well defined entity types and relationship types between such entity types can be isolated. One decides which attributes are relevant to be stored and which serve to characterize the entities as well as the relationships. In its second role the conceptual model essentially defines the semantics of the data captured in the tables formally described in the data-base's schema.

For software systems there exists a host of literature dealing with requirements analysis and architecture definition. For establishing conceptual models of information systems, developers can also choose between various approaches. But irrespective of whether deductive or inductive approaches are followed [6], the modelling task involves to unite different perspectives. In non-trivial applications the perspectives to be combined into a coherent picture are often only obliquely related. It is normal that different stakeholder have different perspectives on issues dealt with in the information system. The modeler has to analyze and relate them and distil a coherent core which is understood by all parties such that they can agree upon it. Whether this is done in an informal ad-hoc creative act of the modeler or whether it is done by a formal analysis of statements expressed in natural language as done in the NIBA-project [10] does not change the basic nature of the end result. In all cases this should be a well defined and consistent conceptual model out of which a conceptual schema can be established.

The (conceptual) schema thus derived from the conceptual model is not only *"crisp"* in contrast to a usually fluffy and somehow fuzzy real world. It is also a *petrification* of the facts covered by the conceptual model. This is considered quite normal in the data-base community and, in fact, this is most adequate if we consider the usual introductory examples of suppliers shipping parts, professors teaching classes, or employees being hired and assigned tasks. – Considering an evolving world with shifting semantics for the terms denoting the information captured in a data base, one is reminded of Nepejvoda's observation that reality and theory are just different banks of the same river. Sticking to the analogy, erosion might lead to situations where one of the two banks comes out of sight of those walking on a given bank. The bridge of the conceptual model, once established, might lose its linking power between reality and its image kept in the information system.

2 Conventional Data Base and Schema Evolution

It seems interesting to note, that the community developed different strategies for ensuring that an information system remains valuable even if the domain or application it represents experiences changes.

If too many suppliers ship too many parts, the physical design of the database will be changed and, interestingly, on the physical level one might migrate to a data-structure that is even dynamically adapting to number, size, or query frequency of certain data records. The change is reflected in the physical schema. But there is no comparable dynamics foreseen for the conceptual model and hence, for the conceptual schema to be understood as model of indirection [1]. On first glance, one might claim that this is not necessary, as it will never happen that a supplier gradually shifts to become a consumer. But such an argument is too short sighted and specifically has too static a perspective.

Of course, as the business of a company grows, it might also want to represent their customers in their IS. Hence, schema evolution will enlarge the supplier-parts-schema to a supplier-parts-customer-schema. This happens as a discrete maintenance step. If some of the suppliers also buy some goods they will be represented twice, as suppliers and as customers.

Schema evolution will be more involved if the parts relation is not only vertically partitioned due to performance reasons but more precise modelling demands to distinguish conceptually between different kinds of parts and possibly different kinds of ready made goods. But still, this schema evolution will happen at a specific, well planned maintenance or redevelopment venture of the IS. Such an operation will lead to a structural change in the (operational) data base. For data warehousing applications different strategies have been devised to cope with such discretionary evolutions which might entail in certain situations also shifts in semantics of the values stored in the operational data base [5]. A typical example were shifts of semantics have be to taken care of is studying the population dynamics of European countries, given splits and mergers of countries throughout the last century.

However there are domains where it is quite inappropriate to stick the label once attributed to an object to this object forever. Likewise there are domains where a relationship once identified may loose its strength over time and should therefore be replaced by a strong relationship to some other entity. If the dynamics are confined to the gradual belonging of an entity (relationship) to a particular entity (relationship) type, this can be taken care of on the attribute level by attaching the adherence degree to the respective attribute value. This allows for fuzzy representation of data and, consequently, requires explicit updates if the degree of membership changes. If, however, such membership values would spread over different entity types, such changes should reflect back on the conceptual model. This paper will address this issue. To my knowledge there exist currently no mechanisms accounting for such semantic shifts.

In the next section the paper points to some domains where such gradual shifts occur in the real world. On the basis of these reflections the scope of such shifts will be delineated. Afterwards an approach is sketched of how to account for such shifts in the conceptual model of an information system.

3 Examples of Intrinsically Dynamic Domains

Without striving for comprehensiveness, just a few domains are mentioned where shifts in attribution cannot be confined to the level of individual instances but should also be reflected on the conceptual level.

In static domains, obviously, such need would not arise. In dynamic domains though, gradual shifts need to be represented. As a trivial, still benign and admittedly constructed example, one may consider the changing relationship between persons entering into divorce and remarriage with another partner afterwards. Assuming an underlying triangular relationship between John, Peter and Mary, the initial state might be given as

$loves(John,\ Mary)$, and possibly
$likes(John,\ Peter)$, and
$knows(Peter,\ Mary)$.

After divorce between John and Mary and Mary marrying Peter, the new situation might be captured by the following representation:

$knows(John,\ Mary)$,
$knows(John,\ Peter)$,
$loves(Peter,\ Mary)$.

Unfortunately this will neither correctly cover the new situation nor would this be suitable for expressing the complexity of the various intermediate states that occurred throughout this transition.

To avoid three different reflexive relations and assuming symmetry, the initial state could be modelled as

$loves(John,\ Mary,\ 1.00)$,
$loves((John,\ Peter,\ 0.75)$, and
$loves(Peter,\ Mary,\ 0.25)$.

After divorce and assuming remarriage the respective values could be (for simplicity still assuming the unlikely case of symmetry):

$loves(John,\ Mary,\ 0.10)$,
$loves((John,\ Peter,\ 0.00)$, and
$loves(Peter,\ Mary,\ 1.00)$.

The two versions of relation *loves* still just realize the final state of a gradual process. In this particular case, one would probably not have appropriate observations and the related values to accurately express the shifts in degree of membership occurring throughout the process. Hence, the example might seem irrelevant even if one might see applications that are not interested in the legal status between two persons but in capturing shades as expressed. The relationship *loves* will always remain to be a relationship between two persons and it seems fair to assume that society (or the particular application) has a firm definition of what is represented by the concepts of "*love*" and of "*person*".

Even if the societal definition of "*love*" changes, an appropriate update of the glossary attached to the conceptual model and a re-valuation of the membership values according to the new semantics of "*love*" would solve the issue, assuming that there is some (so far extraneus) source that can provide the correct membership value.

The problem would change though, if the conceptual model foresees different relations for lovers and people living in some other relationship also expressing some form of personal attachment (for simplicity, assume: hate) such that the two concepts are not orthogonal with respect to each other (which would be the case between hate and love) and if we further assume that the membership value is not directly defined by some particular explicitly given value but by a set of operational indicators also stored in the information system. Persons loving each other are inserted into the *loves* relation. Persons hating each other are inserted into the *hates* relation. In case of hate-love, they can be inserted in both. Thus, it seems, we are back to the supplier-parts-customer example.

However, the semantics of what constitutes a supplier, a part, or a customer remain stable. What constitutes membership attribution to psychological categories or social classes is not stable. Hence, over time, people with quite different characteristics might be inserted into one or the other category. While this might still be not dramatic in the example of $loves(Person,\ Person)$ it might be already relevant in cases of $likes(Person,\ Good,\ DegPreference)$ in some marketing application.

With shifts of market preferences, the degree of preference might drop from a very high value to a low one, finally to 0. Hence presence of a good in relation *likes* does not imply that this good is to be favored by the group of customers a particular person belongs to. We may have $potentialCustomers = SELECT\ Person\ FROM\ \ likes\ WERE\ DegPreference > 0.8$ Likewise, one might group those, who expressed dissatisfaction into a set *unlikelyCustomers* if the "$DegPreference$" is below a certain threshold value.

If "$DegPreference$" is the result of some market survey, it seems clear how to deal with this. The survey was levied at a given time and in case it is repeated, one either has to replace the old entries in the data base by the new results or, possibly more appropriate, one might time-stamp the entries by the date of the survey. Thus, a history of preferences for goods at some discrete points in time will be established.

But how would we deal with this issue, if "*DegPreference*" is a computed value, based on the entries of the (stable and historic) relation *bought(Person, Good)*? If every query will recompute the value of *"DegPreference"*, the value will be current and one has only to worry about not looking too deep into the past to reach a correct interpretation of the results to this query. If, however, for performance reasons, "*potentialCustomers*" gets materialized, semantic shifts in *likes* might go unnoticed.

This case will be even more serious if *likes* or some similar relation capturing all customers is split into several sub-relations in order to allow different marketing approaches for each sub-category of *customers* and establishing relations between these sub-categories. In this case misinterpretation of results becomes likely. For the sake of demonstration, assume: *"affluent", "middle-class", "lower-middle"*, and *"substandard"* to be labels describing categories of customers. Such labels might also be attached to geographical marketing areas such that part of the territory is attached to the category of customers predominantly dwelling in this area.

We know that the financial status of people changes over time and we also know that the demographic characteristics of areas change likewise. Hence, what used to be *"suburban"* might become *"town"* and what used to be *"country"* might become *"grease-belt"* or *"suburban"*. In contrast to the physical level, where, for instance, a B*-tree is a mechanism to adapt itself to different load characteristics, there exist currently no mechanisms on the conceptual level to account for shifting semantics of the categories (relations as well as attributes) kept at the semantic level.

While this issue might be considered of minor importance for typical business applications where data warehouses provide a set of mechanisms that somehow deal with the issue of changing semantics [7], it is a problem in several administrative applications. Here one has to distinguish between static factual information and information about interest driven issues. While the former usually captures atomic facts and thus has relatively stable semantics, interest driven issues usually map a complex bundle of facts into a given term. Thus, the semantics of the term has several shades and the dominance of these shades changes over time. Hence, the overall semantics of such information items experience shifts. Notably in administrative applications such multi-facetted generic terms enter into the operational domain and officers have to place their decisions on the current interpretation of law, regulations, and societal rules.

4 Mechanisms for Dealing with Dynamic Domains

This section first tries to define the scope of the problem resulting from evolving domains and then propose solutions to address this challenge.

4.1 Stability Versus Dynamism

The previous section concluded with a distinction between information that is factual, referring to attributes having atomic properties in both, the real world as well as in the information system, versus information which is derived form the aggregated perception of some non-elementary facts. Examples of the former might be the price, the age, or the color of some device. Examples of the latter might be the characterization of a person as old, wise, friendly or unfriendly.

For the former category, distinct values with stable semantics can be given. It might be necessary to pin down the semantics by some additional definitions, e.g. for "price" we have to clarify whether it includes or excludes sales tax. But aside from this it is clear that "price" refers to the amount of money that has to be transmitted in exchange for the good bought. This price might change over time. Hence, the definition has also to mention whether the number stored is the price which had been paid when the good was bought or whether it is its current price. With age it is even more evident that the information has also to tell when the value has been assessed to come to a correct interpretation. But the semantics of age as "time passed since the thing came into existence" is stable[1]. The situation becomes slightly different when it comes to nominal values as with color. To interpret a given color, one has to know between how many kinds of color this information system distinguishes. To correctly classify borderline cases, say between green and blue or orange and red, the spectral definition of the ranges of individual colors might be given. But the spectral composition can be measured. Hence, finally, one arrives again at numbers defining the range throughout which a nominal value is defined. Moreover, the definitions of price, age, and color are orthogonal.

The situation is different with nominal descriptions that obtain their definitions by perception. In certain situations, "old" might be legally defined. The slightly more polite term "senior" is for instance used for retired persons or persons after a given age with this age being defined differently for men and women. In this case only at discrete well defined points in time the new rules leading to the classification of a person as "senior" will come into effect and consequently these occasions will trigger a corresponding update of the data base. But if a person enters my office, I would have no idea about this persons age. I have to assess this person's age according to his or her appearance. Different people might use different indicators. Amount of grey hair, facial expression, properties of the skin, upright posture and many other indicators will allow me to conclude whether this person is old or young. Highly subjective indicators might play also a role, sometimes even a dominant role. E.g., while I was 20, a person of age 40 was already old. Being now well beyond 40, I might classify a professor aging about 40 years as young. On the other hand, I'd consider a student of the same age still as rather old. Derived values such as assessed (in contrast to measured) age quite often are not fully orthogonal with respect to each other. The classification of "wise" might serve as example. One might easily conceive of a "wise, old person" but it is not common to speak of a "wise young person". The young person might be rather "smart" than "wise". But while this statement might be

[1] Most designers would avoid this update problem by storing the time of birth or manufacture and computing the age on demand by building difference to actual time.

correct in some cultural context and incorrect in another, it is in any case highly subjective. This subjectivity becomes even more apparent when considering the pair “friendly” and “unfriendly”.

So far, the subjectivity of derived concepts has been discussed on the value domain. As long as it remains there, shifts in the definition of these nominal values have little repercussions for the conceptual level. However, when the conceptual model allows for generalizations – and according to Mylopoulos [12,13], generalization is one of the elementary dimensions of a conceptual model – such attributes with unstable semantics might, and quite often do, become discriminatory attributes in classification hierarchies. Thus, we have generalization hierarchies such as

{*friendlyPerson, unfriendlyPerson*} *ISA person,*
{*towns, suburb, grease-belt, countryside*} *ISA region,*
{*affluent, middle-class, lower-Middle, substandard*} *ISA customer*

The effect of this is that what used to be attribute values became definitional properties of (sub-)entity types. This is generally considered as an enrichment of the conceptual schema and therefore it is discussed as a sign of progress that conceptual schemata, notably conceptual schemata of object oriented information systems, allow for sub-typing of entity types. It is to be seen though that in those cases where the sub-typing criterion is not based on atomic terms reflecting atomic real world properties, the classifiers are bound to have shifting semantics. With such shifting semantics, the bridging function of the semantic model becomes progressively eroded. The next sub-section will address the issue how such erosion can be identified. The section concludes in pointing to possible automatic approaches to cope with such shifts.

4.2 Recognizing Shifting Semantics

Everybody notices when inundations moved a river’s banks. If bridges are involved, some emergency actions will take place. But is there a strategy if the regular flow of water undermined some pillar? Let’s hope that regular inspection processes ensure that repair is done in safe intervals. In the current context it might be interesting to see whether one could imagine a hydraulic pillar that automatically grows to compensate for the soil that has been partially washed away.

The analog shows that any compensation of deterioration requires that the amount of deterioration can be recognized first. Focussing on the information systems domain, it becomes clear that there are basically only two alternatives:

- *periodic inspection*, ensuring that some system-external authority recognizes that the conceptual shift of some classifying attribute has become drastic enough to warrant redefinition and reclassification; or
- *actively using internal redundancies* to recognize such conceptual shifts. The advantage of this approach is that the necessary review can be done automatically and it can be triggered aperiodically, depending on the frequency of update operations in the section of the IS that bears on the respective classifying attribute.

As in conventional data base systems redundancy is considered as poor design, one might assume that one is bound to use periodic inspection with all its problems. However, this is giving up too quickly. Firstly, one has to see that many current information systems deal with semi-structured data. Here the information contained in the structured part serves quite often to categorize the information contained in the leaves of XML-trees, containing lengthy natural language text or even non-textual information. Retrieval mechanisms have been devised to benefit from both, the structured and the free-form parts of XML documents [8].

Another form of weak redundancy can be seen between information aggregated from (possibly several) operational data bases to a strategic level by OLAP operations. While the relationship between *Person* and *Product* and the relationship between *Person* and *Address* are strictly orthogonal and therefore free from redundancies in the operational customer data base, a statistical relation between these relationship emerges when raised to the data warehouse level of $CustomerCategory \longleftrightarrow ProductCategory$ seen in conjunction with $CustomerCategroy \longleftrightarrow MarketingArea$. In this case, a statistical link can be established via the not explicitly present attribute *"Income"*. This fuzzy relationship is conceptually fairly close to the redundancy between the tagged information in an XML document and the full text appearing in this document.

It would go beyond this paper to present the specific algorithms necessary to benefit from such explicit but hidden or implicit statistical redundancies. However, the rest of the paper supposes that such redundancies exist. Therefore, it distinguishes between the multi-valued categorizing attribute currently under focus and the semantic support it has within the same information system due to the redundancies just mentioned. Following Zadeh's arguments and terminology [15], the information to be derived from the admittedly not readily visible redundant information assumes the role of the IDS (initial data set), providing the actual definition of the categorizing term. Slightly paraphrasing Zadeh's example, one would have to look at the heights of all Swedes in the IDS and into the definition of the modifier *most* to assess whether *"most Swedes are tall"* and to determine the fuzzy membership of Swedes in *"tallness"* [15].

Coming back to the problems of this paper, a customer will be categorized as *middle-classCustomer* if the properties characterizing this person as *middle-class* dominate properties that would tend to put this person into some other social category. At a given instance in time the previous sentence suffices to correctly classify this person. As this definition ensures that only persons which have their supremal membership value in *middle-class* are categorized as *middle-classCustomer*, the information system is solid.

But, as mentioned already, the definition of *middle-class* rests on various parameters and the interpretation of these parameters changes over time. Hence, a person that was once classified as "dominant middle-class" might under today's interpretation of the term no longer be put into *middle-classCustomer*. In extreme cases, when the Lorenz-curve changes drastically, the whole category might vanish. Hence, it is called for to qualify not only the individual entities classified as *middle-classCustomer* but that the fuzzyness of this concept is raised to the conceptual level, highlighting that "middle-classness" of *middle-classCustomer*s has only a support reflecting the average membership value of the attribute-value *middle-class* of its members. Generically, one might refer to this aggregated membership value as *"own membership"*. Whether foreign memberships (i.e., memberships to other values of the multi-valued attribute *DegrAffluence*) are propagated as individual average membership values or whether this is captured in a single aggregated indicator called *"foreign membership"* is an aspect involving detailed performance considerations which are beyond the scope of this paper. Here it suffices that a relationship between *own* and *foreign* membership can be computed such that the system can highlight situations when new entries point to the fact that semantic shifts have occurred and some action is necessary to avoid misinterpretation of the data contained in this information system.

Thus, while this might still fall short of an analogy to the self-adapting hydraulic pillar mentioned initially, the flexibility of this concept allows that at least no periodic manual inspections are necessary. The system will become capable of observing the evolution of its own domain.

4.3 Categorization Versus Re-classification

The conceptual approximation to the automatically adapting hydraulic pillar of a bridge solves some aspects of the erosion problem. The problem of moving banks, however, still remains to be considered. The spacing of components used to allow for thermal expansion at high temperatures won't suffice to solve this problem. Could a boat-bridge be a solution or do we need to build a ferry?

In the previous subsection basically an alerting mechanism has been presented. Now the focus will be on the question as to whether an automatic healing mechanism can be foreseen.

The answer to this question is less of a technical but rather of a pragmatic nature. On the technical level, one has to see that the propagation of (true!) fuzzyness to the conceptual level cannot be blindly back-propagated to the entities classified under this categorization. This would lead to oscillations that might become very hard to control. Avoiding this, one has to consult extraneous sources leading to a re-interpretation of the information contained in the support vectors of the individual entities.

The trigger highlighting the potential need for such re-interpretations can be found in updates of the *own* membership value of categories figuring in the support vector of items considered to belong to the given entity sub-type. But here, the pragmatic question enters: Should a transaction of a person that was once classified to be *affluent* be moved to the category *middle-class* if it did satisfy the criteria for *affluent* when the transaction took place but not according to current standards?

There is no general answer to this problem. One might conceive of situations were stability of classification is important. But one might also conceive of situations were reclassification would be called for. To cope with both types of queries, time-stamping of the classification and of any reclassification might provide a satisfying answer.

It seems appropriate to readjust the conceptual membership after any such reclassification step. As the reclassification was done by some extraneous source, this can be a rather straight forward operation. This will notably be the case, if the extraneous source was fully independent of the data kept in the information system. In case a panel of experts provided the revised rules for categorizing entities, this condition will be satisfied. If, however, some remote aspects of the same information system provided the semantic update, stability problems might arise again. Discussing them in detail would go beyond the scope of this paper.

Another strategy for reclassification which avoids oscillations would be to use only a time window of entries serving as definitional items for the rest of the whole data base. For further discussions related to such issues of evolutionary versus revolutionary learning, readers are referred to [4] and [3].

5 Conclusion

It has been demonstrated that shifting real world semantics may cause the semantics of data captured in crisp and, therefore, static information systems to deteriorate over time. Fuzzy data representation allows to express a broader spectrum of complex semantics than the token-semantics of conventional data base entries.

To fully account for the dynamics of evolving domains, fuzzyness has to be propagated to the schema level and thus to the conceptual model. This will allow to distinguish between the actual token-semantics of a designator and the semantic shades this designator has on the basis of the semantic support kept at the extensional level of the information system. Explicitly representing this fuzzyness allows changes in the semantics of instances to be reflected on the conceptual level too.

As a consequence of integrating ideas of "computing with words" into a conceptual model, it can preserve the shades of natural language utterances that normally get lost when these statements are categorized and expressed in some formal language. By making explicit the support a conceptual term finds on the extensional level, users not only obtain better interpretation of the data represented, they also have the chance to monitor how interpretations shift over time.

References

1. ANSI/X3/SPARC Study Group on Data Base Management Systems: Interim Report. FDT (Bulletin of ACM-SIGMOD) 7(2) (1975)
2. Bennett, K.H., Rajlich, V.T.: Software Maintenance and Evolution: A Roadmap. In: Finkelstein, A. (ed.) The Future of Software Engineering 2000, May 2000, pp. 73–87. ACM Press, New York (2000)
3. Bouchachia, A.: Advanced Soft Computing Techniques, Habilitationsschrift, Univ. Klagenfurt (2006)

4. Bouchachia, A.: Incremental Learning. In: Wang, J. (ed.) Encyclopedia of Data Warehousing and Mining, 2nd edn., Idea-group (2007)
5. Eder, J., Koncilia, Ch., Wiggisser, K.: Maintaining Temporal Warehouse Models. In: Proc. of the IFIP International Conference on Research and Practical Issues of Enterprise Information Systems, pp. 21–30. Springer, Heidelberg (2006)
6. Eder, J., Mittermeir, R., Wernhart, H.: Induktive und deduktive Ermittlung des Informationsbedarfs. In: Proc. EMISA Fachtagung Informationsbedarfsanalyse und Informationsbedarfsermittlung, Linz, Informatik Fachberichte Nr. 143, Juli 1987, pp. 89–118. Springer, Heidelberg (1987)
7. Eder, J., Wiggisser, K.: Data Warehouse Maintenance, Evolution and Versioning. In: Liu, L., Özsu, T. (eds.) Encyclopedia of Database Systems, Springer, Heidelberg (to appear, 2008)
8. Hassler, M., Bouchachia, A., Mittermeir, R.T.: Classification of XML Documents. International Journal of Information Technology and Intelligent Computing 2(4) (2007)
9. Lehman, M.M.: Programs, Life Cycles and Laws of Software Evolution. Proceedings of the IEEE 68(9), 1060–1076 (1980)
10. Kop, Ch., Mayr, H.C.: Mapping Functional Requirements: From Natural Language to Conceptual Schemata. In: Proceeding of the 6th IASTED International Conference Software Engineering and Applications (SEA 2002), Cambridge (USA), November 2002, pp. 82–87 (2002)
11. Mittermeir, R.T.: Software Evolution: A Distant Perspective. In: Proc. 6th International Workshop on Principles of Software Evolution, IWPSE 2003, pp. 105–112. IEEE-CS Press, Los Alamitos (2003)
12. Mylopoulos, J., Bernstein, P.A., Wong, H.K.T.: A Language Facility for Designing Database-Intensive Applications. ACM Trans. on Database Systems 5(2), 185–207 (1980)
13. Borgida, A., Mylopoulos, J., Wong, H.K.T.: Generalization/Specialization as a Basis for Software Specification. In: Brodie, M.L., Mylopoulos, J., Schmidt, J.W. (eds.) On Conceptual Modeling, pp. 87–114. Springer, Heidelberg (1984)
14. Nepejvoda, N.N.: A bridge between constructive logic and computer programming. In: Bjorner, D., Kotov, V. (eds.) Images of Programming - Dedicated to the Memory of A.P. Ershov, pp. 253–270. North-Holland Publ., Amsterdam (1991)
15. Zadeh, L.A.: Fuzzy Logic = Computing with Words. IEEE Trans. on Fuzzy Systems 4(2), 103–111 (1996)

An Ontological-Based Approach to Analyze Software Production Methods

Óscar Pastor, Sergio España, and Arturo González

Department of Information Systems and Computation
Valencia University of Technology
Camino de Vera s/n, 46022 Valencia, España
{opastor,sergio.espana,agdelrio}@dsic.upv.es
Tel.: +34 96 387 7000; Fax: +34 96 3877359

Abstract. In the Information Systems (IS) area, countless modelling techniques exist to deal with the different perspectives of reality. Many software development methods propose many languages to make descriptions. How to align them correctly following an ontological approach is still an open problem. Instead of facing how to incrementally improve pre-existing methods, more and more proposals are introduced time after time, based on different paradigms or combining (in an apparently innovative way) current IS concepts. We seek to establish a conceptual framework for IS that helps to reason about different software development approaches. In this work, starting from a conceptual framework built on the basis of the FRISCO proposal, we explain how to trace the conceptual alignment of software production methods; that is, how the modelling primitives proposed by the method map to the elements of our reference framework. As a practical example, we align part of the Rational Unified Process, a widespread general-purpose software development method, to show how the different pieces of the method fit conceptually in the proposed ontology.

1 Introduction

In recent years, the use of ontologies is seen as a need for rigorously developing Information Systems. Behind any ontology we find a process of conceptualisation and we argue that appropriate and well-defined concepts are essential for the IS arena. We need precise concepts to understand which are the basic building blocks of the different models that compose a given software production method, and how they can be correctly combined. Accepting that we need those concepts, we are strongly demanding to understand which ontology underlies each method. The lack of such ontology hinders the comparison, understanding and evaluation of the different components of software production methods.

This is the challenge that we want to face in our work. Starting from what we consider a sound framework of Information System concepts –the FRISCO report [1]-, we have defined our basic ontology for IS understanding by enriching the initial framework with some new, required concepts. Once this ontological framework is defined, we introduce what we refer to as conceptual alignment: a strategy that, given a particular software production method, allows us to understand it correctly. This is achieved by analyzing how the modelling primitives proposed by the method map to the elements of the framework.

We have been working on a concrete proposal of such a generic framework for IS concepts, and our initial results can be found in [10]. Additionally, we have applied it in the context of our OO-Method software development process [11]. In this paper, we go one step forward and, using a complete integrated view of our ontological approach, we apply it to show its usefulness.

The ontology plays a basic role in the conceptual alignment. Having a precise conceptualisation provided by the ontological background (the model of the IS), we can argue (through the subsequent conceptual alignment) on the strengths and problems of a particular method. We even expect that, by aligning several methods, they can be compared and, consequently, the potential benefits and drawbacks of each method can be discussed.

The structure of the paper reflects our objectives. After introducing and extending the original FRISCO proposal (section 2), we introduce in section 3 the notion of conceptual alignment as a practical application of the extended conceptual framework for IS. This notion provides a way for aligning the concepts used in software production methods, always according to the proposed reference model. After that, as a practical proof of concept, we apply these ideas to a well-known software process: the RUP [12]. Our intention is to show in section 4 how the application of our ontological framework makes really possible to dissertate on the main issues related with the methodological properties of (in this case) the RUP proposal. Section 5 closes the work with the final conclusions and the presentation of further works.

There are three main ideas in this paper:

- The definition of a conceptual framework for Information Systems that starts from (and enriches) the FRISCO proposal, from an ontological perspective.
- The introduction of the concept of Conceptual Alignment as a useful application of the previous framework, intended to understand the semantics underlying any software production method.
- The particular application of the framework to RUP, a well-known and widely used software process, to demonstrate all the previous ideas.

We argue that reasoning about the ontology that underlies the development of organisational management software is of much value for the Software Engineering community, since it allows gaining deeper knowledge of this area, and that it also benefits the Ontology community because it focuses on some practical needs concerning software development methodology.

2 An Extended Conceptual Framework for Information Systems

Information Systems (IS) aim to support the mission of a particular Organisational System (OS) by means of meeting its needs for communication. Commonly, both the IS and the OS are framed in a Subject System which acts as the environment from where stimuli come.

Organisations are often very complex systems, characterised by the existence of actors who require the adequate knowledge and resources to carry out their tasks. When something of interest happens in the environment of (including within) the organisation, this event is reported to the IS by means of interacting with it. Since IS are systems, a systemic approach to their understanding is adequate.

In order to reason unambiguously about Information Systems, a framework of well defined and related concepts is desirable. In the early 80's, an ISO report [2] presented a conceptual framework for Information Systems, defining a basic interaction architecture where every IS reacts to messages with respect to different objectives. Information System's own objective is to memorise reported facts. Organisational System's higher objective is to react to these facts (events).

Some years later, the IFIP WG 8.1 Task Group issued the FRISCO report [1]. FRISCO models the dialog occurring in an interaction, disregarding the input or output intention. In the FRISCO model of the IS, users are actors who specify input messages to the processor and receive output messages from it. Processors are actors responsible for checking input messages, maintaining the domain model denotation (i.e. the information base) logically consistent with the language representation (i.e. the conceptual model) and retrieving information to produce output messages; in short, a processor is responsible for the IS reaction to external stimuli.

Ontological evaluation of methods has been argued to be strongly founded on theory [14]. However, the desired conceptual alignment of methods is only feasible if a common reference model is shared. The Bunge-Wand-Weber (BWW) ontological model [15] has been typically used for ontological evaluations. The BBW approach considers Information Systems as a mere representation of the Subject System; the authors see IS as technological systems. We claim that IS are socio-technological systems. Therefore, the BWW ontology lacks many social, human and communicational concepts that we consider characteristic of IS. Following what we could call the FRISCO tradition, and based on FRISCO's conceptual framework, we have presented in [10] an extended and refined Information System model intended to achieve a better understanding of the complexity of Information Systems. In Fig. 1 we depict the model graphically.

The change that some external actions induce in the state of the Subject System is of great interest to the Organisational System. We refer to these external action occurrences as events (the term is used as in [1],[2],[16]). A Primary actor is aware of that event and he/she informs the organisation by formulating an input message (e.g. a client ordering some products). However, Organisational Systems usually define a specific shared code [13] in which to express the messages (e.g. business forms), a shared knowledge [4] is also needed to perform a communicative act (e.g. product names and availability), and messages can be quite complex. Therefore, the message must be edited and encoded in order to be understood by the Information System (e.g. the clerk uses the graphical interface). Then, the IS reacts. The reaction typically involves updating the information base and conveying information to those organisational actors that need it to carry out their activities (e.g. the warehouse operator needs to receive the production order; the clerk needs to tell the client that his order will be served). A detailed description of the conceptual framework can be found in [10].

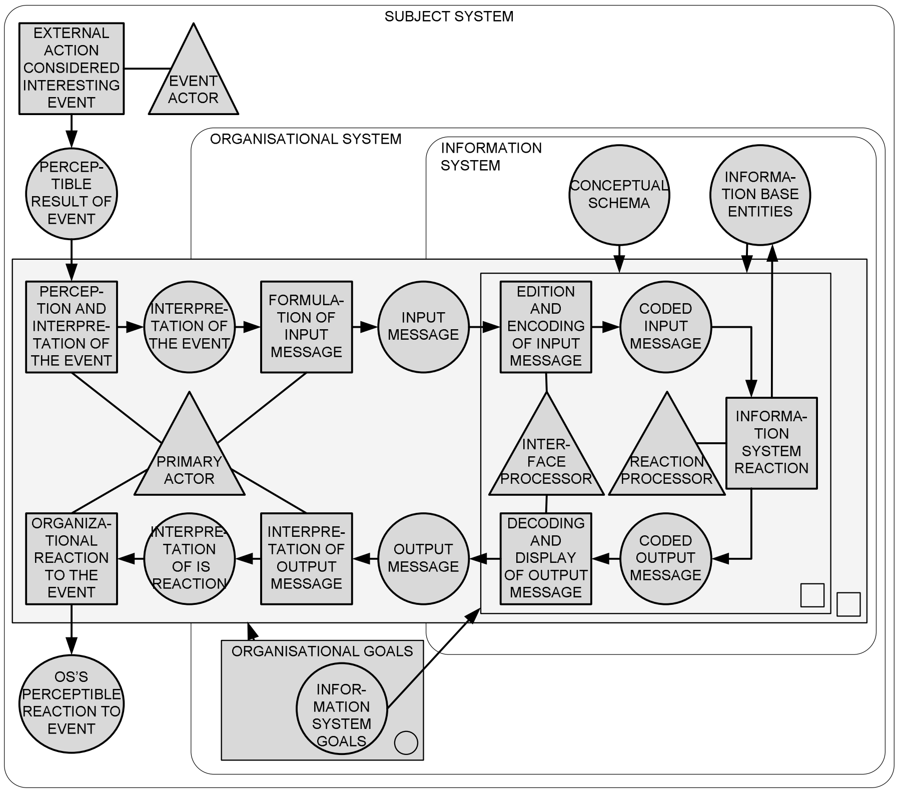

SUBJECT SYSTEM
EXTERNAL ACTION CONSIDERED INTERESTING EVENT
EVENT ACTOR
PERCEPTIBLE RESULT OF EVENT
ORGANISATIONAL SYSTEM
INFORMATION SYSTEM
CONCEPTUAL SCHEMA
INFORMATION BASE ENTITIES
PERCEPTION AND INTERPRETATION OF THE EVENT
INTERPRETATION OF THE EVENT
FORMULATION OF INPUT MESSAGE
INPUT MESSAGE
EDITION AND ENCODING OF INPUT MESSAGE
CODED INPUT MESSAGE
PRIMARY ACTOR
INTERFACE PROCESSOR
REACTION PROCESSOR
INFORMATION SYSTEM REACTION
ORGANIZATIONAL REACTION TO THE EVENT
INTERPRETATION OF IS REACTION
INTERPRETATION OF OUTPUT MESSAGE
OUTPUT MESSAGE
DECODING AND DISPLAY OF OUTPUT MESSAGE
CODED OUTPUT MESSAGE
ORGANISATIONAL GOALS
INFORMATION SYSTEM GOALS
OS'S PERCEPTIBLE REACTION TO EVENT

Fig. 1. A generic Information System model

To prove the validity of this conceptual framework, it should be reviewed by other experts in the area of IS (this is further work). The generic IS model can also be applied to well-known software production methods, to evaluate if it really allows to understand the strengths and weaknesses of a given method, and to assess that a semantically valid comparison is feasible. As commented before, these are two main results of this work: 1) it is possible to define a generic Information System model; 2) the Conceptual Alignment of any method using the previously defined reference model is possible.

3 On Perspectives and Conceptual Alignment of Methods

Software methods contain families of languages. Each language has a reduced set of concepts and some syntactic patterns to perceive and describe reality. These languages produce descriptions which are related in two ways. The descriptions have horizontal relations as they depict complementary aspects (perspectives) of reality. They have vertical relations as they give us different degrees of detail about reality. The refinement into a deeper level of detail is called decomposition. These horizontal and vertical abstraction mechanisms are often combined to describe reality. Since covering both perspectives and decomposition would be lengthy, in this paper we focus on perspectives. The issue of decomposition is left for further work.

We call **perspective** the way we project aspects of reality using a syntactic pattern of concepts. Perspectives are also called models, descriptions and specifications. Each perspective describes reality by means of a series of modelling primitives used as a filter through which to see reality (see Fig. 2). These modelling primitives can be mapped to the elements of the Information System framework. Modelling primitives are also called modelling constructs.

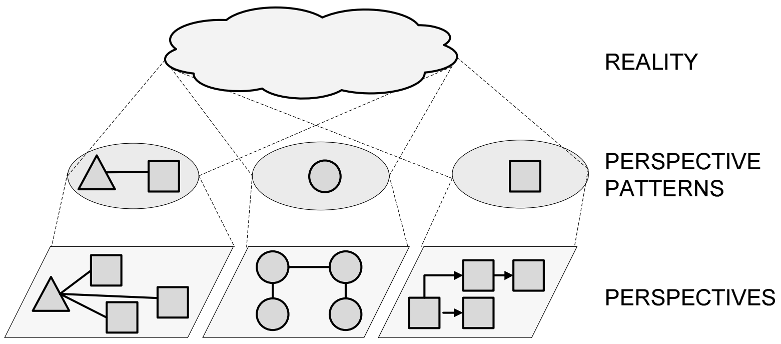


Fig. 2. Perspective patterns act as a lens through which we see reality

We consider these modelling primitives to be a pattern. We use the term **perspective pattern** to refer to the concepts and syntactic patterns that are used to obtain a certain description of reality. A perspective pattern is, for example, an ontological view derived from the proposed conceptual framework (the IS ontology). See Fig. 3.

The **conceptual alignment** of a given method is the establishment of mappings between the elements of the IS ontology and the modelling primitives of the modelling languages proposed by the method. The modelling primitives can be conceptually aligned either to the IS ontology (IS conceptual framework) or to an ontological view (perspective pattern).

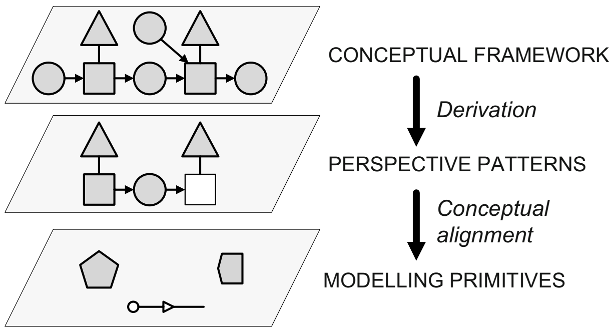


Fig. 3. Perspective patterns are ontological views that we can use to align modelling primitives

Our motivation for doing this work is to analyze and obtain deeper knowledge about the concepts underlying software production methods. We aspire to reason and, if possible, help other practitioners reason about the methods they use or propose. However, we do not seek to fix an ontological domain that holds as a universal reference. This is why we ruled out the specification of conceptual alignments in a more formal way. We expect to use our framework to support a series of methodological research lines. According to McGuinness' characterisation of ontologies, our conceptual framework should be taken as a "simple ontology" [7] addressing basic concepts.

4 Aligning the Underlying Concepts of RUP

This section is structured in two parts: first, a necessary brief introduction of RUP is provided. After that, the corresponding conceptual alignment of some of the most prominent the RUP high-abstraction models is presented. Concretely, we will focus on the Use-Case Model and the Analysis Model.

4.1 RUP at a Glance

The Rational Unified Process (RUP) [12] is a software engineering process that provides a disciplined approach to software development by defining responsibilities, tasks and artefacts to be created. RUP is a commercial product developed and marketed by Rational Rose[1], an IBM company. It defines four phases (Inception, Elaboration, Construction and Transition) which are enacted by means of iteratively working on different disciplines (Business Modelling, Requirements, Analysis, Design and Implementation among others).

Each discipline is described in terms of workflows (a temporal ordering of groups of activities), roles (a way of classifying stakeholders depending on their responsibilities), activities (a unit of work a role may be asked to perform) and artefacts (modelling primitives and documents that activities produce, evolve, maintain or use as input).

The authors of RUP remark the following main characteristics [5]:

- Use-Case driven. Use Cases are a requirements elicitation technique which allows the specification of fragments of system functionality that provides some value to the users. The RUP employs them not only as functional requirements capsules but also to drive the design, implementation and testing of the software application.
- Centred on architecture. The architecture of a system is a view of the organisation and structure of its most relevant parts, which involves both static and dynamic aspects of the system. It is related with how the system is to be built and in which order. Non functional requirements such as quality, performance and reuse need to be taken into account. The architecture is influenced by the chosen hardware and software platform, operating system, database management system, networking protocols, legacy systems to be interacted, etc.
- Iterative and incremental. The RUP proposes the division of work into manageable parts. Each iteration can be seen as a mini-development project where the main discipline workflows are enacted. This allows the Use Cases and the architecture to balance out progressively.

[1] http://www-306.ibm.com/software/rational/

The RUP recommends certain modelling techniques to build the artefacts; these are mainly based in the Unified Modelling Language (UML) [8], a general-purpose modelling language that includes a standardised graphical notation used to create perspectives of a (software) system.

The RUP offers a wide range of modelling activities and techniques; it is intended to be tailored to suit the needs of a particular project. We will focus on the models that specify the Information System in a more abstract way to present next their corresponding conceptual alignment. Whenever the RUP offers some leeway to interpret the concepts behind a modelling primitive, we have reduced the degree of freedom in order to map the primitive to the appropriate framework concepts.

4.2 Conceptual Alignment of the RUP High Abstraction Models

The next sequence of diagrams describes some of the techniques proposed by the RUP in terms of its corresponding conceptual alignment. At each step, the same strategy is followed: we display a perspective pattern derived from the IS ontology; then we "colour" the relevant elements depending on the specification technique that it is being considered. The mappings between IS elements and modelling primitives are traced. For the sake of understandability, it is important to select an adequate notation. The notation that we use contains (see Fig. 5):

- Elements currently being specified at a certain step. A thick-lined shape with background represents an IS element that is being described at that very step.
- Elements which have not yet been specified. These are denoted by a soft grey line and font.
- Elements that have already been specified. A normal dark-lined shape with background represents IS elements which have been specified in previous steps and which are not relevant in the current specification step.
- A thick, dashed line represents the mapping between a particular element of the framework of reference (or a generalisation of elements of the same type, or a heterogeneous aggregation) and the modelling primitives of the given technique.

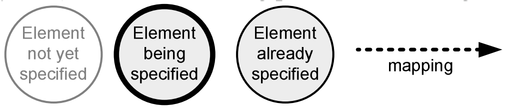


Fig. 4. Notation for the conceptual alignment

According to this notation, we develop next the conceptual alignment of some of the modelling techniques RUP recommends in order to describe the Information System and its computerisation. To exemplify the modelling techniques we adapted the Wiley College case study[2], which documents the development of a student registration system. The resulting client-server application should allow professors to access

[2] IBM offers the Wiley College case study as an example of a tailoring of RUP; more info can be found in http://www-128.ibm.com/developerworks/rational/library/4315.html. We have adapted the case study when necessary.

the system to sign up to teach courses as well as record grades. Students should be able to register for courses and view report cards.

4.2.1 Use-Case Model

The Use Case Model specifies the Computerised Information System functions and its relation with the users. This model is composed of actors (also referred by RUP as system actors to acknowledge the fact that they interact with the IS) and use cases (the capsule of functionality that will guide the rest of the software development process).

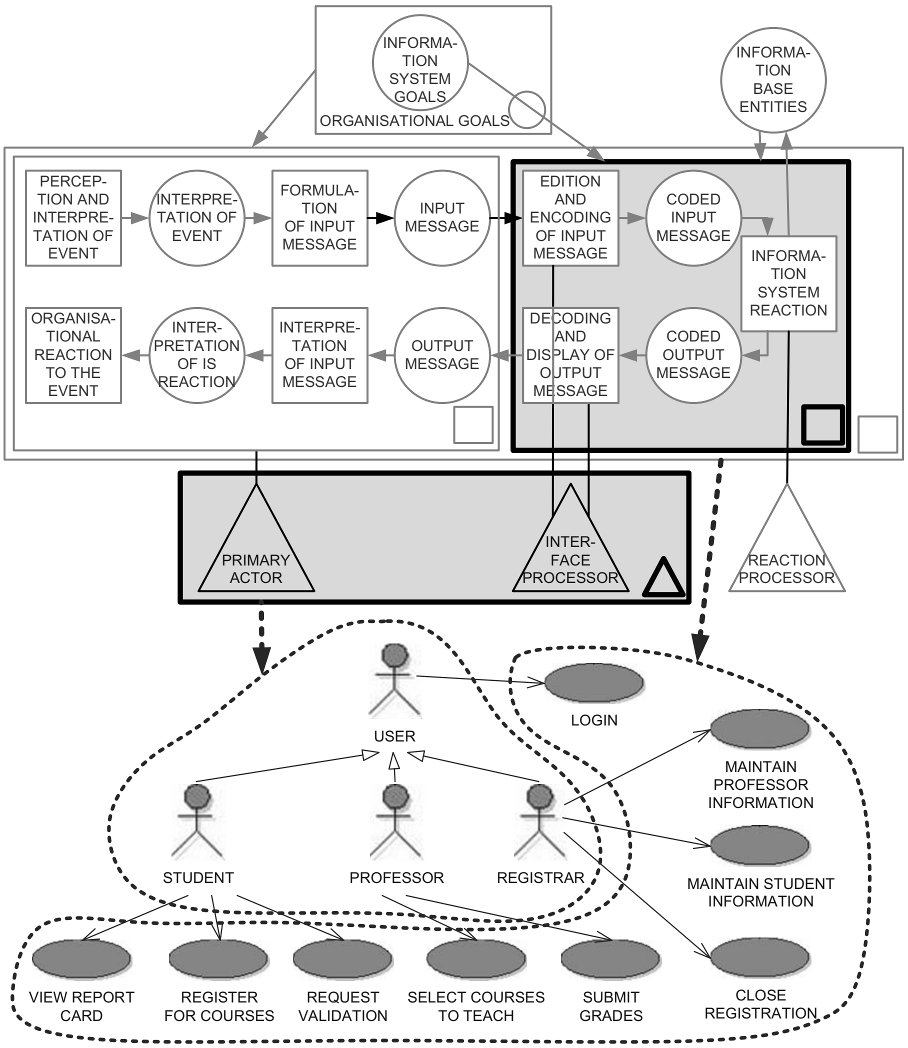

INFORMATION SYSTEM GOALS
ORGANISATIONAL GOALS
INFORMATION BASE ENTITIES
PERCEPTION AND INTERPRETATION OF EVENT
INTERPRETATION OF EVENT
FORMULATION OF INPUT MESSAGE
INPUT MESSAGE
EDITION AND ENCODING OF INPUT MESSAGE
CODED INPUT MESSAGE
INFORMATION SYSTEM REACTION
ORGANISATIONAL REACTION TO THE EVENT
INTERPRETATION OF IS REACTION
INTERPRETATION OF INPUT MESSAGE
OUTPUT MESSAGE
DECODING AND DISPLAY OF OUTPUT MESSAGE
CODED OUTPUT MESSAGE
PRIMARY ACTOR
INTERFACE PROCESSOR
REACTION PROCESSOR
USER
LOGIN
MAINTAIN PROFESSOR INFORMATION
STUDENT
PROFESSOR
REGISTRAR
MAINTAIN STUDENT INFORMATION
VIEW REPORT CARD
REGISTER FOR COURSES
REQUEST VALIDATION
SELECT COURSES TO TEACH
SUBMIT GRADES
CLOSE REGISTRATION

Fig. 5. Defining actors and use cases in the Use Case Model

The Business Analysis model (done previously in the development process) has already identified those actors which belong to the OS and take part in the realisation of business use cases; these actors are referred to as business workers. Now, an actor of the system will be defined for each business worker who will actually use the IS. In the case study the professor will use the system to select the courses he/she is going to teach and to submit the student grades. Other business workers will not become system actors, if the processes they take part of are not going to be computerised. Also, we may decide that some business actors (who are not part of the OS) are going to interact with the IS. This is the case of the students, who will interact with the system to register for the offered courses and to view the report card, at the end of the semester. Some additional actors of the system may appear at this step, typically those of the administrator type (e.g. the registrar). Since system actors define "a coherent set of roles that users of the system can play when interacting with it" [12], they correspond to *Interface processors* that will communicate messages with the IS. In any case, they also may be playing the role of *Primary actors* if they provide the new information to the IS.

Use cases are an abstraction of actions performed within the IS. They define the scope of the software implementation, since only the functionality they define will be computerised.

The perspective pattern of use cases shows an ellipsis of some communicational elements. The *Primary actor* is assigned an aggregate function of the IS, but that relation overlooks the *Input message* and the *Output message* (see Fig.5). The relationship between a system actor and a use case is purely a trigger that only takes into account the phatic function of communication [4]. Even when use cases are refined by means of use case descriptions or scenarios (a description of the sequence of actions done to carry out the use case), the message that is specified (if any) is expressed in terms of the application interface.

4.2.2 Analysis Model

Just as the Business Analysis Model describes the realisation of business use cases, the Analysis Model is an object model describing the realisation of use cases in terms of analysis classes. Analysis classes may be of three different kinds:

- Boundary classes act as an interface to actors outside the IS, whether they are human (then these classes model the user interface), devices (then they model peripheral device/sensor drivers) or other systems. Conceptually, all three uses of the boundary classes model the *Edition and encoding of input message* and *Decoding and display of output message*, independently of whether the actor is a sensor, a human or a whole integrated subsystem.
- Control classes model control behaviour concerning one or several use cases. These classes correspond to an encapsulation of the *Information System reaction.*
- Entity classes model the IS memory; they represent the information that needs to be stored persistently to keep track of changes in the Subject System and the Organisational System. However, since these classes also model behaviour associated to these pieces of information, they are mapped to a heterogeneous aggregation of *Information base entity* (the static view of the class) and *Information System reaction* (its dynamic view).

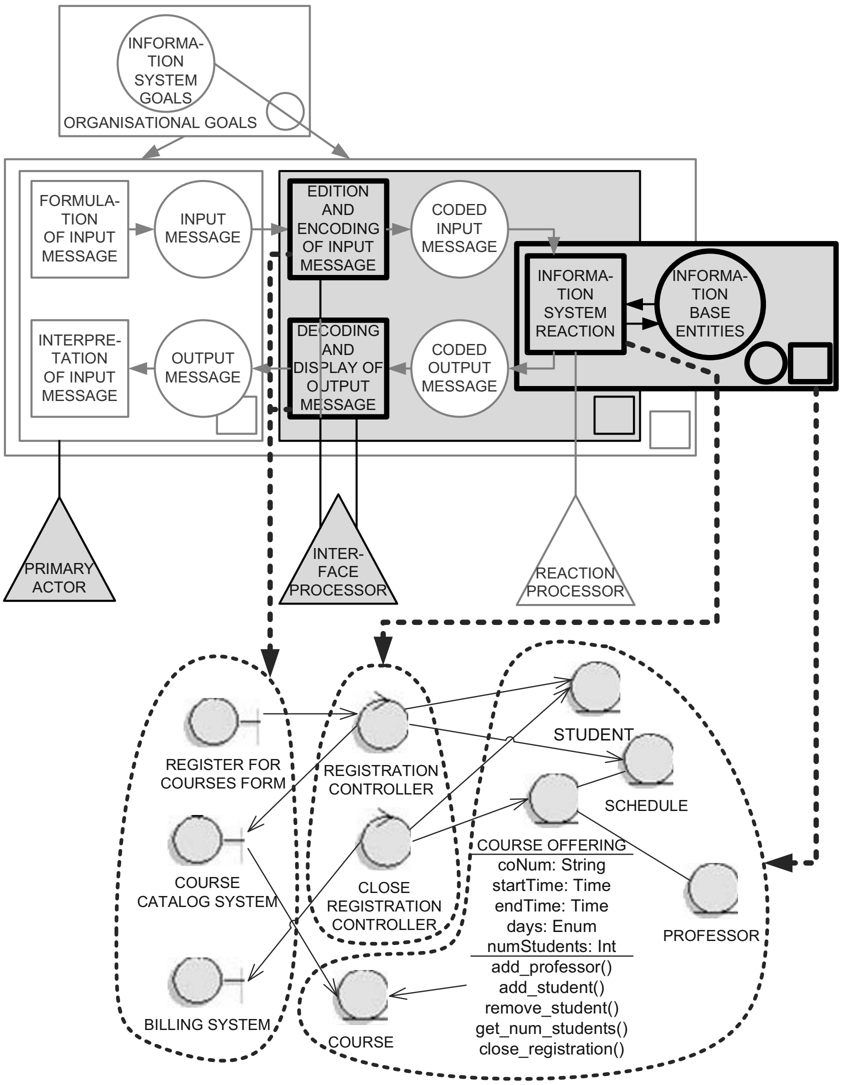


Fig. 6. Specifying the memory and the reaction of the system in the Analysis Model

Fig.6 shows this alignment; note that, for the sake of brevity, only one of the analysis classes (the Course Offering) has been expanded to see its attributes and methods.

RUP continues with the specification of the Design Model, in which the system is further refined in terms of the chosen platform and programming language. Then the implementation is carried out, obtaining the final software application. Many other models support the rest of the process (e.g. architectural decisions, testing, deployment and process management) but we have restricted the conceptual alignment to those with higher abstraction.

5 Conclusions and Further Work

Starting from a conceptual framework -taken as a simple but complete ontology about Information Systems (IS) that includes the most important concepts related to IS-, a rigorous conceptual alignment can be accomplished to characterise any software production method. As a proof of concept, we have applied these ideas to the Use-Case and the Analysis Models of the RUP.

Software production methods propose a set of techniques, as well as some guidelines to put them in practice. Some techniques have an associated modelling language that helps to describe reality. This language will offer a set of modelling primitives. With a language it is possible to make a description of reality; that is, models (like the ones in the case study). However, each language is only capable of describing a series of aspects of reality (by using them one can not take in the whole reality); that is, models are perspectives of reality. In principle, the ontology we have proposed is wider than the particular techniques for describing IS (if we ever find a lack in this sense, we will surely extend the framework). See Fig. 7 for an overview of these terms and the layers[3] they belong to [8].

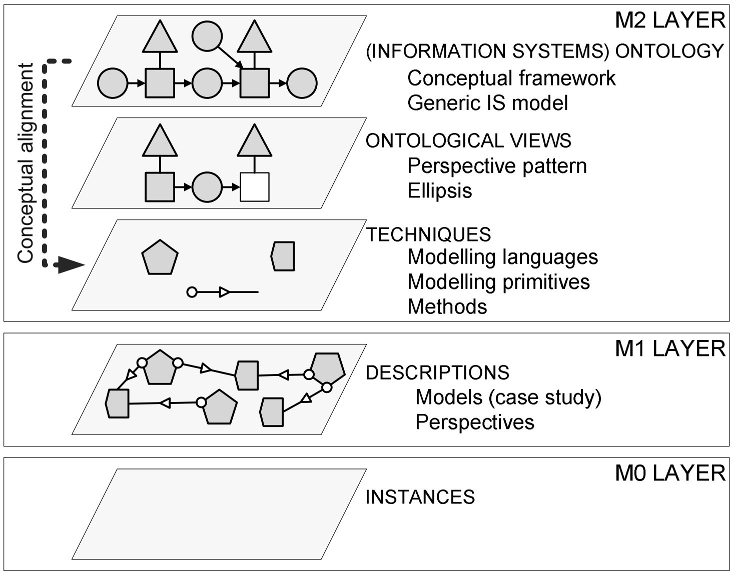


Fig. 7. Organisation of the discussed concepts on layers

The aspects of reality that a technique allows to describe define a view over the proposed IS ontology. This view is what we call perspective pattern; it is a filter over the framework, a partial navigation over its elements. The perspective pattern is a syntactic

[3] Whether the IS ontology belongs to the M2 layer is a matter of modelling intention. Although we do not expect our conceptual framework to be used as a metamodel, it can definitely be used for this purpose. Also, this allows us to organise the concepts clearly.

pattern used to describe reality. It is expected that, when used in combination, the set of techniques proposed by a method accomplish to describe enough aspects of reality (enough in the sense that the IS can be later successfully implemented).

Conceptual alignment is the establishment of mappings between the concepts of the ontology and the modelling primitives.

It must be clarified that in order to show the conceptual alignment of RUP and, at the same time, exemplify its proposed techniques with a case study, we have used descriptions (perspectives) instead of language definitions (modelling primitives). That way, when we trace mappings between an element of the ontology (concept) and an element of a description, we are implicitly tracing the mapping between the concept and the modelling primitive used to instantiate that element of the description. A little effort needs to be done in order to mentally classify the elements of the description, but we are killing two birds with one stone.

Defining a reference conceptual framework in the area of Information Systems has allowed us to align the concepts behind some techniques proposed by a well-known general-purpose software development method, the Rational Unified Process. We expect that this reasoning tool that we are proposing will help to better understand the strengths and weaknesses of development methods. As an example, we noted a total ellipsis of some communicational elements of the IS; this appears to be common in use case-based techniques. Furthermore, we expect that it will allow comparing different methods on a common conceptual basis.

One could dare to ask the following question: Which is the most adequate election of perspectives? That is, which is the best software production method? But that is definitely not our purpose. We believe that there is not “the best” method, since the adequate election of perspective patterns is conditioned to the problem we are trying to solve and the knowledge of the development team. Surely no one holds “the truth” in Software Engineering. However, an improvement could be done in the criteria of the existing techniques.

The Information Systems ontology that we are proposing could help the Conceptual Modelling community to revise their proposals. Our future work has this aim. We plan to carry out a validation of our conceptual framework using the Delphi method ([6]). Starting with a sound conceptual background, we aspire to reflect on commonly accepted techniques in order to align their concepts and identify criteria of use. Our concern is to improve the quality of conceptual models. In this field, there is strong need of methodological experimentation to build an empirical base of knowledge. We are starting to drive methodological experiments both on learners and experts using different modelling techniques or even using the same modelling language but with different criteria. This research line is promising.

We encourage the Conceptual Modelling community to approach Ontology. The study of conceptual alignments helps to reason about the view of the world that underlies software development methods. Modelling techniques are used to describe reality and, therefore, they are based on some ontological preconceptions that should surface and be explicit.

Acknowledgments

This work has been supported by the Ministerio de Educación y Ciencia (Ministry for Education & Science, Spain) under the project SESAMO TIN2007-62894.

References

[1] Falkenberg, E., Hesse, W., et al.: FRISCO. A Framework of Information Systems Concepts (IFIP WG 8.1 Task Group Report) (1998)

[2] van Griethuysen, J.J. (ed.) ISO TC97/SC5/WG3. Information processing systems - Concepts and terminology for the conceptual schema and the information base. TR9007 (1982)

[3] Jacobson, I., Booch, G., Rumbaugh, J.: The unified software development process. Addison-Wesley Longman Publishing Co. Inc., Boston (2000)

[4] Jakobson, R.: The speech event and the functions of language. In: Monville-Burston, M., Waugh, L.R. (eds.) On Language, pp. 69–79. Harvard Univ. Press, Cambridge (1990)

[5] Kruchten, P.: The Rational Unified Process: An Introduction. Addison-Wesley, Reading (2000)

[6] Linstone, H.A., Turoff, M. (eds.): The Delphi Method: techniques and applications (2002) (Last access: January-2008) Available online at: `http://www.is.njit.edu/pubs/delphibook/`

[7] McGuinness, D.L.: Ontologies come of age. In: Fensel, D., Hendler, J., Lieberman, H., Wahlster, W. (eds.) Spinning the Semantic Web: Bringing the World Wide Web to its full potential, MIT Press, Cambridge (2003)

[8] OMG. Unified Modeling Language: Superstructure version 2.0 (2005) (Last access: December-2007) Available: `http://www.omg.org/docs/formal/05-07-04.pdf`

[9] OMG. Meta object facility (MOF) core specification v2.0 (2006) (Last access: December-2007) Available: http://www.omg.org/cgi-bin/apps/doc?formal/06-01-01.pdf

[10] Pastor, O., González, A., España, S.: Conceptual alignment of software production methods. In: Krogstie, J., Opdahl, A.L., Brinkkemper, S. (eds.) Conceptual modelling in information systems engineering, Springer, Heidelberg, Germany (2007)

[11] Pastor, Ó., Gómez, J., et al.: The OO-method approach for information systems modeling: from object-oriented conceptual modeling to automated programming. Information Systems 26(7), 507–534 (2001)

[12] Rational Software Corporation. Rational Unified Process Version 2003.06.00.65 (2003)

[13] Shannon, C.E.: A mathematical theory of communication. Bell System Technical Journal 27, 379–423, 623–656 (1948)

[14] Siau, K., Rossi, M.: Evaluating of information modeling methods – a review. In: 31st Hawaii International Conference on Systems Science (1998)

[15] Wand, Y., Weber, R.: On the deep structure of information systems. Inf. Syst. J. 5, 203–223 (1995)

[16] Yourdon, E.: Modern structured analysis. Computing Series. Yourdon Press, Upper Saddle River, New Jersey, USA (1989)

Towards a Theory of Services

Wolfgang Reisig

Humboldt-Universität zu Berlin
reisig@informatik.hu-berlin.de

Abstract. Service-oriented Computing and Service-oriented Architectures aspire to better exploit existing middleware technologies. To this end, a more flexible, platform independent software design methodology is suggested, to develop rapid, low cost, interoperable, evolving and massively distributed software systems. Intended application areas include electronic commerce, information systems, supply chain control, and many other areas.

Time is ripe to underpin this effort by theoretically well-founded concepts to *model* services, and to *analyze* such models. In this paper we suggest first proposals and principles for a comprehensive theory of services.

Keywords: Service-oriented Computing, Service Models, Theory of Services.

1 Introduction

"Constantly growing complexity of information systems in industry and society already today approach the limits of our planning capacity. A formally founded tool box for planning, design development and operation of such systems is urgently needed. Methods are needed to determine the requirements at information systems. Likewise, we need formal concepts to describe, specify and analyze such systems, and to assess their behavior." This quote dates back almost 30 years, published as the first section of the introduction to the volume "Formal Models for Information systems" [1], authored by Heinrich Christian Mayr. Many of his demands still remain to be fulfilled adequately. A promising candidate to adequately address those problems is the emerging paradigm of *Service Oriented Computing* (SOC). The basic notion of this paradigm are *services.* A service is an autonomous, platform independent component; frequently a software component. The most fundamental aspect of a service is its ability to *communicate* with other services.

The functionality of a service can broadly vary, from quite simple activities such as answering a question, to involved exchanges of messages, as they occur in business processes, involving many partners and resources and including many alternatives, variants and iterations. A typical example of a service was a travel agent, communicating with other services such as flight carriers , hotels, ticket offices, etc. A client of a travel agent may want, say, to book a weekend flight from Klagenfurt to Berlin, to attend a Philharmonics performance. He would likewise behave as a service.

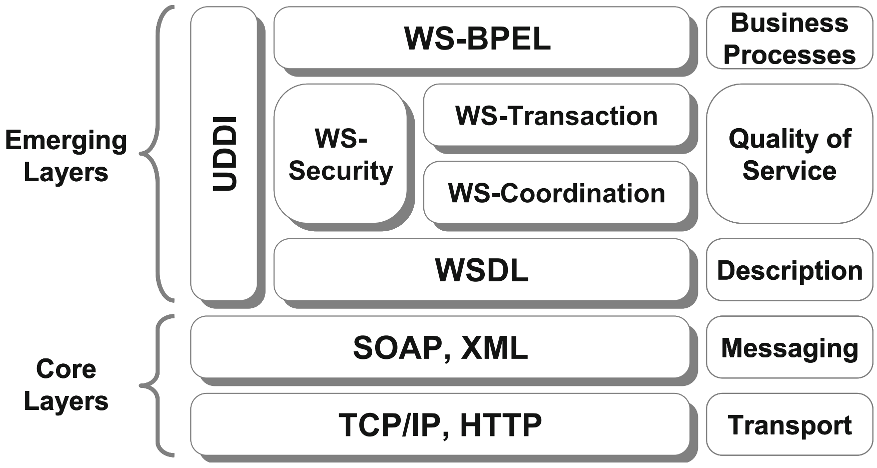


Fig. 1. SOC Technology Stack

Service oriented computing has always been triggered by available technology: Standard middleware technology is combined in a *technology stack*, reaching from basic technical layers for transport and messaging over layers for the description of services and service quality up to layers to orchestrate services, thus implementing business processes, electronic commerce, etc. Fig. 1 shows a typical such technology stack. The internet can be used as the underglying communication medium. Electronic commerce exploits just this technology.

This way, service-oriented computing emerged as a paradigm for new software architectures during recent years. Time is more than ripe to underpin this paradigm by a solid theoretical framework of notions to model services and to analyze relevant properties of such models. First ideas for such a framework will be compiled in the sequel.

The rest of this paper is organized as follows: In Chapter 2 we motivate basic assumptions and properties of services, followed in Chapter 3 by some first assumptions of a general framework for services. A canonical notion of *equivalence* is discussed in Chapter 4. Chapter 5 introduces a concrete modeling technique for services, which is refined in Chapters 6 and Chapter 7.

2 A Wish List for a Theory of Services

We start out with the observation that the classical model of computation does not apply to service-oriented Computing.

The classical model of computation abstracts computational systems down to *functions*, as shown in Fig. 2: Consumption of input is followed by a phase of computing, which ends with the delivery of the computation's output. Equal input for computations always yields equal output. The computing phase may input → computation → output

Fig. 2. The classical Model of Computation

diverge, i.e. fail to terminate. This case is conceived as an error. Consequently, there is no reason to distinguish different kinds of diverging computations.

In contrast to this model, a service S is "always on". An instance of cooperation of S with a service T may start at some states during the runs of S and T. A service is certainly not intended to terminate. In contrast, termination is conceived as failure. The essence of services is that they communicate *during* computation. Nevertheless, a concrete instance of communication usually does not run forever.

Summing up, a service is a *reactive* system. An output item of a service A may serve as an input item of a service B. In turn, B may send an item back to A. This way, A may impact its own input.

An adequate theory of services must cover those aspects. It furthermore has to clarify fundamental notions such as the semantics, equivalence notions, normal forms, etc., of services. More details can be found in [4] and [8].

3 Some First Assumptions

Communication of services is conceptually to be organized as a service *composition*. Composed services together behave with regard to their joint environment like *one* service. For example, a travel agent composed with a flight carrier and a hotel may jointly offer week-end trips from Klagenfurt to Berlin.

Each instance of communication between services may terminate. For example, a travel agent's communication with a client may terminate either with a contract, or with the understanding that the travel agent fails to meet the client's requirements. An example for irregular termination was a client's request never answered at all.

In a more refined setting, one may replace the requirement for termination by some more sophisticated "beauty predicate", or by additional conditions. As an example, each service may come up with its own predicate. Summing up, in the set $\mathcal{S}$ of all services we assume a binary *composition* operator

$$\oplus : \mathcal{S} \times \mathcal{S} \rightarrow \mathcal{S}$$

which is *commutative* i.e. for all services R, S holds

$$R \oplus S = S \oplus R.$$

Furthermore we assume one or more "beauty" predicate, shaped

$$p \subseteq \mathcal{S}.$$

In most cases, p denotes weak termination. This already lays the ground for a canonical, rich theory of services, covering a wealth of important notions, questions and properties, as they are particular important in the framework of Service-oriented computing.

Two services R and S are *partners* iff their composition meets the "beauty" contest, i.e. if

$$R \oplus S \in p.$$

As services are made to communicate, i.e. are to be composed, the *semantics* of a service S is the set

$$sem(S) =_{def} \{R|R \text{ is a partner of } S\}$$

of all partners of S.

The fundamental notion of *partners* of a service S gives rise to a lot of questions:

***Composability*:** Given another service R, is R a partner of S?
***Controlability*:** Does S have partners at all?
***Most liberal partner*:** Is there a canonical partner of S?
***Operation guideline*:** How characterize all partners of S?
***Adapter generation*:** Given another service R, construct a service T such that R is a partner of $S \oplus T$.

4 Equivalent Services

The above definition of the semantics of a service implies a canonical comparison of the comprehensiveness of the capabilities of services: The capabilities of a services R are at most as comprehensive as those of S (written $R \leq S$) iff each partner of R is a partner of S. Hence, comprehension is a partial order

$$\leq \subseteq \mathcal{S} \times \mathcal{S}$$

on the set of all services, defined by

$$R \leq S \text{ iff } sem(R) \subseteq sem(S).$$

A typical step towards a more comprehensive service was a travel agent, additionally offering ship cruises.

Consequently, to services S and S' are *equivalent* (written $S \sim S'$) iff they comprehend each other.

$$S \sim S' \text{ iff } sem(S) = sem(S').$$

This equivalence is in fact the canonical counterpart of functional equivalence in the classical setting: Two systems are equivalent whenever their environment can not distinguish them.

5 A Modeling Technique for Services

The above considerations provided a conceptual framework which is now to be substantiated by a concrete modeling technique for services. It seems to be obvious from the previous considerations that programming languages are no adequate candidates for this endeavor.

The most widespread technique to formally model services, are based on Petri Nets. This is motivated by the observation that the communication style of services, i.e. asynchronous message passing, is perfectly met by the semantics of places of Petri Nets. Consequently, a service S can be modelled as a Petri Net with distinguished places to model the interface of S.

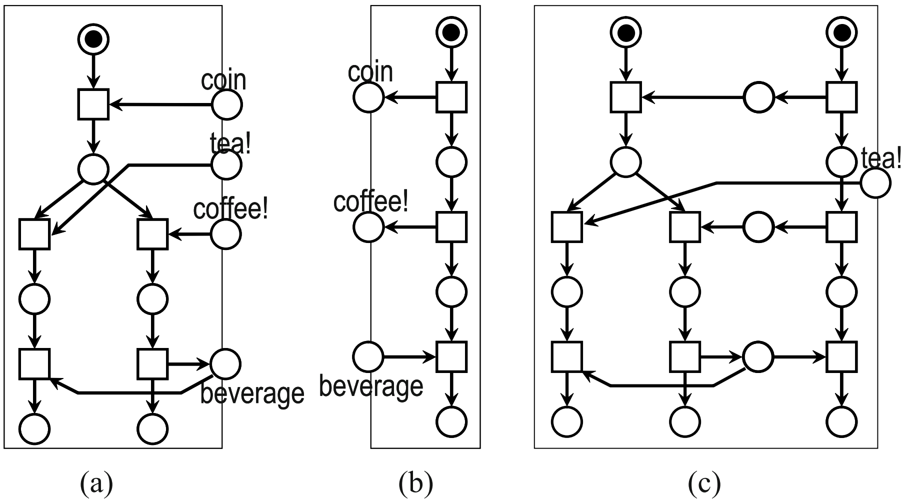


Fig. 3. Vending machine, coffee partner, and their composition

Figure 3 (a) and (b) show tow typical examples of services. Their intuitive meaning is obvious: (a) describes the service of a vending machine which expects its partners first to drop a coin and then to press a button, selecting coffee or tea. Finally, the vending machine provides the drink. Figure 3 (b) shows a corresponding partner. Finally, (c) shows the composition of the vending machine and its partner. The composed system terminates with tokens in terminal places. Technically, Petri Nets of this kind are denoted as *open nets*: An open net is a Petri Net where some places p with $p^\bullet = \emptyset$ are distinguished as *out-places* and some places q with ${}^\bullet q = \emptyset$ are distinguished as *in-places*. Out-places and in-places both are *interface places*. The initial marking of an open net assigns no tokens to the interface places.

The usual conventions for graphical representations for Petri Nets are applied to open nets, too. Additionally, the graphics is enclosed into a box, with the interfaces places on its surface. Figure 3 in fact applies these conventions.

for the composition $R \oplus S$ of two open nets R and S we assume w.l.o.g. that R and S are disjoint except for interface places. The sets of places, transitions, arcs and the initial marking of $R \oplus S$ are just the union of the corresponding sets of R and S. An out-place of R which is coincidently an in-place of S (and vice versa), an in-place of R which is coincidently an out-place of S) turns into an inner place of $R \oplus S$. Figure 3 (c) shows the composition of the two open nets as given in (a) and (b).

Besides the partner shown in Fig. 3 (b) the vending machine has many more partners.

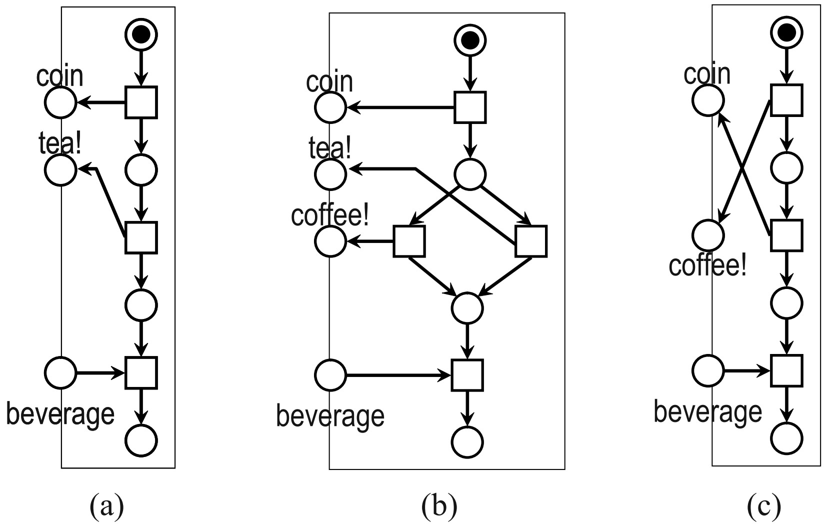


Fig. 4. Three more partners

Figure 4 (a) shows the tea-partner; (b) shows the coffee-or-tea-partner, and (c) shows that a partner may swap the order of dropping a coin and selecting a beverage. There is no need for a service model to do with only *one* thread of control:

Figure 5 shows a partner C with *three* threads of control. In fact, C is the *most permissive* partner: To derive any other partner form this one, extend the order in which actions occur in C, and fix one of the alternatives.

The *operating guideline* $OG(S)$ of a service S describes all possible ways to make use of S. In the above technical framework this means $OG(S)$ describes all partners, i.e. the semantics $sem(S)$. In fact, $OG(S)$ describes all partners, i.e. the semantics $sem(S)$. In fact, $OG(S)$ can finitely be described for each service S. Essentially, this description is an inscribed version of the most permissive partner of S.

It is interesting to observe that there exist services with no partners at all: Figure 6 shows a variant of the vending machine which internally selects either tea or coffee. A partner would have to be able to correctly "guess" this selection.

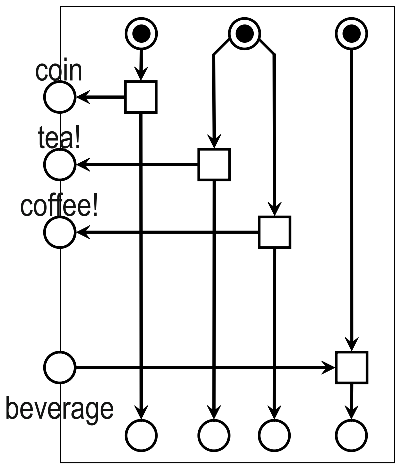


Fig. 5. Partner with three independent threads of control

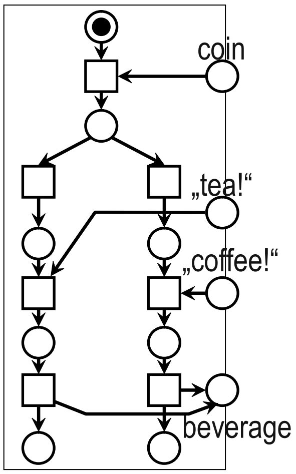


Fig. 6. A variant of the vending machine without partners

This kind of Petri Net models is expressive enough to define a feature complete semantics of the most important specification language for services, WS-BPEL [2],[3],[5],[7],[9],[10]. At the same time it is simple enough for a series of deep analysis techniques. For example, there are algorithmic solutions to all problems mentioned at the end of Chapter 3 [6].

6 Services with Ports

A theory of services must allow for *abstraction*. To this end we suggest *ports* as sets of interface places. Technically, the ports of an open net S define a partition of the interface places of S, i.e. each interface place belongs to exactly one port. Furthermore, each port is *named*, with different ports of a service S named differently. For reasons to become reasonable later, each port is either an *in-port* or an *out-port*.

Figure 7 extends two open nets of our running example by *ports*. *Payment* and *choice* are names of in-ports of the vending machine as well as out-ports of the coffee partner. Vice versa, *supply* is the name of an out-port of the vending machine, as well as of an in-port of the coffee partner. The composition $R \oplus S$ of

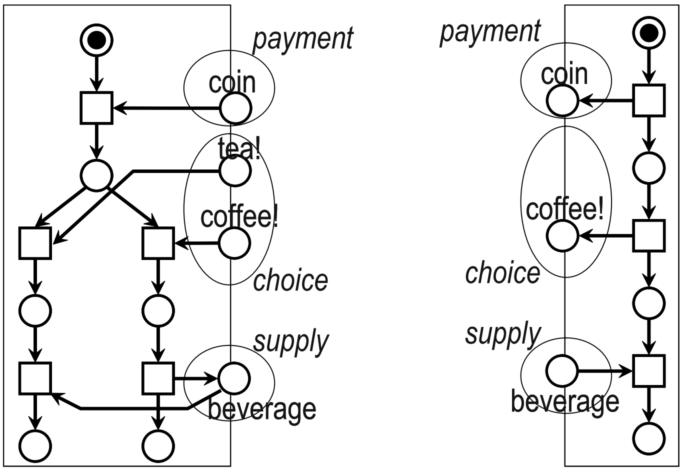


Fig. 7. Vending machine and coffee partner with ports

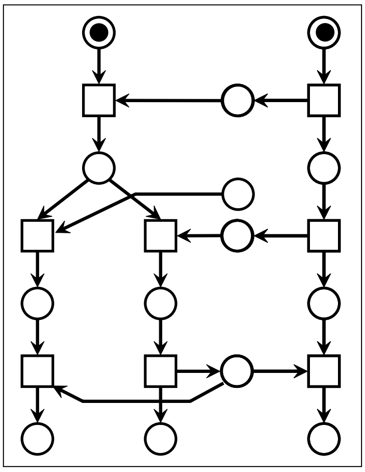


Fig. 8. Composition of the two open nets of Fig. 7

two open nets R and$S$ with ports $r_1, \ldots r_n$ and $s_1, \ldots s_m$ respectively, is again an open net with ports. $R \oplus S$ is composed as described for open nets with interface places. The in-ports of $R \oplus S$ are all

- $(r_i \cup s_j) \setminus P$ if r_i and s_j are equally labelled in-ports and P includes all in-places of r_i which coincidently are out-places of s_j (or vice versa),
- r_i if r_i is an in-port and no s_j is equally labelled,
- s_j if s_j is an in-port and no r_i is equally labelled.

The out-ports of $R \oplus S$ are constructed accordingly. Consequently, an in-port r_i and an out-port s_j with equal names turn their places into inner places of $R \oplus S$. This likewise applies to equally named out-ports r_i and in-ports s_j.

As an example, the composition of the "plain" vending machine and coffee partners in Fig. 3 (b) and [c) resulted in the open net c) with "tea" an in-place. This is counter-intuitive. In fact, the composition of the versions with ports , as in Fig. 7, results in the intuitively satisfying open net of Fig. 8.

7 Many Partners

A service (and hence an open net) may serve more than one partner. As an example, the vending machine's partner in Fig. 5 can be dissolved into three partners, as in Fig. 9.

They together serve the port equipped version of the vending machine in Fig. 7. Each of the three nets can be constructed without considering the other two. This is not possible for all open nets with ports. As an example, Fig. 10 (a) shows a variant where the partners for $port_1$ and $port_2$ must conjointly be constructed.

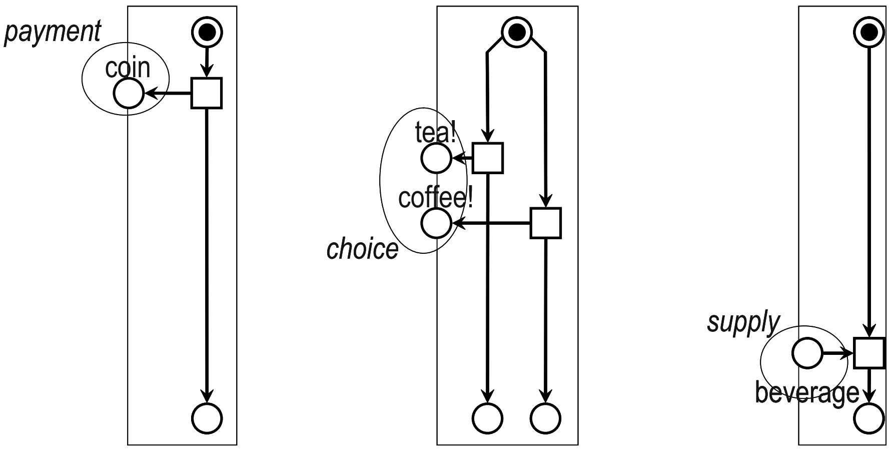


Fig. 9. Three partners of the vending machine

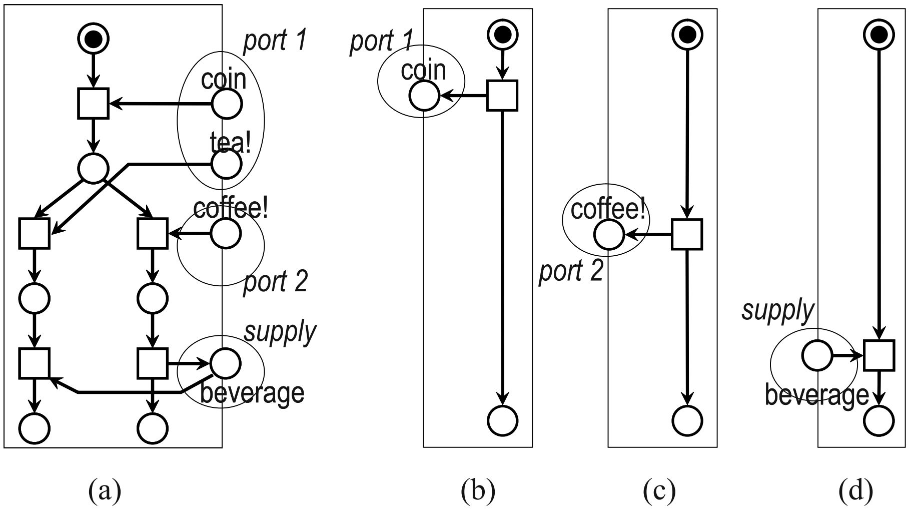


Fig. 10. Partners to be jointly constructed

8 Conclusion

The emerging paradigm of service-oriented computing requires a solid and fertile theory. Many aspects suggest that this theory essentially differs from the conventional theory of computation. The suggestions of this paper may provide a first step toward this theory.

References

1. Mayr, H.C., Meyer, B.E.: Formale Modelle für Informationssysteme, Informatik-Fachberichte Nr. 21. Springer, Heidelberg (1979)
2. Lohmann, N., Massuthe, P., Stahl, C., Weinberg, D.: Analyzing Interacting WS-BPEL Processes Using Flexible Model Generation. Data Knowl. Eng. 64(1), 38–54 (2008)
3. Lohmann, N., Kleine, J.: Fully-automatic Translation of Open Workflow Net Models into Human-readable Abstract BPEL Processes. In: Kühne, T., Steimann, F. (eds.) Modellierung 2008, Berlin, Germany, März 12-14, 2008. Lecture Notes in Informatics (LNI) GI. (to appear, 2008)
4. Reisig, W., Wolf, K., Bretschneider, J., Kaschner, K., Lohmann, N., Massuthe, P., Stahl, C.: Challenges in a Service-Oriented World. ERCIM News 70, 28–29 (2007)
5. Lohmann, N.: A Feature-Complete Petri Net Semantics for WS-BPEL 2.0. In: Dumas, M., Heckel, R. (eds.) WS-FM 2007. LNCS, vol. 4937, Springer, Heidelberg (2008)
6. Lohmann, N., Massuthe, P., Wolf, K.: Behavioral Constraints for Services. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 271–287. Springer, Heidelberg (2007)

7. Moser, S., Martens, A., Görlach, K., Amme, W., Godlinski, A.: Advanced Verification of Distributed WS-BPEL Business Processes Incorporating CSSA-based Data Flow Analysis. In: IEEE International Conference on Services Computing (SCC 2007), pp. 98–105 (2007)
8. Reisig, W., Bretschneider, J., Fahland, D., Lohmann, N., Massuthe, P., Stahl, C.: Services as a Paradigm of Computation. In: Jones, C.B., Liu, Z., Woodcock, J. (eds.) Formal Methods and Hybrid Real-Time Systems. LNCS, vol. 4700, pp. 521–538. Springer, Heidelberg (2007)
9. Lohmann, N., Massuthe, P., Stahl, C., Weinberg, D.: Analyzing Interacting BPEL Processes. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102, pp. 17–32. Springer, Heidelberg (2006)
10. Hinz, S., Schmidt, K., Stahl, C.: Transforming BPEL to Petri Nets. In: van der Aalst, W.M.P., Benatallah, B., Casati, F., Curbera, F. (eds.) BPM 2005. LNCS, vol. 3649, pp. 220–235. Springer, Heidelberg (2005)

Evolving and Implanting Web-Based E-Government-Systems in Universities

Dirk Reiss, Bernhard Rumpe, Marvin Schulze-Quester, and Mark Stein

Institute for Software Systems Engineering
Braunschweig University of Technology
Mühlenpfordtstr. 23
38106 Braunschweig, Germany
`http://www.sse-tubs.de`

Abstract. The Bologna Process [1] has triggered a major restructuring of the current university diploma into a bachelor/master system. As one effect, the administration effort for the new system has increased dramatically. As a second effect, students need and demand a much better information policy, given the new possibilities of the internet. Both to increase efficiency of the university's administration and to provide students as well as lecturers with modern e-government services, it is inevitable to evolve the current IT-infrastructure of a university into a modern web-based landscape of systems that support business processes on campus.

In this paper, we describe the approach taken at the Braunschweig University of Technology to evolve the existing landscape of legacy systems by adding bridges between previously unrelated parts, adding and customizing unused modules of existing software to bring information and services online and to develop new software, where old modules could not serve the necessary purposes. Most of all, both implementation of the results in university's business processes and the resulting quick feedback and wishes for feature enhancement are seen as part of the software development processes and discussed in this paper.

Keywords: E-Government, business process optimization, Hierarchical XP, web-based information systems, agile software development.

1 Current E-Government Situation

Currently German universities undergo major changes in their internal organization. This has several reasons:

– Since the European Countries signed the Bologna Declaration [1], major changes in curricula are necessary. In particular the number of exams has increased by a factor four to six and thus students and lecturers have a higher demand for efficient organization of these exams.
– The introduction of a two-cycle bachelor/master study system goes along with an increased demand for course certificates. This enforces universities to provide online information for students.

- The modularization encompassed by the bachelor/master system partly affects the contents of lectures, but to a much higher degree it affects the organization of curricula. Lecturers have to provide a lot more and detailed information, e.g. to allow students (and other universities) to understand contents of courses in recent and forthcoming semesters.
- While this restructuring goes in parallel with reorganization and optimization of the administration, it is inevitable to optimize administrative processes in cooperation with the development and evolution of the software systems needed.
- Development of new and, in particular, interdisciplinary curricula help modern universities to emphasize particular strengths in teaching and research and thus attract more students. This, however, enforces a university to develop integrated information systems for defining and maintaining module handbooks and examination regulations. This is necessary to prevent redundancies, inconsistencies and in particular to help lecturers to ensure correct conduction of exams.
- And finally, German universities currently undergo a heavy transformation with regard to evaluation. The evaluation burden has heavily increased, both for internal evaluation of teaching and research load and quality as well as for external evaluation and surveys. While each of these evaluations (or at least most) does make sense, in sum there are far too many and researchers/lectures are in total too busy to fill in forms instead of actually being productive. To ease this burden, integrated information systems for statistical and evaluation information will be necessary.

As described in [2], the shortage of resources caused by the implementation of two-cycle study systems is not just a theoretical problem. In future years the quality of education will decrease and the capacity needed for its adminstration will increase dramatically. This effect is to be expected by the year 2010. In order to cope with these problems, universities have started to adopt e-Government principles (see [3]) to improve internal administration efficiency and delivery of public services to all university members.

Evolution and extension of business software systems in most cases go hand in hand with change in business processes. Therefore administration departments, examination offices, lecturers and students most likely have to adapt their procedures to become more efficient. Some old procedures might completely vanish, others need to be extended or evolved. Hence, a close integration of the software development process, its roll-out in form of technical installation and training on the software is inevitable.

On the technical side, the question how to develop a new system, to evolve a given legacy system or to add new modules to a legacy system needs to be tackled. This software landscape needs to be understood and dealt with in cooperation with their maintainers.

Currently existing software systems supporting integrated information and business process flows are often far outdated, inefficient and incompatible for data exchange. Furthermore, modern e-government information delivery models like university-to-student, university-to-lecturer or university-to-staff are often not supported. Worse, universities first have to understand and explicitly define their business processes to be able to optimize them.

The Braunschweig University of Technology decided in 2003 to boldly modernize their software landscape. For that purpose a number of projects were launched, one of them dealing with administration processes serving teaching and teaching organization. Goal of this project was and is to serve the goals listed above, starting with a university-wide module handbook that also serves as a single source for curricula definitions, class schedules, room assignments, university calendar, etc.

In the rest of this paper, we record on the development and deployment of this electronic module handbook. In Section 2 we describe the reorganization of the administration primarily implemented through the introduction of web-based information systems (online portal). Section 3 focuses on the general development approach to implement and introduce these new information systems. Section 4 presents the introduction of the module handbook (MHB) as an example, covering its data model, implemented business functions, and technical interfaces as well as summarizes the lessons learned from the installment procedure. Finally, Section 5 discusses planned future actions and concludes the process of reorganization carried out so far.

2 Situation Analysis and Process/Software Reorganization

Given that the demands discussed above and in particular the restructuring of the curricula induced by the Bologna Declaration will evoke serious capacity problems, the first step is to understand what actually had to be done. This step included the following activities:

- Understand the current business processes within the university, identify problems and inefficiencies and derive possibilities for improvement.
- Understand the currently used solution (which in Braunschweig is based on the HIS system (see [4])) as well as its capabilities and deficiencies and possible alternatives.
- Go public with the project and prepare the university members to accept and welcome these software based process enhancements.

The outcome of the process analysis mentioned above showed clearly that the application formerly in use at Braunschweig Technical University (HIS) was a pure back-office system, capable of administering exams and lectures of single-cycle study systems only. This means that all administrative processes involved could only be done by the faculty administration staff and neither students nor lecturers could get up-to-date information about study progress or lecture assignment instantaneously. An examination of the current solution revealed that a switch from the currently used system to an alternative is currently out of scope. Too many users have learned to adapt to HIS's strengths and weaknesses.

So the decision was to evolve the HIS legacy system, both within our university and together with HIS, while it is also clear that neither technology of the system nor development capabilities of the creators are optimal.

2.1 Selling the Project

To help our clients to understand the purpose of the development project, we used the the metaphor of a plane (as shown in Fig. 1 to define roles and according activities.

Role for airplanes	for software
Airbus	HIS
Interior fittings	our project
maintenance troop	IT administration
steward	application administration
pilot	lecturer
passengers	students

Fig. 1. Analogy between airplane construction and software development

The HIS system in its current form is like an airplane. While Airbus (HIS) is the principal developer, we adapt the existing airplane to fit it to the university's specific needs. Ideally, we only have to define the interior fittings like seating and entertainment system. However, the current HIS system is capable to assist Diploma "flights", but not very well suited for Bachelor/Master flights. Therefore we have to enhance this airplane, like extending the wings (which are quite substantial changes).

2.2 Starting the Development

As indicated above, the most suitable solution was to integrate all relevant information sources for study, lectures and exams into an integrated landscape of web-based information systems with a single portal-based front-end.

A web portal is available from everywhere at anytime for anyone. It serves as a central repository for information shared among university institutions and ideally assists all business processes that are necessary to guide a student from initial enrolment to the final completion of the degree. It regulates the import and export of data from and to subsequent processing systems and enforces the unification of data maintaining processes.

With the introduction of online portal systems for administrative information processing, the following major effects in context of e-government at a university have been identified:

- The use of web systems leads to a unification of information among all organizational units and study programs, while the responsibility for entering and

updating data remains decentralized. For example, all lectures and modules are fed into an electronic study guide and module catalog system (see Section 4) by the individual institutes and faculties, providing the same description elements for all of them and thus allowing their cross-faculty reuse for the definition of different study programs. This approach combines the advantages of decentralized data entry with a common data model and format for all study programs.
- A central web-interface can be used to generate new information objects, like semester catalogs for single institutes, individual timetables or electronic certifications for students and so on. Furthermore it can provide online access to administrative services, e.g. online seminar and lecture enrollment, online exam registration, online submission of grades, etc. Thus administration departments are greatly disencumbered, as many time-consuming processes are relocated to the demanding customers.
- To ensure an efficient, correct and timely administration, business processes need to be explicitly defined and assisted by the web information system. This includes deadlines and reminders for late activities. Synchronization of efforts is necessary e.g. to have an accurate module catalog available in printed as well as online form. Easy to understand and use definitions of these procedures are therefore developed together with the system.

A number of further challenges arise, as legacy systems need to be integrated:

- A central identity management is needed to simplify the use of various subsystems.
- An integrated graphical user interface (GUI) must be capable of presenting all necessary and available information in an intuitive and convenient way, depending on the current user request. Individualization is necessary and must be provided through the portal.
- Performance is necessary to please users as well as effective, pleasent use of provided functionality. Especially the latter is difficult for users with heterogeneous roles and wishes.
- Various legacy systems need to be integrated, either loosely coupled through batch data update or firmly through shared use of the date base, or even through mutual use of service access points. However, this implies that such service access points need to exist, which in legacy systems often is not the case.

3 Development Approach: Hierarchical XP

To face all upcoming challenge, an agile development approach (see: [5], [6], [7]) was inevitable. In particular, the many different user roles, potentially and frequently upcoming new requirements and adaptations need to be handled in an agile way. Therefore the Hierarchical XP (eXtreme Programming) approach, as suggested by [8], [9], [10], seemed to be most appropriate.

Within Hierarchical XP, large projects are decomposed into subprojects of appropriate size with focus on a compact subset of the desired functionality.

In each of these subprojects, an agile development process is used and adapted according to their specific needs. According to [11], agile software development methods are *lightweight*, *iterative* and *flexible*, additionally some are *evolutionary* and *adaptive.* A common charateristic shared among all of them is the availability of key customers during the whole development life cycle. Hierarchical XP however differs in some points to this common understanding of agile projects, as it aims at coordinating several agile subprojects through a moderator function (see [8]). It is feasible when a landscape of individual but related product systems need to be handled, instead of subprojects to be aligned into a synchronized product release. Its adaptation taken at Braunschweig University of Technology is described in Section 3.5 in more detail. Further principles of agile software development, which were partly taken from the eXtreme Programming approach and applied to this project, include pair programming, iterative and evolutionary development in short cycles, code refactoring and requirements refinement between subsequent releases and intensive testing.

With the desired result being a landscape of rather individual subsystems that fit into a larger set of legacy systems, it is useful to embed development into a superordinate project control process. For a stringent development progress, it should be given into the hands of a steering group, whose task is to ensure that the comprehensive requirements are met and to keep track of the overall progress. This steering group is best composed partly of project members and partly of stakeholders that represent various viewpoints throughout the university.

This superordinate process - initiated and controlled by the steering group - consists of five activities, of which the fifth comprises the final breakdown of the project structure into agile subprojects:

1. Understand the administrative business process as they are carried out by faculties, institutes and administration departments at the moment and define how they should look like after the new system has been installed.
2. Map the application landscape as it is currently in use. As mentioned earlier, we encountered a rank growth of different and mostly incompatible systems for processing basically the same information in different data formats.
3. Develop roles according to the processes as they should be carried out in the future supported by the new web application systems to be built.
4. Choose the foremost improvements that will have the greatest effect on bringing administration processes of study and teaching towards modern e-government principles (i.e. improved (online) exchange of information and services between university members).
5. Implement the improvements identified and adequately prioritized in short iterating development cycles.

3.1 Understanding Current Business Processes

To understand which processes have to be supported and where improvements can be achieved by the installation of web-based systems, current workflows and business processes need to be analyzed. For this purpose various administration departments have been interviewed, surveys have been taken and finally documented in a precise manner using UML activity diagrams.

The adminstration processes taken into account were selected due to the impact the Bologna Process will have on them, i.e. the ones where the administration effort will increase or change dramatically. It became obvious that nearly the whole student lifecycle is affected by the changes implied by the introduction of the Bachelor / Master system. Furthermore, some processes varied considerably between different programs of study. Not only the structure of their courses differed, but also the manner of equivalent processes were performed. Hence, it was inevitable to unify those processes in order to provide decentralized web-based applications that alleviate the effort needed to fulfill the adminstration tasks. The main fields identified - and therefore established as subprojects - are students registration and enrollment, lecture planning, lecture-to-student assignment, exam registration, grade submission and the self-service generation of study certifications.

3.2 Mapping Current Application Landscapes

After documenting the processes, we needed to inspect how to support the existing processes, so that the increased effort could become manageable. Therefore, the software landscape present at that time was analyzed. Although identifying a variety of heterogenous software systems with overlapping functionalities developed and installed at different institutes, it crystalized that at the Braunschweig University of Technology, the software system by the HIS GmbH (see [4]) is primarily in use for study administration tasks. Their software packages consist of a backend part (the so-called GX modules) and a web-based part (QIS or LSF). Besides the evaluation of the already existing software system, alternative products were evalutated. As an outcome, it became clear that none of the systems would do a significantly better job at supporting the needs burdened by the mentioned higher education reform. Hence, the already installed system was kept and modified according to the universities specific needs. Speaking of our example mentioned previously: we already bought an airplane, only the interiour has not been modified to suit our customers' needs yet.

3.3 Roles Development

The actors so far participating in the processes have - as they were mostly not supported by modern information technology systems - not been mapped to any role in the forthcoming decentralized web-based system. Therefore, new roles that match the existing fields of administration and the task within each of them had to be indentified and created afterwards. Examples for the activities in the area of course planning and the according roles are lectures who keep their lecturers and modules up-to-date, persons who dispose and maintain rooms and timeslots within a certain course of study as well as deans being responsible for planning and assembling whole courses of study. The decentralized web-application needed to be configured to suite these actors' needs best with regards to usability.

3.4 Choosing Most Important and Helpful Improvements

With a given overview of the processes involved in the students' lifecycle, priorities had to be defined. The expected improvements on the administration of students and study courses in general has been chosen as the crucial factor for the order of precedence. As the availability of highly up-to-date module and lecture information (and accompanied by the existence of high-quality courses of study) is an enabling factor for the installation of the two-cycle study system, this problem area was given a high priority. Besides online registration and enrollment the first focus was put on exam registration and lecture assignment. Enabling students to conduct these administrative transactions online improves their situation significantly while unburdening the faculty administration departments. Furthermore, lecturers should benefit from the new possibilities by allowing them to submit exam grades online and to receive statistical information about their courses and exams taken. Last but not least the need for a central module repository to keep all study guides in sync became more urgent. This also includes the generation of study guides in one consistent format for the whole university. With this approach taken, the planning and management of lectures - done decentralized by each institute on its own - leads to up-to-date database of modules that can be supplied to students and lecturers as well.

In Fig. 2, the subprojects chosen for foremost implementation are referred to as 'ENR' for online registration and enrollment, 'MHB' for the electronic

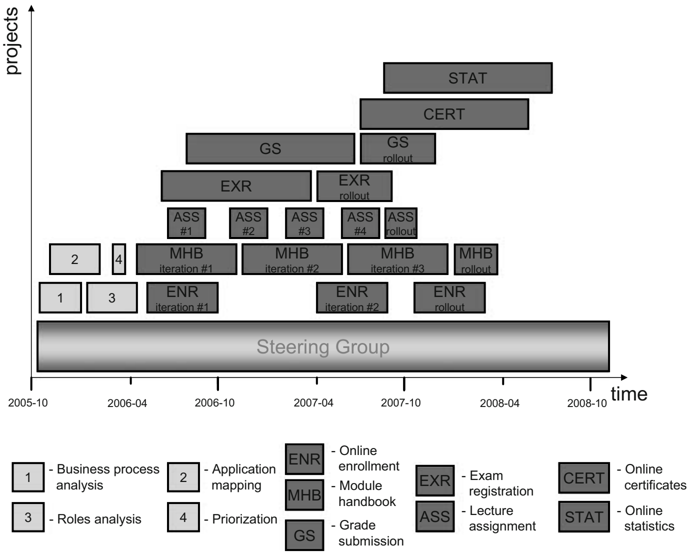


Fig. 2. Project development approach

module handbook and lecture catalog, ‘ASS’ for lecture assignment, ‘EXR’ for exam registration, ‘GS’ for grade submission, ‘CERT’ for online certificates and ‘STAT’ for online statistics.

3.5 Implementing Subprojects in Iterating Cycles

As the university administration departments can be characterized as ‘traditional’ in terms of process organisation, and many targets for optimization have been identified, the overall project goal has been divided into several subprojects as noted above. The general development approach taken within each subproject follows a typical agile path (see: [6], [7]) with several modifications. These were necessary as in case of the most subprojects the customer could not be identified as a single person or department, but as the university (including lecturers, students and administration departments) as a whole, which contrasts the XP approach according to [5]. They claim to have one customer in charge to define all requirements, which implies that this one customer has to know exactly what his or her requirements are. At a complex and heterogenous organization like a university, the different departments, institutes and faculties all have different and sometimes contradicting expectations and want to be supported as good as possible.

With the whole project divided into subprojects, we distinct two different kinds: (1) subprojects that affect the whole university (e.g., students’ enrollment and study guides) and (2) subprojects whose implementation needs adjustment from one program of study to another (e.g., examinations regulation and adjacent exam registrations). Subprojects of the first case needed some more effort to unify the processes involved at first hand. The adapted system is then rolled out in several iterations to certain key users (or at least experts in the field). The latter induced that each course of study had to be handled seperately. Therefore, each subproject is prototypically rolled out to a certain number of pilot users (here: programs of study). This was at least inevitable for processes that were not alike in the past. Hence, one major obstacle was the necessary unification of administration processes. The changes ranged from minor modifications in those processes actually carried out in few courses of study, to a completely different way of dealing with the affected areas of adminstration. During this pilot phase, suggestions for improvement regarding the handling of the web-based application were registered and implemented for better usability. In the next iteration cycle, more courses of study were added to the group of web-assisted ones. This iterative approach was used for the rollout of subprojects such as the online exam and course registration, whereas subprojects like online enrollment and decentralized management of modules and lectures were installed university wide. After several iterations, the subprojects have been stabilized in a way that administration and IT infrastructure departments were ready to run the applications in production and extend them to more courses of study. Hence, this departments’ staff was instructed. Speaking of our metaphor: the plane has been build and modified, now the staff needed to be taught how to operate it. Furthermore, passengers not used to use a vehicle of that kind (or have fear of flying) need to be convinced of the enormous advantages using it than the old way of travelling.

4 Example: Online Module Handbook and Lecture Catalog

4.1 Previous Implementation

One central goal of the new two-cycle study system is the re-organization of study programs into modules. As a consequence, they provide a much more interdisciplinary orientation for students. Therefore the variety of courses to choose from grows immensely. As a disadvantage, the effort needed to keep all study guides in sync (as one change in a module affects more programs of study than before) grows as well. Another problem was the variety of formats in which module handbooks were kept. Some programs of study used Microsoft Word, some Excel and some even LaTeX documents, although the use of the same application did not mean documents were interchangeable. Usual practice demanded every lecturer to submit changes to each program of study that contains the module or lecture respectively, to discuss these changes with the persons responsible and come to a consent about the changes. The submission had to be in a different file format for each program of study and therefore was a time-consuming process.

Until recently, all communication regarding import (asking for a module to be included in one program of study) and export (offering a module to be included in another one) had to be agreed on by the offerer of one module (usually the lecturer of the module) and the dean in whose program of study that module was to be included. This resulted in enormous effort taken on both sides.

4.2 New Approach: Central Online Module Handbook System (MHB)

These inefficiencies described previously - added to the fact that the applications available at that time did not support a decentralized web-based approach to manage lecture and module data and would not cover the requirements identified - resulted in the decision to develop a web-based system from scratch. For implementation, the Java programming language with Apache struts framework (see [12]) and the Velocity template engine (see [13]) was chosen. As a database system, the Postgres database management system is in charge of storing all data. User logins are verified by system calls to the Andrew File System (see [14]) which easily allowed access to the central user database at Braunschweig University of Technology.

This approach enables each lecturer (or his or her delegates respectively) to manage the information describing lectures and modules using a browser-based application. It results in a central repository of up-to-date information about lectures, modules and program of study contents accessible by every interested party. Changes made using the online application will immediately take effect in all adjacent entities (like lecture and module descriptions and course catalogs), not depending on the publication deadline of a paper-based module and course catalog. Furthermore, the many various formats previously necessary have been unified into one central system.

4.3 Data Structure of the MHB System

As shown in Fig. 3, each study program consists of either modules or lectures, depending on the kind of program (single cycle study programs are made up of lectures, two-cycle study programs consist of modules, which themselves consist of lectures). Modules are grouped by categories within two-cycle study programs and use so-called ‘topics’ as static containers for lectures possibly changing between terms. Persons in this data model are associated to lectures (as lecturers), modules (as module responsible and lecturer), study programs (as dean) and institutions (as head or member of an institution). Modules and lectures belong to a certain institution which is primarily responsible for them. Each lecture is held at one or more specific date.

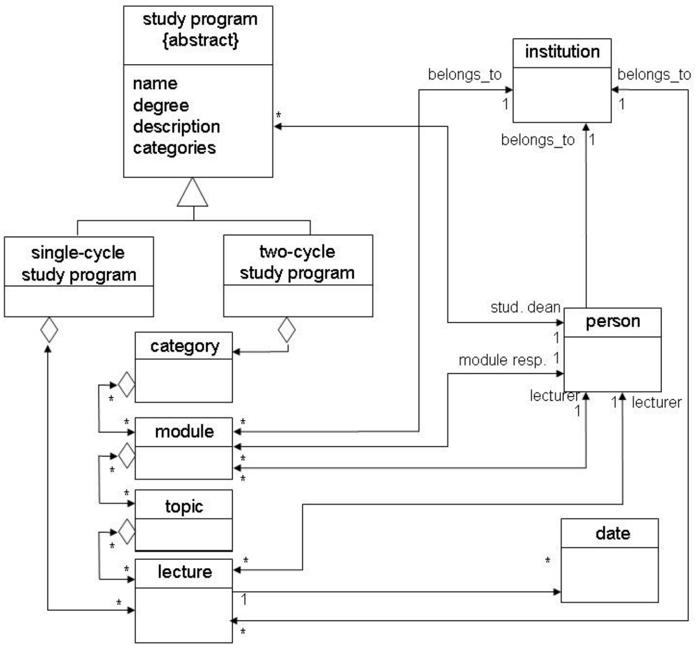


Fig. 3. MHB class diagram

Access rights within the MHB application are generally regulated by associations of users to their institutions. This means, an editor for a certain institution (meant to be the lecturer himself, but according to experience delegated to scientific or secretary staff) only has rights to edit the staff objects, lecture objects and module objects associated with the corresponding institution. Role assignment is backed up by formal grant by first-place responsible persons (like head of an institution for instution rights and dean for course of study rights). For each lecture there is one timetable person in charge, finding times and places to hold the lecture and to fit into the timetables of affected courses of study. Only these persons are allowed to verify and change dates, places and times of lectures after a certain date. Since module catalogs affect legally critical areas like examination regulations, the access to these is limited. Hence, only study program responsible persons (granted by the dean of a course of study) are allowed to alter its contents. As mentioned in section 4.1, assignments of modules to study programs have to be acknowledged by both parties involved. Therefore module inclusion is accepted under reserve. A module not acknowledged by both the module lecturer and the dean responsible for that study program is not included in study guide documents until the other agreed upon its inclusion.

As data storage in an online course management system is a first step, there have to be various ways to utilize the data collected. Therefore, different types of export functionality have been implemented. These range from files of comma seperated values (for use in spreadsheet applications and as data source for different kinds of serial letter inclusion) over predefined annotated lecture documents to complete module catalogs in PDF format.

4.4 Lessons Learned

Round about half a year after the MHB system was introduced at Braunschweig University of Technology, we conducted a survey among its users to assess its acceptance and to explore to which extent the workflow now supported by a decentralized online system is accepted has become second nature to the different roles of users. A total of 137 users answered the poll, which was hold online over a four week's period. The main tenor of the comments and suggestions given was that a general improvement of the quality and availability of information about lectures, modules and study programs could be seen, yet it was consumed by an increased effort for collecting and feeding the data into the system. It is expected that this effort reduces over time, when users become more familiar with the system. Yet we used the following quarter to release a great many of bugfixes and improvements to the system usability according to the users' comments and suggestions in short iterating cycles.

As expected, the introduction of a web based system had an integrating effect on all administration processes that are carried out at the whole university, like building module and lecture catalogs for both semester timetables and study programs. Faculties and institutes now enter this information directly and decentralized, while the data structure and format is laid down by the MHB system. Thus overall processes became more transparent to the institutions involved, as they are now carried out the same way throughout the university. Most of these processes have been documented in form of checklists, which have proven useful to communicate with users inexperienced to more formal notations like UML activity diagrams. At the same time module- and course-of-study information became interchangable between institutions.

The decision towards a system with decentralized data entry was made with respect to the heterogenous organization structure within Braunschweig University of Technology. For the same reason, it was difficult to obtain unified functional and user interface (UI) requirements suitable for all end users and user groups, consisting of faculty, insitute and administration department members. Agile development processes in their original form include the identification of key users, which are to be available for early communication with the developers in all stages of the development process. In our case, we replaced the key user concept with three measures suitable to the Hierarchical XP approach (see Sect. 3) in combination with a widespread, heterogenous end user group:

– We set up an online request ticket system to channel all support requests, then transferred them manually into a central project bugtracking system. This way we were able to integrate feature requests and bug reports from users directly into our development process.
– When the first release of the system was to be launched, we organized three major information events, one for each main user group, in which the system, its capabilities and user interface was introduced in detail to them.
– During the first and second data entry period, we offered special service times, in which users could enter their data with assistance from experienced development team members.

By applying these measures, we were able to uphold the main principle of agile development approaches - early feedback from and intensive communications with customers - without needing to explicitly define pilote users in charge for all others. New or changed user requirements could thus be incorporated as quickly as new features or changes to business models supported could be communicated to the MHB customers spread throughout the university.

5 Conclusions

In this paper, we discussed some of the problems arising when evolving the software landscape to assist e-government processes in a university.

We described the overall development process and how it was established in the actual project. This process is a combination of hierarchical decomposition of the development project into loosely related subprojects, where each of them uses XP-like techniques. This approach, called Hierarchical XP [8], is most appropriate as it combines XP ideas like strong integration of users, anticipation of requirement changes and an iterative approach of development with a larger overall goal of development. We have sketched, which steps are necessary to not only successfully realize, but also introduce a larger online system to a widespread and very heterogenous group of users. As a result, the introduction of this system indeed had the desired effect of unifying administration processes in the field of module and lecture management (e.g. creation of module catalogs for different programs of study), thereby improving the quality and availability of up-to-date information. These processes became more transparent for students and teachers, while the administration's efficiency increased. As a second result, it follows that a hierarchical structure for the evolution of a software landscape in synchrony with the evolution of the business process is indeed helpful to implant new or enhanced e-government systems, when appropriate steps are taken to integrate users into the development process.

References

1. European Ministers of Education. The Bologna Declaration of 19 June 1999 (July 1999), http://www.bologna-berlin2003.de/pdf/bologna_declaration.pdf
2. Hochschulrektorenkonferenz. Statistische Daten zur Einführung von Bachelor- und Masterstudiengängen Wintersemester 2007/2008. Date of last check (Oktober 2007), http://www.hrk.de
3. von Lucke, J.: Heinrich Reinermann. Speyerer Definition von Electronic Government (July 2000)
4. Hochschul Informations System GmbH. HIS Homepage. Date of last check (July 2007), http://www.his.de
5. Beck, K., Andres, C.: Extreme Programming Explained: Embrace Change. Addison-Wesley Professional, Reading (2004)
6. Cockburn, A.: Agile software developent. Addison Wesley, Reading (2002)
7. Eckstein, J.: Agile Softwareentwicklung im Grossen. Dpunkt Verlag (2004)
8. Jacobi, C., Rumpe, B.: Hierarchical XP. In: Succi, G., Marchesi, M. (eds.) Extreme Programming Examined, Addison Wesley, Reading (2001)
9. Rumpe, B., Scholz, P.: A manager's view on large scale XP projects. In: Third International Conference on eXtreme Programming and Agile Processes in Software Engineering (2002)
10. Rumpe, B., Schröder, A.: Quantitative Survey on Extreme Programming Projects. In: Third International Conference on eXtreme Programming and Agile Processes in Software Engineering (May 2002)
11. Fraunhofer-Institut für Experimentelles Software Engineering. Virtuelles Software-Engineering-Kompetenzzentrum. Date of last check (2001-2007), http://www.software-kompetenz.de/
12. Apache Software Foundation. Apache Struts homepage. Date of last check (July 2007), http://struts.apache.org
13. Apache Software Foundation. Apache Velocity homepage. Date of last check (July 2007), http://velocity.apache.org
14. OpenAFS project. OpenAFS homepage. Date of last check (July 2007), http://openafs.org

Facets of Media Types

Klaus-Dieter Schewe[1] and Bernhard Thalheim[2]

[1] Massey University, Information Science Research Centre
Private Bag 11 222, Palmerston North, New Zealand
[2] Christian Albrechts University Kiel, Department of Computer Science
Olshausenstr. 40, D-24098 Kiel, Germany
kdschewe@acm.org, thalheim@is.informatik.uni-kiel.de

Abstract. The concept of media type is central to the codesign approach to Web Information Systems (WISs). By means of views on some database schema it permits a separation of global content from local content that is to support particular scenes of a WIS. By means of various extensions such as cohesion, hierarchies, style options and associated operations it enables interface abstraction and adaptivity to users, channels and end-devices. It can further be used for modelling collaboration, session support and contextual information, and by means of an associated dynamic logic provides the basis for formal reasoning about a WIS design. In this paper we give a brief survey of the potential of media types as a very powerful abstraction mechanism for WISs.

1 Introduction

A Web Information System (WIS) is a data-intensive system that is accessible via the world-wide web (WWW). As such the design of such systems has attracted a lot of attention by researchers in the area of conceptual modelling, and various modelling languages and design frameworks have been developed so far [2,3,4,6,7,15,16], all having their individual merits but none giving a complete answer, how all the challenges in WIS modelling (see [13]) are addressed.

In this paper we give a brief survey of the potential of the concept of media type, the central notion in the codesign approach to WISs [15]. The naming of the concept relates to the common understanding of "media" as means for mass communication suggesting that the WWW has become such a means. "Types" then refer to the classification and formal abstraction from content and functionality of such communication means.

As such a media type is first an interface abstraction related to and extending the notion of dialogue type introduced in [14]. The rough idea is that the global data content of a WIS is captured by means of an instance of a conceptual database schema, while content needed at the interface is modelled by means of views. As emphasized in [13] this includes the modelling of the navigation structure, which in various other approaches is treated as a separate add-on feature. Furthermore, same as with dialogue types operations expressing the functionality of the WIS, are coupled with the views. In this way, media types capture form-based interfaces [5,9] in an abstract way.

In doing so, media types provide a perfect mechanism to refine the scenes and actions of the storyboard that result from analysing and modelling the usage of the WIS [15], thus linking the conceptual model to the identified systems requirements. By means of associated style options [11] they also provide the mechanism for system implementation including a structural part (XML clusters), a behavioural part (scripts) and a layout part associated with page grids.

However, media types are more than just interface abstractions. By means of hierarchical versions they subsume OLAP-like features allowing users to switch between representations at various levels of granularity. By means of cohesion they allow an automatic split of information adhering to the restrictions imposed by channels, end-devices and user preferences. In this sense media types enable derivable adaptivity. Furthermore, as outlined in [12] consistency and personalisation with respect to content and functionality can be achieved by logical reasoning about media types.

Most WISs are centred around the actions of individual users, whereas the need may arise to support collaboration. This can also be captured by media types using associated exchange frames [1] that support cooperation, communication and coordination within user groups. This is related to the fact that media objects, i.e. instances of media types, can be used to capture information about sessions, which are linked to complex scenes in the storyboard.

Finally, media types permit dealing with the often mentioned phenomenon of “loss in cyber space”, which actually is a loss of navigation context. Adapting an idea from Contextual Information Systems [18] we obtain query macros that allow a user to traverse back a path in the storyboard and to explore alternative access paths, respectively [8].

2 Interface Abstraction

In the following let $\mathcal{S}$ denote some conceptual database schema. The introduction of media types in [15] was based on the higher-order Entity-Relationship model (HERM) [17], but we already emphasised that the choice of data model is of minor importance. A model (such as HERM) with richer structuring mechanisms will simplify the definition of views, while a poorer model – even if it has the same expressive power – will require more sophistication in the queries that define views.

Furthermore we assume a type system comprising base types and various type constructors, e.g.

$$t = b \mid \mathbb{1} \mid (a_1 : t_1, \dots, a_n : t_n) \mid \{t\} \mid [t] \mid \langle t \rangle \mid (a_1 : t_1) \oplus \cdots \oplus (a_n : t_n)$$

would define a type system with not further specified base types b, a trivial type $\mathbb{1}$, and constructors for records, finite sets, lists and multisets, and disjoint unions. Usually, we assume that among the base types there is a type *URI* representing abstract surrogates for URIs.

The semantics of such types is primarily defined by their domains, i.e. with each type t we associate a set of values $dom(t)$. For a base type the domain is assumed to be countably infinite. For the trivial type the domain is a singleton set

$\{\mathbb{1}\}$. For constructed types we use the usual recursive definitions. Furthermore, we associate basic (fixed) operations with the type system, e.g. projection on record types, union for sets, etc., but we dispense with listing them here.

The type system forms the basis for the definition of *content type expressions*. We simply extend the type system permitting $r : M$ to appear in lieu of a base type, with r and M being names for a reference and a media type, respectively, both taken from some unspecified countable pool of names. Replacing $r : M$ by *URI* results in a proper type, called *content representation type*.

The core of a media type M is defined by an *interaction type*, which comprises a content type expression $ct(M)$ (with corresponding content representation type t_M, a query q_M defined on $\mathcal{S}$ with output type $(id : URI, value : t_M)$, and a finite set Op_M of operations – more formal details can be found in [15]. An *interaction schema* $\mathfrak{I}$ is a finite set of interaction types that is closed under references, i.e. whenever $r : M$ appears in a content type expression $ct(M')$ with $M' \in \mathfrak{I}$, then $M \in \mathfrak{I}$ must also hold.

The semantics of interaction types and schemata is defined by means of the defining queries. Take an instance db of the database schema $\mathcal{S}$ and execute the queries q_M for all $M \in \mathfrak{I}$, which will result in sets $db(M)$ with values that are actually pairs (u, v) with a URL u and a value v of type t_M. We request that whenever a URL u' appear in such a value corresponding to the reference $r : M'$ in the content type expression $ct(M)$, then $db(M')$ must contain a value (u', v'). Furthermore, URLs must be globally unique. In this way we obtain an abstraction from the structure at the interface. The value $(u, v) \in db(M)$ represents the content v at the abstract URL u including references to other units of content, i.e. including the navigation structure. In its simplest form this is abstraction from web pages, but as we shall see later it can be more than that. As emphasised in [15] and discussed with respect to different query languages, the abstract definition of interaction types requires the use of query languages that are able to create abstract identifiers, i.e. the URLs, and in many cases will need a fixed-point construction.

Operations associated with an interaction type permit updating the underlying database and to open and close interaction objects. For this, we require a signature describing input- and output-parameters as well as a selection type, and an operation body that can be defined by means of usual programming language constructs. The selection type specifies, which part of the content value must be selected as a precondition for the operation being excuted. In [9] we illustrated that this form of interaction types, which is more or less the same as the *dialogue types* in [14] captures form processing in a natural way and on a reasonably high level of abstraction, but it is not limited to this.

For this, consider storyboarding, the precursor of conceptual modelling in the codesign approach to WISs. A storyboard consists of three parts: (1) the story space describing scenes, the transition between scenes, actions associated with these transitions, and the plot, i.e. the detailed action scheme, (2) the actors describing roles, right and obligations associated with roles, user profiles, and preferences associated with these profiles, and (3) the tasks describing meaningful chunks of WIS usage, the actors involved, their goals, the scenes associated with tasks, and the actions required for task execution. Scenes are hierarchically organised in the sense that each non-elemetary scene gives rise to a scenario, which is composed in the same way as the whole story space, in particular giving rise to sub-scenes.

Thebasicinterfaceabstractionmechanismdescribedaboveassociatesamediatype with an elementary scene leading to page abstraction as outlined above. However, there is no formal reason not to associate media types also with non-elementary scenes, thus leading to inferface abstraction on a higher-level. We will discuss the benefits of this high-level interface abstraction in the following sections.

Before doing this, let us first take a look at some other facet of media types. Media types are extended interaction types, and one of the extensions refers to layout and playout style options. For the layout the primary question is how the abstract content of a media object, i.e. a value of type t_M, is to be represented. Exploiting that types give rise to a lattice, this first leads to a decomposition by means of a Sperner set of types $t_M^1, \dots, t_M^k$ with $\bigsqcup_{i=1}^{k} t_M^i = t_M$. Using canonical projections from $dom(t_M)$ to each $dom(t_M^i)$ we obtain values $v_1, \dots, v_k$ that jointly represent the content of some WIS unit. For each of the types t_M^i a style option specifies how records, sets, lists and multisets are to be represented using maybe tables, itemized lists, forms or other layout elements, which colours are to be used, which font families and sizes are to be used, which adornments such as background images should be added, etc. In the same way the playout of operations is handled, i.e. how are operation names to appear, how is input from the user collected using forms or dialogue boxes, how is correctness of data entry ensured, etc.

With respect to the actual system implementation, the content representation type of a media type gives rise to a cluster of XML templates, while the defining query has to be mapped to a query resulting in a cluster of XML documents. The style options then give rise to translators from XML to HTML, while the operations give rise to scripts implemented in the preferred scripting language. In this way, most of the actual implementation of the WIS results from page generation with the actual content defined by the current database instance.

Furthermore, media types permit the easy use of the container metaphor. By using this metaphor the content representation type t_M is only considered as the type describing the content at the server-side, while the demand at the client side may be expressed by a different type t. In this case, the meet $t_M \sqcap t$ defines the part of the content that is to be shipped in a “container” to this client. Any missing part has to be obtained from other media types. Style options on the client side then define how the content of various containers is to be assembled.

3 Adaptivity

Two other extensions of interaction types (turning them into media types) address the granualarity of the represented data, and the adaptivity to users, channels and end-devices. The former one adopts ideas from on-line analytical processing (OLAP), while the latter one addresses splitting and condensation of information. Both extensions exploit the type lattice.

In order to address granularity we take the lattice defined by the supertypes t' of t_M. To stay consistent with the use of join and meet above let the order be such that $t' \leq t_M$ holds. In particular, t_M is the smallest type in that lattice. Now take a subset $\mathcal{H}(M)$ of types in the lattice that defines a tree with root t_M – this will be called a *set of hierarchical versions*. In earlier versions of our work we actually requested a chain instead of a tree. Obviously, be means of canonical projections from $dom(t_M)$ to $dom(t)$ for each $t \in \mathcal{H}(M)$, we obtain for each value v of type t_M and each $t \in \mathcal{H}(M)$ a version v_t of type t. Furthermore, due to the fact that we use a tree, we also get canonical up and down navigation operations between these versions. As t_M itself represents very condensed information, the hierarchical versions v_t represent lighter versions. Which version is the preferred one to start with depends on the preferences of the user. We should remark that there is no need to define specific style options for the hierarchical versions, as these carry over by means of the meet operation in the type lattice. In the same way, other OLAP-like operations, e.g. slicing and dicing are already defined by the content type expression, and there is no need to specify them.

With respect to adaptivity, we start from restrictions on the amount of data that is to be presented to a user. A value v of type t_M corresponds to the full amount of data represented by the media type M, but this may not be needed or requested by a user, or considered to be not suitable due to network or end-device restrictions, e.g. restricted band-width, limited presentation capacity in videotext or simply user preference. The extension to interaction types dealing with these restrictions is *cohesion*, which permits a controlled splitting of a value v of type t_M into a sequence of values $v_1, \dots, v_\ell$ of types $t_1, \dots, t_\ell$ such that $\bigsqcup_{i=1}^{\ell} t_i = t_M$, v_i is the canonical projection of v, i.e. the decomposition is lossless, each v_i contains a reference to v_{i+1}, and each v_i is a higher importance than v_{i+1}. As discussed in [15] this can be achieved in basically two different ways, either by means of a cohesion pre-order, which extends the order defined by the type lattice to a total pre-order, or by means of proximity values for each pair of types in a specified maximal antichain in the type lattice.

The main difference between the two approaches to cohesion is that proximity values use an a priori decomposition of t_M, i.e. an anti-chain of types $t_1, \dots, t_k$ such that $\bigsqcup_{i=1}^{k} t_i = t_M$. Then the types in the cohesion sequence are either these a priori given types or result from applying the join operator to some of them. The guideline for defining the types in the cohesion anti-chain is that the joint type must be maximal within the limitations defined by the end-device, channel or user preferences, and the sum of proximity values must be maximal. In this it is guaranteed that those t_i with the highest cohesion will stay together.

The alternative of using a cohesion pre-order does not assume an a priori decomposition of t_M. Then the guideline for selecting t_1 and thus define the value

v_1 is to be maximal within the limitations defined by the end-device, channel or user preference and also maximal with respect to the cohesion pre-order. The process is then repeated recursively with a maximal type t' complementing t_1, i.e. $t' \sqcup t_1 = t_M$.

Formal details of hierarchies and cohesion were presented in [15]. It is not too difficult to see that cohesion, hierarchies and style options are orthogonal extensions that can be mutually combined, i.e. for each hierarchical version the cohesion decomposition can be applied, and for each value in a cohesion sequence (for whatever hierarchical version) the style options can be applied.

4 Session and Context Support

As already sketched above media types enable interface abstraction non only on the level of elementary scenes, but can also be used for non-elementary scenes in the storyboard. In doing so media types can be used to support the concept of a session. Informally, a session comprises all scenes visited by a user from entering the systems until leaving it. Exploiting the hierarchies of scenes, a session can be represented itself by a non-elementary scene. That is, entering a session refers to the creation of the corresponding media object, which is linked to the elementary media objects used for defining subscenes. Technically, we model this by links from the media types associated with the elementrary scenes to the overarching session scene. Such links can be modelled as being hidden, i.e. they actually do not appear as part of the navigation structure. The session object may represent data that is carried around through the session. A typical example is the shopping cart in e-commerce systems. It is deleted by means of garbage collection, i.e. when no more media object exists that links to it.

Another use of media types for complex scenes is the support of navigation context, for which the basic idea from contextual information bases has been adopted [18]. According to this a context is a set of object identifiers each having several names, and each of these names may be coupled with a reference to another context. There may be names for objects that are not referencing to other contexts. More formally, a *context* C is a finite set of triples (o, n, r), where o is an object identifier, i.e. a value of some base type ID, n is a name, i.e. a value of type $STRING$, and r is either a reference $\rightarrow C'$ to a context C' or *nil*, the latter one indicating that there is no such reference. We write $C = \{n_1 : o_1 \rightarrow C_1, \dots, n_\ell : o_\ell \rightarrow C_\ell\}$. If there is no reference for the i'th name, i.e. we have (o_i, n_i, nil) we simply omit $\rightarrow C_i$ and write $n_i : o_i$.

The idea of working with contextual information bases is that a user queries them and thus retrieves objects. In order to describe these objects in more details s/he accesses the context(s) of the object, which will lead to other objects by following the references. In addition, a particular information encoded by the name is associated with each of these references.

Bringing together media types and contextual information bases, the obvious questions are: What are the object identifiers that are required in contextual information bases, if we are given media types? What are the references that are required in contextual information bases? Is it sufficient to have names for describing objects in a context or should these be replaced by something else?

The natural answer to the first question for generalising the notion of object identifiers in contexts is to choose the unique URLs of the media objects. Supporting navigation context requires access to path information, so we may want to reference back to the various media objects that we have encountered so far. These media objects are placed in several contexts, one of which is the right one corresponding to our path. However, we may also have different references, which lead to different contexts. So, the contexts we asked for in the second question are just the contexts for the media objects, which themselves are represented by media objects for paths, i.e. again non-elementary scenes.

As to the third question, we definitely want to have more information than just a name. Fortunately, the concept of media type is already based on the assumption of an underlying type system. Thus, we simply have to replace the names by values of any type permitted by the type system. Having defined such extended contexts, we can exploit query macros to traverse back a path in the story board and to explore alternative access paths, respectively.

However, one important aspect of media types is the use of classification abstraction. Conceptually, we do not define a set of media objects, but we generate them via queries defined on some underlying database schema. Therefore, we also need a conceptual abstraction for contexts. In order to obtain this conceptual abstraction, we assume another base type *Context*, the values of which are context names. Instead of this, we could take the type *URL*, but in order to avoid confusion we use a new type.

Then a *context* consists of a name C, i.e. a value of type content, a type t_C and a defining query q_C, which must be defined on the media schema, i.e. the set of media types, such that

$$(\{(\mathrm{id} : \mathit{URL}, \mathrm{value} : t_C, \mathrm{reference} : \mathit{Context})\}, q_C)$$

defines a view. Thus, executing the query q_C will result in a set of triples (u, v, r), where u is the URL of a media type, v is a value of type t_C, and r is the name of a context. If this context is undefined, this is interpreted as no reference for this object in this context.

5 Collaboration

In [1] we started an investigation of collaboration frameworks for distributed WISs addressing the problem that we may have tasks that are to be solved by collaboration of several users, e.g. using group work in e-learning systems. Collaboration is understood as combining cooperation, communication and coordination. The ground idea for capturing collaboration in WISs therefore is to define another extension of media types by means of *exchange frames*, which combine an exchange architecture addressing communication and cooperation, and a coordination specification, which consists of supporting programs, cooperation style coordination facilities, roles of collaborators, their responsibilities and rights, and the protocols they rely on.

Exploiting again the idea of media types supporting complex scenes we can abstract from the actor representing an individual user to actors representing a group of users. The exchange architecture for cooperation and communication within a group can be modelled by means of a workspace, which again is nothing but a media object associated with the group action, and consequently media object used by individuals in the group contain hidden links to the group object.

6 Reasoning about Media Types

Following our previous work in [12] we can formalise the operations associated on media types using the following language of abstract programs:

– 1 and 0 are abstract programs meaning `skip` and `fail`, respectively.
– An *assignment* $x := exp$ with a variable x and an expression of the same type as x is an abstract program. The possible expressions are defined by the type system. In addition, we permit expressions $\{\mathcal{P}\}$ with a logic program $\mathcal{P}$, assuming that $\mathcal{P}$ contains a variable ans. The expression $\{\mathcal{P}\}$ is interpreted as the result of the logic program bound to ans.
– If p, p_1 and P_2 are abstract programs, the same holds for the *iteration* p^*, the choice $p_1 + p_2$ and the sequence $p_1 \cdot p_2 = p_1 p_2$.
– If p is an abstract program and φ is a condition, then the *guarded program* φp and the *postguarded program* $p\varphi$ are also abstract programs.
– If x is a variable and p is an abstract program, then the *selection* $@x \bullet p$ is also an abstract program.

On these grounds we introduce a higher-order dynamic logic, where the order comes from the intrinsic use of the set constructor and the logic programs in queries. In fact, instead of using logic programs with a semantics defined by fixed-points, we could use directly higher-order logic enriched with a fixed-point operator. As a consequence, we may consider a logic program $\mathcal{P}$ as a representative of a higher-order logical formula, say $\varphi_{\mathcal{P}}$. If $\{\mathcal{P}\}$ is used as the right-hand side of an assignment, then it will correspond to a term $\mathbf{I}ans.\varphi_{\mathcal{P}}$ denoting the unique ans satisfying formula $\varphi_{\mathcal{P}}$. That is, all conditions turn out to be formulae of a logic $\mathcal{L}$, which happens to be a higher-order logic with a fixed-point operator. From the point of view of expressiveness the fixed-point operator is already subsumed by the order, but for convenience we do not emphasise this aspect here. Furthermore, by adding terms of the form $\mathbf{I}x.\varphi$ with a formula φ and a variable x all assignments in operations are just "normal" assignments, where the left-hand side is a variable and the right-hand side is a term of $\mathcal{L}$.

We now extend $\mathcal{L}$ to a dynamic logic by adding formulae of the form $[p]\varphi$ with an abstract program p and a formula φ of $\mathcal{L}$. Informally, $[p]\varphi$ means that after the successful execution of p the formula φ necessarily holds. In addition, we use the shortcut $\langle p\rangle\varphi \equiv \neg[p]\neg\varphi$, so $\langle p\rangle\varphi$ means that after the successful execution of p it is possible that the formula φ holds. Using our recursive definition of abstract programs the following rules apply to $[p]\psi$ for a complex abstract program p:

$$\begin{aligned}
[1]\psi &\equiv \psi \\
[0]\psi &\equiv 0 \\
[x := t]\psi &\equiv \psi\{x/t\} \text{ (substitute all free occurences of } x \text{ in } \psi \text{ by } t) \\
[p_1 p_2]\psi &\equiv [p_1][p_2]\psi \\
[p_1 + p_2]\psi &\equiv [p_1]\psi \wedge [p_2]\psi \\
[p^*]\psi &\equiv \text{ the weakest solution } \varphi \text{ of } \varphi \leftrightarrow \psi \wedge [p]\varphi \\
[\varphi p]\psi &\equiv \varphi \rightarrow [p]\psi \\
[p\varphi]\psi &\equiv [p](\varphi \rightarrow \psi) \\
[@x \bullet p]\psi &\equiv \forall x.[p]\psi
\end{aligned}$$

As outlined in [12] we can formulate proof obligations for the operations that result from the the specification of the story space, in particular for consistency with respect to static and dynamic integrity constraints on the underlying database schema. Furthermore, we can extend WIS personalisation in the light of dynamic logic.

7 Conclusion

In this survey we outlined in compact form the potential of the concept of media type emphasizing its use for interface abstraction supporting adaptivity to users, channels and devices, collaboration among user groups, session and context support, and serving as the conceptual bridge between identified requirements, the usage model of the WIS, the layout and playout, and the implementation on the basis of XML clusters and scripts. The versatility of media types are a major reason for the success of the codesign approach to WIS design as demonstrated in various large-scale applications. Furthermore, it is worth exploring the concept of media types in more depth, in particular with respect to the logical inferences as started in [12] on the basis of dynamic logic, the connection to deontic constraints on the storyboard and their refinement to the level of media types [13], and their embedding in layout and playout by means of screenography [10].

References

1. Binemann-Zdanowicz, A., Schewe, K.-D., Thalheim, B.: Development of collaboration frameworks for distributed web information systems. In: Kotsis, G., et al. (eds.) Proc. 7th International Conference on Information Integration and Web-Based Applications and Services (IIWAS 2005), vol. 2, pp. 551–562. Österreichische Computer Gesellschaft (2005)
2. Ceri, S., Fraternali, P., Bongio, A., Brambilla, M., Comai, S., Matera, M.: Designing Data-Intensive Web Applications. Morgan Kaufmann, San Francisco (2003)
3. Conallen, J.: Building Web Applications with UML. Addison-Wesley, Boston (2003)
4. De Troyer, O., Leune, C.: WSDM: A user-centered design method for web sites. In: Computer Networks and ISDN Systems – Proceedings of the 7th International WWW Conference, pp. 85–94. Elsevier, Amsterdam (1998)

5. Draheim, D., Weber, G.: Form-Oriented Analysis –A New Methodology to Model Form-Based Applications. Springer, Heidelberg (2004)
6. Güell, N., Schwabe, D., Vilain, P.: Modeling interactions and navigation in web applications. In: Mayr, H.C., Liddle, S.W., Thalheim, B. (eds.) ER Workshops 2000. LNCS, vol. 1921, pp. 115–127. Springer, Heidelberg (2000)
7. Houben, G.-J., Barna, P., Frasincar, F., Vdovjak, R.: HERA: Development of semantic web information systems. In: Cueva Lovelle, J.M., Rodríguez, B.M.G., Gayo, J.E.L., Ruiz, M.d.P.P., Aguilar, L.J. (eds.) ICWE 2003. LNCS, vol. 2722, pp. 529–538. Springer, Heidelberg (2003)
8. Kaschek, R., Schewe, K.-D., Thalheim, B., Zhang, L.: Integrating context in conceptual modelling for web information systems. In: Bussler, C.J., Fensel, D., Orlowska, M.E., Yang, J. (eds.) WES 2003. LNCS, vol. 3095, pp. 77–88. Springer, Heidelberg (2004)
9. Ma, H., Noack, R., Riaz-ud-Din, F., Schewe, K.-D., Thalheim, B.: Capturing forms in web information systems. In: Innovations in Information Technology 2007 – Web Applications and Services, IEEE, Los Alamitos (2007)
10. Moritz, T., Noack, R., Schewe, K.-D., Thalheim, B.: Intention-driven screenography. In: Information Systems Technology and its Applications (ISTA 2007), GI. Lecture Notes in Informatics, vol. P-107, pp. 128–139 (2007)
11. Moritz, T., Schewe, K.-D., Thalheim, B.: Strategic modelling of web information systems. International Journal on Web Information Systems 1(4), 77–94 (2005)
12. Schewe, K.-D.: The power of media types. In: Zhou, X., Su, S., Papazoglou, M.P., Orlowska, M.E., Jeffery, K.G. (eds.) WISE 2004. LNCS, vol. 3306, pp. 53–58. Springer, Heidelberg (2004)
13. Schewe, K.-D.: The challenges in web information systems development. In: Kaschek, R., Mayr, H., Liddle, S. (eds.) Information Systems Technology and its Applications (ISTA 2005), GI. Lecture Notes in Informatics, vol. P-63, pp. 204–215 (2005)
14. Schewe, K.-D., Schewe, B.: Integrating database and dialogue design. Knowledge and Information Systems 2(1), 1–32 (2000)
15. Schewe, K.-D., Thalheim, B.: Conceptual modelling of web information systems. Data and Knowledge Engineering 54(2), 147–188 (2005)
16. Schwabe, D., Rossi, G.: An object oriented approach to web-based application design. TAPOS 4(4), 207–225 (1998)
17. Thalheim, B.: Entity-Relationship Modeling: Foundations of Database Technology. Springer, Heidelberg (2000)
18. Theodorakis, M., Analyti, A., Constantopoulos, P., Spyratos, N.: Context in information bases. In: Proc. CoopIS 1998, pp. 260–270. Springer, Heidelberg (1998)

e-Skills and Standards – Prerequisites to Fully Exploit the Potential of ICT in Europe

Wolffried Stucky and Peter Weiß

Institute for Applied Informatics and Formal Description Methods
University of Karlsruhe (TH), D-76128 Karlsruhe, Germany
{stucky,weiss}@aifb.uni-karlsruhe.de

Abstract. The paper highlights the importance of platforms such as CEN and takes in particular focus on ongoing developments in the area of ICT skills. The authors look at current activities and some of the challenges ahead of ongoing standardisation endeavours in the field of ICT skills, qualification and training in Europe. Information and Communication Technology (ICT) today forms already an integral part of all industrial and service markets through the integration of ICT in goods or service offers. The availability of the right skills is beyond doubts one of the key success factors and important milestone on our way towards the knowledge-based economy. Knowledge, skills and competences are the cornerstone of Europe's competitiveness. ICT has always been subject to continuous advancement and change. Due to the explosion of innovations in ICT, the shortening technology life-cycles and newly emerging trends, ICT professionalism undergoes constant change and requires regular updating of acquired knowledge, skills and competences.

Keywords: e-skills, e-competences, e-learning, ICT standardisation, IT professionalism, ICT practitioners, lifelong learning.

1 Introduction

Information and Communication Technology (ICT) supports networked digital businesses across Europe and beyond and connects European companies in a digital business ecosystem. Digital business ecosystem is a new e-business paradigm which stands for the ability of companies to integrate their systems into highly complex value chains. In Europe there are roughly 20 million small and medium-sized enterprises (SMEs) in the European Union. They make up more than 99% of all European companies by number. Furthermore, SMEs make approximately 50% of European GDP (DBE, 2007). One of the characteristics in the knowledge-based economy is the increasing collaboration among enterprises and economic and social agents (DBE, 2007) – especially SMEs. Knowledge lies at the heart of the European Union's Lisbon Strategy to become the "most dynamic competitive knowledge-based economy in the world" (EC, 2007). Hence, qualified people with right knowledge, skills and competences (KSC) are the cornerstone of Europe's competitiveness. The availability of the right skills is beyond doubts one of the key success factors and important milestone on our way towards the knowledge-based economy. KSC together with ICT standards are important prerequisites to fully exploit the potential of ICT in Europe. Europe is still characterised today through diverging systems and paradigms demarcated through national borders. Europe is developing a research area which supports collaborative research across national borders through financial incentives such as funding opportunities to conduct collaborative research in one of the specific strategic objectives defined by the Seventh Research Framework Programme (FP7). ICT standards allow working towards the harmonisation of existing approaches and supporting the broader uptake of e-business solutions and applications in Europe. The European Single Market is a recent success story in political and economic history. Promoting a European Single Market requires in essence the following three priorities (DBE, 2007):

- Promote a favourable environment and framework conditions for electronic business and entrepreneurship
- Facilitate the take-up of electronic business
- Contribute to providing related ICT skills.

The three priorities stress the necessity of an integrated approach which is based on creation of an environment, a business ecosystem and need for ICT skills. The latter will be subsequently in focus of this paper and will be looked at in more detail.

E-Skills is a broader concept than ICT skills. It encompasses in essence three skill areas: (1) practitioner skills, (2) user skills and (3) e-business skills. To get acquainted in more detail with the concept "e-skills" we recommend to consult (Dixon and Beier, 2006), (Winterton et al., 2006), (Weiß et al., 2006).

The first author was president of the Council of European Professional Informatics Societies (CEPIS) from 2001 to 2003 and acted as past president from 2003 to 2004. CEPIS is a non-profit organisation seeking to improve and promote a high standard among Informatics Professionals in recognition of the impact that Informatics has on employment, business and society. CEPIS represents 37 Member Societies in 33 Countries across greater Europe. CEPIS is to provide a coordinated European voice that is able to represent the views of European Informatics Professionals on major issues to European Institutions (CEPIS, 2008). Furthermore, he was chair of the CEN/ISSS Workshop on ICT Skills in its first phase (2002-2004) and co-chair of the second phase (2004-2006). Since 2006 the authors are both active members of the now running third phase of the Workshop. The second author has co-ordinated the initiative "ICT Certification in Europe" within the Workshop which involves amongst others some of the key players of industry-based ICT certification business such as Microsoft, Cisco, CompTIA, EXIN, ECDL-F, etc.

The paper highlights the importance of platforms such as CEN[1] and takes in particular focus on ongoing developments in area of ICT skills. Activities of IFIP[2], CEPIS and CEN as well as initiatives supported and launched by the European Commission in the field of ICT skills are presented. The paper looks at current activities and some of the challenges ahead of ongoing standardisation endeavours in the field of ICT skills, qualification and training in Europe, namely (see section 4 for further details):

[1] Comité Européen de Normalisation (The European Standards Committee).
[2] International Federation for Information Processing: http://www.ifip.org, [31 January 2008].

- European e-Skills Forum and Follow up Activities
- ICT Professionalism
- ICT Certification in Europe
- European e-Competence Framework.

2 The European Standards Committee

Many various working groups exist at European level dedicating their efforts and work to overcome existing barriers for the broader and more efficient use of ICT in Europe. The European Committee for Standardization, CEN, with its voluntary standards, provides against this background an important consensus-based platform which supports the dialogue and information exchange of the various stakeholders. European Standards are one of the constituent elements to fully exploit the potential of ICT such as for e-business applications for the European Single Market (CEN, 2008). ICT is a global business and the major players of the business accordingly operate on global level. Looking at the European dimension of ICT allows different views and perspectives. Increasing the competitiveness of the European information and communication industries requires giving interoperability highest priority on all fronts (EC, 2006). CEN provides an open consensus platform to meet market needs. CEN/ISSS[3] (2007) is the name given to CEN's ICT sector activities dealing with ICT standards. It provides market players with a comprehensive and integrated range of standardization services and products, in order to contribute to the success of the Information Society in Europe (CEN, 2008).

CEN is a non-profit association, whose members are national standards bodies (BSI[4], DIN[5], AFNOR[6], etc.). It is one of three European Standardization Organizations recognized by the European Commission. CEN, together with CENELEC[7] and ETSI[8], has to address the challenges of converging technologies. The borderlines between different technologies are disappearing more and more. European standardization has to respond accordingly and has to ensure the respective infrastructure. Industry expects an efficient and timely production of standards that meet market needs. Results of the various working groups and standardisation activities are published as CEN Workshop Agreements (CWAs) which represent the workshops' main end-product. CWAs are not "formal standards", but appear in catalogues of CEN Member bodies (CEN, 2008).

CEN's self-understanding is to be a business facilitator in Europe, removing trade barriers for European industry and consumers. Its mission is to foster the European economy in global trading, the welfare of European citizens and the environment. Through its services, it provides a platform for the development of standards and other technical specifications. CEN's 30 National Members work together to develop voluntary European Standards (ENs) in various sectors to build a European Single

[3] ISSS stands for Information Society Standardization System.
[4] BSI British Standards is the National Standards Body of the UK.
[5] The German Institute for Standardization.
[6] Association Française de Normalisation.
[7] European Committee for Electrotechnical Standardization.
[8] The European Telecommunications Standards Institute.

Market for goods and services and to position Europe in the global economy. More than 60,000 technical experts, as well as business federations, consumer and other societal interest organizations, are involved in the CEN network that reaches over 460 million people (CEN, 2008).

3 Current Development and Challenges ahead

As already mentioned ICT is central to "boosting productivity and improving competitiveness" (EC, 2007). The European Commission underlines in an actual report dated November 2006 that the ICT sector added 5.3% value to EU GDP and 3.6% of EU employment. It accounted for 25% of total EU research in business (EC, 2006). Furthermore the report argues that for 10 years technological progress and investment in ICT have accounted for half of the EU economy's productivity gains. Hence, the availability of required KSC is crucial for the competitiveness of European industry. Looking at current estimates of supply and demand of ICT skills, experts are concerned about future shortages of ICT professionals in Europe.

A current study published by CEPIS forecasted demand and supply of ICT skills comparing and analysing the various supply channels of ICT professionals with expected demand. The study strived for predicting future developments of demand and supply. The experts have created various future scenarios based on quantitative evidence and various indicators, and have estimated supply and demand levels in 2010 and 2015. The study concludes that there is concrete evidence of facing shortages of up to 70,000 IT practitioners per year, as supply limitations fail to satisfy high demand (CEPIS, 2007). In order to avoid future ICT skills shortages, public-private initiatives are key. CEPIS recommends: "[…] to get the balance of supply and demand right, policymakers in education, in regional and national governments and at EU level must be very attentive to likely workforce needs. At a policy level, this extends to research and development and to immigration policy; and it requires professional bodies and trade unions to work with policymakers to ensure a greater supply of ICT professionals. The ICT industry itself has a key role to play in coordinating these efforts" (CEPIS, 2007).

The European e-Skills Forum (ESF) highlighted in its declaration from 2006 (ESF, 2006) that it is essential to develop a long-term strategy for e-skills in Europe. The declaration calls upon relevant stakeholders to support a set of actions and to include them into a respective long-term strategy. Subsequently, we look at some of the key issues raised and describe the major pillars which lay the foundation of concerted future activities of stakeholders in Europe.

4 Towards a Long Term Strategy for e-Skills

In the previous sections we highlighted some of the challenges ahead for Europe to fully exploit the potential of ICT in Europe. Here we present now some of the key messages and actual key priorities in the area of e-skills. As already mentioned, the CEN/ISSS Workshop on ICT Skills together with the ESF are two important open-consensus platforms for stakeholders to agree on common ICT standards and to launch concerted actions. Various other working groups such as EU ICT-Task Force (Task Force, 2007), E-Skills Competences Consortium (E-SCC, 2007), IFIP Task Force on ICT professionalism, Working Group on Skills and Employability, European Alliance on Skills and Employability (EA, 2007) etc. have been established with support of public stakeholders such as the European Commission to address specific areas of some of the identified problems. In essence, actual measures gravitate around the central aim to establish an EU e-Career Portal which offers career services for the members of the ICT practitioner community. To prepare the ground for such a portal, the European Commission has launched in 2006 a first study which looked into the possibilities of setting up such a portal. Within the current phase 3 of the CEN/ISSS Workshop on ICT Skills (CEN/ISSS, 2007) a major initiative "EU e-Career Services" will be launched in March 2008.

Many experts today see one of the key challenges in minimising actual mismatches in ICT skills supply and demand between university and industry (CompTIA, 2004), (E-SCC, 2007). Higher education and industry have already noticed the importance to intensify their dialogue and information exchange. A common language would ease such regular information exchange between industry and higher education. The European e-Competence Framework will be an integral element of an overarching (reference) framework that allows the flow of information between industry and education and training.

Another activity which will be launched in March 2008 overviews the actual ICT professional certification market. Furthermore, the Workshop plans to launch accompanying projects such as a methodological study to elaborate the necessary theoretical background and methodological foundations which will be required to strengthen and support launched initiatives. Subsequently, we look at some of the ongoing activities and most influential actions to be carried out in 2008.

4.1 European e-Skills Forum and Follow Up Activities

The ESF and its current follow-up activities (ESF, 2004, 2006) is an important platform which stimulates multi-stakeholder partnerships in the field of e-skills in Europe. The Forum originated in response to severe shortages of ICT practitioners at the end of the 1990s. The Commission supported industry-led initiatives and established expert groups, in particular the ESF, with representatives of Member States and stakeholders. The ESF attracts major stakeholders stemming from ICT industry, ICT user industry, professional associations, social partners, public authorities, educational bodies, etc. The long-term objective of the ESF is to work towards the ability to create, manage, plan and develop ICT competences that will be needed in a long term perspective across Europe (ESF, 2008). Every two years the stakeholders meet at the European e-Skills Conference in Thessalonica. Next conference will be held in October 2008.

In 2007, the European Commission has launched a study "e-Skills Foresight Scenarios" which elaborated a framework with respective indicators (CEPIS, 2007). The framework helps to continue the collaboration between industry and policymakers through better understanding of the impacts of cyclical market effects on the supply of and demand for practitioners (CEPIS, 2007). One may consider that usually reactions on occurring changes in supply and demand are often rather slow and delayed in time.

Educating and training ICT professionals requires generally up to 3 to 5 years. ICT skills "[…] must be imparted more evenly throughout all phases of the business cycle. The pressures of business often make it difficult to allocate enough time to training, while in slack phases, industry is cautious about investing in training" (CEPIS, 2007).

4.2 ICT Professionalism

There is an ongoing discussion on ICT professionalism which relates to the availability of performance and occupational standards for ICT practitioners. A lack of professional standards is often referred to in the context of software failures and induced pressure to reduce the risk and cost of failure.[9]

Knight (2005) stresses that many engineers lack adequate education and most and worst of all lack a comprehensive professional culture that encompasses responsibility of every engineer when dealing with software as an entity and with software engineers as professionals. It is not just a question of education but of professionalism. He demands enforcement of necessary care and monitoring by professional societies in this case in the area of software development. He also suggests that government regulation and insurance requirements should be considered. Hughes and Thompson (2007) describe the fundamentals of professionalism in ICT as developed in the UK and the new IFIP programme which is designed to establish a global IT profession with clear and consistent internationally recognised standards and qualifications for ICT practitioners. The ongoing discussion about European standards and solutions in the field of e-skills especially for the harmonisation of ICT professional certification requires looking at the broader context of the ICT profession.

IFIP has launched a programme which aims to build the international IT profession. In the meanwhile, a task force has been established by IFIP that determined a "[…] lack of clarity around international professional skills and qualifications and worldwide mobility of IT professionals" (Hughes and Thompson, 2007). ICT professionalism needs to be built on clearly defined standards and refers to three fundamental parts to build the ICT profession: (1) infrastructure, (2) governance & administration and (3) required standards.

An *infrastructure* needs to be put in place which encompasses important elements such as relevant frameworks providing common reference points and models to identify skills needs (qualification and skills/competence frameworks) as well as the definition of scope of the ICT profession and respective specialist disciplines.

Another important element is the *Body of Knowledge* which ought to be discussed not only in relation to the establishment of ICT professionalism but also in the context of harmonisation of ICT professional certification. The Body of Knowledge provides the guiding principles on which certification ought to be based. The Body of Knowledge builds the foundation for creating courseware and planning and design of exams (Welch, 2007). Tasks in relation to governance and administration span from delivery of development services, support processes to a respective admission process and putting in place an appropriate governance structure. All stakeholders, led by professional associations and major ICT vendors ought to work towards defining standards and making the administration arrangements as well as contributing to the

[9] See (Hughes and Thompson, 2007), (Knight, 2005), (Welch, 2007).

professional infrastructure mentioned above through products and services (Hughes and Thompson, 2007). ICT professionalism requires clearly defined and broadly recognised standards. “The progressive attainment of professional ethos necessitates a hierarchy of standards and qualifications to acknowledge progress and achievement as well as to set public expectations” (Hughes and Thompson, 2007). Standards encompass educational and industry-oriented performance standards which in turn are preferably expressed through a common language as competence and skills standards, as well as clearly defined standards in the sense of a code of conduct every ICT professional is committed to. Code of conduct encompasses professional ethics and behaviour recognising the public obligations that ICT professionals also have. The essential elements of professionalism are three-fold (Hughes and Thompson, 2007): (1) competence, (2) integrity, responsibility and accountability, and (3) recognition.

While ICT professionalism ought to be the uncontestable long-term objective driving the harmonisation process, the network of relationships of ICT certifications is apparently the key to success in the short-term. Our research showed that there is general interest and willingness of stakeholders to co-operate and to join a round table. Again CEN offers an excellent platform for stakeholders to meet and discuss the challenges ahead and steer common measures.

4.3 ICT Certification in Europe

Figure 1 displays the major milestones and roadmap towards agreed European standards to exploit mutual benefit. On the right side required measures towards ICT professionalism is shown. On the left side the figure refers to corresponding objectives. Harmonise (2007) has identified ICT professionalism as one fundamental part of the harmonisation framework so it stands on top of our “pyramidal model“ forming the harmonisation roadmap outlining an evolutionary path demarcated through five stages. As depicted the roadmap describes objectives and measures to reach the goal of harmonisation of ICT professional certification. While on the left side of the figure the objectives are derived “top-down” aiming at creating the right climate in the sense of an environment which allows for exploitation of mutual benefit and working toward the harmonisation of ICT professional certifications, on the right side measures were outlined following a “bottom-up” approach with the prevailing competitive environment seen as our starting point. In the middle the figure illustrates the different stages that demarcate the necessary steps to be taken and describes the milestones to be met on the way towards harmonisation.

Harmonisation of certification ought to strive for a learner-centred credentialing system which is transparent with regard to content, technologies, methods and actors. Such a system ought to offer clear guidance as well as recognition of learning (whether formal, non-formal or informal) and acquired knowledge, skills and competences. ICT professional certifications as a concept appears to be able to contribute significantly to increase the mobility of the ICT workforce in Europe. From this perspective, certification programmes are competing for being part of the body of knowledge of an international ICT profession and should be considered for becoming an integral part of an individual’s continuous professional development plan and career path (Povalej and Weiß, 2007). Certification of ICT professionals comes to the fore when discussing three major requirements to achieve an international ICT profession:

- common language to describe professional skills and competences
- standard means of measurement for professional skills and competences
- mechanism for independent recognition and assurance of quality of professional skills and competences.

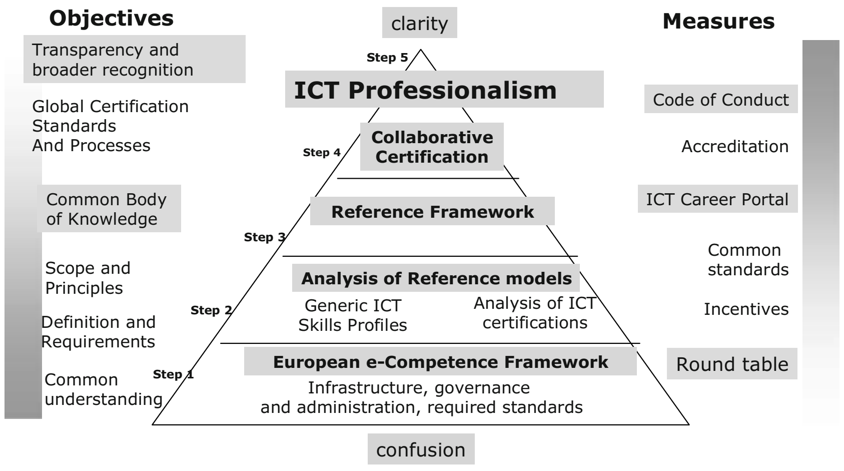

Objectives
Transparency and broader recognition
Global Certification Standards And Processes
Common Body of Knowledge
Scope and Principles
Definition and Requirements
Common understanding
clarity
Step 5
ICT Professionalism
Step 4
Collaborative Certification
Reference Framework
Step 3
Analysis of Reference models
Generic ICT Skills Profiles
Analysis of ICT certifications
Step 2
European e-Competence Framework
Step 1
Infrastructure, governance and administration, required standards
confusion
Measures
Code of Conduct
Accreditation
ICT Career Portal
Common standards
Incentives
Round table

Fig. 1. Roadmap towards Harmonisation (Harmonise, 2007)

As shown the first step aims to reach a common understanding and to agree initially on definitions and requirements in order to achieve a consistent base that allows for subsequent concentration on and turning to the more complex issues and problems to be addressed in the harmonisation process. For this purpose a stakeholder needs to join a round table that allows for the formation required partnerships and to achieve an information exchange and knowledge flow between stakeholders, foremost between industry and educational institutions.

Transparency and broader recognition of ICT professional certification forms another important goal of the harmonisation endeavour. Additional derived objectives are to agree on the scope and principles of the harmonisation process, to create a body of knowledge, and to arrive at certification standards and processes at global level (Harmonise, 2007). CEPIS recommends that "bridges should be built between ICT industry-based certifications (CompTIA, 2004), (E-SCC, 2007) and formal education and vocational training courses. At the same time, students should not be moulded just for the immediate needs of the workplace" (CEPIS, 2007), (Stucky, 2004). Competence-based professional profiles are a referred concept to define and exchange information about respective professional standards. To manage competence-based professional profiles an organisation needs both: competences and profiles. Accordingly two constituent elements are needed to set up the necessary infrastructure: a technical competence dictionary and a set of profiles based on technical competences (Harmonise, 2007).

4.4 European e-Competence Framework

Higher education and industry started working on developing a common language to communicate about skills requirements and needs. The work is currently being carried out within the actual phase of the CEN/ISSS Workshop on ICT Skills. Better understanding about ICT skills can be useful in a number of ways, but extensive discussions as part of the Workshop process have concluded that the greatest value from this common language can be gained from its use as meta-framework (Dixon and Beier, 2006):

- a tool for structured comparison between existing ICT practitioner skills/ competence frameworks;
- a guidance resource on which those considering the possibility of developing their own frameworks can draw;
- a conceptual basis for planning future developments that would help assure a greater supply of competent ICT practitioners to European employers; and
- a starting point from which the proposed European Qualifications Framework (EQF) can be applied to, and evaluated for, ICT practitioner work, both by employers and by practitioners planning their careers.

The framework aims to facilitate a link between national structures; additionally it provides a set of Europe-wide jointly defined ICT core competences. These competences will be classified according to main ICT work areas and link directly to the EQF (EQF, 2008), providing a European basis for internationally efficient personnel planning and development. EQF is a translation grid for qualifications around Europe (ECF, 2007). It has two principal purposes: (1) to promote mobility between countries, and (2) to facilitate lifelong learning. Both are indispensable for achieving more and better jobs and growth, as Europe faces the challenges of becoming an advanced, knowledge-based economy. All countries have a qualifications system but a qualifications framework is a more systematic way of classifying qualifications, usually by a hierarchy of levels. Among the benefits are greater readability of qualifications and easier progression between levels (EQF, 2006). The primary users of the EQF will be bodies in charge of national and/or sectoral qualification systems and frameworks. Once they have related their respective systems to the EQF, the EQF will help individuals, employers and education and training providers compare individual qualifications from different countries and education and training systems.

ICT practitioner work, and the skills and competences that are required to carry it out, arise from the way in which work on ICT products, services and systems is organized. Just as ICT themselves are highly complex and continuing to evolve, so the skill-sets needed in relation to deploying and using ICT are both very complex and not yet stable or mature in terms of coherence of their classification. As a result, efforts to clarify and codify the structures of these skills have not yet reached a level of stability that enables adequate agreement at the European level on classification frameworks that could be thought of as a possible future standard for the European Union (Dixon and Beier, 2006).

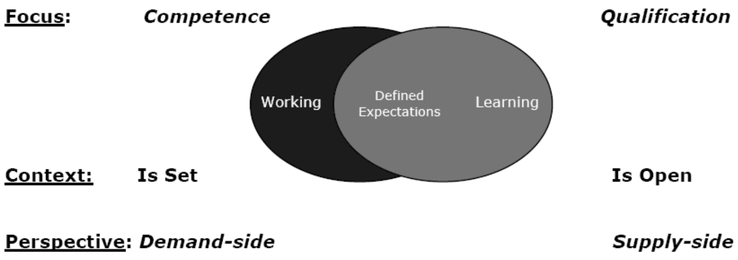


Fig. 2. Relating Qualification and Competence (Dixon and Beier, 2006)

Ongoing work within (CEN/ISSS, 2007) aims at producing such a common European reference framework of ICT core competencies adopted by industry. It is expected to bring major benefits to education, training, industrial and social partners across Europe (see Figure 2). This European multistakeholder effort is co-funded by the European Commission and driven by ICT vendor and user industry's needs. It will provide significant support to ICT skills development across the European Union. The framework is scheduled to be ready for Europe-wide use by autumn 2008. The framework addresses participants seeking to maximise effectiveness of ICT services and infrastructure through competence development (CEN/ISSS, 2007).

5 Summary and Outlook

The paper has argued the importance of ICT skills and European standards to fully exploit the potential of ICT in Europe. ICT standardisation is a part of the general standardisation activities, and contributes to policy objectives to improve the competitiveness of European industry, as specified in the Lisbon strategy. CEN provides an important platform for stakeholders to meet and to discuss how to overcome existing barriers through launching concerted actions. The ESF – a multi-stakeholders effort and partnership – is one important initiative in the ICT standardisation domain working towards a long-term strategy for e-skills in Europe. We have presented important initiatives launched from the CEN/ISSS Workshop on ICT Skills. The European e-Competence Framework constitutes an important step forward towards a common language to exchange information about ICT skills in the field of ICT. In 2008 the framework will be available. The result of the work will represent a sector specific application of the EQF. The endorsed EU e-Career Portal is likely to achieve better transparency and visibility of what is available and is expected to leverage the attractiveness of the ICT profession through highlighting possible career paths for individuals by linking available training offerings and related ICT qualifications. Furthermore, it will offer required flexible entry points into the ICT profession to respond to emerging shortages and to react faster on mismatches of demand and supply in industry.

Added-value services for ICT practitioners will support the continuing professional development of individuals. ICT certifications offer "opportunities-to-learn" and can contribute to a larger transition process towards individualised learning paths. These learning paths or opportunities ought to be preferably linked to an overarching common qualifications structure or framework broadly recognised by employers and ICT professionals in Europe or beyond. The generally tolerated "mystification" of some ICT professional certification programmes can be identified as one of the pivotal problems that needs to be solved along the way towards harmonisation. Emerging myths are generally stimulated through marketing campaigns of products. Better transparency of certification requirements and contents through open access policy appears counter-productive as it contributes to "demystification" of popular ICT certification programmes.

References

[CEN, 2008] The European Standards Committee: CEN Strategy 2010, Key Objectives (January 2007), http://www.cen.eu [January 28, 2008]

[CEN/ISSS, 2007] CEN/ISSS Workshop on ICT Skills (Phase 3): 2006 - 2008, http://www.cen.eu/cenorm/businessdomains/businessdomains/isss/activity/wsict-skills.asp [January 31, 2008]

[CEPIS, 2007] E-Skills in Europe: Matching Supply to Demand, CEPIS Council of European Professional Informatics Societies (October 2007), http://www.cepis.org [January 30, 2008]

[CEPIS, 2008] The Council of European Professional Informatics Societies (CEPIS), http://www.cepis.org [January 31, 2008]

[CompTIA, 2004] CompTIA (The Computing Technology Industry Association) (2004). The Situation and the Role of E-Skills Industry Certification in Europe (August 2004), http://eskills.cedefop.europa.eu/download/ESCC%20report%20for%20e-Skills%202004.pdf [January 30, 2008]

[DBE, 2007] Digital Business Ecosystems, European Commission, DG Information Society and Media, Luxembourg: Office for Official Publications of the European Communities, book, ISBN 92-79-01817-5 (2007), http://www.digital-ecosystem.org/ [January 31, 2008]

[Dixon and Beier, 2006] Dixon, M., Beier, Y.: CWA 15515: European ICT Skills Meta-Framework - State-of-the-Art Review, Clarification of the Realities, and Recommendations for Next Steps, ftp://ftp.cenorm.be/PUBLIC/CWAs/e-Europe/ICT-Skill/CWA15515-00-2006-Feb.pdf [January 31, 2008]

[EA, 2007] European Alliance on Skills for Employability, http://www.e-scc.org/alliance/default.aspx [January 31, 2008]

[EC, 2006] European Commission: More interoperability needed to boost the European ICT industry, competitiveness. IP/06/1635, Brussels, p.2 (November 27, 2006), http://europa.eu/rapid/pressReleasesAction.do?reference=IP/06/1635&format=PDF&aged=1&language=EN&guiLanguage=en [January 31, 2008]

[ECF, 2007] European e-Competence Framework: Towards a European e-Competence Framework, http://www.ecompetences.eu/ [January 31, 2008]

[EC 2007] European Commission: e-Skills for the 21st Century: Fostering Competitiveness, Growth and Jobs. COM(2007) 496 final, Brussels, Communication from the Commission to the Council, the European Parliament, the European Economic and Social Committee and the Committee of the Regions (September 07, 2007), http://ec.europa.eu/enterprise/ict/policy/ict-skills/2007/COMM_PDF_COM_2007_0496_F_EN_ACTE.pdf [January 31, 2008]

[EQF, 2006] Towards a European Qualifications Framework for Lifelong Learning, http://ec.europa.eu/education/policies/2010/doc/consultation_eqf_en.pdf [January 31, 2008]

[EQF, 2008] The European Qualifications Framework, http://ec.europa.eu/education/policies/educ/eqf/index_en.html [January 31, 2008]

[E-SCC, 2007] E-Skills Competences Consortium, http://www.e-scc.org/default.aspx [January 31, 2008]

[ESF, 2004] European E-Skills Forum: E-Skills in Europe: Towards 2010 and Beyond, http://ec.europa.eu/enterprise/ict/policy/doc/e-skills-forum-2004-09-fsr.pdf [January 31, 2008]

[ESF, 2006] European E-Skills Forum: European e-Skills 2006 Conference Declaration, http://eskills.cedefop.europa.eu/conference2006/Thessaloniki_Declaration_2006.pdf [January 31, 2008]

[ESF, 2008] European e-Skills Forum and Follow-Up Activities; E-Skills Online Virtual Community (Cedefop), http://communities.trainingvillage.gr/esf [January 31, 2008]

[Harmonise, 2007] HARMONISE Project: Survey of Certification Schemes for IT Professionals across Europe towards Harmonisation, Technical Report, CEPIS Council of European Professional Informatics Societies (September 2007), http://www.cepis-harmonise.org [January 31, 2008]

[Hughes and Thompson, 2007] Hughes, C., Thompson, C.: The International IT Professional Practice Programme. UPGRADE Journal, Special Issue on ICT Certifications for Informatics Professionals VIII(3) (June 2007), http://www.upgrade-cepis.org/issues/2007/3/upgrade-vol-VIII-3.html [January 31, 2008]

[Knight, 2005] Knight, J.C.: Focusing Software Education on Engineering. ACM SIGSOFT Software Engineering Notes 30(2), 3 (2005)

[Povalej and Weiß, 2007] Povalej, R., Weiß, P.: Survey of ICT Certification Systems for ICT Professionals in Europe. UPGRADE Journal, ICT Certifications for Informatics Professionals VIII(3) (June 2007), http://www.upgrade-cepis.org/issues/2007/3/upgrade-vol-VIII-3.html [January 31, 2008]

[Stucky, 2004] Stucky, W.: E-Skills in the Knowledge-Based Economy. In: Creating Jobs in the Knowledge-Based Economy, European e-Skills Conference 2004, Thessalonica (2004), http://eskills.cedefop.europa.eu/eskills2004/presentations/Wolffried_Stucky_esfConf2004_cepis.ppt [January 31, 2008]

[Task Force, 2007] EU ICT-Task Force (Task Force on ICT Sector Competitiveness & ICT Uptake), http://ec.europa.eu/enterprise/ict/taskforce.htm [January 31, 2008]

[Weiß et al., 2006] Weiß, P., Dolan, D., Stucky, W., Bumann, P.: ICT Certification in Europe. Cedefop Dossier Series, vol. 13. Office for Official Publications of the European Communities, Luxembourg (2006), http://www.trainingvillage.gr/etv/Information_resources/Bookshop/publication_download.asp?pub_id=431&dl_id=1269&pub_lang=en [January 31, 2008]

[Welch, 2007] Welch, J.: Certifications for Software Professionals Helps to Assure Safety, Reliability in Vital Systems. UPGRADE VII(3) (June 2007), ICT Certifications for Informatics Professionals

[Winterton et al., 2006] Winterton, J., Deist, F.D.-L., Stringfellow, E.: Typology of Knowledge. In: Skills and Competences: Clarification of the Concept and Prototype. Cedefop Reference Series, vol. 64. Office for Official Publications of the European Communities, Luxembourg (2008), http://www.trainingvillage.gr/etv/Information_resources/Bookshop/publication_details.asp?pub_id=434

ASM Foundations of Database Management

Bernhard Thalheim[1] and Klaus-Dieter Schewe[2]

[1] Christian Albrechts University Kiel, Department of Computer Science
Olshausenstr. 40, D-24098 Kiel, Germany
[2] Massey University, Information Science Research Centre
Private Bag 11 222, Palmerston North, New Zealand
thalheim@is.informatik.uni-kiel.de, k.d.schewe@massey.ac.nz

Abstract. Database structuring is well understood since decades. The operating of databases has been based in the past on temporal logics and did not yet get an easy to understand formal underpinning. Therefore, conceptions like transaction and recovery are mainly discussed at the logical or operational level. This paper shows that database structuring and functionality can be defined within a uniform language. We base database semantical on the operational semantics of *abstract state machines* (ASM). This uniform mechanism allows to define the structuring, the functionality, the distribution and the interactivity of a database system in a way that supports abstract consideration at various layers of abstraction, that supports refinement of specifications to more detailed ones and that support proof of properties.

1 Adequacy and Deficiencies of Database Technology

1.1 Strength and Weaknesses of Database Technology

Database systems are currently broadly used for support of data-intensive services. These broad usage is based on advantages such as the following:

Consistent storage of data: Data are uniquely stored in the most actual version. Each user gets the same data. Inconsistency can be avoided. Furthermore, redundancy can be reduced and standards can be enforced.

Multi-user support: Data can be consistently shared among different users. Also, conflicting requirements can be balanced. Security is enforced by restricting and managing access to data. Data can be consistently distributed within a network.

Integration into component-ware: Currently, database systems are turned into middle-ware components in information-intensive applications. Database operating is based on the transaction paradigm. A transaction is a logical unit of work. Database systems are designed to support transactions.

Nevertheless, database engines do not completely support complex applications such as internet services, real-time applications, stream information systems and web information systems.

Content and information instead of data: With the development of more complex application sophisticated support for various abstraction levels on data is required. Instead of raw or micro-data users want to retrieve condensed data or macro-data and to work on them.

Transaction semantics: The transaction approach is based on atomicity, consistency, isolation and durability requirements on transaction application. User action are sometimes not atomic and require sophisticated support for intermediate actions. The isolation level may vary over time. The are different degrees for durability.

View support: Each object in the database needs to be identifiable. Query languages allow to construct macro-data from the micro-data of the objects contained in the database. These macro-data may be composed into complex derived objects. Due to the query language some of them may be not identifiable. Further, their connection to the micro-data may be not injective. Therefore, data manipulation on macro-data cannot translated to data manipulation on micro-data.

Missing operational semantics: Semantical treatment of structuring is based on the predicate logic since the advent of the relational database model. Entity-relationship modelling can be based on a generalized predicate logic. Object-oriented models are often defined in a fuzzy manner without existing models. Moreover, database operation is still not well based.

In this section we show now that the weaknesses of database models and technology for internet application can be reduced by the ASM approach.

1.2 Well-Founded Structuring and Operating Transparency

Database systems are typically specified through a specification of database structuring and the assumption of canonical database operating. Structuring of databases is given by a signature of the database and a set of static integrity constraints. They form the schema of the database. The signature is used as an alphabet of a canonical first-order (or first-order hierarchical [13]) predicate logic. Static integrity constraints may be expressed as formulas of this logic. The functionality of a database system is defined on the basis of an algebra that is generically defined over the signature. This algebra is extended to a query algebra and to modification operations for the database system. Database management systems provide services that are assumed to be given whenever we are talking on database operating. Their behaviour and their operating is assumed to be canonically given.

This assumption might be appropriate for the predesign or business user layer [5,13] of specification. The implementation independence of the business user layer and the operating transparency supports concentration application specific aspects of a database system. It is not appropriate for the conceptual layer since we need to consider database behaviour as well. This inappropriateness causes many problems for database programming and operating. This paper shows how the ASM approach can be used to overcome this gap.

Information systems extend database systems by a support for users, by interface systems and by content management. Web information systems [9,12] augment classical information systems by modern web technologies. They aim in supporting a wide variety of users with a large diversity of utilisation stories, within different environments and with desires for personal web-enhanced work spaces. The development of WIS is therefore adding the user, story and interaction dimension to information systems development. So far information systems development could concentrate on the development of sophisticated structuring and functionality. Distributed work [10]has already partially been supported.

1.3 Necessities for Specification of Interactive Information Systems

Despite the presence of an active research community that studies conceptual models for information systems, interaction still lacks precise formal underpinnings. There are several approaches which can be generalized for development of a formal basis of interactive information services:

Workflow and pattern: Workflow approaches allow to combine dataflow and computation. Workflows can be constructed on the basis of basic processes by application of basic control constructors such as sequence, parallel split, exclusive choice, synchronization, and simple merge, by application of advanced branching and synchronization control constructors such as multiple choice, multiple merge, discriminator, n-out-of-m join, and synchronizing join and by application of structural control and state-based control constructors.

Wegner's interaction machines: The approach is based on four assumptions: User rather observe machine behavior instead of having a complete understanding of machines I/O behavior. Modelling of systems cannot be entirely inductive. Instead of that co-induction needs to be used, at least for the earlier design phases. Users compete for resources. Thus, modelling includes an explicit specification of cooperation and competition. Interactive behavior cannot be entirely modelled on the basis of I/O behavior but rather in term of interaction streams.

We use the interaction machine approach[6] in order to reason formally on information services. An interaction machine[15] can be understood as Turing machine with multi-user dynamic oracles (MIM) or with single-user dynamic oracles (SIM). An interaction machine can be specified as follows [3]:

```
if $condition$
      then $state = Nextstate(state, database, input)$
           $database = Modification(state, database, input)$
           $output = output \circ Out(state, database, input)$
```

Statechart diagrams: The statechart approach has been proved to be useful for state-oriented modelling of database applications and more generally for use-case diagrams. Despite its usage in UML it has a precise semantics which can be easily integrated into ER modelling approaches[13].

User interaction modelling: Interaction involves several partners (grouped according to characteristics; group representatives are called 'actors'), manifests itself in diverse activities and creates an interplay between these activities. In order to develop usable websites the story of the application has to be neatly supported [9,12,10,14]. Interaction modelling include modelling of environments, tasks and actors beside modelling of interaction flow, interaction content and interaction form [7].

1.4 Achievements of the ASM Approach

The abstract state machine (ASM) method nicely supports high-level design, analysis, validation and verification of computing systems:

- ASM-based specification improves industrial practice by proper orchestration of all phases of software development, by supporting a high-level modelling at any level of abstraction, and by providing a scientific and formal foundation for systems engineering. All other specification frameworks known so far only provide a loose coupling of notions, techniques, and notations used at various levels of abstraction.

  By using the ASM method, a system engineer can derive a general application-oriented understanding, can base the specification on a uniform algorithmic view, and can refine the model until the implementation level is achieved. The three ingredients to achieve this generality are the notion of the ASM itself, the ground model techniques, and the proper treatment of refinement.
- Abstract state machines entirely capture the four principles [16] of computer science: structuring, evolution, collaboration, and abstraction.

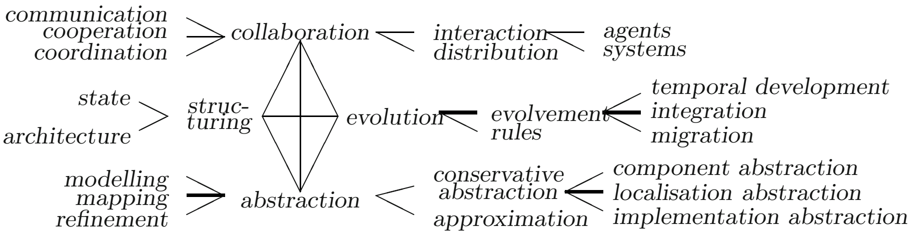


Fig. 1. The four principles of Computer Science

This coverage of all principles has not been achieved in any other approach of any other discipline of computer science. Due to this coverage, the ASM method underpins computer science as a whole. We observe the following by comparing current techniques and technologies and ASM methods: ASM are running in parallel. Collaboration is currently mainly discussed at the logical or physical level. Evolution of systems is currently considered to be a hot but difficult topic. Architecture of systems has not yet been systematically developed.

– The ASM method is clearly based on a number of postulates restricting evolution of systems. For instance, sequential computation is based on the postulate of sequential time, the postulate of abstract state, and the postulate of bounded exploration of the state space. These postulates may be extended to postulates for parallel and concurrent computation, e.g., by extending the last postulate to the postulate of finite exploration.

2 ASM Specification of Database Systems

Database systems are designed to operate in parallel. Operating of databases systems can be understood in a state-based approach.

2.1 ASM Specification of Databases and Database Structuring

The abstract state machine approach allows simple and refinable specification of parallel processes based on states and transitions.

The ASM signature $\mathbb{S}$ is a finite collection of function names.

– Each function name f has an arity, a non-negative integer.
– Nullary function names are called constants.
– Function names can be static or dynamic.
– Every ASM signature contains the static constants *undef, true, false.*

A database schema may be based on a collection of (predicative) functions representing the structures of the database. Typically such functions are dynamic. Static functions are those functions that do not change over time. Most generic database computation functions such as the aggregation functions can considered to be static. They are defined as static higher-order functions. Their concretisation to database functions may lead to a dynamic functions or may still be static. Database state functions are however dynamic.

The signature $\mathbb{S}$ is the main component of alphabet for the logical language $\mathcal{L}_{\mathbb{S}}$. We assume that this language is specified in the canonical way used in predicate logics. An ASM database schema $\mathcal{S}$ is given by a signature $\mathbb{S}$ and by a finite set Σ of formulas from $\mathcal{L}_{\mathbb{S}}$.

We restrict the consideration in this paper to the tuple (or product) constructor. The set, list, multiset, disjoint union, labelling and naming constructors can be treated in a similar way.

A database state $\mathcal{D}B$ (or database instance) for the signature $\mathbb{S}$ is a non-empty set Val called the superuniverse of $\mathcal{D}B$, together with an interpretation $f^{\mathcal{D}B}$ of each function name f in $\mathbb{S}$.

– If f is an n-ary function name of $\mathbb{S}$, then $f^{\mathcal{D}B} : Val^n \to Val$.
– If c is a constant of $\mathbb{S}$, then $c^{\mathcal{D}B} \in Val$.
– The superuniverse Val of the state $\mathcal{D}B$ is denoted by $Val(\mathcal{D}B)$.

Relations are functions that have the value *true, false, undef*

$$(\bar{a} \in R \quad \text{iff} \quad R(\bar{a}) = true).$$
$$dom(f^{\mathcal{D}B}) = \{(a_1, ..., a_n) \in Val(\mathcal{D}B)^n \mid f^{\mathcal{D}B}(a_1, ..., a_n) \neq undef\}$$
$$rel(f^{\mathcal{D}B}) = \{(a_1, ..., a_n) \in Val(\mathcal{D}B)^n \mid f^{\mathcal{D}B}(a_1, ..., a_n) = true\}$$

The superuniverse can be divided into subuniverses represented by unary relations. These unary relations form the basic data types under consideration. A data type is typically given by a set of values and a set of (static) functions defined over these values.

A database state $\mathcal{D}B$ is a model of Σ ($\mathcal{D}B \models \Sigma$) (or is called consistent database state) if $[\![\Sigma]\!]_{\zeta}^{\mathcal{D}B} = true$ for all variable assignments ζ for Σ. We typically consider only models. Since transactions may have intermediate database states that are not models we distinguish the two notions. The distinction between *undef* and *false* allows to separate for relational functions the case that an object does not belong to the database from the case that an object is known to be false.

2.2 Specification of Database Operating

Database state modifications can be described as changes to the dynamic functions. Changes we need to consider are either assigning a value *undef, true, false* to a relational function or changes of the functions themselves. We thus may consider that a change can be given through a set of changes to the functions for the values of the domain of a function. This detailed view on the content of the database allows the introduction of database dynamics. Roughly speaking, we introduce memory cell abstractions (called locations) and consider the database to consist of those objects which locations evaluate to *true*. The functions of the database system can also be treated on the basis of locations.

A database location of $\mathcal{D}B$ is a pair

$$l = (f, (a_1, ..., a_n)) \qquad \text{((relational) function, object).}$$

The value $\mathcal{D}B(l) = f^{\mathcal{D}B}(a_1, ..., a_n)$ is the content of the location l in $\mathcal{D}B$.

The tuple $(a_1, ..., a_n)$ represents a potential object. The current value is given by a location $\mathcal{D}B(l)$ of the database system. The database consists of those objects $(a_1, ..., a_n)$ for which a relational function $f(a_1, ..., a_n)$ evaluates to *true*.

An modification (l, u) to $\mathcal{D}B$ is the assignment of a value u to a location l. For instance, an insert changes the value $\mathcal{D}B(l)$ at a location l to *true*. The delete operation changes the value at a location l to *undef*. A basic database update assigns v to another location and changes the value of original location to *undef*. A modification is called *trivial* if $v = \mathcal{D}B(l)$. A *modification set* consists of a set of modifications.

A modification set U is *consistent*, if it has no clashing modifications, i.e., if for any location l and all elements v, w if $(l, v), (l, w) \in U$ then $v = w$.

The result of firing a consistent modification set U is a new database state $\mathcal{D}B + U$

$$(\mathcal{D}B + U)(l) = \begin{cases} v & \text{if } (l, v) \in U \\ \mathcal{D}B(l) & \text{if there is no } v \text{ with } (l, v) \in U \end{cases}$$

for all l of $\mathcal{D}B$.

For value-based types we define generic functions beyond *insert, delete, update* and evaluation functions such as *average, min, max, sum*. Additional functions $(F_f)_{f \in \mathcal{F}}$ are defined in the same manner on the type system. These functions can be used to construct a $\mathbb{S}$-algebra (e.g., operations such as *projection, join, selection, union, difference, intersection*). We combine these functions into an $\mathbb{S}$-algebra.

Databases change over time, We may represent the change history of a database by a sequence of database states. The changes are defined by an application of database systems operations to the current database state. We may require that any change applied to a model shall either transform the model to a new model or shall not be considered otherwise. This operating requirement leads to *transaction semantics* of database operating. We can use the logic $\mathcal{L}_{\mathbb{S}}$ for the definition of transition constraints.

A transition constraint consists of a pair of formulas $(\psi_{pre}, \psi_{post})$ from $\mathcal{L}_{\mathbb{S}}$. The transition constraint is valid for a database modification from $\mathcal{DB}$ to $\mathcal{DB}'$ if $\mathcal{DB} \models \psi_{pre}$ and $\mathcal{DB}' \models \psi_{post}$. Static integrity constraints from a finite set Σ can be mapped to transition constraints $(\bigwedge_{\alpha \in \Sigma} \alpha, \bigwedge_{\alpha \in \Sigma} \alpha)$. Let $\Sigma_{dynamic}$ be the set of transition constraints.

A sequence of database states $\mathcal{DB}_0, \mathcal{DB}_1, ..., \mathcal{DB}_{i+1} = \tau_A(\mathcal{DB}_i), ...$ satisfying $\Sigma_{dynamic}$ is called a *run* of the database system. The run can be defined though $\mathbb{S}$-algorithms A or transformation rules that impose a one-step transformation $\tau_A^{\mathcal{DB}}$ of a database state to its successor in the run.

A database program is given by a rule name r of arity n is an expression $r(x_1, ..., x_n) = P$ where P is a database transition operation and the free variables of P are contained in the list $x_1, ..., x_n$.

Database transition operations are either basic operations from the $\mathbb{S}$-algebra or are constructed by inductively applying the following construction rules:

– Skip rule:	`skip`
– Update rule:	$f(s_1, ..., s_n) := t$
– Parallel execution rule:	P `par` Q
– Conditional rule:	`if` ϕ `then` P `else` Q
– Let rule:	`let` $x = t$ `in` P
– For all rule:	`forall` x `with` ϕ `do` P
– Choose rule:	`choose` x `with` ϕ `do` P
– Sequence rule:	P `seq` Q
– Call rule:	$r(t_1, ..., t_n)$

A database system consists of a database management system and of a number of databases. Let us consider only one database.

An abstract database system $\mathcal{M}$ consists of a

- a signature $\mathbb{S}_{\mathcal{M}}$ of the database system that embodies the signature $\mathbb{S}$ of the database,
- a set of initial states for $\mathbb{S}_{\mathcal{M}}$,
- a set of database programs,
- a distinguished program of arity zero called *main program* of the system.

We denote the current state of the database system by $\mathcal{DBS}$ and by $\mathcal{DB}$ the current state of the database. The transition operation P yields the modification set U in a state $\mathcal{DBS}$ under the variable assignment ζ: `+yields+(P,`$\mathcal{DBS}$`,`ζ`,U)`.

Semantics of transition operations defined in a calculus by rules:

$$\frac{\text{Premise}_1, ..., \text{Premise}_n}{\text{Conclusion}} \text{ Condition}$$

A query is an (open) $\mathbb{S}$-formula. Moreover, a view is a query adorned by names (labels) for free variables. Due to our definitions, views are hierarchical too.

2.3 ASM Specification of Database System Behaviour

The database system state $\mathcal{DBS}$ is structured into four state spaces:

(input states $\mathcal{I}$, output states $\mathcal{O}$, DBMS states $\mathcal{E}$, database states $\mathcal{DB}$).

The input states $\mathcal{I}$ accommodate the database input to the database systems, i.e. queries and data. The output space $\mathcal{O}$ allows to model the output data and error messages. The internal state space $\mathcal{E}$ of the DBMS represents the DBMS states. The database content of the database system is represented in the database states $\mathcal{DB}$. The four state spaces are typically structured. This structuring is reflected in all four state spaces. For instance, if the database states are structured by a database schema then the input states can be structured accordingly.

The main database functions are modelled by modification, retrieval or internal control programs that are run in parallel:

Modification programs allow to modify the database state if is enabled:

```
if $\mathcal{I}(req) \neq \lambda \wedge \mathcal{E}(modify) = enabled \wedge \mathcal{I}(req) \in Update$
    then $\mathcal{I}(req) := \lambda$ , $\mathcal{O}(errMsg) := ...$, $\mathcal{D} := ...$ , $\mathcal{E} := ...$
```

Retrieval rules allow to retrieve the content of the database:

```
if $\mathcal{I}(req) \neq \lambda \wedge \mathcal{E}(retrieve) = enabled \wedge \mathcal{I}(req) \in Update$
    then $\mathcal{I}(req) := \lambda$ , $\mathcal{O}(errMsg) := ...$, $\mathcal{O}(answer) := ...$
```

DBMS control rules allow to change the database state and the internal state of the database:

```
if $\mathcal{E}(DBMSstateChange)$ and $\mathcal{E}(modify) = disabled$
    then $\mathcal{D} := ...$, $\mathcal{E} := ...$
```

The description can be extended in a similar fashion to sets of inputs instead of a singleton input.

The ASM programs are based on ASM rules
ModifyInput(request,DBMS_state, DB_state),
ModifyOutput(request,DBMS_state, DB_state),
ModifyDB(request,DBMS_state, DB_state),
ModifyControl(request,DBMS_state, DB_state),
RetrieveOutput(request,DBMS_state, DB_state),
RetrieveDB(request,DBMS_state, DB_state),
RetrieveControl(request,DBMS_state, DB_state),
ControllerDBMS(DBMS_state, DB_state), and
ControllerDB(DBMS_state, DB_state).

These rules run in parallel. They express general DBMS functions for state modification:

`modify :`	*(req,_,s,d)*	$\mapsto$	*(_,errMsg,s',d')*
`retrieve :`	*(req,_,s,d)*	$\mapsto$	*(_,answ$\cup$errMsg,s,d)*
`controller :`	*(_,_,s,d)*	$\mapsto$	*(_,_,s',d')*

Modifications may cause deadlock. In order to overcome them the controller enables scheduling, recovery, and optimization. An modification can be imposed completely to the database for support of transaction semantics.

Summarizing we observe that ASM's can be used for definition of operational semantics of database systems.

2.4 Co-design of Structuring, Interaction and Behavior

Information systems design starts with database structuring and bases functionality on structuring. Typically, it uses various abstraction layers [13]: application domain layer for a rough specification of the application domain and its influence on the information system, requirements acquisition layer for the description of requirements such as main business data and business processes, business user layer or the predesign layer [5,13] for the specification of business user objects and business workflows, conceptual layer for the conceptual description of structuring and functionality and implementation layer describing the implementation, i.e. code, SQL structures, interfaces and user views. Nowadays application tend to be distributed and components collaborate with each other [10]. Information systems provide services to users depending on their application tasks, portfolio, user stories and context [11,12]. Therefore, we represent the four dimensions of information systems specification and their layers in Figure 2.

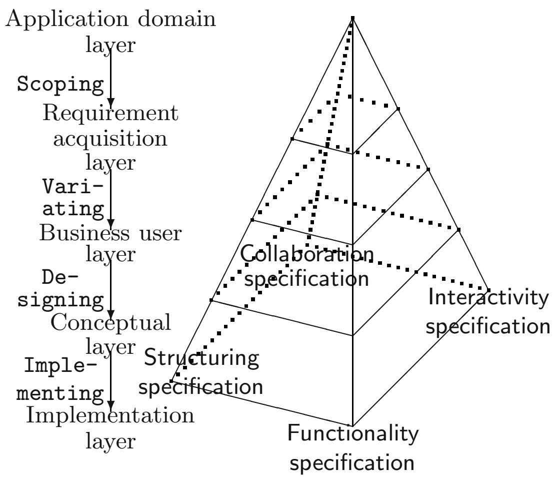

Application domain layer
Scoping
Requirement acquisition layer
Vari-ating
Business user layer
De-signing
Conceptual layer
Imple-menting
Implementation layer
Collaboration specification
Interactivity specification
Structuring specification
Functionality specification

Fig. 2. The abstraction layer model for information systems specification

The structuring and functionality can be described as an abstract state machine. The stories of users, their profile, context and portfolio can be combined into an ASM-based description of the story space [2]. Interaction description can be based on notions developed in linguistics and in movie business [9].

According to the co-design approach we can distinguish two levels of consideration:

Database system development layers: The specification of database systems can be of different granularity and levels of detail. Typical development layers are motivation and requirements layers. A development layer (which is at the same time a database system run layer) is the conceptual layer. These layers display a database specification on different levels of detail and should be refinements of each other.

Database systems operating layers: Database operating is observed, maintained and administrated at various levels of detail: Business users have their own view onto the database application. These views may be different from each other. They are considered to be external views on the conceptual layer. The conceptual layer integrates all different aspects of database systems specification. The implementation layer is based on the logical and physical database schemata. Additionally, database systems functionality is added to the application on the basis of programs etc.

The ASM specification should match all different layers:

Level 1 (Point of view of business users)**:** Database system defined by three state space: input state, database state, output state. The *input state* is based on algebraic structure with ground terms defined on the values and the names. The *database state* is based on the (object-)relational structure with well-defined composition operators. The *output state* is a general database defined on the values and names.

Level 2 (Conceptual point of view)**:** Database systems are defined as an extension of level 1 by transactions, constraints, views and integrity maintenance.

Level 3-1 (Logical point of view)**:** The logical database system defined as an extension of level 2 by states of the database management system by the transaction and recovery engine, by the synchronization engine, the logging engine and the query translating engine.

Level 3-2 (Physical point of view)**:** The physical database system is defined as an extension of level 3-1 by specific functions of the DBMS.

Level 4 (DBMS point of view)**:** On level 4, the storage engine is modelled in detail with the buffers and the access engine.

This separation allows to model database systems at different levels of detail. Each higher level should be a refinement of the lower levels.

3 Faithful Refinement of Database Systems Specification

The co-design approach can be based on the abstraction layer model. We need a correctness criterion for the faithfulness of specifications among these different layers. We may distinguish between specifications at the requirements acquisition layer, at the business user layer and at the conceptual layer. (Static) integrity constraints typically limit the state space for the instances of database schema.

Functionality can be provided within a framework for specifying the functions. The database system development process aims in stepwise refinement of structuring and functionality of a database system. This refinement process needs a formal underpinning. We can develop a theory of specification refinement for database applications based on ASM.

3.1 Refinements of Database Systems

Given two abstract database systems $\mathcal{M}$ and $\mathcal{M}^*$. The refinement of $\mathcal{M}_{\mathcal{D}B}$ to $\mathcal{M}^*$ is based on

a refinement of signatures $\mathbb{S}_{\mathcal{M}}$ **to** $\mathbb{S}_{\mathcal{M}^*}$ that associates functions and values of $\mathbb{S}_{\mathcal{M}}$ with those on $\mathbb{S}_{\mathcal{M}^*}$

a states of interest correspondence between the states of interest $\mathcal{D}BS$ and $\mathcal{D}BS^*$ defined over $\mathbb{S}_{\mathcal{M}}$ and $\mathbb{S}_{\mathcal{M}^*}$ correspondingly,

abstract computation segments $\mathcal{D}BS_1, ..., \mathcal{D}BS_m$ on $\mathcal{M}$ and $\mathcal{D}BS^*_1, ..., \mathcal{D}BS^*_n$ on $\mathcal{M}$ and $\mathcal{M}^*$,

locations of interest $(\mathcal{D}B, \mathcal{D}B^*)$ defined on $\mathbb{S} \times \mathbb{S}^*$ and

an equivalence relation $\equiv$ on locations of interest.

$\mathcal{M}^*$ is a correct refinement of $\mathcal{M}$ if there for each $\mathcal{M}$-run $\mathcal{D}BS^*_0, ..., \mathcal{D}BS^*_k, ...$ there is an $\mathcal{M}$-run and sequences $i_0 < i_1 < ....$ and $j_0 < j_1 < ...$ such that $i_0 = j_0 = 0$ $\mathcal{D}B_{i_k} \equiv \mathcal{D}B^*_{j_k}$ for each k and either

- both runs terminate and their final states are the last pair of equivalent states, or
- both runs and both sequences are infinite.

A refinement is called complete refinement if $\mathcal{M}_{\mathcal{D}B}$ is a correct refinement of $\mathcal{M}^*_{\mathcal{D}B^*}$ and $\mathcal{M}^*_{\mathcal{D}B^*}$ is a correct refinement of $\mathcal{M}_{\mathcal{D}B}$.

We distinguish between

- structure refinement that is based on a notion of schema equivalence and state equivalence,
- functionality refinement that is based on a notion of schema equivalence and a notion of coherence, and
- state refinement that uses the notion of equivalence and coherence.

3.2 Deriving Plans and Primitives for Refinement

The perspectives and styles of modelling rule the kind of refinement styles. As an example we consider structure-oriented strategies of development:

Inside-out refinement: Inside-out refinement uses the current ASM machine for extending it by additional part. These parts are hocked onto the current specification without changing it.

Top-down refinement: Top-down refinement uses decomposition of functions in the vocabulary and refinement of rules. Additionally, the ASM may be extended by functions and rules that are not yet considered.

Bottom-up refinement: Bottom-up refinement uses composition and generalisation of functions and of rules to more general or complex. Bottom-up refinement also uses generation of new functions and rules that are not yet considered.

Modular refinement: Modular refinement is based on parqueting of applications and separation of concern. Refinement is only applied to one module and does not affect others. Modules may also be decomposed.

Mixed skeleton-driven refinement: Mixed refinement is a combination of refinement techniques. It uses a skeleton of the application or a draft of the architecture. This draft is used for deriving plans for refinement. Each component or module is developed on its own based on top-down or bottom-up refinement.

These different kinds of refinement styles allow to derive *plans* for refinement and and *primitives* for refinement.

3.3 Generic Refinement Steps and Their Correctness

An engineering approach is based on a general methodology, operations for specification evolution, and a specification of restrictions to the modelling itself. Each evolution step must either be correct according to some correctness criterion or must lead to obligations that can be used for later correction of the specification. The correctness of a refinement step is defined in terms of two given ASM together with the equivalence relations. Already [8] has observed that refinement steps can be governed by contracts. We may consider a number of governments [1]. We should however take into account the choices for style and perspectives.

Given a refinement pattern, perspectives, styles and contract, we may derive generic refinement steps such as data refinement, purely incremental refinement,

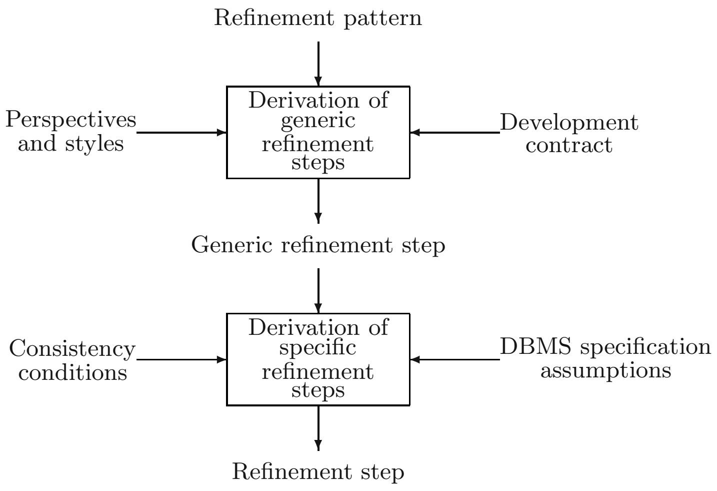


Fig. 3. The derivation of correct refinement steps

submachine refinement, and (m,n) refinement. The generic refinement is adapted to the assumptions made for the given application and to consistency conditions. Typical such consistency are binding conditions of rules to state and vocabulary through the scope of rules. The general approach is depicted in Figure 3.

4 Conclusion

The ASM approach allows the development of a theory of database system operating. It has the following advantages:

– The signature of object-relational databases can be specified based on hierarchical predicate logic. Therefore, the theory of static integrity constraints can be entirely embedded into the theory of sequential abstract-state machines [4].
– The run of a database system is parallel. As an example that is not covered in this paper we elaborated transaction semantics that handles conflicts in concurrency. Parallel runs of transactions must behave in the same manner as a sequential run of the transactions. This concept is entirely covered by partially ordered runs.
– Interaction of database systems with user or information systems can be handled by ASM as discussed above.
– Database systems are considered on a variety of abstractions layers. Users consider a database system as an input-output engine with a powerful memory. At the conceptual layer, a database system is far more complex. The implementation is even more complex. Therefore, the consideration of database systems semantics requires also a powerful theory of refinement as provided by ASM.

Our main aim in this paper is to develop a theory of database systems at the user layer. The other abstraction layers may be seen as refinements of the business user layer. We omitted all examples to the space limitations.

The next abstraction level must consider specifical approaches of database engines at the implementation layer. In this case, we need a general notion of database semantics. Database dynamics has not yet been well understood. Transactions may serve as an example. So far we have only used sequential ASM. We might however use power ASM that allow to abstract from intermediate computational steps.

Acknowledgement

The authors would like to thank Egon Börger and to Andreas Prinz for their helpful comments and discussions. Egon Börger proposed to use power ASM instead of sequential ASM for the treatment of transactions. We than P. Schmidt for proofreading.

References

1. Bienemann, A., Schewe, K.-D., Thalheim, B.: Towards a theory of genericity based on government and binding. In: Embley, D.W., Olivé, A., Ram, S. (eds.) ER 2006. LNCS, vol. 4215, pp. 311–324. Springer, Heidelberg (2006)
2. Binemann-Zdanowicz, A., Thalheim, B.: Modeling of information services on the basis of ASM semantics. In: Börger, E., Gargantini, A., Riccobene, E. (eds.) ASM 2003. LNCS, vol. 2589, pp. 408–410. Springer, Heidelberg (2003)
3. Börger, E.: Discrete systems modeling. In: Encyclopedia of Physical Sciences and Technology, Academic Press, London (2001)
4. Börger, E., Stärk, R.: Abstract state machines - A method for high-level system design and analysis. Springer, Berlin (2003)
5. Fliedl, G., Kop, C., Mayerthaler, W., Mayr, H.C.: NTS gestützte Vorentwurfsschemaeinträge aus natürlichsprachlichen Anforderungsdefinitionen. In: Natürlichsprachlicher Entwurf von Informationssystemen, Konstanz, pp. 260–279. Universitätsverlag Konstanz (1996)
6. Goldin, D., Srinivasa, S., Thalheim, B.: IS = DBS + interaction - towards principles of information systems. In: Laender, A.H.F., Liddle, S.W., Storey, V.C. (eds.) ER 2000. LNCS, vol. 1920, pp. 140–153. Springer, Heidelberg (2000)
7. Lewerenz, J.: Human-computer interaction in heterogeneous and dynamic environments: A framework for its conceptual modelling and automatic customization. PhD thesis, Brandenburg University of Technology at Cottbus, Faculty Mathematics, Natural Sciences and Computer Science (2000)
8. Schellhorn, G.: ASM refinement and generalizations of forward simulation in data refinement: A comparison. Theor. Comput. Sci. 336(2-3), 403–435 (2005)
9. Schewe, K.-D., Thalheim, B.: Conceptual modelling of web information systems. Data and Knowledge Engineering 54, 147–188 (2005)
10. Schewe, K.-D., Thalheim, B.: Component-driven engineering of database applications. In: APCCM 2006, vol. 49, pp. 105–114 (2006)
11. Schewe, K.-D., Thalheim, B.: Design and Development of Web Information Systems. Springer, Heidelberg (forthcoming, 2008)
12. Schewe, K.D., Thalheim, B.: Web information systems: Usage, content, and functionality modelling. Technical Report 2004-3, Christian Albrechts University Kiel, Institute of Computer Science and Applied Mathematics, Kiel (2004)
13. Thalheim, B.: Entity-relationship modeling – Foundations of database technology. Springer, Berlin (2000)
14. Thalheim, B.: Readings in fundamentals of interaction in information systems. Binder, C., Clauß, W., Düsterhöft, A., Feyer, T., Gutacker, T., Heinze, B., Lewerenz, J., Roll, M., Schewe, B., Schewe, K.-D., Seelig, K., Srinivasa, S., Thalheim B.: BTU-Cottbus (reprint, 2000), `http://www.is.informatik.uni-kiel.de/~thalheim`
15. Wegner, P., Goldin, D.: Interaction as a framework for modeling. In: Chen, P.P., Akoka, J., Kangassalu, H., Thalheim, B. (eds.) Conceptual Modeling. LNCS, vol. 1565, pp. 243–257. Springer, Heidelberg (1999)
16. Zimmermann, W., Thalheim, B.: Preface. In: Zimmermann, W., Thalheim, B. (eds.) ASM 2004. LNCS, vol. 3052, pp. V–VII. Springer, Heidelberg (2004)

Software Quality – Experiences in System Development and Operation

Helmut Thoma

Helmut Thoma, Software Engineering and IT Consultancy,
Postbox 255, CH-4008 Basel, Switzerland

Abstract. Described are findings for the software quality together with their evidences, resulting from observations in practice and experience: The board of managers and not software engineering exponents should be responsible for activities, concerned with software engineering and software quality activities. Basics of methodical process skills are necessary to implement quality processes in practice. Software development processes have to be intensively educated and trained. Requirements engineering is a crucial factor in software development, requirements should be managed. To conserve a high quality standard all along is not easy, it is difficult. Software engineering activities are not only a business or a technical matter; they are also a challenge for the industrial psychology and applied economics.

Keywords: Software quality, software development, software maintenance, software process, methodical skills, requirements engineering, quality improvement, quality assurance, incentive system, software process research.

1 Introduction

This paper describes findings and conclusions for the software quality together with their evidences and stories, resulting from observations in practice and experience, referring to the last thirty years viewed from the position of a Software Engineer.

In nearly thirty years working as a Software Engineer, IT Consultant, and IT Architect in industry and IT services, I got a wide experience in designing, developing and maintaining of information systems as well as in supporting such activities. During projects to be a member of a Software Engineering Process Group, I got also deep insight into working processes of other software development projects, supporting and preparing software appraisals on behalf of IT organizations and IT projects.

Nearly at the same time, I had the opportunity to be integrated in a network of colleagues in IT associations, special interest groups, and IT conferences. I often held the position of an executive member. Moreover, I evaluate publications and projects in topics of applied computer science and business information technology. Since more than twenty years, I am a University Lecturer in business information technology. Thanks to these teaching and scientific activities, I am able to compare concepts used in my own practice with some concepts of science and technology being state-of-the-art.

This invited talk does not report results from scientific research projects. It fulfills the function of an analysis report and it is ought to give some remarks and informations regarding projects in applied research and in practice in the field of software quality and software engineering – always related to application software for information systems.

2 Observations and Conclusions

In the following, I delineate some aspects of my observations, findings, and conclusions regarding the software quality in system development and operation: Organization, process consultancy, methodical skills, education and training, requirements engineering, quality improvement, communication system, quality assurance, incentive system, research, and maintenance.

2.1 Organization

Findings and Conclusions

If a software development organization appoints a person or a group, concerned with software engineering activities and software quality improvement or assurance, don't make this person or group responsible for software quality. Always the board of managers should be responsible for such activities.

Story and Evidence

One member of staff is not able to support a few tens development projects, and a few trained software method exponents are insufficient, if the developer community was not committed to attend education or training courses and if managers of project and development groups don't understand, that software quality is one of the most crucial dimensions in a development project. Customers are sensitized to software quality.

Another story illuminates my experience during the preparation of a development organization for a Class A appraisal of CMMI Maturity Level 5 (ML 5), [1]. The organization received a CMM M L 3 some years before. An engineering process group got the mandate to prepare the organization for a Class B appraisal, afterwards for a Class A appraisal. Before the Class B appraisal, the engineering process group worked independent from the management of the development organization. Outcome of the Class B appraisal: The different development project groups worked independently of each other according a culture "we have experience and we know our customer". The customers were satisfied with the quality, but regarding the CMMI appraisal, the development organization was fallen back more or less from ML 3 to ML 1.

A reorganization of quality-project's processes was successful. We organized some process improvement sub-projects, staffed with colleagues from development projects and their managers and led by members of the engineering process group. Outcome: Less than one year later, the development organization got CMMI ML 5, prepared with the same members of the engineering process group preparing the appraisal, and led with the same lead appraiser as in Class B appraisal. Development's organization culture changed to "we share our experience and improve our processes". Accompanied was the transition (the phase after Class B appraisal) by an extensive and intensive training initiative for the developer community.

2.2 Process Consultancy

Findings and Conclusions
The set up of a process consultancy for software quality improvement and assurance is necessary, but it is not sufficient.

Story and Evidence
Some evidences are described in other chapters of this paper.

2.3 Methodical Skills

Findings and Conclusions
Many peoples with studies in IT, other colleagues or career changer often have a good product-specific education and training. Basics of methodical process skills for software development are not widely spread. But the basics are necessary to implement quality processes in practice.

Story and Evidence
To train software engineering products is very effective and efficient, done in practice. To teach basics should be effected in IT education of colleges and academies. My own experience reflects me, it is better to teach this content of the curriculum together with problems and solutions in workshops and tutorials – preferable with some practical application – than in lectures. To establish an awareness of the problems is fundamental.

2.4 Education and Training

Findings and Conclusions
Software development processes have to be intensively educated and trained. The encouragement, the support, and the sponsorship of the executive board and the management body are of utmost importance and should be mandatory. Education and training are necessary, but they are not sufficient to earn a mature organization for operating according the sate-of-the-art of software engineering. If you strive for a maturity level in CMMI, you have to enforce "organizational training" for the whole system development organization.

Story and Evidence
In a big IT system development organization, on one's own account, it was created a manual for system development standards as the result of a project. One member of staff has to support a few tens development projects and to collect conceptual models. Another big organization was engaged to use a worldwide established development method. Method exponents were trained, but the developer community was not committed to attend education or training courses. Success in both organizations depended on the competence and heroics of the people in the organization and not on the use of proven processes. Often the organizations produced products and services that worked, however they frequently exceeded the budget and the schedule of their projects.

During the preparation of a development organization for a Class A appraisal of CMMI ML 5, the software process engineering group arranged an extensive and intensive training initiative for the developer community, accompanied by a modification of the quality-project's processes. Outcome: The organization culture changed. There was a transition from "we have experience and we know our customer" working processes to "we share our experience and improve our processes". One reason of the change was the organization-wide training of the developer community.

Introducing processes in compliance with a development methodology (e.g. CMMI), often an organization has to create and to introduce organizational process assets. They support e.g. the developers to be in accordance with the methodology: Process assets escort developers from "what to do" of the methodology via "how to do" in the process asset to "do it". The application of the process assets must be trained.

2.5 Requirements Engineering

Findings and Conclusions

Requirements engineering is one of the crucial factors in software development. Both business requirements and system requirements are fundamental for the success of developing a system. Requirements should be managed; starting out with the kickoff of a development project, and during the software is alive. The vital aspect: No system or component requirement should be elaborated without a correspondent business requirement and vice versa.

Story and Evidence

For the corporate safety and ecological data representation of a big group of chemical industry towards public authorities (ecological authorities, customs, safety and hazard authorities, service providers for transportation via air, sea, road, river, medical emergency services etc.), an information system had to be developed. This raw data were ascertained by different business divisions (creating pharmaceutics products, agricultural products, dyestuffs, plastics, etc.). Each division had one's one information system with one's one data base structure. The raw data had to be interchanged between business divisions, and specific data had to be attached from experts for classification data regarding transport type, customs etc. Therefore the corporate safety and ecological information system has to be implemented. To find a common business model, a common database structure representing the corporate used data etc., comprehensive effort to develop business and system requirements was required. Outcomes: Development time approx. 5 years, approx. EUR 2.3 million development cost, high user satisfaction, since 10 years in worldwide use with approx. 2000 users.

On the other hand, there is an example for the development of an information system for news. The news has to be published with different media. An important section of the news is sport data. Structured sport data are stored and maintained in a Relational Database System, unstructured news data like reports and comments etc. are managed in a Content Management System. For the use of the news system, end-user has to distinguish between different activities: Data are collected and inserted; they have to be retrieved and/or modified and deleted by editorial staff. Another activity is to produce output for the TV, cell phones, Fax devices etc. Real-time data feeding from sporting grounds or from news agencies is an alternative solution for news input, if available.

Important are the business processes, particularly with regard to the history: Before working with an information system supported by databases, editorial staff used the data recording system of the correspondent emitter or broadcast system; no data were stored. The executive board of the user did not like to review the current used processes – not yet existing in written form – and to design new business processes, harmonized with the new database supported system. Consequently only technical system requirements could be reviewed and implemented. Outcomes: Satisfied members of the editorial staff with the results of the phase "sport data", completely dissatisfied executive board of the customer by reason of extensive exceeded budget and schedule. The executive board discussed the additional phase "unstructured data" with the service provider. It would still avoid the development of any business requirements; the service provider would not go on working without business requirements; the customer changed the service provider.

Let us consider the organization of the development department for a Class A appraisal of CMMI ML 5. This organization applies an exemplary requirement management. Helpful are the close collaboration from end users and service providers as well as clearly defined reviews after defined baselines or project phases. This organization uses a review board staffed with the project management of end user department and development department. Reviews decide about "start the next phase or rework the focused phase". These reviews occur at the end of the concept phase (solution outline), the plan phase (macro design), and before the rollout (deployment). Systems Engineering and Architecture, a software engineering initiative of IBM defines reviews after well defined baselines: Business requirements review after customer baseline (solution startup), system requirements review after system baseline (solution outline), preliminary design review after component / application baseline (macro design), critical design review after design baseline (micro design), all of them led by a systems engineer, test readiness review in the build cycle, led by the test team, and a production readiness review starting before deployment and led by the project manager.

Requirements engineering – in particular requirements management – makes a important contribution to the creation and the management of new releases. From this point of view, requirements management also plays a role in preparation for the preventive maintenance. During the execution of requirements management discussions and reviews after defined project phases, new business and system requirements may be created: The maintenance of software can together with business transformation and with new findings in information-techniques feed to new development activities.

2.6 Quality Improvement

Findings and Conclusions

For what reason, a software development or maintenance organization – a service provider – wants to initiate any quality improvement procedures? I see primarily three motivators: The customer, the market or a deeper insight, that to improve quality is cheaper than to go on with the current used mode of operation. It is not easy to win the executive board to initiate activities of quality improvement; today all companies operate under cost, time, and work pressure.

Story and Evidence
Basically, it is sometimes difficult for developers to argue the management of a project or a department into quality improvements. If a customer or a community of potential customers – a market sector – of a service provider demand the confirmation that a service will be created or has been created using a defined software development process or using a defined development methodology (e.g. CMMI) or a defined IT service management (e.g. ITIL, [2]), this challenge is easy to perform. Otherwise, developers often have to argue that changing the current procedure to a well-defined development process is cheaper than to continue with the approach at present – usually being ad hoc and chaotic.

2.7 Communication System

Findings and Conclusions
If a company is changing to a new software development process, the change should be escorted by communication activities. The employees should be well-informed repeatedly about the current state of the implementation, the next steps, the individual and team objectives, the agenda, etc. They have to become acquainted with the new process.

Story and Evidence
Well-informed members of staff are better motivated and more willing to accept new procedures doing their duties and responsibilities and to complete orders. In distributed work environments we made positive experience with teleconferencing to acquaint colleagues with new facts, to discuss new working processes with the intended purpose for education and training; we preferred to arrange peer-to-peer discussions. Important topics are process assets and their utilization.

We made positive experience with a modern approach of knowledge management: Story telling [3]. With the method of exchanging positive and negative experiences from projects – converted to good stories, intuition and impressions can be communicated. Video clips integrated in slides of a presentation or slides presenting a joke can also initiate a person's associativity.

2.8 Quality Assurance

Findings and Conclusions
To conserve a high quality standard all along is not easy, it is difficult.

Story and Evidence
A quality-conscious organization – fit for software quality too; a process consultancy – qualified to support; a software engineering team – provided with methodical skills, well educated and trained; a requirements engineering – implemented for the purpose of business and system requirements as well as for requirements management; a quality improvement initiative – which has been effective carried out; a communication system – well implemented and being en vogue in the developer community: These are prerequisites for a effective quality assurance.

There are some other tools to perform quality assurance. One of them are some process areas of CMMI, being engaged in process measurement and process performance, in collaboration with organizational innovation and deployment Another approach is the implementation of reviews after defined baselines or project phases, discussed already in the section "requirements engineering". A third approach is the periodical iteration of maturity level appraisals; since version 1.2 of CMMI, a Class A appraisal must be re-done each three years. Otherwise an organization's achieved maturity level gets lost.

2.9 Incentive System

Findings and Conclusions
It seems to me that current incentive systems applied to evaluate the contribution of managers to the operating success of a company or organizations don't encourages the quality thinking and a faultless process implementation. In addition some software developers feel a process controlled approach as a threat for their creativity.

Story and Evidence
Today's evaluations of the contribution of manager's success tend to focus the rapid percentage increase of earnings, sales volume, transaction volume, share earnings, etc. It seems to me there is only a marginal space for long-term objectives, required to initiate quality improvement and assurance activities. I think software engineering activities are not only a business or a technical matter; they are also a challenge for the industrial psychology and applied economics.

Techniques and humans act together in the process of designing, implementing and operating the software product. I bear the two factor theory in mind, the motivation-hygiene theory [4, 5], developed by Frederick Herzberg, a psychologist who found that job satisfaction and job dissatisfaction acted independently of each other: The two factor theory distinguishes between motivators (which give positive satisfaction, arising from intrinsic conditions of the job itself, such as recognition, achievement, or personal growth), and hygiene factors (which do not give positive satisfaction, although dissatisfaction results from their absence). Hygiene factors are extrinsic to the work itself, and include aspects such as company policies, supervisory practices, or wages/salary.

2.10 Research

Findings and Conclusions
We must not forget the scope of software engineering and software quality is the appliance in practical experience. In this spirit findings from research projects should be intended for transformation into quality assurance and the improvement of the software development and maintenance processes. Research projects for software quality and quality assurance should be subject of engineers, business economists, and industrial psychologists.

Story and Evidence
The capacity to understand that to increase the productivity (including Herzberg's two factor theory) and to improve the software quality goes hand in hand, gives some managers a lot of trouble. Fortunately there exist examples bringing research and practice together. One example is CMMI: Together with deputies from numerous companies, the Software Engineering Institute (SEI), a US-federally funded research

and development center at the Carnegie Mellon University, developed CMMI as a software engineering methodology of best practices and support IT organizations in their implementation. Another example is "Systems Engineering and Architecture": Created from the "International Council of Systems Engineering" (INCOSE, with membership of technology companies, Stevens Institute of Technology, MIT, SEI of CMU), it is deployed to some IT service provider organizations.

To cite a research at the university level as an example, I adduce projects performed at the University of Klagenfurt [6]. Although these researches on a pre-design level could be very important for IT practice, nearly they don't find ways of doing their implementation in IT practice organizations without sponsoring.

2.11 Maintenance

Findings and Conclusions
Activities for software development and for software maintenance are interdependent.

Story and Evidence
I assume maintenance activities are often based on the IT Infrastructure Library (ITIL), the framework for IT service management, developed by the OGC (Office of Governance Commerce) in Norwich (England) on behalf of the British government. Organizations developing application software should have in mind that maintenance must be done lifelong of an application, that a traceable documentation must be available, no complicated and bad documented dependencies must not constrain to supersede the application. Organizations which operate maintenance processes should be concerned with software development processes and standards: To maintain IT applications can suddenly lead to software development activities.

Acknowledgments. Thanks a lot to Prof. Dr. H.C. Mayr and to Dr. C. Kop for their invitation and support.

Referencess

1. Chrissis, M.B., Konrad, M., Shrum, S.: CMMI. Guidelines for Process Integration and Product Improvement, 2nd edn. Addison-Wesley, Boston (2006)
2. ITIL.org Homepage, `http://www.itil.org/en/index.php`
3. Brown, J.S., Groh, K., Prusak, L., Denning, S.: Storytelling in Organizations. Butterworth-Heinemann (2005)
4. Herzberg, F., Mausner, B., Snyderman, B.B.: The Motivation to Work. John Wiley, New York (1959)
5. Herzberg, F.: One more time: how do you motivate employees? Harvard Business Review 46(1), 53–62 (1968)
6. Fliedl, G., Kop, C., Mayr, H.C., Mayerthaler, W., Winkler, C.: Linguistically based requirements engineering – the NIBA project. Data & Knowledge Engineering 35, 111–120 (2000)

Adaptive Macro-designing Technology for Complex Process Control Systems

Mykola Tkachuk

Computed-Aided Management Systems Department,
National Technical University "Kharkiv Polytechnic Institute",
Frunze str., 21, Kharkiv, Ukraine
tka@kpi.kharkov.ua

Abstract. The new framework for macro-designing and implementation of complex process control systems (PCS) is presented. The core of this approach is integrated multi-dimensional modeling and technological toolkit, which provides adaptive and scalable software solutions with respect to system performance and reliability. This framework was used successfully in real-life PCS-projects performed for some gas-production enterprises in Ukraine.

Keywords: Adaptive software, control system, software cybernetics, information space, fuzzy-logic, performance, simulation.

1 Introduction: Research Actuality and Aims

Nowadays a Web-based designing and implementation approach is recognized for different kinds of information systems, which could be considered as so-called *process control systems* (PCS) operating in remote mode (see, for instance, in [12,13]). Some Web-PCS obligatory features are already well-defined and described for single PCS-configuration: e.g., to display a plant operating in form of graphical (mnemonic) schemes, to provide data trends analyzing, to perform notification of system alarms, etc. But much less attention is paid to such problem as following: what basic principles should be taken into account for large-scale PCS's development, especially with respect to the permanent changes in system requirements (SR), which have to be met during their designing and functioning?

Basing on our own project experience collected, and analyzing some modern trends in software engineering, we have elaborated and would like to present in this paper the *Adaptive Macro-Designing Technology* (AMDT) for multi-level distributed PCS. AMDT is the *inter-disciplinary approach*, which is based on some fusion of software engineering methods and control theory concepts.

The paper is structured as following: Section 2 depicts briefly some modern trends in this research domain, in Section 3 the methodological background and formal definition of AMDT are introduced. Section 4 represents the core of proposed AMDT-concept, which is based on the information meta-space metaphor. The short overview about some data models, design methods and software tools elaborated for AMDT-handling is given in Section 5, and some experimental results are discussed there as well. Finally Section 6 concludes with short outlook on future work to be done within this domain.

2 Adaptive Software Development: Some Modern Trends and Related Works

If we consider some publications about adaptive and reflective software designing methods and technologies, e.g. in [8,1,2], the following "pure" software-centered points can be mentioned:

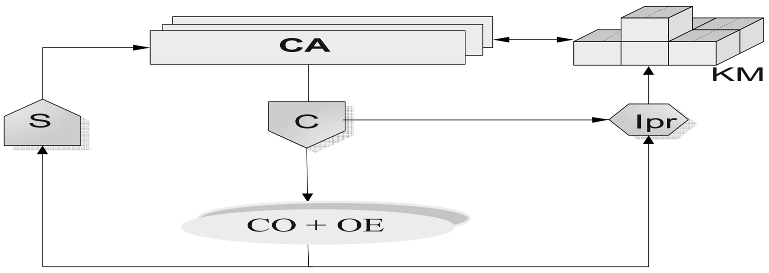
- *compositional* (or structural) *adaptation* enables software to modify its structure and behavior dynamically w.r.t. changes in its execution environment, e.g. the class *java.lang.reflect.proxy* in the standard Java API v.1.3 has been added which can be used to modify the dispatching method mechanism;
- *introspective reflection* provides the system structure can be accessed but not modified: e.g. if we take Java as an example, with its *java.lang.reflect* package, we can get information about classes, objects, methods and fields at runtime mode;
- *linguistic reflection* defines some advanced system's features, as an example, with *OpenJava* reflective language, the own problem-oriented language can be enhanced to be adapted to specific design patterns.

From the other hand, last years in the domain of adaptive software the special attention is paid to some new conceptual approaches, which explore the interplay between software engineering and control theory, also called as "software cybernetics (SC)" (see e.g. in [7]). From our point of view, these research activities had actually impacted by Herring's work [4], where he introduced the general intelligent control paradigm for adaptive and adaptable software systems (or *viable systems*) designing. The principles of SC are applied successfully for modeling and implementation of several kinds of software systems. E.g., in [19] a process to construct adaptation models, automatically generate adaptive programs from these models, and additionally verify the program functions during and after adaptation is proposed. The authors of paper [10] applied the typical mechanism of control theory, namely a Bayesian Belief Network to capture the probabilistic relationships between software architecture elements and decisions to be made in a designing process. In [17] the principles and concepts of SC are used to deploying and scheduling workflows with timing and resource constraints in service-based software systems.

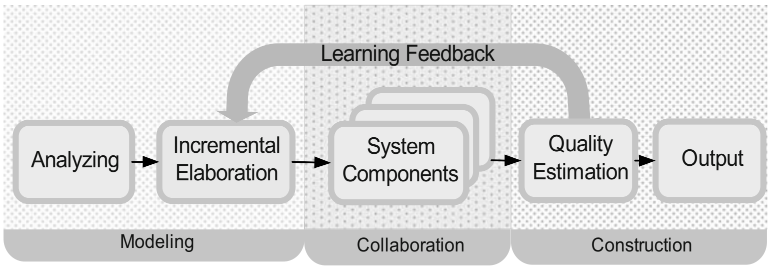
We are deeply convinced that similar interdisciplinary fusion of concepts and technologies allow us to elaborate new effective approaches for designing and implementation of complex PCS.

3 AMDT: Cybernetic-Centered Methodological Background and Formal Definition

Taking into account already mentioned trends in adaptive software development domain (s. Section 2) we also had tried to elaborate the methodological background for our AMDT-framework, to formalize its structure and functionality as well. For this purpose we had compared two well-known conceptual schemes, namely: the typical scheme of adaptive control system given in [9], Fig. 1(a), and proposed by Highsmith [5] the incremental life-cycle (LC) model used by component-centered software development, Fig. 1(b).



(a)



(b)

Fig. 1. Control system's scheme (a), and incremental LC-model (b)

Table 1. The comparison between software components and control system's artifacts

Control System's Components	SW- development Process Artifacts
Control Object (CO) / Operating Environment (OE)	Software Components / System Requirements (SR) to be analyzed for their elaborating
Control Algorithms (CA), Controllers (C)	Design Procedures (DP), which are used in development process
Knowledge Models (KM) used for adaptive process control algorithms	Domain Models (DM), and data resources for system designing (best praxis cases, reference architectures, design patterns, etc.)
Feedback Sensor (S), Interpreter (Ipr)	Estimation Models (EM): e.g., SW-quality metrics for performance, reliability, etc.

The result of this comparative analysis is shown in the Table 1. As we can see there are the logically-interconnected components and artifacts in both concepts, e.g.: control objects / operating environment from the one side, and software components / system requirements to be analyzed - from the other side, etc. (Table 1).

Therefore summarizing the artifacts given in the right column of the Table 1, and using the set - theoretical notation, we can propose the following formal definition for ADMT-framework

$$AMDT \Rightarrow \langle DM, SR, DP, EM \rangle,$$

where DM is a set of domain models representing an appropriate application area of PCS to be designed, SR is a set of system requirements which have to be analyzed for this purpose, DP is a set of design procedures used in the software development process, and EM is a set of estimation models which allow to evaluate some PCS-features, e.g. such their quality attributes like performance, reliability, maintenance ability, etc.

4 AMDT-Core: Information Meta-space Metaphor

Our concept considers all AMDT-objects and processes in software development within an adaptive control scheme, including some metrics and estimation models in its feed-back loop. They are integrated in the multi-dimensional information meta-space, which consists of 4 local information spaces: 1) *P1-space* for system architecture construction basing on the set DM; 2) *P2-space* for requirements traceability inside of the set SR; 3) *P3-space* for representation of designing methods and project solutions included in the set DP; 4) *P4 - space* for structuring of evaluation models and metrics, which are encapsulated in the set EM. All these spaces *P1-P4* are connected via some *system trajectory*, which represents the appropriate designing variant of PCS to be elaborated (see Fig. 2). Let's consider these local spaces more detailed.

The space *P1* is structured basing on the ontological specifications for general PCS-topology and functionality, which reflects the well-known "4+1" – software architectural vision given by Ph. Kruchten [6]. These projections are the following (see in [15] for more details]):

– *Process Control System - Technological Process*: it defines the organizational and technical structure of an appropriate PCS,
– *Technological Process - Control Facility*: it describes the infrastructure of the soft- and hardware solutions, which implement process control functions,
– *Control Facility - User Profile*: it captures the information needs and requests of all system user groups,
– *User Profile - System Conflicts*: it focuses on problem situations and errors, which can arise during system operation,
– *System Conflict - Pattern Solution*: it depicts possible reusable solutions have to be used for system conflicts resolving.

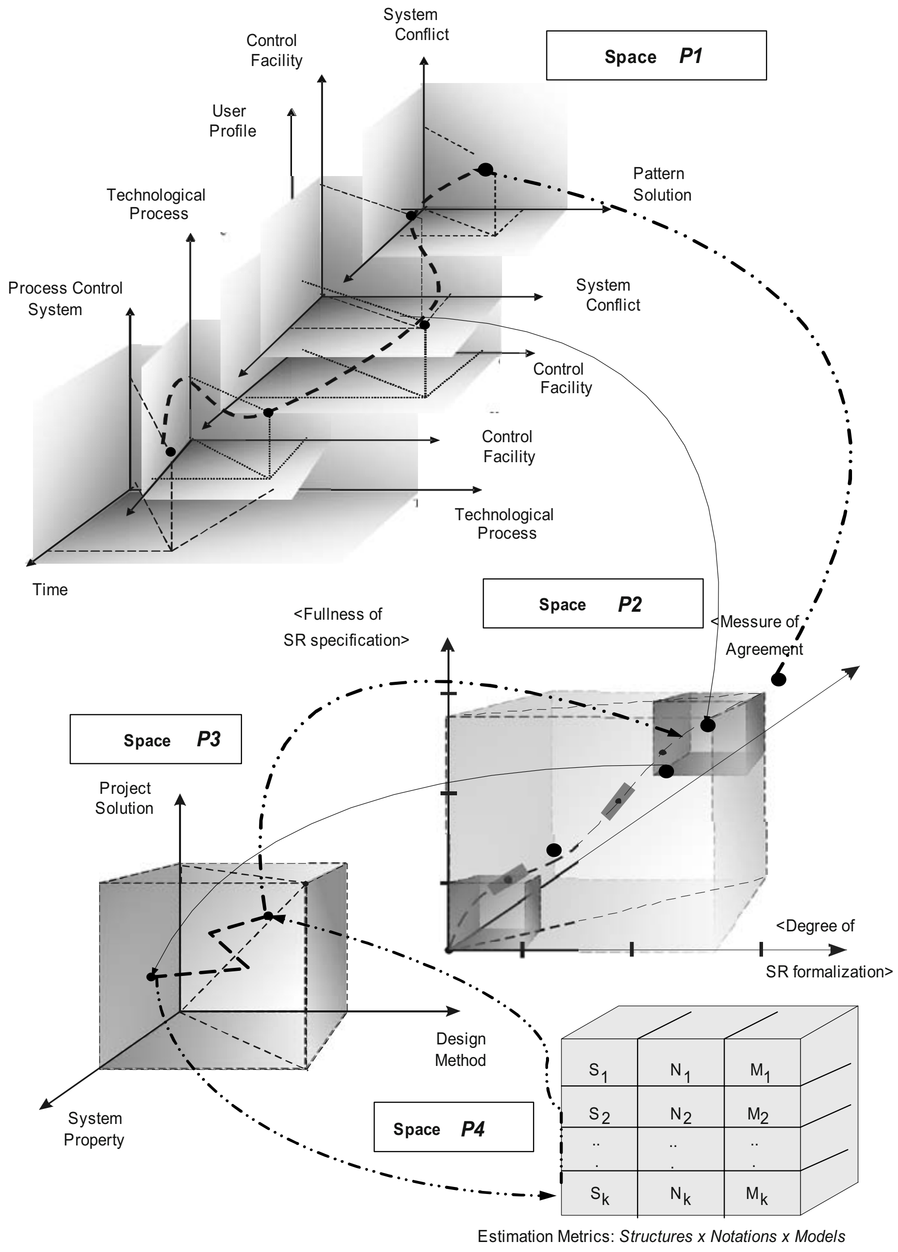


Fig. 2. The information meta-space metaphor for AMDT

The space *P2* is built basing on the 3 criteria for SR - estimation, which are invariant regarding any application domain, namely:

C1: *Fullness of Specification*, it defines the degree of requirements completeness in some project;
C2: *Degree of Formalization*, it indicates the formalizing degree of given SR;
C3: *Measure of Agreement,* it shows to which degree the stakeholders (domain experts, analysts, programmers, etc.) are consolidated from their points of view to SR to be analyzed in the project.

These criteria C1-C3 are really complex and weakly formalized. For instance, for the criteria C2 - *Degree of formalization* for given SR - the range of possible values could be defined as following: the initial value $C2_{min}$ is the description of SR given in a pure verbal form (in natural language to be understood for all project's participants); the final value $C2_{max}$ is the description of this SR presented in any kind of formalized notation (e.g., in form of UML diagrams), which allows to generate a source code in target programming language, and finally to get an appropriate software solution accordingly to this SR. Basically, the criteria values C1-C3 are *orthogonal logically* because some project's state is possible, when there is the completed functional description of given SR (it means $C1=C1_{max}$), but the level of it's formalization is low ($C2=C2_{min}$), and in addition to this the measure of coordination is also very poor (i.e., $C3= C3_{min}$); in the same way all others logical combinations of criteria values C1-C3 could be constructed, etc.

If criteria values of C1-C3 are fuzzy defined then the *P2-space* is a fuzzy set, and it could be defined as subset from Cartesian product of appropriate fuzzy sets, namely: $\Pi \subseteq D(C1) \times D(C2) \times D(C3)$ where: $D(C_i), i \in [1,3]$ - is a fuzzy set (domain) of each criteria.

We can describe the criteria C1-C3 using *linguistic variables* (LV), which are defined in fuzzy set *P2*. Basing on well-known definition for LV [18], each of them can be given as a tuple like: $< \beta, T, X, G, \mu >$ where: β is the *name* of LV; T is the *set of terms* for this LV; X is *universal set* of basic variables, which define the given LV; G is the set of *syntax rules* used for constructing of new terms in T (in particularly, it could be an empty set); and $\mu_T(X)$ is the *membership function* (MF) for given LV.

Accordingly to this general definition the criteria introduced above can be described in the *P2-space* using the following LVs:

1) LV for criteria C1: β_1="*Fullness of Specification*", T_1= <"INITIALLY","PARTLY", "FULLY"> $X = [0,1]$, the MF is $\mu_{C_1}(X)$, it has the triangular form and is shown on Fig. 3;
2) LV for criteria C2: β_2="*Degree of Formalization*", T_2=<"NON-FORMALIZED","SEMI-FORMALIZED","FORMALIZED">, $X = [0,1]$, the MF $\mu_{C_2}(X)$ can be defined analogically;
3) LV for criteria C3: β_3="*Measure of Agreement*", T_3=<"NON-CORRECTED","CORRECTED","AGREED">, $X = [0,1]$, and the MF $\mu_{C_3}(X)$ (the same assumption as by $\mu_{C_1}(X)$).

The fuzzy-logic based approach for SR - analyzing and evaluation in *P2*-space was elaborated in [11] that improves efficiency their usage in PCS-designing process.

The space *P3* to be a space for decision making for *Project Solutions* which have to be established taking into account some *System Property* to be implemented using an appropriate *Design Method.* In particular, *P3- space* includes the catalog of design patterns for scalable system software architecture. Finally, the space *P4* provides a collection of simulation models and metrics for system performance and reliability evaluation.

Taking into account this information meta-space metaphor, the AMDT-framework can be realized within the following algorithm (see Fig. 2):

Step 1. In the *P1*-space in point of time t some current PCS - architecture version is established, and for this one the appropriate set of system conflicts has to be defined $C = \{c_i\}, i = \overline{1,n}$, and if this set is not empty then

```
IF $C \neq 0$ THEN
        GOTO to Step 2;
ELSE
        GO TO Step 5;
ENDIF.
```

Step 2. In the *P2*-space each set member: $c_i \in C$ can be represented as a fuzzy description of an appropriate SR, and using the method elaborated in [11], the alternative description for this SR has to be constructed, which belongs to so-called acceptable area $A^{(2)}$ and

```
IF $a_k^{(i)} \in A^{(2)}$ THEN
        GOTO Step 3;
ELSE
        REPEAT Step 2;
ENDIF.
```

Step 3. Basing on the alternative fuzzy-specification of SR its non-fuzzy value (e.g., using Mamdany's method [3]) is generated, and it constitutes as some system property to be met $p_k^{(i)} = defuzzy\left(a_k^{(i)}\right)$. Basing on this one an appropriate design method in the *P3*-space can be chosen, which produces a collection of possible project solutions $S = \{s_j\}, j = \overline{1,l}$, and this set is not empty:

```
IF $S \neq 0$ THEN
        GOTO Step 4;
ELSE
        GOTO Step 2;
ENDIF
```

Step 4. In the *P4*-space for each set member of S an estimation procedure for such system characteristics as performance, reliability, etc. is provided and

```
IF <testing results are satisfied> THEN
        GOTO Step 5;
ELSE
        GOTO Step 3;
ENDIF.
```

Step 5. The achieved system design version has to be documented, and if other design criteria (financial, temporal, etc.) are satisfied then designing process can be finished else the steps 1- 4 have to be repeated.

Using this workflow we can provide an integrated and multi-criteria adaptive software development process for complex PCS.

5 Some AMDT – Models, Methods and CASE-Tools

The local spaces *P1-P4* from the technological point of view are represented in AMDT-approach as a collection of appropriate data models, algorithms and software tools. E.g., each projection of *P1*-space is specified in form of UML-diagram shown on Fig. 3, and basing on this one the corresponding relational database was implemented (see in [15]).

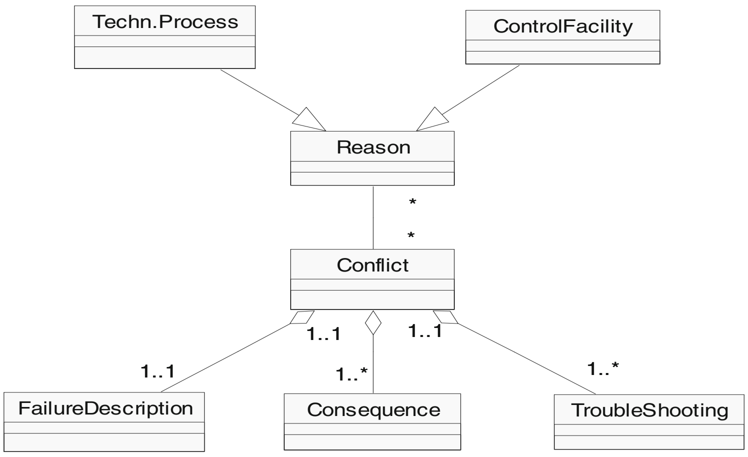


Fig. 3. UML-specification for one projection in the *P1*-space

As mentioned above (s. Section 4) for SR-traceability in the *P2*-space the fuzzy-logic based method was elaborated, which utilizes some MF for appropriate LV like shown on Fig. 4.

The *P3*-space provides a collection of project solutions for PCS-architecture, which can be implemented as reusable software components. On Fig. 5 one of such component-based software solution (CBSS) is presented, which realizes distributed multi-level PCS for real-time data processing with respect to some programmable logical controllers (PLC) configuration.

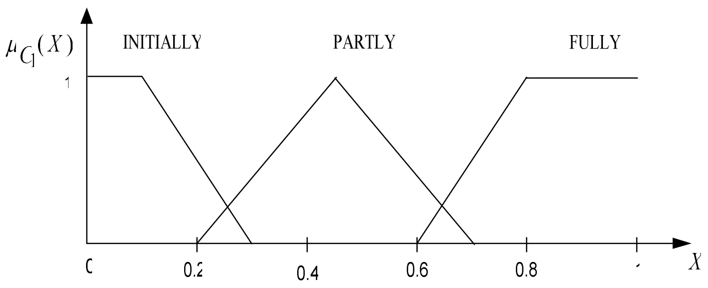


Fig. 4. MF for the LV: "Fullness of SR specification"

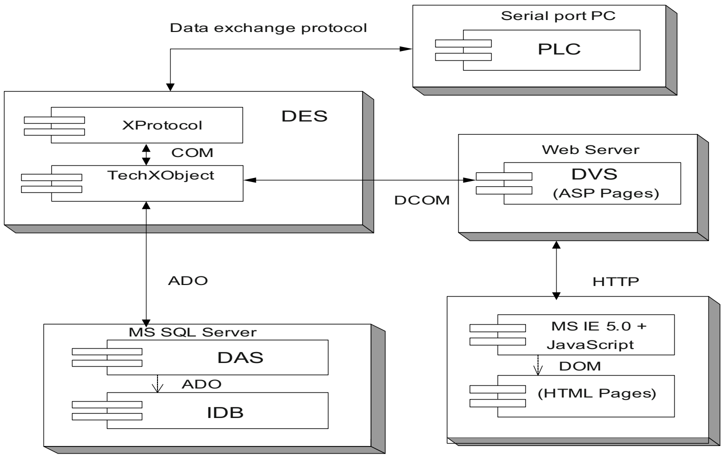


Fig. 5. The CBSS for distributed Web-based PCS

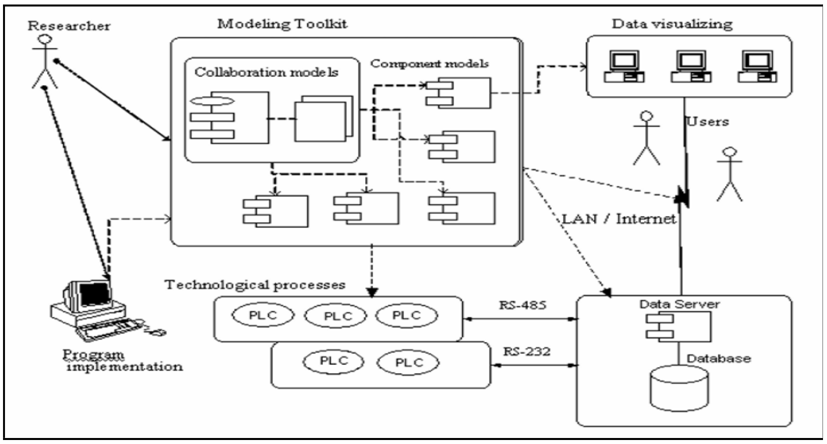
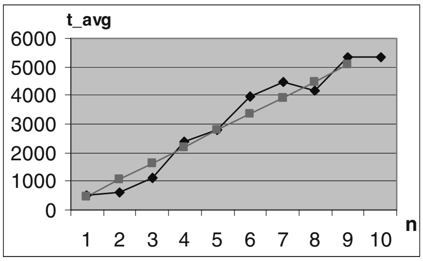


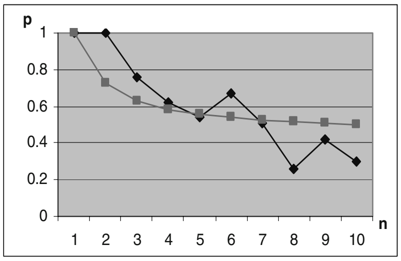
Fig. 6. The architecture of simulation environment

There are the following typical software components elaborated and running together in this CBSS, namely: *Data Exchange Server* (DES) including 2 sub-components *XProtocol* and *TechXObject*; *Data Visualizing Service* (DVS), *Data Archiving Service* (DAS), *Integrated Database* (IDB). The *P4*-space includes the models and algorithms to prove CBSS concerning their non-functional SR. It can be presented as the integrated software package, which includes both real PCS components to be tested, and CBSS-models with the appropriate simulation CASE-environment as well. It's general architecture is shown on Fig. 6 (see in [14,16] for more details).

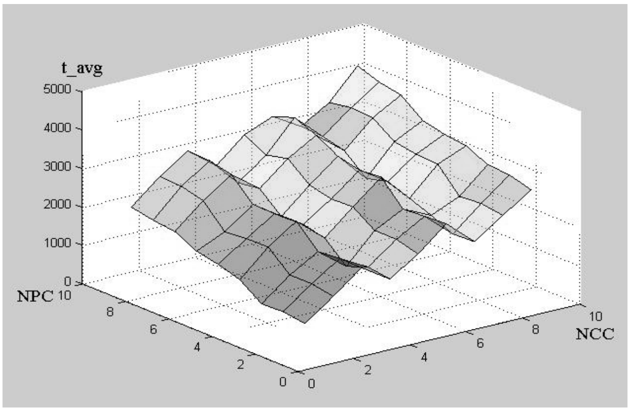
Simulation modeling of CBSS allows us to investigate and to adapt the appropriate target PCS - parameters for new SR to be fulfilled by their real-life operating. E.g., we can investigate configurations with various *numbers of client components* (**n**) in an appropriate CBSS, and can estimate such their operating characteristics as an average *response time* (**t_avg,** given in ms), and a *request / response rate* (**p,** given in %) by client-server interaction in CBSS. The results of these simulations are presented below on the Fig. 7(a)-(c).



(a) average response time (in ms);



(b) request / response rate (in %)



(c) 3D-representation for the dependency: "clients – controllers – response time"

Fig. 7. The results of CBSS performance simulation

In particular, the 3D-chart given on Fig. 7(c) represents the simulated relationship between the discrete variable *Number of Clients Combination* (NCC), the variable *Number of PLCs Combination* (NPC), and the *average response time* (**t_avg**). In this way we can predict a performance of the DES component in the CBSS shown on Fig. 5, and to adapt an appropriate PCS to possible changes in its operating environment.

6 Conclusions and Future Work

We have presented the new framework for adaptive designing, implementation and simulation of multi-level distributed PCS, which provides reusable, scalable and efficient software solutions on a Web-oriented technological platform. The proposed AMDT-approach has been already used successfully in the real-life projects performed at the oil- and gas-production enterprises in Ukraine. We are going to improve some its features through usage of knowledge-based methods in the AMDT-subsystems.

Acknowledgments. The author would like to express the gratitude to Prof. Dr. D.h.c. Heinrich C. Mayr from Alpen-Adria University of Klagenfurt for his stimulating discussions of some problems related to the ideas presented in this paper. I would like also to thank all my post-graduates and my students of the Computed-Aided Management Systems Department at the National Technical University “Kharkiv Polytechnic Institute” for their support by the designing and implementation of the topics outlined in this report.

References

1. Bhowmick, S., Kaushik, D., McInnes, L., et al.: Parallel Adaptive Solvers in Compressible PETSc-FUN3D Simulations. In: Proceedings of the 17th International Conference on Parallel Computational Fluid Dynamics, University of Maryland, College Park, MD, May 24-27, 2006 (2006)
2. Bosilca, G., Chen, Z., Dongarra, J., et al.: Self-Adapting Numerical Software SANS Effort. UTK CS Technical Report UT-CS-05-554, IBM Journal of Research and Development 50(2/3), 223–238 (2005)
3. Dubois, D., Prade, H.: Theorie des Possibilites, Masson, Paris (1988)
4. Herring, C.: Viable Software: The Intelligent Control Paradigm for Adaptable and Adaptive Architecture. : PhD dissertation, Dept of Computer Science and Electrical Engineering, University if Queensland, Brisbane (2002)
5. Highsmith, J.: Adaptive Software Development. Dorset House Publishing (2000)
6. Kruchten, Ph.: Architectural Blueprints: The “4+1” View Model of Software Architecture. IEEE Software 12(6), 42–50 (1995)
7. Miller, S.D., DeCario, R.A., Mathur, A.P.: A Software Cybernetic Approach to Control of Software Test Phase. In: Computer Software and Applications Conference, Proceeding of the 29th Annual International, vol. 2, pp. 103–108 (2005)
8. Norris, B.: Software Architecture Approaches for Adaptive Scientific Computing. In: Dongarra, J., Madsen, K., Waśniewski, J. (eds.) PARA 2004. LNCS, vol. 3732, pp. 629–636. Springer, Heidelberg (2006)
9. Pospelov, D.: Logical-linguistic Models in Control Systems, Moscow, Energoizdat (1981)

10. Tang, A., Jin, Y., Han, J., Nicholson, A.: Predicting Change Impact in Architecture Design with Bayesian Belief Networks. In: Proceeding of the 5th IEEP/IFIP Conference on Software Architecture (2005)
11. Tkachuk, N., Godlevsky, M., Zemlyanoy, A., Gamzayev,: Fuzzy-Logic Modeling Approach and Software Solutions for System Requirements Management. In: Lecture Notes in Informatics (LNI) Proceedings. Series of the German Informatics Society (GI), vol. P-84, pp. 185–188 (Printed in Bonn, 2006)
12. Tkachuk, N., Mayr, H.C., Kuklenko, D., Godlevsky, M.: Web-based Process Control Systems: Architectural patterns, Data Models, and Services. In: Shafazand, H., Tjoa, A.M. (eds.) EurAsia-ICT 2002. LNCS, vol. 2510, pp. 721–729. Springer, Heidelberg (2002)
13. Tkachuk, N., Pester, A., Kuklenko, D., Zemlyaniy, A.: Remote Process Control Framework in Gas-Production Domain: Web-based Software Solutions, Intelligent Data Processing and Simulation. In: Carinthia Tech Institute, 1st International Symposium on Remote Engineering and Virtual Instrumentation (REV-2004), Villach, Austria (2004)
14. Tkachuk, N., Polkovnikov, S., Al-Hassanie, Z.: Multi-level Composed Framework for Simulation and Estimation of Component-based Software Solutions. In: Radio-electronic and Computerized Systems, National Aerospace University, Kharkiv, vol. 4, pp. 93–99 (2006)
15. Tkachuk, M., Sokol, V., Mayr, H.C., Godlevsky, M.: A Knowledge-based Approach to Traceability and Maintenance of Requirements for Information Systems. J. Problems of Programming, Kiev (2-3), 370–378 (2004)
16. Tkachuk, N., Zemlyaniy, A., Soloshuk, V.: Architecture and Implementation Technology of Simulation Tools for Component-based Software Solutions. In: Computerized Control Systems and Devices, National University of Radio-Electronic, Kharkiv, vol. 126, pp. 71–77 (2004)
17. Yau, S., Huang, D., Zhu, L., Cai, K.-Y.: A Software Cybernetics Approach to Deploying and Scheduling. In: Proceedings of the 11th IEEE International Workshop on Future Trends of Distributed Computing Systems (2007)
18. Zadeh, L.: The concept of a Linguistic Variable and its Application to Approximate Reasoning. Part I: J. Inf. Sci. 8, 199–249 (1975)
19. Zhang, J., Cheng, B.: Model-Based Development of Dynamically Adaptive Software. In: International Conference on Software Engineering (ICSE 2006), Shanghai, China, May 20-28 (2006)

Mokum for Correctness by Design in Relation to MDA[*]

Reind van de Riet

Emeritus, Dept. of Computer Science
Vrije Universiteit in Amsterdam

Abstract. The use of the Mokum system for correctly designing Information Systems and how these designs can automatically be translated into implementations, is the theme of this paper. Similarities and differences with tools from the field of Model Driven Engineering/Architecture, such as UML, will be demonstrated.

Keywords: Model Driven Architecture, Knowledge base systems.

1 Introduction

In the old days one believed in the possibility to design correct Information Systems (IS) and to implement them. Currently, one is more realistic, and is concerned with tools to find errors and to remedy these systems.

The Mokum system is from the old days; the acronym stands for Manipulating Objects with Knowledge and Understanding in Mokum (Mokum is another name for Amsterdam). We want to show in this paper what its principles are, and why we believe that these principles are still valid, because one can automatically generate an implementation from a design, proven to be correct. The following principles are being used:

1. **the IS encompasses all players in the field**, not just the data. So in a hospital IS, these are the doctors, the nurses, the financial people, etc. In an IS for a garage, not only cars to be repaired or sold are represented but also the technicians, the selling people, their bosses, the managers and the administrative employees. The ground for this principle is that we want to be able to prove correctness of access and usage of software tools.
2. **design and implementation are one and cannot be considered independently**.
3. **all players and data to be defined are ordinary objects**, in the realm of Object-Oriented programming.

During design phase and implementation phase the **type tree** is of crucial importance. The type tree is a tree in which all the types of the objects to be considered are defined. It is used to define access from one object to another, and also to pieces of software.

[*] MDA is the trademark of the Object Management Group OMG; MDE is free.

This last criterium is translated in the rule that an object can only execute a piece of software in the form of a **trigger**. All triggers are defined within a type(s) of the object and react upon a message sent by another object (or itself). Dependent on the state an object is in it will/can react. Suppose it is asleep, then only a timer message can awake it. Suppose it is waiting for a reaction of another object, it may be only susceptible to a message of this other object.

We will see that it is (in principle) easy and straightforward to automatically translate in Mokum a design, in the form of WorkFlow diagrams, on a high level into the definitions of these triggers. That means that with a correct design comes a correct implementation. For more on WorkFlow diagrams see [1].

How do we achieve this goal of having a correct design? There are two reasons: due to the simple principles upon which access control is based, it is possible to use simple tools (in Prolog) to prove that acccess for security and privacy is guaranteed; the second is that we use semantic knowledge, in the form of Ontologies, to validate the design. This validation does not have the power of a proof of correctness, but it helps to ascertain the high quality of a design. For reading about Mokum in Cyperspace and Security & Privacy issues see [11].

2 The Mokum System

As said above, Mokum is an object-oriented system, with special notions to handle access control. (See also [4-16]). There is a type hierarchy, with the usual inheritance of properties. Each type has a number of attributes, which themselves may be of simple nature, such as text and integers, or of a newly defined type, such as for the father of a person. In addition a type has several states and a script, in which triggers are defined which react on incoming messages, by changing states, values of attributes, and sending messages to (other) objects. There is the notion of time, so that timers can be set. The is_a relation defines the inheritance of properties. If T is_a S then type T has all the properties of type S, not only attributes but also states and script. So an object being an instance of type T is automatically an instance of type S. All types are is_a related to the special type *thing*. The only property of *thing* is that it defines unique identifiers for its instances, the so-called object identifiers: OID. All types can be put in a tree, the **type tree**, structured according to the is_a relation as hierarchical principle. In the diagram for the type tree the is_a relation is indicated by means of an arrow with a closed head. The is_a relation is transitive.

What makes Mokum special is the notion of **collection** and its **keeper**. In order to make it possible to define access control, these notions have been introduced. Suppose that an object, like a doctor in a hospital system, needs single access to a set of other objects, of a certain type, in this case type patient, we make this object keeper of that set. So in the example case: we make the doctor keeper of the collection of (his or her) patients, to which only he or she has access. To indicate the relation between keeper and collection, there is a special relation between types the **coll_of** relation, indicated as follows: suppose the keeper has type T and the collection is a set of objects of type S, then in the type tree there is a special arrow from T to S, an arrow with open head. The coll_of relation is non-transitive. In a case where there is only one coll_of relation for a certain type, the drawing of the coll_of arrow is enough, otherwise each coll_of relation needs to get a unique name. In principle, a compiler can now already prevent many accesses: when object O_T, as an instance of type T, wants access to an attribute A of object O_S, as an instance of type S, and T and S are not is_a or coll_of related, then access can immediately be denied. We say that A is not visible to T and therefore to O_T. Of course, one has to take into account that the coll_of relation can be inherited:

T **is_a** T_1 **is_a** T_2 **is_a** .... T_n **coll_of** S_1 **is_a** S_2 **is_a** ... S

So an algorithm for visibility is: (Note that the basic facts all have a "_" in their names and are simply given beforehand, e.g. in a diagram).

(Note in Prolog "," means: "go on if OK", ";" means "or" and "." "end of definition").

```
isa(T,S):- T=S;(is_a(T,T1),isa(T1,S)).
attr(A,T):- isa(T,S),attr_of(A,S).
vis(A,T):- attr(A,T);
       (isa(T,T1),coll_of(T1,S),attr(A,S)).
```

That visibility is only a necessary condition for accessability is obvious: one doctor should not be able to see a patient of another doctor. Therefore, Mokum adopts the following rule for accessability: it is based on the ad hoc situation: is *this* doctor keeper of a collection of patients in which *that* particular patient is currently a member. We call this an ontological rule, as it is based on **existence** in collections. The visibility rule is a rule which we call epistemical, as it is based on **reasoning**. So as accessability criterion, Mokum uses:

```
acc(CO,A,O,T):-(CO=O,attr(A,T));
  (isa(T,T1),keeper_of(CO,T1,O,S),attr(A,S)).
```

`CO` is the calling object, `O` the object and `A` the attribute in which `CO` is interested, and `T` the type according to which `CO` is calling (in one of `T`'s triggers). The type `S` is the type of `O`. It looks a bit complicated, but one has to take into account the situation that the calling object is e.g. a doctor, who in another function is also a docent and `O` may be a patient, but can also be a student. `CO` may be keeper of some collection of which `O` is a member (as an `S`), or this is not the case. In the first case the algorithm gives precisely the criterium, in the other case, the algorithm correctly denies access.

In Fig. 1 we have depicted a type tree for objects in four different areas: University, Hospital, Insurance Company and the Civil Administration, where personal data are being stored. This type tree forbids that Nurses access Med-patients, while Doctors, in principle can (dependent on whether they have Med_patients in their collection). Employees of an IC located in the hospital (Hosp-IC-empl) can access, in principle, short_fin_records of IC-patients which also are IC_clients and have a policy with this IC, but they cannot access (full) fin_records, to which only Fin_administrators may have access. These are also supposed to supply data for the short_fin_records. If a person, John, happens to be a doctor as well as a docent, he can only access the medical record of a patient, if that patient is in his doctor's collection, not when that patient happens also to be a student of John and is in John's student collection. As you see there is no problem with an object having mutiple types, such as John above. You can also see that Mokum really is meant to deal with mutiple sites. There is a special protocol, described in [9], which makes sure that the communication between sites is such that security is guaranteed.

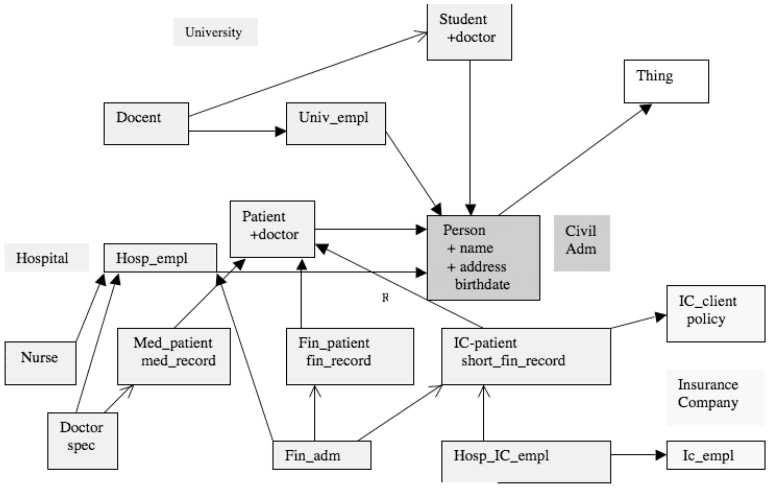


Fig. 1. A hierachical type tree

One remark about the non-transitiveness of the coll_of relationship: suppose it was transitive, then a doctor having a patient in its collection could also access his collection of cars, if the patient has one.

3 WorkFlow Diagrams

The Mokum system encompasses also a kind of WorkFlow (or MDE) system, called ColorX, designed and implemented by Hans Burg; the name stands for **Co**nceptual **l**inguistically based **o**bject oriented **r**epresentation Language for Information and Communication Systems (ICS = **X**), in which diagrams can be described defining dynamic and static properties of an information system IS. It is a graphical system for drawing boxes and arrows, together with a connection to WordNet or an Ontology to validate the design by comparing the labels used in the diagrams with their (possible) meaning according to WordNet or the Ontology. There are tools to check internal structural consistency of the diagrams themselves and their combination. We will illustrate the use of ColorX with a few examples. We have more examples from the area of Hospitals, combined with Insurance Company and DKE's refereeing process.

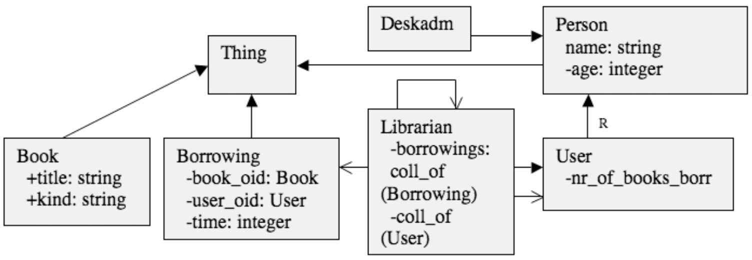


Fig. 2. Type tree for library example

The example is a library system, with one librarian and a number of users borrowing books. These have to be administrated, using Borrowings. There are some rules: a user may not borrow more than four books (simultaneously) and there is also a restriction on the kind of books a user may borrow dependent on the user's age. The Librarian is the only one who can see the Borrowings of a user, being the keeper of the collection of users and Borrowings, as indicated in the type diagram for this library situation, as shown in Fig. 2. All actions of user and librarian are specified in the ColorX dynamic diagram shown in Fig. 3. A combination of both diagrams, as a kind of semantic diagram, is shown in Fig. 4. In this diagram all the actions of the dynamic diagram in combination with the types in the type tree are shown.

Finally, a direct translation of the dynamic diagram in the form of a State Machine Diagram is shown in Fig. 5.

The reason that Librarian has a coll_of relation to itself is that there is a rule in Mokum: *a keeper of a collection cannot insert or delete members*. Usually, there is a kind of manager who is in charge of this function. In the case of the Librarian, we want to keep things simple. We therefore have designed the Librarian to be his or her own manager. Attributes with a "+" are public and with a "-" are private. The whole process of borrowing a book is shown in the next dynamic ColorX diagram.

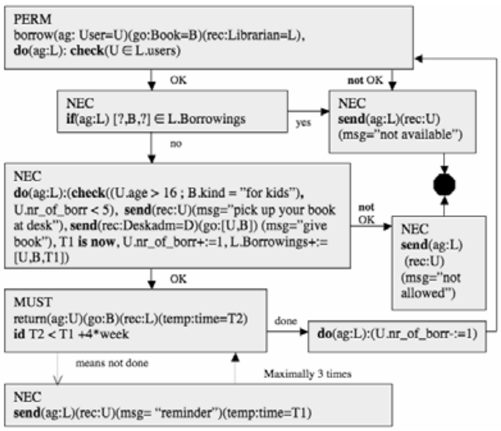

PERM
borrow(ag: User=U)(go:Book=B)(rec:Librarian=L),
do(ag:L): check(U ∈ L.users)
OK
not OK
NEC
if(ag:L) [?,B,?] ∈ L.Borrowings
yes
NEC
send(ag:L)(rec:U)
(msg="not available")
no
NEC
do(ag:L):(check((U.age > 16 ; B.kind = "for kids"),
U.nr_of_borr < 5), send(rec:U)(msg="pick up your book at desk"), send(rec:Deskadm=D)(go:[U,B]) (msg="give book"), T1 is now, U.nr_of_borr+:=1, L.Borrowings+:= [U,B,T1])
not OK
NEC
send(ag:L)
(rec:U)
(msg="not allowed")
OK
MUST
return(ag:U)(go:B)(rec:L)(temp:time=T2)
id T2 < T1 +4*week
done
do(ag:L):(U.nr_of_borr-:=1)
means not done
Maximally 3 times
NEC
send(ag:L)(rec:U)(msg= "reminder")(temp:time=T1)

Fig. 3. The dynamics of borrowing a book

The diagram deals with one Borrowing at a time only. When more than one users are borrowing books, and the Librarian necessarily has to divide time between them, there may be a different situation. For the simulation it does not matter much as the Mokum system can deal with multiple objects simultaneously.

Following the arrows. each box is labeled with a so-called mode: PERM for "permitted" or "no initial condition", NEC for a "necessary activity" and MUST for an "obligation for an object to which the library system does not have control", such as the user who has to return the book he borrowed. Usually there is a time limit involved as the four weeks in our diagram. Objects playing a role in this diagram are denoted by a single letter, such as U for the argument of a User in borrow.

do(ag:L):**check**(U∈L.Users) means that a condition is checked; if the condition is met, nothing is done, otherwise the system stops after producing an appropriate error message. In this case it is checked whether the triple [?,B,?] is a member of L's collection of Borrowings, in which case the book is not available. The notation x+:=y is used to denote that y is to be added to x; Similarly for -:=. So **do**(ag:L): (U.nr_of_borr-:=1) means decrease U.nr_of_borr with 1.

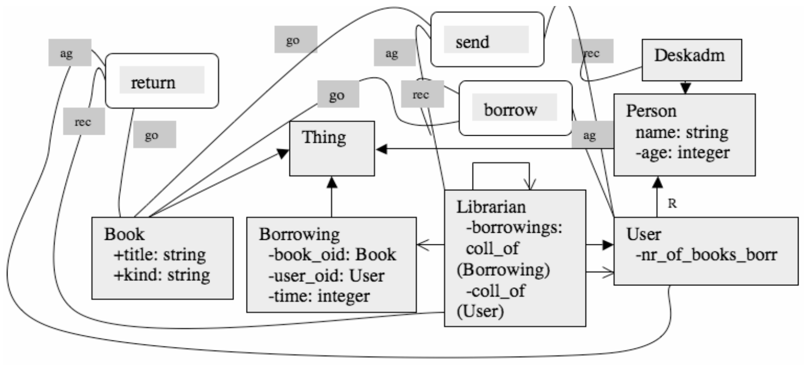


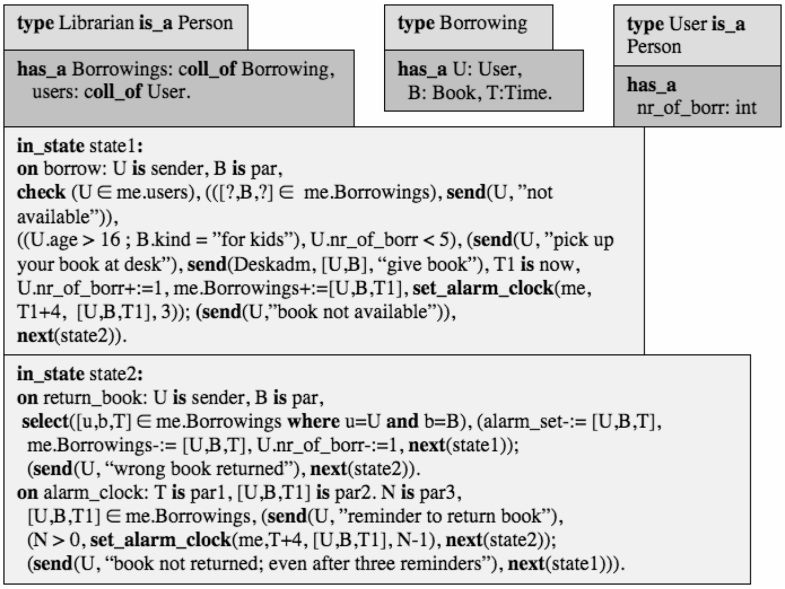
Fig. 4. Semantic Diagram for Librarian

In the semantic diagram, depicted in Fig.4, we see that it is a copy of the type tree, shown in Fig. 3, but a few boxes and arrows are added. For each action in the dynamic diagram of Fig. 2, except for **check** and **do**, an action box is added, in our case only for send, borrow and return. The arguments of these actions are labelled with roles: **ag**, **rec**, and **go**, for agent, recipient and goal, having the following meaning:

a) **ag (agent)**: entity controlling the action;
b) **rec (recipient)**: entity into whose possession something is transferred;
c) **go (goal)**: entity effected by the operation of ag.

Example: WordNet says that the agent in borrow must be a living thing, and a user is_a person, known to WordNet, so this usage is approved!

These are linguistic notions invented by Simon Dik in his linguistic theory Functional Grammar. This is heavily used in the tool ColorX. It has a connection with WordNet which is a Lexicon defined by linguists, containing the meaning and usage of some 200 000 words. ColorX may also be connected to an Ontology for the area for which the IS is being designed, in our case the Library system. There is another semantic check on the dynamic diagram: ColorX checks if after a borrow operation also a return action is foreseen, which in our case it is. The verbs “borrow” and “return” are complementary antonyms, another notion borrowed from Linguistics.



```
type Librarian is_a Person
has_a Borrowings: coll_of Borrowing,
  users: coll_of User.

type Borrowing
has_a U: User,
  B: Book, T:Time.

type User is_a
Person
has_a
  nr_of_borr: int

in_state state1:
on borrow: U is sender, B is par,
check (U ∈ me.users), (([?,B,?] ∈ me.Borrowings), send(U, ”not
available”)),
((U.age > 16 ; B.kind = ”for kids”), U.nr_of_borr < 5), (send(U, ”pick up
your book at desk”), send(Deskadm, [U,B], “give book”), T1 is now,
U.nr_of_borr+:=1, me.Borrowings+:=[U,B,T1], set_alarm_clock(me,
T1+4, [U,B,T1], 3)); (send(U,”book not available”)),
next(state2)).

in_state state2:
on return_book: U is sender, B is par,
 select([u,b,T] ∈ me.Borrowings where u=U and b=B), (alarm_set-:= [U,B,T],
  me.Borrowings-:= [U,B,T], U.nr_of_borr-:=1, next(state1));
  (send(U, “wrong book returned”), next(state2)).
on alarm_clock: T is par1, [U,B,T1] is par2. N is par3,
  [U,B,T1] ∈ me.Borrowings, (send(U, ”reminder to return book”),
  (N > 0, set_alarm_clock(me,T+4, [U,B,T1], N-1), next(state2));
  (send(U, “book not returned; even after three reminders”), next(state1))).
```

Fig. 5. Triggers for the Library case

From the dynamic diagram it is, in principle, easy to automatically generate the specifications of the triggers for the types of objects participating in the diagram. Take as example the first actions specified:

borrow(ag: User=U)(go:Book=B)(rec:Librarian=L),
do(ag:L): **check**(U ∈ L.users)

Evidently, there must be a trigger in the User type which reacts on a message expressing the wish to borrow a particular book from the library. This message originates most probably from a real user. The connection between real and virtual objects is not described here, but it is very straightforward. Also, evidently, the virtual User U sends a “borrow” message to the Librarian L. How he knows the object identifier of L is again not shown here, but it has to do with a proper initialization of the system. For the type Librarian it means that a trigger has to be provided which reacts upon the message “borrow”. This trigger is shown in Fig. 5 and discussed now. After the usual things in a trigger, such as putting sender and message parameters in specific locations, the trigger checks whether the sender of the message is a member of the collection of users of this object (“me”). Note that in the dynamic diagram different objects occur, such as L and U, but in a trigger the object usually meant is “me”. If the condition comes out yes the next thing L is doing is defined in: **“send**(ag:L)(rec:U)(msg=“not available”)”, which is translated into: "**send**(U, "not available")", very natural indeed. In the other case the check on age and type of books and on nr_of_borr is performed and if successful:

"**send**(U, "pick up your book at desk"), **send**(Deskadm, (U,B), "give book"), T1 **is now**, U.nr_of_borr+:=1, me.Borrowings+:= [U,B,T1]),"

is performed, as a direct translation of:

"**send**(rec:U)(msg="pick up your book at desk"), **send**(rec:Deskadm=D)(go:(U,B))(msg="give book"), T1 **is now**, U.nr_of_borr+:=1, L.Borrowings+:= [U,B,T1])".

The following instruction**: "set_alarm_clock**(me,T1+4, [U,B,T1], 3))" is generated because of the next MUST box, in which it is specified that the User must return the book, within four units of time after the current time. The number 3 indicates the number of times a timer message is being sent by the Librarian. The last thing the trigger is doing is to change the "state of me" into "state2". That is the end of the actions in this box.

In state 2 it is waiting for either a message from the User returning the book or a message sent by itself, as a result of setting a timer in the form of the alarm_clock.

In the first case it is doing the following:

"select([u,b,T] ∈ me.Borrowings **where** u=U **and** b=B), alarm_set-:= [U,B,T], me.Borrowings-:= [U,B,T], U.nr_of_borr-:=1, **next**(state1)); (**send**(U, "wrong book returned"), **next**(state2))."

This is not directly translated from actions in the dynamic diagram, but rather from a general structure where alarm_set contains all the alarms which have been set.

In the second case it is doing:

"([U,B,T1] ∈ me.Borrowings), (**send**(U, "reminder to return book"), (N > 0, **set_alarm_clock**(me,T+4, [U,B,T1], N-1), **next**(state2)); (**send**(U, "book not returned; even after three reminders"), **next**(state1)))"

where N is the number of times a reminder is being sent.

We will now deal with some other interesting examples. For the refereeing example we have the triggers here for the Researcher, shown in Fig.6. He behaves exactly as a State Machine with five states. The figure at the bottom right of Fig. 6, is taken from the original ColorX diagram. (See the original DKE50 article [3]).

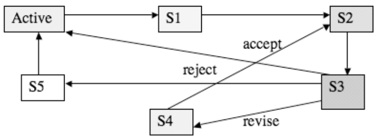
The whole process can take several months. If one wants that the Researcher is able to do other things, the above triggers have to be changed in such a way that the researcher is in several states simultaneously, which is a facility provided in Mokum. Of course, it is not possible for the Researcher to deal with more than one paper of one author at a time; such a facility would demand a kind of multi-threading, Mokum does not provide.

A third example is one which was initiated by a question about the relationship between Mokum and UML. The direct question was whether it is possible in Mokum to represent the fact of life that persons are divided in two groups which don't overlap: male and female persons. We show first in the next picture a type diagram where persons are divided indeed in these two groups, but where a keeper in the form of an angel (Gabriel) is needed to see to it that the groups remain separated.

As this has to do with births and families, the angel is also a keeper of families, while a Divine being, say G, is responsible for the creation and deletion of persons. Also the case of transgendering is taking into account. Note that a Person can be a (bank) Customer and a Patient (in a Hospital).

Note (again) that in Mokum a keeper cannot add or delete members of its collection, therefore we need a keeper of a keeper (manager) to do this, in our case G.

in_state(active): **next**(state1), **send**(me, "want_to_submit_paper", i).
in_state(state1): **on** want_to_submit_paper: PA **is** par, send(ESS, msg="submit_paper", me.coll_of Paper[PA]), **next**(state2).
in_state(state2): **on** ack_from_Elsevier: **next**(state3).
in_state(state3): **on** aswer_from_Elsevier: SP **is** me.Subm_paper[1], **print**(SP.decision), **for_all** i (**print**(SP. ref_rep[i])), ((SP.decision=revise), **next**(state4); ((SP.decision=accept), **next**(Active)); **next**(state5))).
in_state(state4): **on** need_to_revise_paper: PA1 **is** par, PA2 **is** revise(PA1), send(ESS, msg="submit_paper", PA2), **next**(state2).
in_state(state5): **on** need_to_rewrite_paper: PA1 **is** msg.par, PA2 **is** rewrite(PA1), **send**(another journal, msg="submit", PA2), **next**(active).



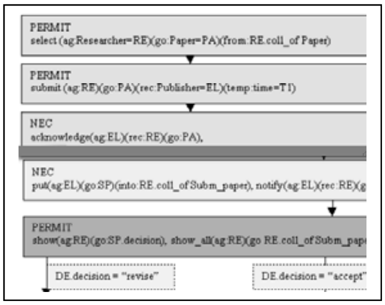

PERMIT
select (ag:Researcher=RE)(go:Paper=PA)(from:RE.coll_of Paper)
PERMIT
submit (ag:RE)(go:PA)(rec:Publisher=EL)(temp:time=T1)
NEC
acknowledge(ag:EL)(rec:RE)(go:PA),
NEC
put(ag:EL)(go:SP)(into:RE.coll_of Subm_paper), notify(ag:EL)(rec:RE)(g
PERMIT
show(ag:RE)(go:SP.decision), show_all(ag:RE)(go RE.coll_of Subm_pap
DE.decision = "revise"
DE.decision = "accept"

Fig. 6. Triggers for a researcher

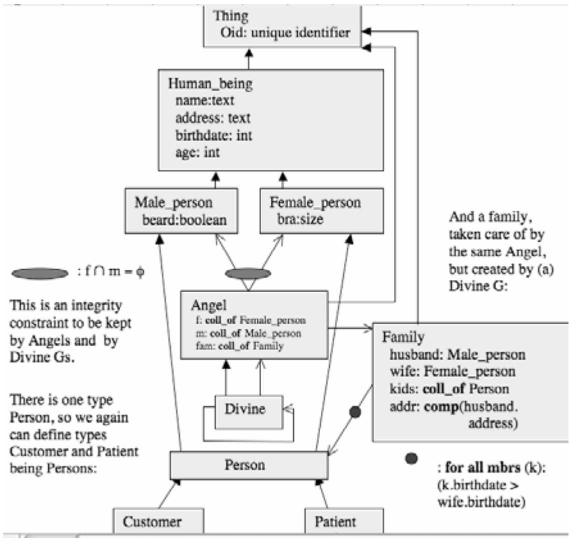
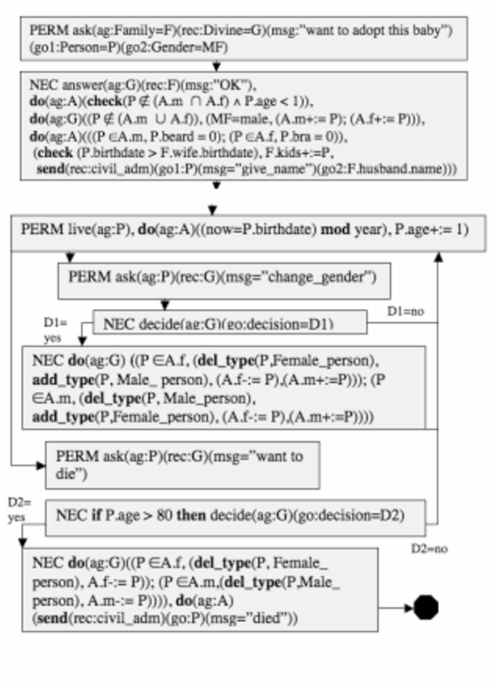
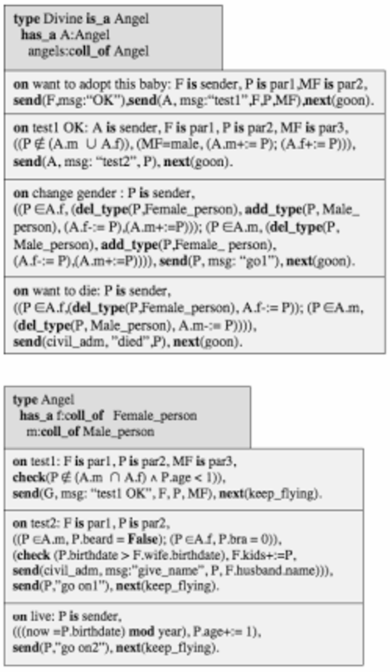

Thing
Oid: unique identifier
Human_being
name:text
address: text
birthdate: int
age: int
Male_person
beard:boolean
Female_person
bra:size
And a family, taken care of by the same Angel, but created by (a) Divine G:
: f ∩ m = ϕ
This is an integrity constraint to be kept by Angels and by Divine Gs.
Angel
f: coll_of Female_person
m: coll_of Male_person
fam: coll_of Family
Family
husband: Male_person
wife: Female_person
kids: coll_of Person
addr: comp(husband. address)
Divine
There is one type Person, so we again can define types Customer and Patient being Persons:
Person
: for all mbrs (k): (k.birthdate > wife.birthdate)
Customer
Patient

Fig. 7. The type tree of Persons, separated in Male and Female

The dynamic diagram in ColorX for the life of a person is depicted in the Fig. 8 where the process starts with a family asking to adopt a given child and ends with his or her death. The picture is followed by a picture of the triggers for G and the angel Gabriel in Fig. 9.



```
PERM ask(ag:Family=F)(rec:Divine=G)(msg:"want to adopt this baby")
(go1:Person=P)(go2:Gender=MF)

NEC answer(ag:G)(rec:F)(msg:"OK"),
do(ag:A)(check(P ∉ (A.m ∩ A.f) ∧ P.age < 1)),
do(ag:G)((P ∉ (A.m ∪ A.f)), (MF=male, (A.m+:= P); (A.f+:= P))),
do(ag:A)(((P ∈A.m, P.beard = 0); (P ∈A.f, P.bra = 0)),
 (check (P.birthdate > F.wife.birthdate), F.kids+:=P,
  send(rec:civil_adm)(go1:P)(msg="give_name")(go2:F.husband.name)))

PERM live(ag:P), do(ag:A)((now=P.birthdate) mod year), P.age+:= 1)

PERM ask(ag:P)(rec:G)(msg="change_gender")

NEC decide(ag:G)(go:decision=D1)

NEC do(ag:G) ((P ∈A.f, (del_type(P,Female_person),
add_type(P, Male_ person), (A.f-:= P),(A.m+:=P))); (P
∈A.m, (del_type(P, Male_person),
add_type(P,Female_person), (A.f-:= P),(A.m+:=P))))

PERM ask(ag:P)(rec:G)(msg="want to
die")

NEC if P.age > 80 then decide(ag:G)(go:decision=D2)

NEC do(ag:G)((P ∈A.f, (del_type(P, Female_
person), A.f-:= P)); (P ∈A.m,(del_type(P,Male_
person), A.m-:= P)))), do(ag:A)
(send(rec:civil_adm)(go:P)(msg="died"))
```



```
type Divine is_a Angel
  has_a A:Angel
    angels:coll_of Angel

on want to adopt this baby: F is sender, P is par1,MF is par2,
send(F,msg:"OK"),send(A, msg:"test1",F,P,MF),next(goon).

on test1 OK: A is sender, F is par1, P is par2, MF is par3,
((P ∉ (A.m ∪ A.f)), (MF=male, (A.m+:= P); (A.f+:= P))),
send(A, msg: "test2", P), next(goon).

on change gender : P is sender,
((P ∈A.f, (del_type(P,Female_person), add_type(P, Male_
person), (A.f-:= P),(A.m+:=P))); (P ∈A.m, (del_type(P,
Male_person), add_type(P,Female_ person),
(A.f-:= P),(A.m+:=P)))), send(P, msg: "go1"), next(goon).

on want to die: P is sender,
((P ∈A.f,(del_type(P,Female_person), A.f-:= P)); (P ∈A.m,
(del_type(P, Male_person), A.m-:= P)))),
send(civil_adm, "died",P), next(goon).

type Angel
  has_a f:coll_of   Female_person
    m:coll_of Male_person

on test1: F is par1, P is par2, MF is par3,
check(P ∉ (A.m ∩ A.f) ∧ P.age < 1)),
send(G, msg: "test1 OK", F, P, MF), next(keep_flying).

on test2: F is par1, P is par2,
((P ∈A.m, P.beard = False); (P ∈A.f, P.bra = 0)),
(check (P.birthdate > F.wife.birthdate), F.kids+:=P,
send(civil_adm, msg:"give_name", P, F.husband.name))),
send(P,"go on1"), next(keep_flying).

on live: P is sender,
(((now =P.birthdate) mod year), P.age+:= 1),
send(P,"go on2"), next(keep_flying).
```

Fig. 8. Dynamic Diagram of the life of a person

Fig. 9. Divine and Angeline triggers

The main reason for showing these diagrams is to show that the (UML) State Machine Diagram in the form of the Mokum triggers can (almost) automatically be generated from the Dynamic Diagram made by the ColorX system.

The Divine object is supposed to be in the "goon" state, while the angel's state is "keep_flying".

4 The Garage Case

We also want to show that Mokum is very fitting to represent the Information System of some multi-national general purpose company, for which we have chosen a garage company, as it deals with managers giving tasks to personnel. In this example we give the type tree only, as it is the main structure determining mechanism. We have depicted the type tree for the garage in Fig. 10.

There are a few peculiarities we want to emphasize:

1. Dealing with tasks is an important thing in a company. In this case a task consists of a general piece of data, such as begin- and end time of the task, certain skills

needed for an employee to perform the task, and a special part for a technician who has to repair the engine of a car or for a sales person for the price of a car. These tasks are defined by a Prod Mgr (or Dep Prod Mgr) and assigned to an employee, that is put in the employee's collection of tasks. With this task in his/hers collection the employee now can open the task to see what he/she has to do. No one else can do that, which provides for the basic security measure. Note also that the manager can easily delegate certain responsibilities (assigning tasks to technicians) to "his/her" deputy manager.

2. As employees are also part of the system, it is possible, as in an ERP system to combine the personnel administration with the client and work administration. So the bonus for a technician can be computed from the tasks he/she has performed, provided they are in the data base.

3. Using the ontological rule for the financial administrators we can take care that a financial administrator cannot be responsible of his own salary, by carefully putting his object identifier in another one's collection (e.g. his manager's).

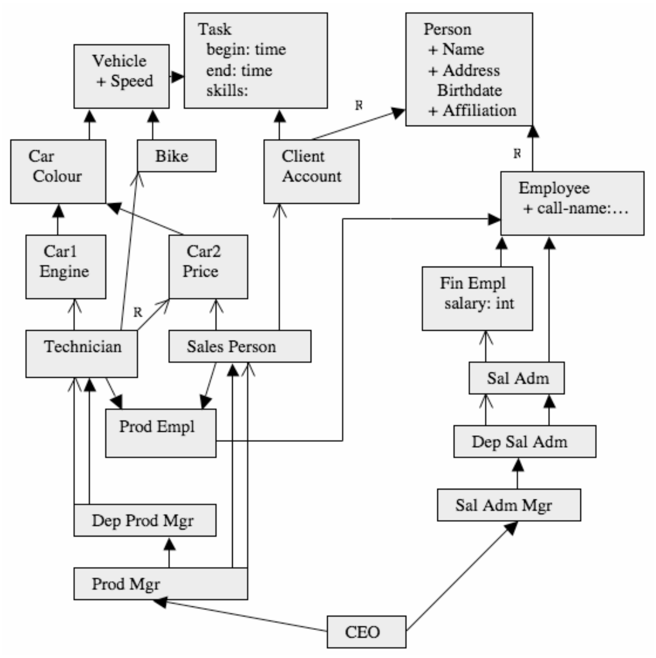
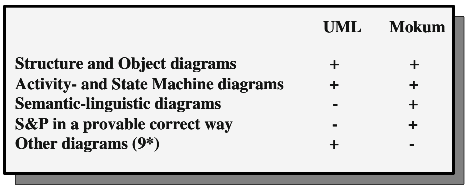


Fig. 10. Type tree for Garage case

5 Comparison with UML

From Booch et al. [2] we quote: "UML is a standard language for writing software blueprints; it may be used to visualize, specify, construct and document the artifacts of a software-intensive system". From this we see a great similarity with the Mokum system, although the appearance is different. For more on MDA see also [3]. Below we have indicated the similarities and the differences:



	UML	Mokum
Structure and Object diagrams	+	+
Activity- and State Machine diagrams	+	+
Semantic-linguistic diagrams	-	+
S&P in a provable correct way	-	+
Other diagrams (9*)	+	-

Fig. 11. Comparision – UML / Mokum

The main differences between the Mokum system and UML is the lack in UML of correctness improving tools, such as in Mokum the tools for defining access control and to validate semantically the design of an Information System.

References

1. van der Aalst, W.M.P., Wieske, M., Verbeek, H.M.W.: Advances in Business Process Management. In: DKE on BPM, Data and Knowledge Engineering, vol. 50(1), July 2004, pp. 1–115, Elsevier, Amsterdam (2004) (special issue)
2. Booch, G., Rumbaugh, J., Jacobson, I.: The Unified Modelling Language, User Guide, Addison Wesley (1997)
3. Brown, A.: An Introduction to Model Driven Architecture, Part I: MDA and today's systems, IBM (February, 2004)
4. Caminada, M.W.A., Steuten, A.A.G., van de Riet, R.P.: An evaluation of linguistically based modeling approaches on the basis of the EANCOM EDI standard. In: Goldkuhl, G., Lind, M., Seigerroth, U., Aegerfalk, P.J. (Eds.) Proceedings of the Fourth International Workshop The Language Action Perspective on Communication Modelling. Jonkoping International Business School, pp. 29-42 (1999)
5. Dehne, F.: Steuten, A.A.G., van de Riet, R.P.: WORDNET++: A Lexicon for the Color-X Method. Data and Knowledge Engineering, 38(1), 3–30 (2001)
6. Dehne, F., Steuten, A.A.G., van de Riet, R.P.: Linguistic and Graphical tools for the COLOR-X Method, Computer Science Department, IR482, Vrije Universiteit (December 2000)
7. Gudes, E., van de Riet, R.P., Burg, J.F.M., Olivier, M.S.: Alter-egos and Roles-Supporting Workflow Security in Cyberspace. In: Lin, T.Y., Qian, X. (Eds.), Proc. IFIP WG11.3 Work. Conf. on Database Security, Lake Tahoe, pp.152–166 (1997)
8. Olivier, M.S.E., Gudes, R.P., van de Riet, J.F.M.: Burg: Specifying Application-level Security in Workflow Systems. In: Wagner, R. (ed.) Database and Expert Systems Applications, IEEE Computer Society, pp. 346–354. IEEE Computer Society, Los Alamitos (1998)
9. van de Riet, R.P.: The Future Refereeing Process in Cyberspace Data&Knowledge Engineering Journal, An Attempt in Guaranteeing Security&Privacy on three levels, Data and Knowledge Engineering, 50(3), 305–339 (2004)

10. van de Riet, R.P., Burg, J.F.M.: Modelling Alter Egos in Cyberspace: Using a Work Flow management tool: who takes care of the Security and Privacy? In: Lobodzinsky, S., Tomek, I. (eds.) Proceedings of WebNet 1997, Toronto, Association for the Advancement of Computing in Education), Charlottesville, USA, pp. 582–588 (1997)
11. van de Riet, R.P.: 25 years of Mokum for Data&Knowledge Engineering. In: DKE25Y, celebrating 25 years of publishing Data&Knowledge Engineering (2008) (special issue)
12. Steuten, A.A.G., van de Riet, R.P., Dietz, J.L.G.: Linguistically based Conceptual Modeling of Business Communication. In: Proceedings of the Fourth International Conference on Applications of Natural Language to Information System, University of Klagenfurt, Austria, 17-19 June 1999, pp. 91–106 (1999)
13. Steuten, A.A.G., van de Riet, R.P., Dietz, J.L.G.: Linguistically based Conceptual Modeling of Business Communication. In: Métais, E., Mayr, H.C. (eds.) NLDB 1999 (special issue), Data & Knowledge Engineering, vol. 35(2), pp. 121–136. North Holland, Amsterdam (1999)
14. Steuten, A.A.G., Dehne, F., van de Riet, R.P.: WORDNET++: A Lexicon Supporting the COLOR-X Method. In: Bouzeghoub, M., Kedad, Z., Métais, E. (eds.) NLDB 2000. LNCS, vol. 1959, pp. 1–16. Springer, Heidelberg (2001)
15. Dehne, F., Steuten, A.A.G., van de Riet, R.P.: WORDNET++: A Lexicon for the Color-X Method. Data & Knowledge Engineering 38(1), 3–30 (2001)
16. Teepe, W., van de Riet, R., Olivier, M.: WorkFlow Analyzed for Security and Privacy in using Databases. Journal of Security 11(3), 353–364 (2003)

Conceptual Model – Meeting of Cultures

Tatjana Welzer[1], Zala Volčič[2], Boštjan Brumen[1], Izidor Golob[1], Aida Kamišalić[1], and Marko Hölbl[1]

[1] Faculty of Electrical Engineering and Computer Science, University of Maribor
Smetanova 17, Maribor, Slovenia
{welzer,bostjan.brumen,izidor.golob,aida.kamisalic, marko.holbl}@uni-mb.si
[2] University of Queensland
Brisbane, Australia
z.volcic@uq.edu.au

Abstract. Electronic availability of data and their usage in various domains have upgraded the importance of internationalization. In general, we could say that internationalization has a syntactic and a semantic component. The syntactic component is relatively easy reached, mostly is supported by tools (software translated to national languages), while the semantic component requires further research in the connection with the cultural dialog. Special attention needs to be paid to data upon which critical decisions are met. In the paper we will focus on internationalization in the connection with conceptual modeling including needs for intercultural dialog integrated into the process of conceptual modeling.

Keywords: Conceptual modeling, internationalization, cross-cultural communication, intercultural dialog, information society, database design.

1 Introduction

Nowadays all around the word we are faced with an increasing demand for more and more complex applications on databases with the main goal to support the informatization of different activities helping people by every day life in different domains. Bank, post, medicine, pharmacies, local and state administration, voting systems, libraries are just some of them. We are a part of information society in which information and communication technology is coming closer and closer to people who are not experts in information and communication technology even more, sometimes they have even a very low level of general education. Independent to these facts they are users of data and databases have to fulfill their needs.

At the beginning when databases first entered the information system market, database designer needed to invest a lot of hard work in the development of databases which was supported only by very rough tools. But nowadays, however, designing databases has become a popular activity performed not only by database designers but also by non specialists, what can bring also a doubt into the solution and quality either of the database, either of the data saved in the database, even of the whole application or information system. Beside all this, solutions have to be user friendly and they have to take into account also the intercultural dialog which is important especially in multicultural societies. In the near past, as multicultural societies mostly have been described only big societies with a lot of immigrants, but nowadays with increased mobility also small societies are more and more multicultural and international.

The rapid growth and expectations have stimulated the need for high level concepts, tools and techniques for database design and development. Starting point for the database design is an abstract and general description of the reality (domain) named the conceptual model. In the context of the database design the conceptual model has several usages [1]. At the start of the database design it should integrate various interests and views of the end users. Further, it is useful description for communication with users as well as non specialists, then it also helps the database designer to build a more durable database system and at the last but not least it enables efficient introduction of the already designed database.

To achieve the above mentioned usages of the conceptual model as well as to design an effective and high quality model we must take into consideration that the conceptual database design is extremely complex and iterative [2]. It can be greatly simplified by using different database design aids (methodologies and tools). Design methodologies for conceptual design should be rigorous as well as flexible. They should be based on a formal approach but on the other hand they should be applicable to a variety of situations and environments. Finally, they should take care also on problems of internationalization and influence of different cultures on understanding and using data as well as its importance.

In the last time some of approaches for improving conceptual modeling take into account also "meeting of cultures". They have been developed from different points of views and have been implemented in various situations. Most of these approaches are only familiar with the fact that internationalization is a prerequisite for the final success. In our contribution, we will concentrate on the problem of internationalization and cross-cultural communication influencing developing of conceptual model.

In the following chapters we will be concentrating on internationalization in general, as well as on possible intercultural dialog and its influence on database design especially conceptual modeling. Further research and final remarks will be presented in the conclusion.

2 Internationalization

Today companies and other organizations have for different reasons – benefits of information society - an unprecedented opportunity to optimize and expand their businesses exponentially. With the advent of the World Wide Web applications (Internet and Intranet), the global exposures have become possible.

According to this fact, most of the international internet and e-community is non-native English speaking. But at the same time, this same community represents the fastest growing consumers group [3]. Statistically, customer is four times more likely to make a "purchase" and other activities if the "content" is displayed in their native language. So, the challenge to be successful in the global market is to have multilingual enabled applications, opposite to the recently used only English applications and commercial computer software. Commercial computer software is usually asking more investment and knowledge for translations opposite to specific applications that can be developed in natural languages, for example for a known customer.

These resent facts could lead to the simple but not correct conclusion that only translation into selected languages is needed to reach the internationalization [4]. The internationalization is not just another feature it is much more and request participation of experts like industry practitioners, software engineering experts, experts in internationalization and cross culture, experts in selected domains (medicine, intelligent systems, economics,..) language technology experts and some others experts from closely related fields.

For success the cooperation of the academic community is expected as well. Why? Internationalization is comparatively new arrival in all fields including also computer science curricula. More or less English was language in operation for computer science and informatics. As it seems, not problematic for all users, and customers, but last changes on mobility and importance of intercultural dialog change a lot the whole situation. Every user expects the product close to his needs not only in functionality but also in language and culture.

3 Intercultural Dialogue

As already mentioned before, internet and ubiquitous information and communication technology have enabled new possibilities to promote research, development, business and administrative activities to and in new geographical locations as well as cultures [5]. Work with people remotely is in the sense of efficiency almost more efficient as work face to face. But cross-cultural communication is more and more present not only because of technology but also because of peoples' mobility, global enterprises and organizations which are more and more multilingual as well as also multicultural [6].

In our work the term "culture" is presented by the definition given by Hofstede [7],[8]: "Culture is a collective phenomenon, because it is shared with people who live or lived within the same social environment. Culture consists of unwritten rules of social game. It is the collective programming of the mind that distinguishes the member of one group or category of people from others". That means that in international (multicultural, multilingual) development, research and business, participants also share culturally integrated space. For some participants in this space could be hard to understand cultural dependent behavior of others. Adjusting to cultural differences people can face the challenge and become better in their own work. Cultural competence became one of the most important characteristics of candidates for different positions or jobs [9] within enterprises and organizations, also for conceptual modeling.

According to the before presented definition of culture by Hofstede, we have to point also on Alvesson and Berg who introduced different levels within the concept of culture [10]. They have defined seven levels which reflect intensively also in conceptual modeling. Those levels are:

- Culture in societies and nations
- Regional and local cultures
- Business cultures

- Organizational and corporate cultures
- Functional subcultures at organisational level
- Social groups in the organization
- Professional and functional cultures

Actually, those levels are presenting a kind of subgroups or better to say specific cultures according to the social life, geographical location and business domain including the enterprise and organization culture. The last two (professional and functional) are those which mostly influence also on the conceptual modeling in the phase of database design.

4 Conceptual Modeling – Do Not Forget on Internationalization and Cross-Cultural Communication

Phases and steps in modeling of conceptual models are very well defined in a long list of the literature [1],[2],[11],[12]. All this steps are oriented first of all on detecting main objects of the enterprise which are later presented as entities and attributes in the entity-relationship diagram which is mostly used for presenting the conceptual model. Relationships between objects are defined as well.

The most important rule by conceptual modeling is according to our experiences satisfying user's needs and requests in the frame of defined domain [13]. Expert knowledge of domain is desired. Mostly (except in global enterprises and organizations) are projects local or national oriented. Also in international research projects usually each partner is working on limited working packages with well defined domain [14], what means that the problem of internationalization and/or cultural influence is usually not taken into account. Exceptions are international (global) projects, projects for the customer from another lingual or cultural environment or any multilingual and/or multicultural oriented projects, which have been already as multilingual and/or multicultural requested by the customer definition.

How we have to act in such a case? First of all our conceptual modeling team will grow up. Besides conceptual modeling experts and domain experts, we will need language experts and/or experts that are well known in cultural-communication, to avoid wrong understanding of same or similar objects and theirs relationships in the domain. According to the previous chapter we have to consider first of all professional and functional cultures, business culture and organizational and corporate cultures. Not to forget on one of it or lingual part we have to add some steps to the conceptual model procedure. We are proposing the following steps:

- Introduction of users requests and domain (purpose, users, definitions, related work, basic approach, responsibilities, comments, languages, cultures)
- Detecting elements for conceptual model (entities, relationships, cardinalities, attributes, limitations)
- For selected elements for conceptual model we are preparing translations in requested languages (entities, relationships, attributes) – *valid only for multilingual projects*
- For selected elements for conceptual model we are detecting culture sensitive elements (entities, relationships, attributes)

- List of culture sensitive elements have to be discussed and decisions have to be taken (elimination of some elements, exchange of some elements, renaming of some elements (entities, relationships, attributes)
- Reconstruction of conceptual model according to changes
- Solving open problems

Following the above defined steps we can get confidence and assurance that we will not forget on internationalization or cross cultural influence in the phase of conceptual modeling. Of course new open questions will follow:

- What kind of support we can get from available tools?
- Are available tools appropriate?
- What are benefits what and what are imperfections?
- When and why we are selecting cultural depending conceptual modeling?
- How we will teach users about intercultural dialog?

The list of questions is not closed as well as not all questions would be necessarily considered.

5 Conclusion

In our paper we introduced the problem of internationalization and cultural communication by conceptual modeling. Besides the presentation of the general problems we have also introduced a possible approach in witch some multilingual and multicultural solutions are included for conceptual modeling.

For future research we will concentrate more on detailed presentation of the approach with the working name culture dependent conceptual modeling (design). Formal description and graphical presentation of conceptual model elements will be introduced. We expect that also some tests in multilingual and multicultural environment will be done. Most probably the first tests will be done with help of both students and researchers on exchange.

References

1. Frost, R.: Introduction to knowledge base system. Macmillan Publishing Company, New York (1986)
2. Ramamoorty, C.W., Who, B.W.: Knowledge and Data Engineering. IEEE Transaction on Knowledge and Data Engineering 1(1), 9–17 (1989)
3. Oracle9i Database: The Power of Globalization Technology, Oracle (January 2001)
4. Hogan, J.M., Ho-Stuart, C., Pham, B.: Current Issuess in Software Internationalisation (2003)
5. Heimbuerger, A.: When Cultures Meet: Modelling Cross-Cultural Knowledge Spaces. In: Jaakkola, H., Kiyoki, Y., Tokuda, T. (eds.) Proceedings of the 17th European-Japanese conference on information modelling and knowledge EJC 2007, Tampere University of Technology, Pori, vol. 7, pp. 280–285. Tampere University of Technology, Pori (2007)

6. Lewis, R.D.: When Cultures Collide. Managing Successfully Across Cultures. Nicholas Brealey Publishing, London (1999)
7. Hofstede, G.: Culture's Consequences, Comparing Values, Behaviors, Institutions and Organizations Across Nations. Sage Publications, Thousand Oaks (2001)
8. Hofstede, G., Hofstede, G.J.: Cultures and Organizations: Software of the Mind: Intercultural Cooperation and its Importance for Survival. McGraw-Hill, New York (2004)
9. Magnan, M.: Valeurtech, a Leonardo Pilot Project: Highlighting Professional Experience Acquired in Undergraduate Technology Programmes; EAIE Vienna Conference, Session 4.14 (2003)
10. Christensen, S., Delahousse, H., Profession, B.: Culture and Communication. Institute of Business Administration and Technology Press, Herning (2003)
11. Elmasri, R., Navathe, S.: Fundamentals of Database Systems. Addison Wesley, Boston (2004)
12. Connolly, T., Begg, C.: Database Systems. Addison Wesley, Boston (2004)
13. Welzer, T., Družovec, M.: Similarity search in Database Reusability – a Support for efficient design of conceptual models. In: Hello, P., Welzer, T. (eds.) Contemporary Applications and Research Issues in Industrial Product Modeling, pp. 23–34. University of Vaasa, Vaasa (2000)
14. Kiyoki, Y., Inakage, M., Satoh, M., Tomita, M.: Future Directions of Cyber Knowledge and Databases. In: Proceedings of the IEEE 2004 International Symposium on Applications and the Internet Workshops, pp. 1–6. IEEE Computer Society Press, Los Alamitos (2004)

Operational Business-IT Alignment in Value Webs

Roel Wieringa

University of Twente
Department of Computer Science
roelw@cs.utwente.nl
http://is.cs.utwente.nl/∼roelw

Abstract. Value webs are constellations of profit-and-loss responsible actors that have independent decision-making authority and that have decided to cooperate for a specific purpose. To the extent that the actors are independent, they each decide independently whether to participate in the network, and because they are profit-and-loss responsible, this decision will be based on economic sustainability of the participation. This sustainability depends on the balance between the costs and benefits of participating. The costs are generated by the coordination process and IT infrastructure required to participate; the benefits materialize in the form of commercial transactions enabled by the participation. In this paper I summarize results of recent research into conceptual modeling techniques to design economically sustainable IT-enabled value webs.[1]

Keywords: Value webs, value models, business-IT alignment, coordination, IT services.

1 The Alignment Problem in Value Webs

A value web is a network of profit-and-loss responsible actors that cooperate for the purpose of producing something of value. Examples are value chains, value constellations, joint ventures, or even business units in one company if these are profit-and-loss responsible. To the extent that the units are independent, they will decide on their own whether to participate in the network, and because the actors are profit-and-loss responsible, this decision will based on economic sustainability of the participation for each of the actors itself. In general, value webs are dynamic: Actors may join or leave the network at any time subject to contract restrictions; this is a consequence of the goal of each actor to engage in economically sustainable activities only. Therefore, to design value webs, economic profitability for each of the actors must be taken into account.

[1] This paper reports about work done in the VITAL and COOP projects by Jaap Gordijn, Sybren de Kinderen en Vincent Pijpers of the Free University of Amsterdam and by Lianne Bodenstaff, Pascal van Eck, Andreas Wombacher, Novica Zarvic and myself at the University of Twente. See http://www.vital.org

This is true for value webs in general, and therefore also for the value webs that we are interested in, namely IT-enabled value webs. Examples of such value webs are groups of companies that cooperate to sell tickets on-line, to sell music on-line, or to provide news on-line. The resulting webs have been variously classified as value constellations, virtual organizations, on-line auctions, portals, etc. [11,12]. The characteristic shared by all of these value webs is that they consist of profit-and-loss responsible business actors as well as customers, exchanging services with each other using IT in order to deliver a service to the customer.

The problem in the design of IT-enabled value webs is that an integrated commercial and technical design must be produced. Commercial design cannot be done without sufficient knowledge of IT properties, and IT design cannot be done without a thorough understanding of the commercial impact of the design. The required technical and commercial insight is rarely combined in one person, and even when it is, in order to make the design process manageable, the commercial aspect of the design should be distinguished from the IT aspect.

Since we are dealing with value web design, and not with value web strategy, we will model the commercial and IT aspects of a value web at an operational level. This means that we should include coordination of the actors in the value web as a third aspect to be specified in a value web design. We therefore deal with three aspect models of an IT-enabled value web.

- **Value Model.** The value model of an IT-enabled value web design shows which economic actors participate in the web, what value objects they exchange, and what the net present value of these exchanges for each of the actors is over the intended validity period of the model. The actors represented by the value model are profit-and-loss responsible businesses and their customers. The value objects they exchange are goods, services, experiences, and money.
- **Coordination Model.** The coordination model shows how the commercial exchanges in the value model are operationalized in actions by each of the actors. Where the value model shows that something of value is exchanged for money, the coordination model shows how orders are placed, changed, canceled, confirmed, delivered, paid for, etc. The coordination aspect is present in all value webs, IT-enabled or not. Designing it explicitly has the advantage of showing why coordination activities take place, namely to realize a commercial exchange, and allows value web designers to cut down unnecessary coordination activities.
- **IT Model.** The IT model of an IT-enabled value web shows how coordination is realized by IT-services that the actors deliver to each other. In the extreme case, the entire coordination process is automated by IT. The IT model shows how the coordination activities are mapped to the IT systems owned by the various actors in the web. The IT model is also closely linked to the value model because the value objects exchanged in the value model map to quality properties in the IT model. For example, the value of an

e-service in the value model is often determined by the quality of the IT service that implements the e-service.

The models represent three aspects of an IT-enabled value web for an intended period of time. For example, the designer may decide that a value web is going to evolve over time, as represented by a series of models valid in successive periods of time. Each of the models describes the value web for a particular period of time.

We now frame the operational alignment problem for IT-enabled value webs as the problem of keeping these three aspects models mutually consistent. This is a problem must be solved during value web design, although we should realize that value web design is never finished. Value web dynamics involves periodic redesign of the web based on evaluation of past performance. In the following sections I review some techniques for aligning coordination models and IT models with value models developed in our recent research.

2 Value Models

We use the e^3-*value* notation for representing value webs [5,6]. Figure 1 shows a simple value model consisting of two business actors, seller and carrier, and one market segment consisting of buyers. The seller sells a good to a buyer for a price, and it acquires a transport service from the carrier against a fee. The bullet represents the occurrence of a customer need, and the dashed line, called a dependency path, connects the customer need with all the commercial transactions that are executed to satisfy this need. The dependency path is used to do profitability computations because it allows us to connect estimations of the number of consumer needs in a time period with estimations of the number and value of commercial transactions needed to satisfy this need. The bull's eye represents the final point of the profitability computation.

By focusing on commercial transactions only, each actor must make clear what it receives in return for the value objects it delivers to other actors. This turns out to be surprisingly difficult, and very useful, because it forces every actor to reconsider in commercial terms why they participate in this value web. Case studies with e^3-*value* have been done in the music business [9], health care [8], and the energy distribution business [3,7], among others.

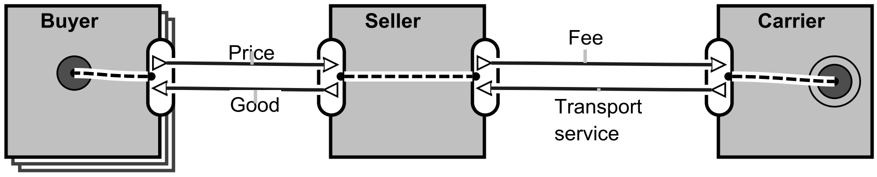


Fig. 1. A value model representing the sale of a good and of a transport service. When a customer need occurs, both transactions take place in the time period represented by the model.

3 Aligning Value Models and Coordination Models

The value model shows that the carriers delivers a transport service to the seller but it does not show how this service is delivered: By carrying a good from buyer to seller. This is shown in the coordination model of figure 2, where we use a UML activity diagram to represent the coordination process. There is one swimlane for each actor, and the diagram shows that the buyer sends a purchase order (P.O.) to the seller, who sends an acknowledgment back and then starts two parallel processes: Requesting transport from the carrier, and sending ownership documents to the buyer. The buyer pays on receiving these documents, and uses the documents to obtain the goods from the carrier when they arrive.

To obtain a coordination model from a value model, two things must be done.

- We must replace the value exchanges of the value model with physical exchanges that will implement these value exchanges.
- We must add operational information about the actions that perform these physical exchanges.

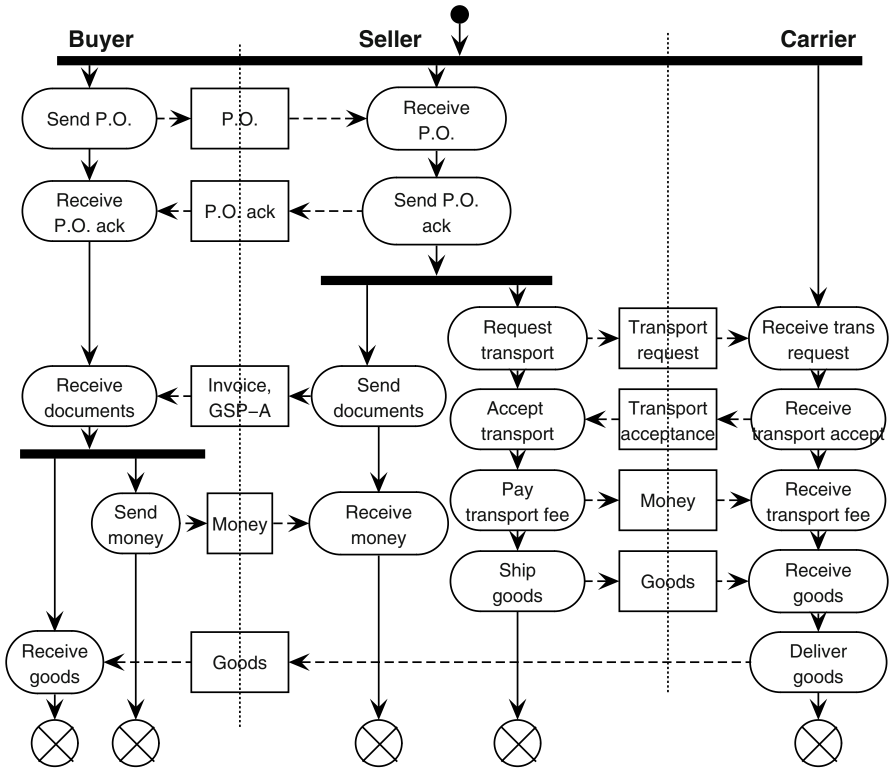


Fig. 2. A coordination process that correctly aligns with the value model of figure 1

The value model shows which value objects are transferred between actors. A value object can be realized by three kinds of operational objects.

- **Physical oBJECTS.** Examples of physical objects are physical goods and sense objects (music, sounds, images, etc.).
- **Documents.** Examples of documentary objects are paper and digital documents (containing text, data, graphics etc.)
- **Social Interactions.** Examples of social interactions are individual interactions (e.g. consultancy, coaching) and group interactions (performances, games, etc.).

Ultimately, these objects are all physical, for they consist of the movement of matter or energy. This is important because in the coordination model we want to include only objects that can be moved around physically. Abstract commercial values and mathematical entities cannot be moved around, but physical objects such as paper or electronic documents, physical goods and human beings can be moved from one place to another. Figure 2 shows physical goods and documents (such as a purchase order, an invoice, transport requests and money).

The relation between the physical objects in the coordination model and the value objects in the value model is that the physical object manipulations realize change of ownership. The value transfers in a value model are really changes of ownership: The receiver obtains the right to claim and use the value object [10]. Ownership is however a legal abstraction and in the physical world we live in, this is realized by changing the physical state of physical objects (e.g. by writing a signature on a document) and by moving these physical objects to the appropriate place. For example, a purchase order will be signed by the buyer and then sent to the seller. The ability to physically manipulate a physical object is called possession. Where the buyer possesses the purchase order when he signs it, the seller possesses it when he receives it. And the physical goods in figure 2 are first possessed by the seller, then by the carrier, and finally by the buyer. But they are never owned by the carrier. Ownership changes from seller to buyer directly, as represented by the value model. Maintaining consistency between physical object transfers in the coordination model and value object transfer in the value model is the first element of the operational alignment of value models and coordination models.

Once we know which physical goods and documents change hands, and which social interactions must take place, we can flesh out the coordination process by including the activities to be performed by the actors to realize these manipulations. This introduces the second consistency requirement between value- and process models, namely that for each sequence of commercial transaction along a dependency path there must be an execution sequence of object transfer activities in the coordination model, and vice versa. This is called static consistency of value– and process models [2,13,15].

A third element of value model– coordination model alignment is dynamic consistency: Are the estimations of the numbers and values of commercial transactions in the value model for a particular time period, consistent with the actually occurrences of physical transfers in a log of all executions of a coordination process in that time period? This is subject of current research [1].

A fourth element of operational alignment concerns the amount of trust that business partners have in each other. For example, in overseas trade, sellers do not want to send the goods before they receive payment but buyers do not want to pay before receiving the goods. This is solved routinely by a letter of credit issued by a bank, which gives the seller the security that he will receive payment from the bank once the goods have arrived, and gives the buyer the security that the seller will ship the goods if he pays. This however introduces an additional party to the model, a bank, which needs to be paid for its services, and so the value model must be adapted to it [13] and the coordination process must then be aligned with this new value model. In general, value– and coordination modeling is a process of mutual adjustment until all actors agree to both models, and the models are mutually consistent.

4 Aligning Value Models with IT Services

IT can facilitate value transfers by playing various roles in the underlying physical object transfers.

- Execution. All document flows, some sense objects (sounds, images), and some social interaction (teaching, gaming), can be implemented through online channels.
- Registration. Any physical transfer can be registered in IT if needed, e.g. for accounting purposes.
- Monitoring. Any physical transfer, and indirectly any value transfer, can be monitored, e.g. to check compliance with contract requirements.
- Control. Some physical transfers can be triggered automatically by the occurrence of temporal events or conditions. For example some transfers can be triggered periodically or when a monitored condition occurs.

These design choices too will reflect on the value model because the chosen implementation technology will affect the estimations of profitability made in value modeling. Expenses made to implement the physical object transfers must be deducted from the revenues generated by the value transfers in the value model [4,14]. IT implementation choices may change the value model, hence the coordination model, and therefore also the IT model. The mutual alignment between the three models goes three ways.

To make trade-offs between the different design choices here, more work needs to be done on the impact of quality of service provided by IT and the estimated value of the value transfers in the value model. Quality is benefit for the user, and benefit for the user is value for the user. For example, provision of payment has a lower value for most users than provision of secure payment. Therefore, a value object like “secure pauyment” maps onto quality requirements in the IT model. This is subject of current research [3].

5 Conclusions

The research reported in this overview is in full steam and as is usual with a generative research problem, we have generated research questions faster than we could solve them. So far, the growing number of real life case studies done with the conceptual modeling techniques sketched in this paper increases our confidence that this approach will have practical use. Performing action research projects—dong real-life cases and reflecting on them—has proven to be a fruitful way of answering research problems that are relevant for practice.

References

1. Bodenstaff, L., Reichert, M., Wieringa, R.: Towards the integration of value and coordination models. In: Proceedings of Workshops and Doctoral Consortium of the 19th International Conference on Advanced Information Systems Engineering, pp. 291–298 (2007)
2. Bodenstaff, L., Wombacher, A., Reichert, M.: Dynamic consistency between value and coordination models – research issues. In: Meersman, R., Tari, Z., Herrero, P. (eds.) OTM 2006 Workshops. LNCS, vol. 4277, pp. 802–812. Springer, Heidelberg (2006)
3. Derzsi, Z., Gordijn, J., Kok, K., Akkermans, H., Tan, Y.H.: Assessing feasibility of it-enabled networked value constellations: A case study in the electricity sector. In: Krogstie, J., Opdahl, A., Sindre, G. (eds.) CAiSE 2007 and WES 2007. LNCS, vol. 4495, pp. 66–80. Springer, Heidelberg (2007)
4. van Eck, P.A.T., Yamamoto, R., Gordijn, J., Wieringa, R.J.: Cross-organizational workflows: A classification of design decisions. In: Fifth IFIP Conference on e-Commerce, e-Business and e-Government, October 26-28 (2005)
5. Gordijn, J., Akkermans, H.: Designing and evaluating e-business models. IEEE Intelligent Systems 16(4), 11–17 (2001)
6. Gordijn, J., Akkermans, H.: Value-based requirements engineering: exploring innovative e-commerce ideas. Requirements Engineering 8, 114–134 (2003)
7. Gordijn, J., Akkermans, H.: Business models for distributed energy resources in a liberalized market environment. The Electric Power Systems Research Journal 77(9), 1178–1188 (2007)
8. Kartseva, V., Hulstijn, F., Gordijn, J., Tan, Y.-H.: Modelling value-based inter-organizational controls in healthcare regulations. In: Suomi, R., Cabral, R., Felix Hampe, J., Heikkila, A., Jarvelainen, J., Koskivaara, E. (eds.) Proceedings of the 6th IFIP conference on e-Commerce, e-Business, and e-Government (I3E 2006), IFIP: International Federation for Information Processing, pp. 278–291. Springer, Heidelberg (2006)
9. Pedrinaci, C., Baida, Z., Akkermans, H., Bernaras, A., Gordijn, J., Smithers, T.: Music rights clearance: Business analysis and delivery. In: Bauknecht, K., Pröll, B., Werthner, H. (eds.) EC-Web 2005. LNCS, vol. 3590, pp. 198–207. Springer, Heidelberg (2005)
10. Pijpers, V., Gordijn, J.: Bridging business value models and business process models in aviation value webs via possession rights. In: Proceedings of the 20th Annual Hawaii International Conference on System Sciences, Computer Society Press (2007)

11. Tapscott, D., Ticoll, D., Lowy, A.: Digital Capital – Harnessing the Power of Business Webs. Nicholas Brealy Publishing (2000)
12. Weill, P., Vitale, M.R.: Place to Space: Migrating to eBusiness Models. Harvard Business School Press (2001)
13. Wieringa, R.J., Gordijn, J.: Value-oriented design of correct service coordination protocols. In: 20th ACM Symposium on Applied Computing, March 13-17, 2005, pp. 1320–1327. ACM Press, New York (2005)
14. Zarvic, N., Wieringa, R., van Eck, P.: Checking the alignment of value-based business models and it functionality. In: 23rd Annual ACM Symposium on Applied Computing (accepted, 2008)
15. Zlatev, Z., Wombacher, A.: Consistency between e3-value models and activity diagrams in a multi-perspective development method. In: Meersman, R., Tari, Z. (eds.) OTM 2005. LNCS, vol. 3760, Springer, Heidelberg (2005)

Towards Virtual Sales Assistants: State-of-the-Art and Research Challenges

Markus Zanker

University Klagenfurt,
Intelligent Systems and Business Informatics Group,
9020 Klagenfurt, Austria
markus.zanker@uni-klu.ac.at

Abstract. The metaphor of a *virtual sales assistant* has become commonplace in many online shops over the past decade. It raises expectations that Web applications serve online customers in the same way as sales persons of brick & mortar businesses do. However, in reality this promise is rarely kept. In many online shops the term is misused and denotes simple search tools or static query forms that do not provide personalized interactive behavior - let alone a social or emotional perspective.

The purpose of this talk is to give an overview on current state-of-the-art techniques for Web personalisation and user interaction with e-commerce platforms and present selected examples. Furthermore, current research results are given and challenges for future work discussed.

An Investigation into Improving the Load Balance for Term-Based Partitioning

Ahmad Abusukhon[1], Mohammad Talib[2], and Michael P. Oakes[1]

[1] School of Computing and Technology
University of Sunderland
Tel.: +44 (0)191 515 3631
ahmad.abusukhon@sunderland.ac.uk, michael.oakes@sunderland.ac.uk
[2] Department of Computer Science
University of Botswana
Private Bag UB 00704, Gaborone
Tel.: +267 355 2129
talib@mopipi.ub.bw

Abstract. In Parallel (IR) systems the query response time is limited by the time of the slowest node in the system, thus distributing the load equally across the nodes is very important issue. In this paper, we propose improving the load balance for term-based partitioning by classifying the terms based on their length then distribute them equally across nodes. The motivation for term length partitioning comes from the observation that the Excite-97 queries have a very skewed distribution of term lengths with some predominant lengths. We also propose the term-frequency partitioning scheme in which the terms are classified based on the total term frequency (F) and then distribute equally across the nodes.

Keywords: Term-partitioning schemes, Term-frequency partitioning, Term-length partitioning, Node utilization.

1 Introduction

The first step in developing information retrieval system is to decide on what access method should be used in order to access large-scale collection efficiently. The most known structures for building the index of large-scale collection are inverted files and signature files. The most common and most efficient structure for building the index of large-scale collection is the inverted files, [1,2].

Zobel[3] compared inverted files and signature files with respect to query response time and space requirements. They found that inverted files evaluate queries in less time than signature files and need less space, thus for efficiency reasons, we use the inverted files in our research.

In general, inverted files consist of vocabulary and a set of inverted lists. The vocabulary contains all unique terms in the whole data collection; while the inverted lists composed of a list of pointers and each pointer consists of document identifier and term frequency. The term frequency in each pair represents how many times term i appears in document j ($F_{i,j}$). Let's suppose that the inverted list for term "world" is:

world 2:5, 6:3, 12:1, 15:1

This means that term world appears five times in document 2, three times in document 6, one time in document 12, and one time in document 15. The numbers 2, 6, 12, and 15 are called the document identifiers while the numbers 5, 3, 1, and 1 are called the term frequencies.

In parallel IR system when term partitioning scheme is used all unique terms in the data collection and their inverted lists reside on a single node called the broker. The broker distributes all terms and their inverted lists across nodes using different approaches. The terms, for instance, may be distributed in round robin fashion. In this case the broker iterates over all terms in the inverted file, and distributes them sequentially across nodes. The aim of round robin partitioning scheme is to balance the load over the nodes by storing nearly equal number of terms on all nodes.

Moffat[4] showed that round robin partitioning scheme results in load imbalance especially when there are heavy loaded terms. Because of this the round robin partitioning scheme does not take care of those terms to be distributed equally across nodes.

Xi[5] proposed the hybrid partitioning scheme in order to achieve load balance. In hybrid partitioning scheme the inverted lists are split into chunks then chunks are distributed across nodes. They investigated partitioning the inverted list into different sizes and they concluded that hybrid partitioning scheme achieves better load balance than the other schemes when the chunk size is small (1024 posting). But when the chunk size is large the hybrid partitioning is worse than the document partitioning.

In this paper, we propose two methods for term partitioning scheme - term length partitioning and term frequency partitioning.

1.1 Related Work

Inverted files can be partitioned by different approaches. In this section we shed light on various strategies for term-partitioning schemes.

Cambazoglu[6] demonstrated two main types for inverted file partitioning - term-based partitioning and document-based partitioning. In term-based partitioning all unique terms in the data collection and their inverted lists reside on a single node. In document-based partitioning the data collection is divided into sub-collections, sub-collections are distributed across nodes, and then each node builds its own index.

Jeong [7] proposed two methods for load balancing when using term-based partitioning scheme. In the first method they proposed to split the inverted list into equal parts and then distribute those parts across nodes instead of distributing equal number of terms across nodes in order to achieve better load balance. In the second method they proposed to partition the inverted lists based on the access frequency of terms in the user query and the inverted list size for each term appears in the query. They studied the performance of the above schemes by simulation under different workloads.

Xi[5] proposed a hybrid partitioning scheme in order to distribute terms across the nodes. Hybrid partitioning scheme avoids storing terms with long posting lists on one node instead of the inverted list of a given term is split into a number of equal size chunks and then distributed randomly across the nodes. They measured the load balance and concluded that the hybrid-partitioning scheme outperforms other schemes when the chunk size is small.

Moffat[4] examined different methods to balance the load for term-distributed parallel architecture and proposed different techniques in order to reduce the net querying costs. They defined the workload as follows:

$$Wt = Qt * St$$

Where Wt is the workload caused by the term t that appears in a query batch Qt and has an inverted list of length equals St bytes. The workload for a given node is the sum of Wt of the terms distributed over that node. In one of their experiments the terms of the queries were distributed over the nodes randomly. The simulation result showed that some nodes were heavy-loaded because they retrieved very large size inverted lists; therefore, some of the nodes in the system were half-idle and affect the system throughput. In order to improve the load balance they proposed distributing the inverted lists equally among the nodes based on the number of pointers P in each inverted list.

1.2 System Architecture

Fig.1 shows our system architecture. It consists of six nodes and one broker. All nodes are connected to the broker via Ethernet switch.

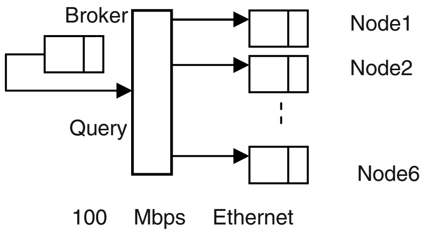


Fig. 1. Distributed IR Architecture

The machine specifications for five nodes are: CPU 2.80Ghz RAM 256MB whereas the specification for the last and the broker are: CPU 3.00Ghz, RAM 512MB. All machines are running in Windows XP environment.

2 Research Methodology

We carried-out a set of real experiments using six nodes and one broker as shown in Fig. 1. In all of our experiments we use the data collection WT10G from TREC-9 and 10,000 queries extracted from the start of Excite-97 log file. The chronology of our research methodology is traced below:

1. We build the global index (called the term-based partitioning) in the following way:
   a. Broker sends the documents across nodes in round robin fashion.
   b. Each node when receiving its document performs these activities.
      - Filters the document it receives from stop words (the stop word list consists of 30 words)

- Filters the document it receives from HTML tags
- Filters the document it receives from non-characters and non-digits terms
- Accumulates the posing lists in main memory until a given memory threshold is reached. At this point the data stored in memory is flushed to on-disk file [8,9, 2,11].
- This process is repeated until all documents in the data collection are indexed
- Merges all on-disk files together into one on-disk file called the local index or the document-based partitioning.

c. Finally, the broker collects all local indices from all nodes and merges them together in order to produce the global index.

2. We partition the terms of the global index across nodes using four different approaches, viz., round robin partitioning, partitioning based on the length of the inverted list, term length partitioning and term frequency partitioning.

Next, we demonstrate the above approaches and then run a set of real experiments in order to compare them with respect to the node utilization.

2.1 Round Robin Partitioning

In round robin partitioning, we distribute the terms of the global index across nodes. If we have three nodes and four terms A, B, C, and D associated with their posting lists then term A may reside on node 1, term B on node 2, term C on node 3, and term D on node 1, and so on [10].

2.2 Term Partitioning Based on the Length of the Inverted List

In this method of partitioning, we pass over the terms of the global index twice. In the first pass, we calculate the length of the inverted list L for each term T, store T and L in a look up file PL after sorting them on L in ascending order.

In the second pass, we distribute the terms and the inverted lists of the global index across the nodes using the PL in round robin fashion in the follow order:

1. Read one record (L, T) from PL
2. Search T in the global index and retrieve its inverted list
3. Send T and its inverted list to a certain node in round robin fashion.
4. If no more records then EXIT else go to step1.

Inverted List length	Term
100	a
100	b
.	.
.	.
1200	z

Fig. 2. Sorted look up file (PL)

We use the above algorithm in order to guarantee that all inverted lists of the same length reside on all nodes equally. Fig. 2 shows an example of the PL file.

2.3 Term Length Partitioning

In this section, we propose to partition the terms of the global index associated with their inverted lists with respect to the term length (in letters).

Our research hypothesis for term length partitioning is based on statistical information collected from the query log file Excite-97. This information is stored in a look up file.

In Fig. 3, we can see that the term lengths are not distributed equally in Excite-97 (i.e. have a very skewed distribution). For example, the number of terms of length 5 equals 360093 while the number of terms of length 11 equals 59927. The total number of terms in Excite-97 is 2235620. Thus the percentage of the terms of length 5 to the total number of terms =360093/ 2235620 = 0.16% while the percentage of terms of length 11 = 59927/ 2235620 = 0.03%.

When users submit their queries, the queries contain terms of different length. Suppose that the majority of those terms are of length 4 and 5 as it is shown in Fig. 3. In addition, we partitioned the terms of the global index in round robin fashion and all terms of length 4 and 5 resided on one or two nodes. This way of partitioning will result in load imbalance because most of the work will be carried out by one or two nodes only while other nodes doing less work or may be idle.

Thus our hypothesis is that all terms of the same length must be distributed equally a cross nodes in order to achieve more load balance.

Our partitioning method requires passing over all terms of the global index twice. In the first pass, we calculate the length W_L of each term T, store W_L and T in a look up file PL after sorting them on W_L in ascending order.

In the second pass, we distribute the terms and the inverted lists of the global index across nodes using the PL in round robin fashion in the following order:

1. Read one record (W_L, T) from PL
2. Search T in the global index and retrieve its inverted list
3. Send T and its inverted list to a certain node in round robin fashion.
4. If no more records then EXIT else go to step1

We use the above algorithm in order to guarantee that all terms of the same length reside on all nodes equally.

Our proposed partitioning algorithm differ from round robin partitioning in that it distributes the terms across the nodes equally, and it also guarantee that all nodes get the same number of terms of the same length. The round robin algorithm distributes the terms across the nodes equally regardless the term length. Therefore, we expect that the term length-partitioning scheme achieves better load balance than the round robin partitioning.

To the best of our knowledge, no previous work investigated partitioning the global index based on the term length, or measured the nodes utilization when using the term length partitioning.

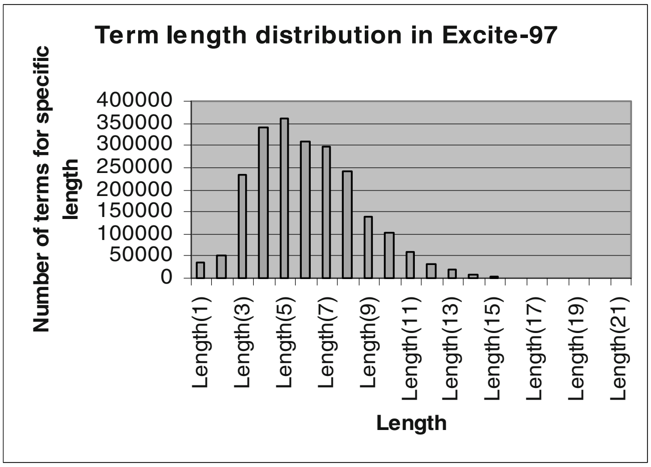


Fig. 3. Term length distribution for Excite-97

2.4 Term Frequency Partitioning

Baeza[1] demonstrated Zipf's law (Fig.4), which is used to capture the distribution of the frequencies of the words in a given text. This law stated that "the frequency of the i-th most frequent word is $1/i^r$ times that the most frequent word", (r between 1.5 and 2.0), thus the frequency of any word is inversely proportional to its rank (i.e. i-th position) in the frequency table.

They showed that the distribution of sorted frequencies (decreasing order) is very skewed (i.e. there were a few hundred words which take up 50% of the text), thus words that are too frequent like stop words can be ignored.

In Fig. 4, graph (A), shows the skewed distribution of the sorted frequencies, while graph (B) is the same as graph (A) but we divided the curve into six clusters (A, B, C, D, E, F) after ignoring the stop words. Cluster (A) has the most frequent terms, then cluster B, then C, and so on. In addition, in graph (B), we assumed that most or all of the query terms appear in cluster (A) that all or most of the terms in cluster (A) may not reside on all nodes but on one or two nodes in the system. In this case, we may have one or two nodes busy answering the query terms while other nodes are idle, and thus cause the load imbalance.

Our hypothesis is that if we filter the data collection from stop words (words like the, in, on, …, etc), then the terms with high total frequency (for example the terms in cluster A) are more likely to appear in the user query (i.e. have higher probability to appear in the user query) than the terms with low total frequency. Thus, the terms with high frequency must be distributed equally across the nodes in order to achieve more load balance.

Here we propose to partition the terms of the global index with respect to the total term frequency calculated from their inverted lists. To make it clear what we mean by term frequency, we demonstrate the following example: Let's suppose we have two terms (A, B) associated with their inverted lists as it is shown in table 1.

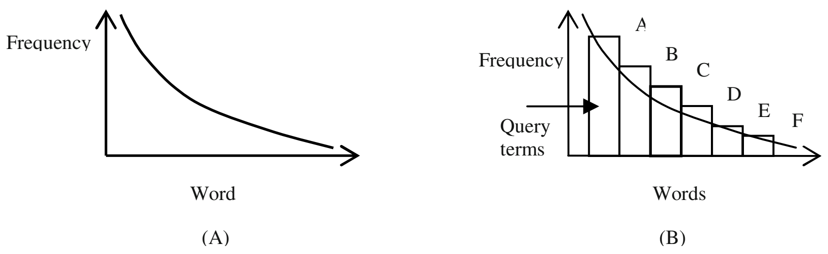


Fig. 4. Zipf's Law

Table 1. Inverted lists

term	Inverted list
A	1:3, 4:1, 6:2, 9:5, 12:5, 13:2, 15:3
B	2:1, 4:1, 7:2, 9:2, 11:1, 12:1

Then, the total frequency F for each term is calculated as follows:

$F_A = 3+1+2+5+5+2+3 = 21$
$F_B = 1+1+2+2+1+1= 8$

Based on the above calculations, the total frequency of term A is higher than the total frequency of term B and thus we expect that term A has higher probability to appear in the user query than term B. We expect that the load imbalance may occur, if the majority of the terms with higher total frequency reside on one or two nodes, as a result of performing some techniques like round robin partitioning, in this case, most of the user query terms are answered by one or two nodes and thus cause the load imbalance.

In the next section, we show how to calculate the probability of a given term to appear in the user query terms.

2.4.1 Calculate the Term Probability

Suppose that we have the document collection Ω where:

$$\Omega = \{D_0, D_1, D_2, \ldots, D_n\}$$

Let $T = \{t_0, t_1, t_2, \ldots , t_n\}$ be the set of terms appears in any document D_i in any combination. Let the term t_j occurs m times in D_i , then we assume that the probability (Pt_{ji}) that the term t_j appears in the query terms is equivalent to how many times it occurs in the whole data collection

$$p_{tj} = \sum_{i=1}^{n} m_{j,i} \tag{1}$$

Where: n is the total number of documents in the data collection, and $m_{j,i}$: is how many times the term j appears in document i.

For example, suppose we have a data collection contains 4 documents (d1, d2, d3, and d4), and three terms (t1, t2, and t3) and that t1 appears in these documents (5, 10, 2, 3) times, t2 appears (1, 3, 0, 1) and t3 appears (1, 1, 1, 2). We assume that the probability of term t1 to appear in the query terms is 20, t2 is 5 and t3 is 5.

To normalize the value of Ptj, we divided it by the summation of the total frequencies of all distinct terms in the data collection, i.e.

$$p_{tj} = \frac{\sum_{i=1}^{n} m_{j,i}}{\sum_{k=1}^{n}\sum_{l=1}^{s} m_{l,k}} \tag{2}$$

where n : total number of documents in the data collection and s : total number of distinct terms in the data collection.

In the above example, after normalization, the probability of term t1 = 20/ 30 (i.e. 0.7), while the probability of term t2 = 5/30 (i.e. 0.17).

2.4.2 Term Distribution Based on the Total Term Frequency

We pass over all terms of the global index twice. In the first pass, we calculate the total frequency F for each term T using equation 1 then store F and T in a look up file PL after sorting them on F in ascending order.

In the second pass, we distribute the terms and the inverted lists of the global index using the PL in round robin fashion in the following order:

1. Read one record (F, T) from PL
2. Search T in the global index and retrieve its inverted list
3. Send T and its inverted list to a certain node in round robin fashion
4. If no more records then EXIT else go to step1.

The above algorithm is used in order to guarantee that all terms of the same total frequency F are distributed across all nodes equally.

To the best of our knowledge, no previous work investigated partitioning the global index based on the total term frequency, or measured the nodes utilization when using the term frequency partitioning.

3 Experiments

In this research, we carried out a set of real experiments using the system architecture shown in Fig. 1. We used the data collection WT10G from TREC-9 in order to build the global index and 10,000 queries extracted from the start of the Excite-97 log file in order to measure the node utilization for each node.

Xi[5] defined the node utilization as – “the total amount of time the node is serving requests from the IR server divided by the total amount of time of the entire experiment”.

We distributed the terms of the global index using the four approaches mentioned in sections 2.1, 2.2, 2.3, and 2.4. We carried out 10,000 queries, each query is sent across all nodes. Each node retrieves and sends the inverted lists of the query terms to the broker for evaluation. We considered the time the node serving the query S_t to be the time required to retrieve all inverted lists of query terms and send them to the broker. We considered the node to be idle if the query term does not exist on its hard disk in that case, the searching time is excluded from S_t.

The total time for each experiment is shown in table 2-A.

Table 2-A. Total time of experiments

Term partitioning method	Total time of experiment (milliseconds)
Round Robin	3349515
Length of inverted list	3259879
Term frequency	3303175
Term length	3528047

For each partitioning scheme mentioned in sections 2.1, 2.2, 2.3, and 2.4, we calculated ΔU:

$$\Delta U = \text{Maximum node utilization} - \text{Minimum node utilization}$$

$$= \text{MaxU} - \text{MinU} \quad (3)$$

Tables 2-B, 2-C, 2-D, and 2-E, show the time taken by each node to serve 10,000 queries sent by the broker as well as the node utilization. We calculated the node utilization by

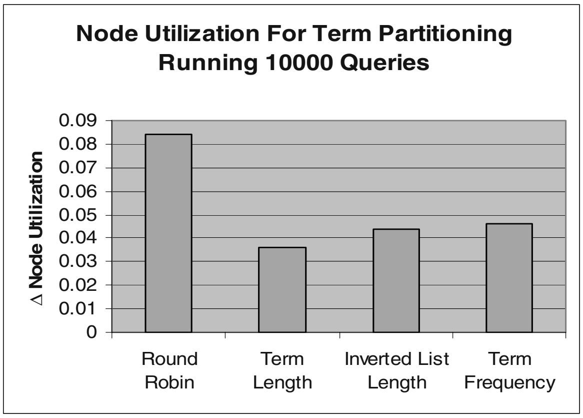


Fig. 5. Comparison between four approaches for term partitioning scheme (Round robin, Term Length, inverted List Length, and Term frequency) with respect to ΔU

dividing the time the node serving queries by the total time of the experiment. For example, the node utilization for node 1 (Table 2-B) is calculated as follows:

$$\text{Node utilization} = 598195 / 3349515 = 0.1785.$$

This calculation step is carried out for all tables (2-B, 2-C, 2-D, and 2-E). Next, we produce table 2-F from the above tables.

For each table we got the minimum and the maximum node utilization (MinU, MaxU). For table 2-B, MinU = 0.1104 and MaxU= 0.1947 then we calculate ΔU

$$\begin{aligned}\Delta U &= \text{MaxU} - \text{MinU}\\ &= 0.1947 - 0.1104\\ &= 0.0843\end{aligned}$$

We used table 2-F to produce Fig. 5 and we calculated the average query response time for each of the above partitioning algorithms by dividing the total time of the experiment by the total number of the executed queries. For round robin partitioning scheme:

$$\begin{aligned}\text{The average query response time} &= 3349515 / 10000\\ &= 334.95 \text{ milliseconds}\end{aligned}$$

Table 3 and Fig. 6, show the partitioning methods and the average query response time.

Table 2-B. Node utilization for Round Robin partitioning

Node #	Time serving queries (milliseconds)	Node utilization
1	598195	0.1785
2	541421	0.1616
3	369798	0.1104
4	652215	0.1947
5	628682	0.1876
6	604870	0.1805

Table 2-C. Node utilization for partitioning based on the length of inverted list

Node #	Time serving queries (milliseconds)	Node utilization
1	667148	0.2046
2	596106	0.1828
3	640117	0.1963
4	699275	0.2145
5	647914	0.1987
6	581225	0.1782

Table 2-D. Node utilization for partitioning based on the total term frequency

Node #	Time serving queries (milliseconds)	Node utilization
1	559151	0.1692
2	640726	0.1939
3	590804	0.1788
4	594265	0.1799
5	667375	0.202
6	711796	0.2154

Table 2-E. Node utilization for partitioning based on term Length

Node #	Time serving queries (milliseconds)	Node utilization
1	596510	0.1690
2	535703	0.1518
3	689623	0.1954
4	625451	0.1772
5	629086	0.1783
6	618969	0.1754

Table 2-F. Δ Node utilization

Term Partitioning Scheme	Δ Node Utilization (Max - Min)
Round Robin	0.0843
Term Length	0.0363
Inverted List Length	0.0436
Term Frequency	0.0462

Table 3. Average query response time (milliseconds)

Partitioning Method	Average Query Response Time (milliseconds)
Length of inverted list	325.9879
Round Robin	334.9515
Term frequency	330.3175
Term length	352.8047

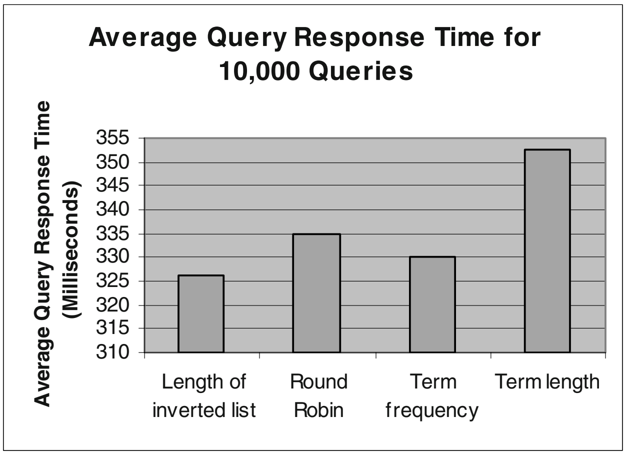


Fig. 6. Average query response time

4 Conclusion and Future Work

In this paper, we carried out a set of real experiments using our parallel IR system in order to improve the load balance for term partitioning scheme. We proposed to partition the terms of the global index based on term length and the total term frequency extracted from the inverted lists.

We compared our proposed methods with round robin partitioning scheme and the partitioning scheme based on the length of the inverted list. Our results showed that the term length-partitioning scheme performed better than other schemes with respect to node utilization. On the other hand, partitioning terms based on the length of the inverted list achieved less average query response time than other schemes (Fig. 5 and Fig. 6).

References

1. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. ACM Press, Addison-Wesley, New York (1999)
2. Zobel, J., Moffat, A.: Inverted Files for Text Search Engines. ACM Computing Surveys (CSUR) (2006)
3. Zobel, J., Moffat, A., Ramamohanarao, K.: Inverted Files Versus Signature Files for Text Indexing. ACM Transactions on Database systems, 453–490 (1998)
4. Moffat, A., Webber, W., Zobel, J.: Load Balancing for Term-Distributed Parallel Retrieval. In: The 29th annual international ACM SIGIR conference on Research and development in information, pp. 348–355. ACM, New York (2006)
5. Xi, W., Somil, O., Luo, M., Fox, E.: Hybrid partition inverted files for large-scale digital libraries. In: Proc. Digital Library: IT Opportunities and Challenges in the New Millennium, Beijing Library Press, Beijing, China (2002)

6. Cambazoglu, B., Catal, A., Aykanat, C.: Effect of Inverted Index Partitioning Schemes on Performance of Query Processing in Parallel Text Retrieval Systems. In: Levi, A., Savaş, E., Yenigün, H., Balcısoy, S., Saygın, Y. (eds.) ISCIS 2006. LNCS, vol. 4263, pp. 717–725. Springer, Heidelberg (2006)
7. Jeong, B.S., Omiecinski, E.: Inverted File Partitioning Schemes in Multiple Disk Systems. IEEE, Transactions on Parallel and Distributed Systems 6(2), 142–153 (1995)
8. Heinz, S., Zobel, J.: Efficient Single-Pass Index Construction for Text Databases. Journal of the American Society for Information Science and Technology 54(8), 713–729 (2003)
9. Jaruskulchai, C., Kruengkrai, C.: Building Inverted Files Through Efficient Dynamic Hashing (2002)
10. Badue, C., Baeza-Yates, R., Ribeiro-Neto, B., Ziviani, N.: Distributed Query Processing Using Partitioned Inverted Files, pp. 10–20 (2001)
11. Lester, N., Moffat, A., Zobel, J.: Fast On-Line Index Construction by Geometric Partitioning. In: Proceedings of the 14th ACM international conference on Information and knowledge management, CIKM 2005, Bremen, Germany, October 31– November 5, pp. 776–783. ACM, New York (2005)

A UML-Rewriting Driven Architectural Proposal for Developing Adaptive Concurrent IS

Nasreddine Aoumeur and Gunter Saake

ITI, FIN, Otto-von-Guericke-Universität Magdeburg
Postfach 4120, D–39016 Magdeburg, Germany
{aoumeur,saake}@iti.cs.uni-magdeburg.de

Abstract. This contribution investigates the application of *behavior-driven* transient architectural connectors to rigorously and coherently unifying UML structural- with behavioral-diagrams. Indeed, existing UML-Architectural proposals drastically neglect behavioural issues, ignoring thereby vital concurrent and evolving behaviors in today's cross-organizational agile complex information systems (IS).

A stepwise approach is put forwards. We first conceptualize complex IS with class-and object-diagrams, OCL constraints and state-charts. In the second main step towards componentization and distribution, we *explicitly distinguish* local properties from to-be observed ones and then leverage the later as component interfaces. By adopting a very appealing simple stereo-typed notations on such structural interfaces, we increment them with *behavioral connectors*, by exploiting OCL pre- post-constraints, state transitions and related business rules. Finally, we endow this new unified UML-compliant architectural modelling proposal with a true-concurrent operational semantics based on rewriting logic.

Keywords: UML, Architectural connectors, Rewriting logic, Distributed IS, Evolution and Adaptability.

1 Introduction

The Unified Modeling Language (UML) has been emerging as the de-facto standard framework for modeling and implementing complex software-intensive systems in general and information systems in particular [2]. At least three main strengths are making most of system developers embracing UML. Firstly, it has been the fruit and the confluence of most successful object-oriented (OO) methodologies and approaches, including Jacobson's OOD, Rumbaugh's OMT and Booch's OOA. Secondly, UML offers several (structural, dynamic, behavioral) views and perspectives through very appealing semi-graphical and textual diagrams such as class- object-diagrams for structure, OCL-constraints for dynamics, sequence- state-charts- and collaborative diagrams for behavioral descriptions. Thirdly, UML is by nature very flexible and extensible via stereotype mechanisms and profiles. Moreover, the UML itself has underwent several improvements and extensions, growing from UML 1.1 to current UML 2.0 [9] which includes most of practitioner / researcher feedbacks and (good and bad) experiences during the last decade. Last but least, UML is supported by advanced tools including Rational Rose, Poseidon for editing, checking and generating Java codes from different UML-diagrams.

The great benefits of bringing together architectural techniques [11] and UML artifacts have attracted several researchers and practitioners giving birth to several variants of UML-ADLs as well as UML component-based framework [5]. Among these benefits we may stress: (1) The high-abstraction level of architectural artifacts (e.g. component interfaces, connectors, configurations) permit leveraging OO concepts and UML-diagrams focus from inter-component computations to explicitly *interacting* components; (2) The exploitation of architectural connectors as first-class modelling artifact leads to more dynamic and evolution, as they allow extracting cross-component (non-)functionalities at the interaction level; (3) The intrinsic support of distribution and concurrent nature of ADLs languages circumvents the centralized holistic and sequential nature of UML diagrams.

Nevertheless, despite these expected benefits of bringing architectural techniques to UML diagrams, when assessing existing UML-driven architectural approaches, we observe that heavy emphasis has been put only on UML *structural features* and how they may fit / be-extended with ADLs—with class-diagrams playing the driving force. We find this state-of-affair very unfortunate as modern IS are extremely behavior-intensive, fully distributed and networked and very adaptive and agile to ensure competitive and quickly respond to highly volatile market and laws via dynamic cross-organizational cooperation.

We thus aim in this contribution leveraging the mostly adopted UML semi-graphical and textual (e.g. OCL constraints) diagrams while developing complex IS towards a more *behavioral*, *rigorous* and *evolving* architectural-driven conceptual modelling. Moreover, we strive for an approach where validation through symbolic proof-based computation is intrinsic. We achieve this by extensively and incrementally exploiting the flexibility of UML-diagrams while bringing to them architectural techniques, with the endeavor of strengthening the following emerging requirements in today's complex adaptive information systems:

Explicit "class"-interfaces: Although UML class- and object-diagrams in principle offer means for distinguishing between private and public attributes and methods, their exploitation for dynamically deriving required component interfaces remain unexplored. In the forwarded approach, we explicitly distinguish intra- from inter-component (with component as complex class-hierarchy) features. We do so first at the structural level and then fully exploit it at the behavioral level.

Behavior-centric interaction: We achieve this by coherently bringing specific OCL pre- and post-method constraints and cross-organizational business rules [3] in general with state-charts towards behavior-intensive stereo-typed *transient connectors* for interacting component interfaces. Further, we semi-graphically construct such connector glues using very appealing Petri nets-inspired stereo-typed graphical notations. In the same spirit, we govern intra-component behavioral computation as optional step for delving inside components.

Rigor and full distribution: The resulting behavior-intensive UML-architectural approach is semantically governed using rewriting logic [7]. More precisely, a tailored new rewrite theory that separate intra-component computation from inter-component interaction is put forwards.

The rest of this paper is organized as follows. Using a very simplified banking example, the next section overviews different UML diagrams we are focusing on, for semi-formally and partially describing information system requirements. In the third section, we present how incrementally leveraging UML class-diagrams towards componentization by explicitly distinguishing intra-class features from inter-classes ones. In the fourth section, we put forwards the new concepts of architectural connector glues and how they coherently bring together OCL constraints, state changes from state-charts and business rules in general, by explicitly and behaviorally incorporating them into inter-class interfaces. In the fifth section, due space limitations, we just hint how this new UML-architectural approach is semantically governed in rewriting logic. In the last section, we recapitulate on the achieved work and discusses future improvements.

2 UML Diagrams of Interest: Simple Illustration

First of all as we already emphasized, the behavior-intensive UML-architectural approach we are forwarding focusses primarily on UML class- and object-diagrams for structural features and OCL pre- post-method constraints for dynamic features and state-charts for behavioral aspects. Methodologically speaking, the construction of such diagrams has to be regarded as a *first phase* in the development of complex information systems using the present UML-compliant architectural approach.

The running illustration concerns a simplified banking system, where we assume having two ‘independent’ (i.e. related at this preliminary phase through an intrusive static relationship) class-diagrams, namely the account and the customer diagrams. Before giving the detail of each diagram, the left-hand side of Figure 1 sketches the ‘generic’ form of UML class diagrams. In this running example, different kinds of accounts (current, saving, loan, etc.) should represent a single compatible hierarchy of classes, whereas different profiled customers (e.g. ordinary, privileged, private and corporate) represent the second hierarchy.

The right-hand side of Figure 1 depicts the class-diagram of an account class hierarchy that interacts with the customer (i.e. Account holder)one. That account hierarchy includes a super-class `current accounts` and a sub-class `Saving accounts`. Attributes of current accounts are the `balance`, the account `owner`’s identity, a limit that the balance must not to go below we denote by `Limit`, and a history attribute for keeping track of transactions, as a list of pairs [Amount.Date]. As methods of this class we consider, besides creating and

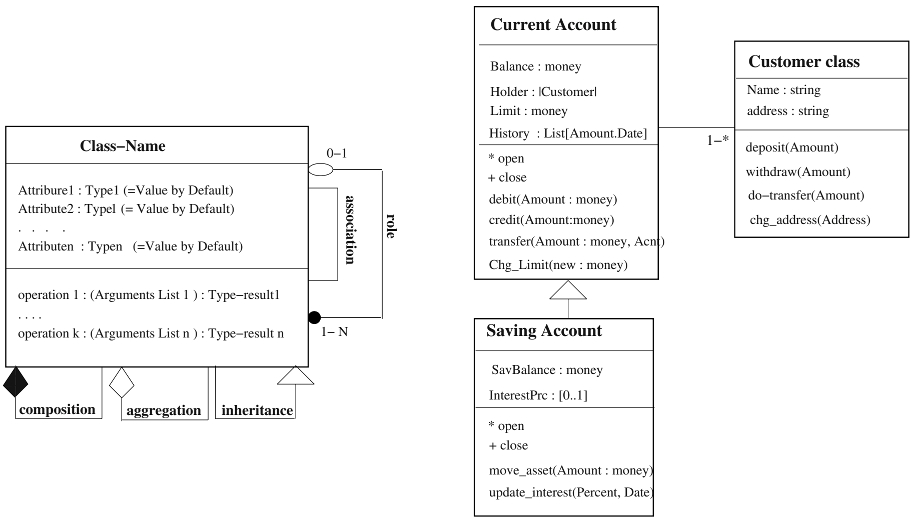


Fig. 1. Generic UML class-diagrams pattern and a Simplified Banking Case

deleting of objects, the crediting (resp. debiting) current account with given amount, the transfer of fund between accounts, and a method to permit changing the Limit. For the saving account subclass, as attribute we associate the saving balance and the current interest. As methods for this saving accounts sub-class, we introduce the possibility of updating the interest and the moving of funds from the current accounts to that saving and vis-versa.

As a sketch of the OCL description part which is of interest, we present the corresponding description to be associated, for instance, with the `debit` method. This description is depicted in left-hand side of Figure 2, where besides the signature of the method and its informal meaning, relevant is the condition `Pre` to be true to perform such a debit, namely the balance has to the greater than the requested amount. Relevant is also, the result of any operation captured through the `Post` constraint.

Finally in the right-hand side of Figure 2 we have also reported on a very simplified state-chart capturing state changes, where after opening an account, the first operation should be a deposit, afterwards all operations are allowed.

keywords	corresponding instantiation
`Operation`	Debit(src:Account, Amount)
`Pre:`	src.balance $\geq$ Amount
`Post:`	src.balance= src.balance$-$Amount$\wedge$ src.balance$\geq$ Limit

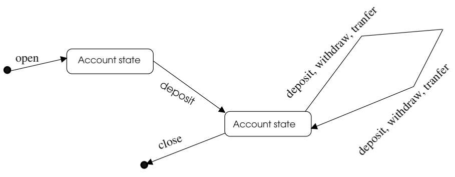
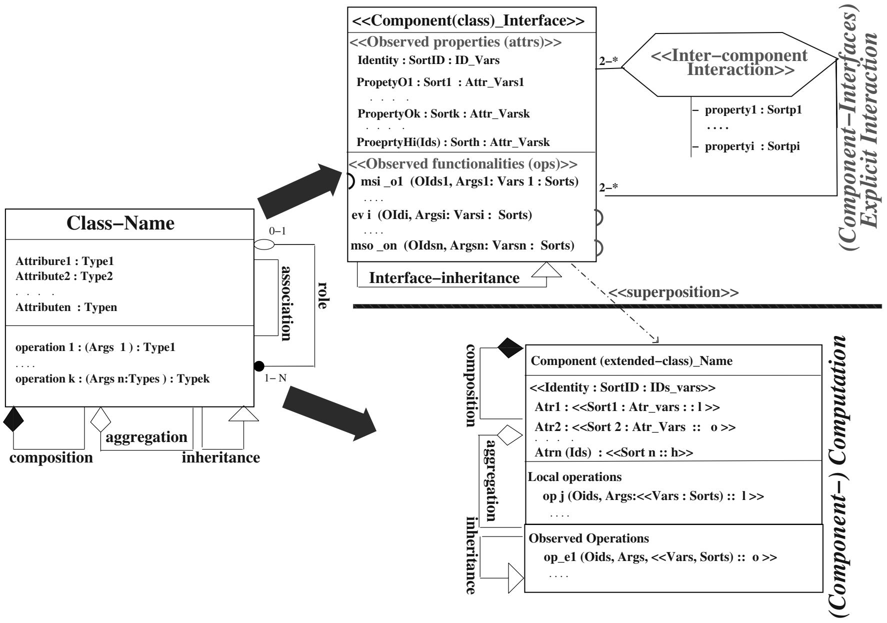


Fig. 2. Debit OCL constraint and Account state machine

3 UML-Compliant Architecture: Structural Features

The usual way consists in directly implementing such UML-based diagrammatical model(using Java/C++ or any other technology), which clearly leads to incoherent and conflicting multi-views. To overcome that, we are thus proposing an "extra" intermediate conceptual phase, where a coherent certified and behaviorally-driven architectural-based specification of the target distributed information systems is forwarded. In the section, we focus on the structural features of this UML-compliant architectural proposal, by smoothly leveraging class diagrams towards more explicit component-orientation.



<<Component(class)_Interface>>
<<Observed properties (attrs)>>
Identity : SortID : ID_Vars
PropetyO1 : Sort1 : Attr_Vars1
PropertyOk : Sortk : Attr_Varsk
ProeprtyHi(Ids) : Sorth : Attr_Varsk
<<Observed functionalities (ops)>>
msi _o1 (OIds1, Args1: Vars 1 : Sorts)
ev i (OIdi, Argsi: Varsi : Sorts)
mso _on (OIdsn, Argsn: Varsn : Sorts)
Interface–inheritance
2–*
<<Inter–component Interaction>>
– property1 : Sortp1
– propertyi : Sortpi
2–*
(Component–Interfaces) Explicit Interaction
Class–Name
Attribure1 : Type1
Attribute2 : Type2
Attributen : Typen
operation 1 : (Args 1) : Type1
operation k : (Args n:Types) : Typek
0–1
association
role
1– N
composition
aggregation
inheritance
<<superposition>>
Component (extended–class)_Name
<<Identity : SortID : IDs_vars>>
Atr1 : <<Sort1 : Atr_vars : : l >>
Atr2 : <<Sort 2 : Atr_Vars :: o >>
Atrn (Ids) : <<Sort n :: h>>
Local operations
op j (Oids, Args:<<Vars : Sorts) :: l >>
Observed Operations
op_e1 (Oids, Args, <<Vars, Sorts) :: o >>
composition
aggregation
inheritance
(Component–) Computation

Fig. 3. Leveraging UML class-diagrams to structural intra-component and interfaces

As illustrated in Figure 3, this progressive seemingly structural shifting from holistic class-diagrams to explicitly interacting components (via interfaces) is governed by a set of coherent light-weight stereotypes, that can be highlighted as follows:

Explicit separation intra-component and interfaces: This first step consists in explicitly conceiving UML-compliant interfaces using all observed (i.e. public) attributes and methods. Inspired by close investigations [1], we further propose to distinguish between imported / exported messages[1] and triggering events. The benefits of this fine-grained separation will be subsequently exploited at the behavioral level. Crucial to emphasize is that the

[1] As we strive for a fully distributed information systems governed by rewriting logic.

interface properties and messages closely dependent on the intended behavior of the concerned interaction (connector glue). That is, depending on what is required for expressing the intended behavior, we select the needed properties and messages from different components as interfaces.

Explicit interaction stereotyped icons: We propose at this structural level an icon (in form of six-corner polygon) for relating different interfaces, where at least different required properties (e.g. variables, invariants and constants) can be defined at this level.

Attribute variables and Sorts: The first step towards algebraically certifying attribute types and method arguments, while allowing complex data types, consists in abstracting them to ADT [6] (abstract data types. In this way, we can define any user-specific data types algebraically and reason on it using rewriting techniques, as a first data-level validation in our approach. The second main upgrading towards behavioral issues concerns the association of (set of) variable names to each attribute and method argument. These variables, as we detail in next section, will play crucial role in bringing OCL constraints and (Event-Conditions-Actions) ECA-business rules to this conceived architectural approach.

Example 1. Figure 4 demonstrates the smooth automatic shifting from the holistic banking class-diagrams to an explicitly interacting account and customer components. In this illustration, we have for instance decided for the balance to

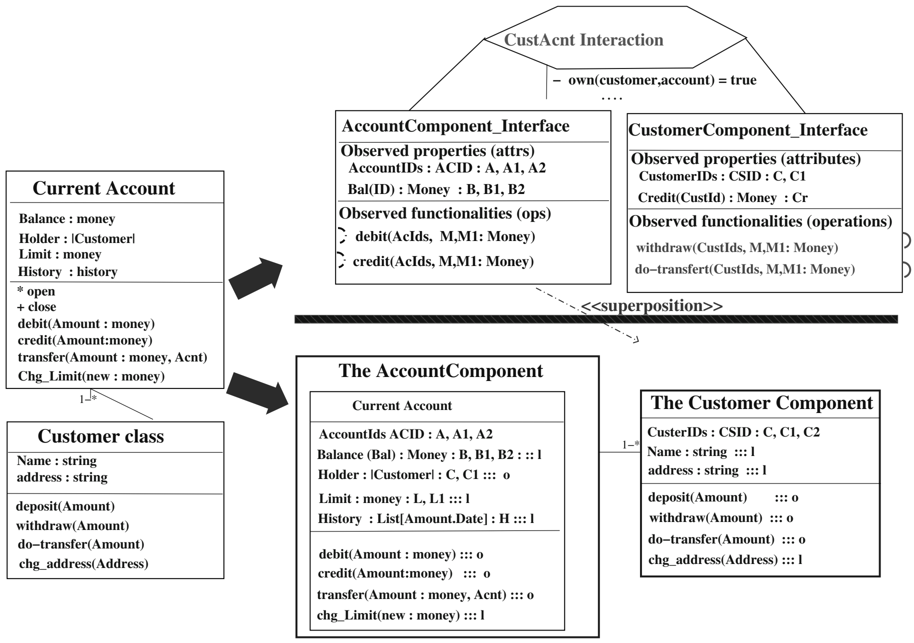

CustAcnt Interaction
– own(customer,account) = true
....
AccountComponent_Interface
Observed properties (attrs)
AccountIDs : ACID : A, A1, A2
Bal(ID) : Money : B, B1, B2
Observed functionalities (ops)
debit(AcIds, M,M1: Money)
credit(AcIds, M,M1: Money)
CustomerComponent_Interface
Observed properties (attributes)
CustomerIDs : CSID : C, C1
Credit(CustId) : Money : Cr
Observed functionalities (operations)
withdraw(CustIds, M,M1: Money)
do–transfert(CustIds, M,M1: Money)
<<superposition>>
Current Account
Balance : money
Holder : |Customer|
Limit : money
History : history
* open
+ close
debit(Amount : money)
credit(Amount:money)
transfer(Amount : money, Acnt)
Chg_Limit(new : money)
1–*
Customer class
Name : string
address : string
deposit(Amount)
withdraw(Amount)
do–transfer(Amount)
chg_address(Address)
The AccountComponent
Current Account
AccountIds ACID : A, A1, A2
Balance (Bal) : Money : B, B1, B2 : :: l
Holder : |Customer| : C, C1 ::: o
Limit : money : L, L1 ::: l
History : List[Amount.Date] : H ::: l
debit(Amount : money) ::: o
credit(Amount:money) ::: o
transfer(Amount : money, Acnt) ::: o
chg_Limit(new : money) ::: l
1–*
The Customer Component
CusterIDs : CSID : C, C1, C2
Name : string ::: l
address : string ::: l
deposit(Amount) ::: o
withdraw(Amount) ::: o
do–transfer(Amount) ::: o
chg_address(Address) ::: l

Fig. 4. The banking class-diagram shifted to UML componentization

be an observed attribute, but with a protected value, that is declared as function $bal(Ids)$ meaning that the balance can be used only for testing. We also declared as balance variables B, B_1, B_2 to be used later. We declared as imported messages, the two methods credit and debit. At the customer side, we have the credit as hidden or protected value and as triggering *event* (of the interaction) the withdrawal.

4 UML-Compliant Architecture: Behavioral Concerns

With the above structural leveraging of UML class-diagrams towards more componentization, the next important step aims at coherently bringing in behavioral issues. We do so by exploiting OCL pre- and post-methods constraints (and business rules in general) and state-charts diagrams.

4.1 UML-Compliant Behavior-Intensive Connectors

At this interaction level, instead of usually local OCL pre- post-constraints, we assume the existence of more complex business rules informally describing the interaction between (business) entities. As general pattern for such Event-condition-business business rules, we propose the following.

ECA-behavioral glue <*glue-Identity*>
  interface participants <*list-of-participants*>
  invariant <*possible interaction constraints to respect*>
  constants/attributes/operations
    <*extra-required elements for the interaction*>
  interaction rules: <*Rule-Name*>
    at-trigger <`(set-of-)events`>
    under <`conditions`>
    reacting <`set-of-actions`>

This generic behavioral business rule requires thus from different participating entities, explicit interfaces including different events, messages and other properties (such as constants, variables, etc). When needed, we explicitly specify such interfaces before giving this glue.

Before showing how to incorporate this cross-entities business rules into our UML-compliant architectural proposal, let us illustrate it with the running example.

Example 2. To stay competitive, banking systems are offering different incitive packages for their customers, ranging from simple agreed-on contracts (e.g. different formulas for withdrawing / transferring moneys) to highly sophisticated complex offers (i.e. staged housing loans, mortgages, etc.) depending on their profiles, trust, experiences, etc. The usual case for any ordinary customer is to check what, we call *standard* withdrawal, where the customer should own the corresponding account and the withdrawal amount should not go beyond the current balance. This standard explicit withdrawal agreement is described in the left-hand below.

```
ECA-behavioral glue Std-Withdraw
participants Ac: Account;
        Cs: Customer
invariants Cs.own(Ac) = True

interaction rule : Standard
at-trigger Cs.withdraw(M)
   under  (Ac.bal() ≥ M)
   acting Ac.Debit(M)
end Std-withdraw
```

```
ECA-behavioral glue VIP-Withdraw
participants Ac: Account;
        Cs: Customer
attribute Cs.credit : Money
invariants Cs.own(Ac) = True

interaction rule : VIP
at-trigger Cs.withdraw(M)
   under  (Ac.bal() +Cust.credit ≥ M)
   acting Ac.Debit(M)
end VIP-withdraw
```

A more flexible withdrawal for privileged customers is to endow them with a credit those amount depends on their profile and trust. Customers enjoying such agreements can now withdraw amounts going beyond the current balance. The modelling of such flexible agreed-on VIP withdrawal takes the following slightly modified behavioral glue. The right-hand side above describe this VIP-withdrawal.

As depicted in Figure 5, the incorporation of such business rules consists in conceiving a Petri-net[10] like transition, with the following characteristics.

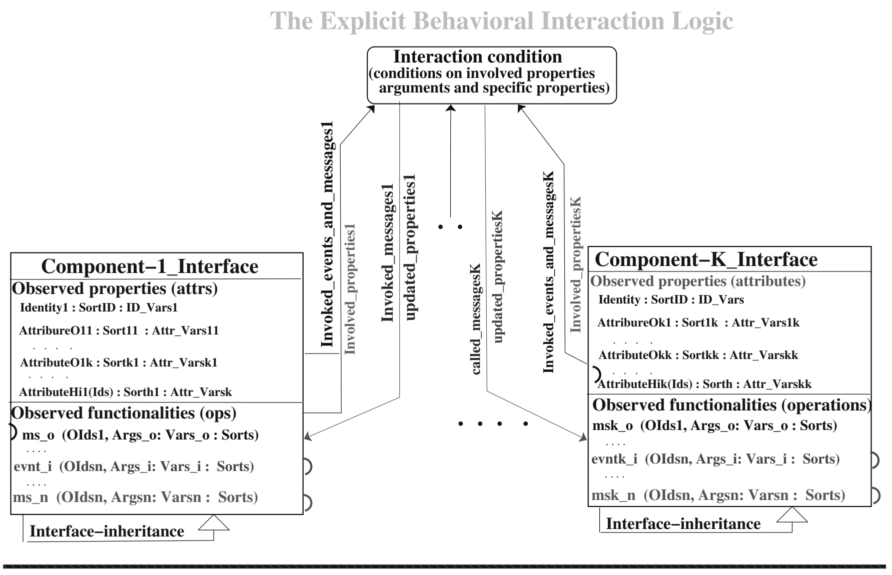

The Explicit Behavioral Interaction Logic
Interaction condition
(conditions on involved properties arguments and specific properties)
Invoked_events_and_messages1
Involved_properties1
Invoked_messages1
updated_properties1
called_messagesK
updated_propertiesK
Invoked_events_and_messagesK
Involved_propertiesK
Component−1_Interface
Observed properties (attrs)
Identity1 : SortID : ID_Vars1
AttribureO11 : Sort11 : Attr_Vars11
AttributeO1k : Sortk1 : Attr_Varsk1
AttributeHi1(Ids) : Sorth1 : Attr_Varsk
Observed functionalities (ops)
ms_o (OIds1, Args_o: Vars_o : Sorts)
evnt_i (OIdsn, Args_i: Vars_i : Sorts)
ms_n (OIdsn, Argsn: Varsn : Sorts)
Interface−inheritance
Component−K_Interface
Observed properties (attributes)
Identity : SortID : ID_Vars
AttribureOk1 : Sort1k : Attr_Vars1k
AttributeOkk : Sortkk : Attr_Varskk
AttributeHik(Ids) : Sorth : Attr_Varskk
Observed functionalities (operations)
msk_o (OIds1, Args_o: Vars_o : Sorts)
evntk_i (OIdsn, Args_i: Vars_i : Sorts)
msk_n (OIdsn, Argsn: Varsn : Sorts)
Interface−inheritance

Fig. 5. Interaction Pattern between independent UML interface-diagrams

– The transition form, we associate with each message is represented as a rounded box. Within each rounded box we associate a condition (i.e. a boolean expression), we denote by `Inter_cond`, which has to be built on the invoked attributes and message arguments using different comparison operators (e.g $=, >, <, \neq, \leq, \geq$) and / or boolean operators (e.g `and, or`).
– The (input) arrows or arcs going from the corresponding interfaces to the rounded box are labeled by two information.
  1. The first inscription denoted as `Invoked_Evnts_Mes` (above the arc) is a set of events and messages participating in the interaction. their general form is $\texttt{ms}_i\texttt{(Id}_1\texttt{, .., var}_1\texttt{,..,var}_k\texttt{)}$;
     where $\texttt{ms}_i$ is any message or event declared there.
  2. The second inscription (below the arc) denoted as `Invoked_properties` has to be of the form: $\texttt{Id}_1\texttt{.atr}_1\texttt{:Var}_1$,... , $\texttt{Id}_k\texttt{.atr}_k\texttt{:Var}_k$
     Where, intuitively each pair $\texttt{Id}_i\texttt{.atr}_i\texttt{:Var}_i$ corresponds to an invoked attribute belonging to an interface $\texttt{Id}_i$ with $\texttt{Var}_i$ to be understood as a current value.
– Finally, the two inscriptions associated with any output arrow, denoted as `called_messages` and `updated_properties` clearly reflect the actions to undertake (i.e. messages to invoke) and possible state changes of the participating properties (i.e. attribute state changes). They take as general form: $\texttt{Id}_1\texttt{.atr}_1\texttt{:Exp}_1$,... , $\texttt{Id}_k\texttt{.atr}_k\texttt{:Exp}_k$.
  Where each expression $\texttt{Exp}_i$ reflects the expected state changes captured from the business rules and corresponding state-charts.

Example 3. Figure 6 depicts the application, to our banking running example, of the above steps for intrinsically incorporating behavioral concerns as architectural connectors. For instance, for the standard-withdrawal, first a withdrawal

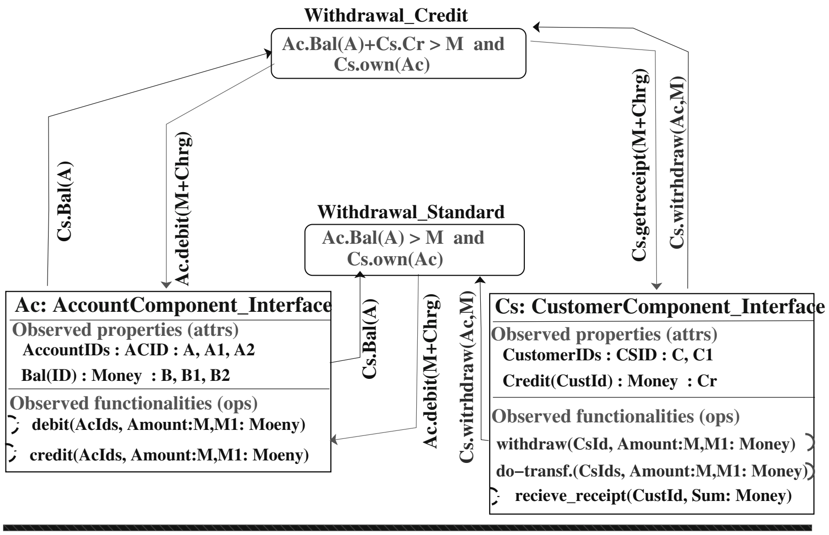

Withdrawal_Credit
Ac.Bal(A)+Cs.Cr > M and Cs.own(Ac)
Cs.Bal(A)
Ac.debit(M+Chrg)
Cs.getreceipt(M+Chrg)
Cs.witrhdraw(Ac,M)
Withdrawal_Standard
Ac.Bal(A) > M and Cs.own(Ac)
Ac: AccountComponent_Interface
Observed properties (attrs)
AccountIDs : ACID : A, A1, A2
Bal(ID) : Money : B, B1, B2
Observed functionalities (ops)
debit(AcIds, Amount:M,M1: Moeny)
credit(AcIds, Amount:M,M1: Moeny)
Cs.Bal(A)
Ac.debit(M+Chrg)
Cs.witrhdraw(Ac,M)
Cs: CustomerComponent_Interface
Observed properties (attrs)
CustomerIDs : CSID : C, C1
Credit(CustId) : Money : Cr
Observed functionalities (ops)
withdraw(CsId, Amount:M,M1: Money)
do−transf.(CsIds, Amount:M,M1: Money)
recieve_receipt(CustId, Sum: Money)

Fig. 6. The interaction between the account and the customer class-diagrams

event have to trigger the interaction from the customer interface. From the account interface, we require the balance property as hidden attribute. The usual condition ensures that the requested amount is less that the available balance. In such case, a debit message is invoked. The same reasoning is to apply to the VIP-withdrawal, where we have added as extra message, `receipt_reception`, to-be invoked on the customer side.

5 Rewriting Logic Operational Semantics

The semantical framework we are proposing for this new UML-compliant architectural conceptual model is based on rewriting logic [7], which has been proved very appropriate for dealing with concurrent systems [8]. Further strengths making this logic very practical is the current implementation of the MAUDE language [4]. Due to the strict imposed space limitations, we restrict ourselves to just highlight the translating steps leading to MAUDE programs.

1. The translation of structural aspects of any intra-component features is directly captured as a MAUDE module.
2. The translation of the behavior, we associated with each operation is expressed as a rewrite rule. More precisely, we have to perform the two following two steps:
   – first, we have to gather all inscriptions in a given arc together as a MAUDE tuple
   $$\langle Id_1|atri_1 : val_1\rangle, \ldots, \langle Id_k|atri_k : val_k\rangle$$
   (resp. $\langle Id_1|atri_1 : exp_1\rangle, \ldots, \langle Id_l|atri_l : exp_l\rangle$);
   – Each Petri-net like transition is then to be expressed as a rewrite rule of the form:
   $\texttt{Invoked\_Mes}_i \ \ \texttt{Invoked\_Attri}_i \Rightarrow \texttt{Result\_change}_i \ \ \texttt{if} \ \ \texttt{Condition}$

Example 4. With respect to these straightforward ideas, the corresponding structural MAUDE part of the account class, for instance, takes the following form:

```
omod Account is
  protecting Money .
  class-loc Account | Limit : Money .
  class-obs Account | Balance : Money, Owner : OId .
  msg-loc  Chg_Limit : OId  Money → Msg .
  msg-loc  Deposit : OId  Money → Msg .
  msg-loc  Withdraw : OId  Money → Msg .

  vars I, I2, I3 : OId .
  vars C, C1, C2 : OId .
  Vars  B, B1, B2, L, L1, L2 : Money .
  vars W, D : Money .
```

On the other hand, following these very simple translating ideas, the corresponding rewrite rules of the messages in the account class-diagram, for instance, are as follows. In these rules we have considered the corresponding message names as rule labels.

`Chg_Limit` : $Chg\_Limit(I, L1)\ \langle I|Limit : L\rangle \Rightarrow \langle I|Limit : L1\rangle$
`Credit` : $Credit(I, D)\ \langle I|Balance : B\rangle \Rightarrow \langle I|Balance : B + D\rangle$ `if` $(D > 0)$
`Debit` : $Debit(I, W)\ \langle I|Balance : B\rangle \Rightarrow \langle I|Balance : B - W\rangle$ `if` $(true$

6 Conclusions

We proposed a UML-compliant behavior-intensive architectural conceptual model for adaptive and concurrent interacting IS. The approach explicitly separates between intra-component and inter-component interaction using architectural connectors, and incrementally constructing them from usual UML structural and behavioral diagrams. Moreover, for rapid-prototyping the proposal is semantically governed using a tailored extensions of the efficient concurrent current implementation of the Maude language. To consolidate the practicability of the approach, we are implementing a tool environment for automatically developing evolving architectural IS. Further, we are exploring the strengths of the Maude reflection capabilities for runtime adaptivity.

Acknowledgment

The authors are very grateful to the three referees. Their insightful comments significantly helped in improving this final version. We are also thankful to the editor, Roland Kaschek, for assisting us in reformatting the paper as requested.

References

1. Andrade, L., Fiadeiro, J.: Agility through Coordination. Information Systems 27, 411–424 (2002)
2. Booch, G., Jacobson, I., Rumbaugh, J.: Unified Modeling Language, Notation Guide, Version 1. Addison-Wesley, Reading (1998)
3. Bajec, M., Krisper, M.: A Methodology and Tool Support for Managing Business Rules in Organisations. Information Systems 30(6), 423–443 (2005)
4. Clavel, M., et al.: Maude: Specification and Programming in Rewriting Logic. Technical report, SRI, Computer Science Laboratory (March 1999)
5. Garlan, D., Cheng, S., Kompanek, A.J.: Reconciling the Needs of Architectural Description with Object-modeling Notations. Science of Computer and Programming 44(1), 23–49 (2002)
6. Ehrig, H., Mahr, B.: Fundamentals of algebraic specifications 1: Equations and initial semantics. EATCS Monographs on Theoretical Computer Science. 21 (1985)
7. Meseguer, J.: Conditional rewriting logic as a unified model for concurrency. Theoretical Computer Science 96, 73–155 (1992)
8. Marti-Oliet, N., Meseguer, J.: Rewriting logic as a logical and semantic framework. In: Meseguer, J. (ed.) Proc. of First International Workshop on Rewriting Logic. Electronic Notes in Theoretical Computer Science, vol. 4, pp. 189–224 (1996)

9. OMG.: UML 2.0: Superstructure Specification. Version 2.0, formal/05-07-04. Technical report, omg.org (2005)
10. Reisig, W.: Petri Nets. EATCS Monographs on Theoretical Computer Science 4 (1985)
11. Shaw, M., Garlan, D.: Software Architecture: Perspectives on an Emerging Discipline. Prentice Hall, Englewood Cliffs (1996)

A Framework for Designing and Recognizing Sketch-Based Libraries for Pervasive Systems

Danilo Avola, Fernando Ferri, Patrizia Grifoni, and Stefano Paolozzi

National Research Council, IRPPS-CNR,
Via Nizza 128, 00198 Rome, Italy
{danilo.avola,fernando.ferri,p.grifoni,stefano.paolozzi}@irpps.cnr.it

Abstract. Sketch-based interfaces enable users to effortless and powerfully communication way to represent concepts and/or commands, in different application domains, on different devices. Developing of a sketch-based interface for recognizing each specific domain is a time-consuming operation that requires the re-engineering and/or the re-designing of the whole recognition framework.

The sketch-based interaction can be exploit in order to have a suitable modality to interact, in pervasive and ubiquitous way, with mobile information systems. Moreover, sketch-based interaction can be particularly profitable if used together with other modalities (such as: speech, gesture, text, and so on). In this paper we propose a general framework to define and to recognize each kind of 2D symbols library, besides the proposed approach allow us to add, by text and speech modalities, an additional semantic meaning to each symbol contained in the built library.

Keywords: Feature Extraction, Mobile Devices, Sketch-based Interfaces, Sketch-based Interaction, SVG.

1 Introduction

People use sketches to express concepts and to represent ideas in an immediate and intuitive way.

A sketch based interface is a pen-centered computer system. The user can draw on a graphic tablet to express “everything” he/she wants (such as: concepts, ideas, objects, and so on). There are several advantages using sketch based interface. For example it is possible to erase a drawn object (or part of it), or save on a file the current draw. In our context, it is possible to use the digital form of the drawing (i.e. the set of pixels) to perform an interpretation of that drawing. For example, it is possible to recognize that a set of pixel with particular geometrical features represents a square or a circle or a more complex symbol.

Therefore, the use of sketch-based interfaces to interact with different devices provides a natural and convenient way to convey concepts and commands. Moreover, it has to taken into account that both the diffusion of new mobile devices (and related usability, pervasively and ubiquitously) and the complexity of the actual services, applications and information systems available on mobile and desktop devices need of a powerfully, simple and effectiveness interaction way. For all these reasons, sketch-based applications have had strong development in recent years.

There are two main tasks, related to the sketch-based interfaces, which have to be analyzed.

The first one regards the applications domains heterogeneity. We are used to adopt always the common domains (such as: data flow diagram (DFD), flow charts diagram (FCD), entity-relationship diagram (ERD), geometrical figures, electrical schemes, and so on). Indeed, the domains are potentially infinite because it is always possible to build a new and personalized domain for a specific device and/or application context. The complexity and the variety of a library that represents a specific domain depends only on the creativity of the library maker. Obviously, the maker will tend to create simple and suitable symbols for a specific library in according to the requirement of the related application and/or service and/or context.

There is an extensive heterogeneous literature on sketch recognition systems. For example in [1] an interesting unified modeling language (UML) recognize system is given. A novel approach to identify the symbols belonging to the FCD is instead given in [2]. Another remarkable approach to recognize a particular domain composed by symbols representing the more common elements in Java graphical user interfaces (GUI) is explicated in [3]. A further particular kind of conceptual design sketch recognition system is given in [4].

The common factor of the existing approaches to recognize a specific domain is the personal way through which every domain is recognized by the related application. Indeed, there is not a unique accepted way to recognize and to distinguish the set of sketched symbols belonging to a specific library. This leads us to the second difficult task related to the sketch-based interfaces: the case-based application. Developing of a sketch-based interface for recognizing each specific domain is a time-consuming operation that often requires the re-engineering and/or the re-designing of the whole framework.

In this paper we propose a multi-domain framework to accomplish the two aforementioned tasks. In particular, we propose a sketch-based framework to define and to recognize each kind of graphical 2D library. The developed prototype provides two distinct environments. In the first one a library editor, through which the user can create 2D libraries, is given. In the second environment the user can load one of the previously created libraries, moreover he/she can make up drawing activity, according to the loaded library, to perform the recognition of the whole sketch.

The proposed framework is different from the other sketch-based systems just for the ability to overcome the impossibility, of these systems, to recognize sketches in multi-domains context. In such a way the proposed approach (and related framework) can be considered more general than others. Moreover, in order to provide an added value to the expressiveness of the sketch-based interface, it is possible to use, during the editor phase, other two modalities: speech and text. The first one can be used to add a semantic explicit information flow to the symbols that compose the library, the second can be used to add a visual additional information to each sketched symbol contained in the library, such as: a label, a note, a comment and so on.

The paper is structured as follows. Section 2 proposes some remarkable related works in sketch-based interfaces area, Section 3 presents the library editor environment. In particular, both the approach used to build a library and the role of the speech and text modalities is given. Section 4 describes the recognition environment, showing how the symbols, sketched by the user, are matched with the related symbols of a given library. Finally, Section 5 concludes.

2 Related Works

The sketch-based interaction to express simple and/or complex concepts and commands to control different mobile devices can be considered, at the moment, the best interaction modality. In fact, other modalities depend strongly on the surrounding environment and on other human factors that can corrupt the right device interpretation of the concept and/or command expressed by the user. For example, the speech modality is prone to interpretation errors caused by the presence of surrounding noise, defects in pronunciation, location and context in which the user performs the concept and/or command, and so on. Similar issues can be considered for other modalities, such as: gesture, handwriting, etc. Moreover, in the last decade, pen-based interfaces have regained popularity thanks to the proliferation of mobile devices such as PDAs, PalmPC, TabletPC and consequently the importance of sketch-based interfaces has been further increased (see for example [5] and [6]).

The sketch-based interaction can be exploited in order to have a suitable modality to interact, in pervasive and ubiquitous way, with mobile information systems. There is an extensive literature about the recognition of single domains, such as: software design, diagrams (UML, DFD, ERD, and so on), finite state machines, mechanical engineering design, and so on.

In [7] a remarkable freehand sketch recognition approach is given. In this work the authors present a method for interpreting on-line freehand sketch and describes a human-computer interface prototype system of freehand sketch recognition (FSR) that is designed to infer designers' intention and interprets the input sketch into more exact 2D geometric primitives: straight lines, polylines, circles, circular arcs, ellipses, elliptical arcs, hyperbolas and parabolas. Another interesting approach is described in [8]. Differently from the previous work, based on geometrical structure of the stokes belonging to the sketch, this work is based on a probabilistic approach to sketch stroke interpretation. In this work domain-specific libraries of Bayesian Network Fragments (BNF) that describe shapes and domain patterns is used. Several mechanisms to control the size of the space of hypotheses are presented, and the technique is applied to the domain of electrical circuit diagram recognition. Another meaningful statistical approach is shown in [9]. In this work the authors propose the use of conditional random fields for labeling box-and-line diagrams for particularly difficult ambiguous examples where constraints must propagate in order to find the most-likely interpretation. One of the most interesting, practical, and popular application about the sketch recognition systems is SILK (Sketching Interfaces Like Krazy), a tool that allows users to sketch interactive user interfaces. SILK [10] was one of the first systems that recognizes a sketch and allows interactive use of the sketch without replacing the strokes with cleaned-up strokes, users can also view and modify their original strokes.

Two other popular systems that exploit geometrical and mathematical models to perform the sketch recognition activity are: SketchIt [11], which is used to support the recognition of both the mechanical engineering designs and hand-drawn diagrams; and Denim [12], a very practical tool that allow the users to sketch and design web pages. A particular mention have to be made for LADDER [14]. The purpose of this application is to enable user interface designers and domain experts who may not have expertise in sketch recognition to build these sketch systems. The authors have created and implemented an interesting framework (FLUID - FaciLitating User Interface Development) in which developers can specify a domain description indicating how domain shapes have to be recognized, displayed, and edited. Therefore, this description is automatically transformed into a sketch recognition user interface for that domain.

Our approach (and related prototype) has several novel and alternative solutions to solve common issues in multi-domain sketch recognition. In particular, the use of the geometrical features to characterize both the library and the user sketch, and the SVG-based (Scalable Vector Graphics) approach, used to perform the matching between them, can be considered an improvement in the sketch recognition field.

3 Making a General 2D Library

In this section we show our approach, and the related developed editor environment, through which it is possible to build each kind of 2D library.

In Figure 1 a simple principle schema about the basic concepts concerning the making of a general 2D library is given. In particular, it is possible to observe that our approach foresees that the user can define a symbol joining three different modalities. More precisely, the user must use the sketch modality (core) to describe the geometrical shape of a symbol, then he/she can also use text and speech modalities (optionals) to add a semantic meaning to the symbol. When all the information (or at least the core information) have been given, the Analyzer Module can provide a detailed description of the object in terms of: (i) geometrical constraints (derived from the sketch) and (ii) additional semantic description (derived from the eventual text and/or speech inputs).

We now describe the details of each aforementioned step regarding the 2D library making process through the analysis of a screenshot of the editor environment shown in Figure 2.

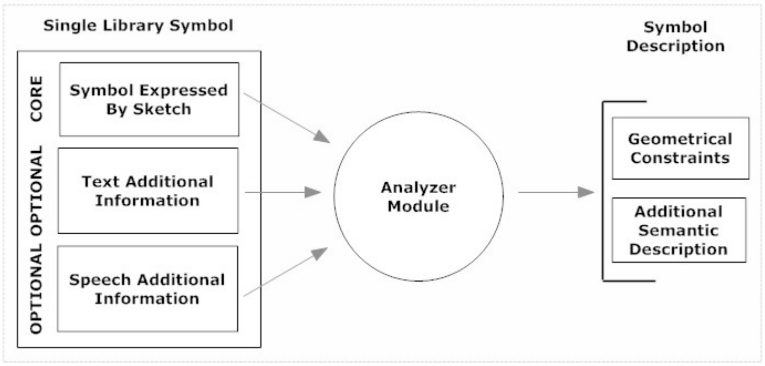

Single Library Symbol
CORE
OPTIONAL
OPTIONAL
Symbol Expressed By Sketch
Text Additional Information
Speech Additional Information
Analyzer Module
Symbol Description
Geometrical Constraints
Additional Semantic Description

Fig. 1. Making a single library symbol

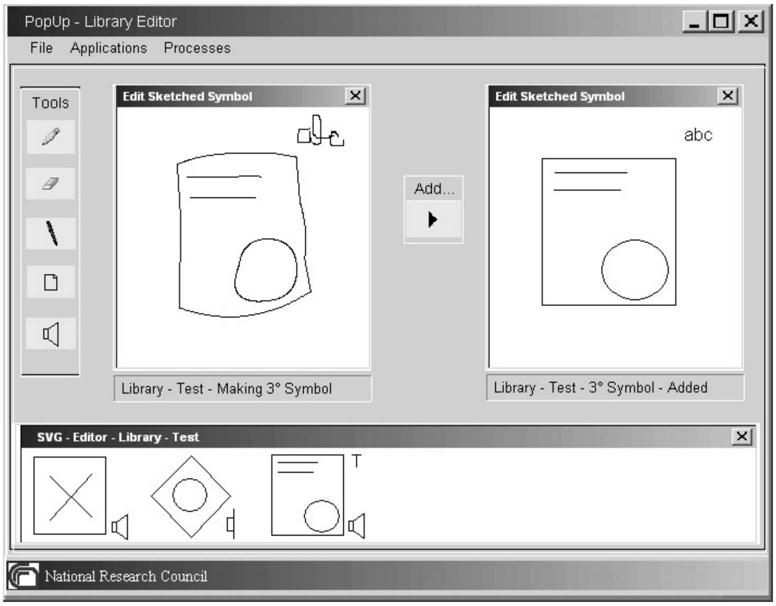

PopUp - Library Editor
File Applications Processes
Tools
Edit Sketched Symbol
Add...
Edit Sketched Symbol
abc
Library - Test - Making 3° Symbol
Library - Test - 3° Symbol - Added
SVG - Editor - Library - Test
T
National Research Council

Fig. 2. Screenshot of the library editor

In Figure 2 the creation of a library called *Test* is shown. In particular the library has three elements, two previously created and one just created. The first previously created symbol has a speech additional information, while the second element does not have any additional information. The speech additional information are shown at the bottom of the recognized symbol inside the SVG-Editor window: loudspeaker “on” or loudspeaker “off”, while the text additional information are shown by the presence (or absence) of the icon T at the top of the recognized symbol inside the SVG-Editor window. The third symbol presents

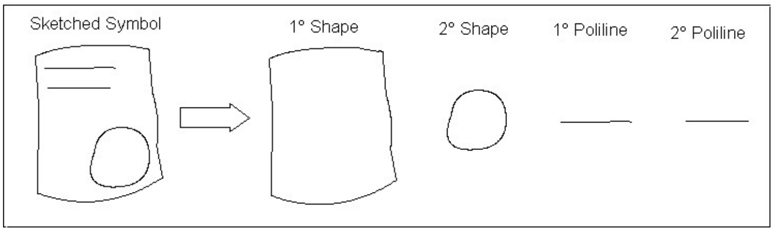
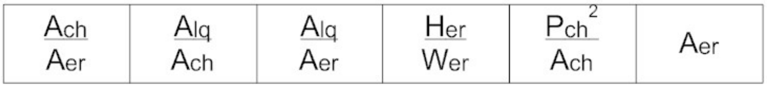


Fig. 3. Polylines and shapes detection



$\frac{Ach}{Aer}$	$\frac{Alq}{Ach}$	$\frac{Alq}{Aer}$	$\frac{Her}{Wer}$	$\frac{Pch^2}{Ach}$	Aer

Fig. 4. Feature Extraction Vector

both speech and text additional information (as shown by the loudspeaker "on" and the presence of the T icon).

The Popup library editor allows the user to build a personalized domain. He/she starts by performing on the left sub-window (edit sketched symbol) the sketch representation of the wished symbol. During this activity the user is supported by several editing tools (such as: rubber, total clearing, undo action, and so on) that allows he/she to build the symbol in a suitable and simple way. When the symbol is built, the user can choose to add additional information to the performed symbol. Indeed, our sketch-based interface is provided of two particular tools to link speech and/or text information upon the just created symbol.

When the user has wholly designed the symbol (and added the eventual additional information) than he/she can add it (by add button) to the library (in this case, the Test library). The result of this last step is a vectorial representation (by scalable vector graphics) of the sketched symbol, moreover this representation takes into account the "physical" link to the eventual informative streams (speech and/or text).

The transformation of a sketched symbol in a vectorial one is the main issue in sketch recognition, while speech and text modalities have been simply integrated in our framework, by using the available open source systems.

Sketch recognition is a complex task that involve several different sub-tasks, such as: objects identification, beautifications, matching mechanisms, etc. These steps are often tied to the application that has been developed for a specific domain. Our framework allows the user to represent each 2D sketched library without changing the recognition process.

The fist step is characterized by the identification of each polyline and closed shape belonging to the symbol performed by the user. In our case, as it is possible to observe in Figure 3 the previous example was composed by two closed shapes and two polylines.

From each recognized closed region and/or polyline a set of geometrical features have been extracted. The purpose of the feature extraction process was to identify the mapping between the recognized objects and the following considerate 2D primitive: line, circle, rectangle, triangle, ellipse, square, diamond, and point. The main concept is that every simple or complex 2D drawing can be always expressed by the composition of these basilar geometrical shapes. This kind of measurements have been performed exploiting the meaningful results obtained by M.J. Fonseca [15]. In these works the authors exploit the convex hull to compute three special polygons: the first two are the largest area triangle and quadrilateral inscribed in the convex hull, the third is the smallest area enclosing rectangle. By these polygons a feature vector, shown in Figure 4, able to recognize the mentioned 2D primitive has been performed.

More specifically, the six image local operators used in the feature extraction vector regard:

– Ach : Area of Convex Hull;
– Aer : Area of the Enclosing Rectangle;
– Alq : Area of the Largest Quadrilateral;
– Her : Height of the (non-aligned) Enclosing Rectangle;
– Wer : Width of the (non-aligned) Enclosing Rectangle;
– Pch : Perimeter of the Convex Hull.

Each of the specific measure is applied to the set of points of each object (close region and polyline) to classify every object as a specific 2D primitive.

A main step performed by the editor environment is the creation of an SVG representation of the given geometrical description of the sketched objects. The SVG file allows the user to have a flexible, comparable and manageable structure to store every kind of 2D objects. Moreover, this XML-based standard allows the user to add, suitably, more additional information (such as, the eventual speech and/or text semantic information). In this phase all geometrical descriptions of an object are computed as SVG containing lines and/or ellipses. According to the previous example we have: four line for the first shape, an ellipse for the circle, and other two lines for the third and fourth shape. Our editor is able to consider also the constraints of the object. The constraint represents the geometrical relationships of each shape/polyline and between shape and/or polylines (such as: inclusion, near, intersection, etc.).

A simple schema about the library making process is introduced in figure 5.

The geometrical description is computed directly on SVG file by a developed "parser" that decomposes SVG primitives (i.e. rect, polygon, etc.) in terms of lines and ellipses (also called basic elements). Afterward an engine to detect the internal constraints between basic elements of a same shape is activated. In this context an internal constraints is a spatial relationship between two or more basic elements. At the moment the following main internal constraints on the same shape have been considered:

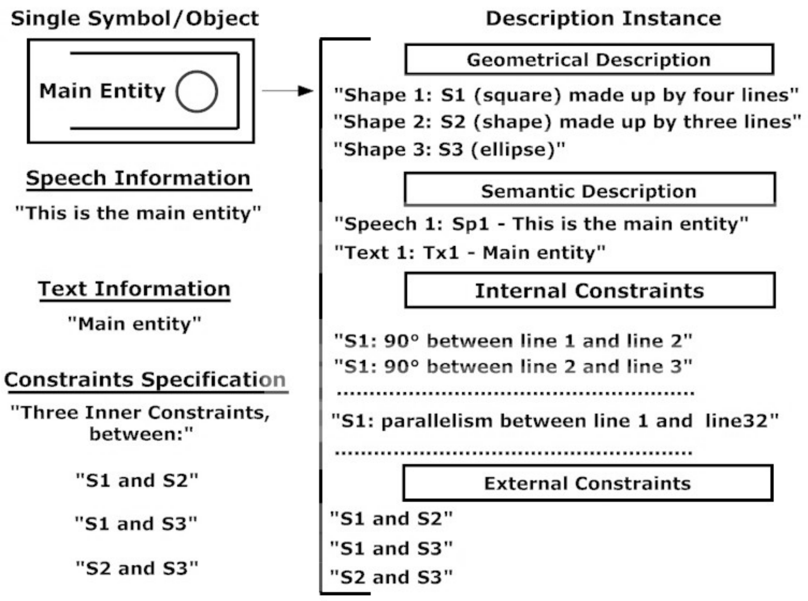

Single Symbol/Object
Main Entity
Speech Information
"This is the main entity"
Text Information
"Main entity"
Constraints Specification
"Three Inner Constraints, between:"
"S1 and S2"
"S1 and S3"
"S2 and S3"
Description Instance
Geometrical Description
"Shape 1: S1 (square) made up by four lines"
"Shape 2: S2 (shape) made up by three lines"
"Shape 3: S3 (ellipse)"
Semantic Description
"Speech 1: Sp1 - This is the main entity"
"Text 1: Tx1 - Main entity"
Internal Constraints
"S1: 90° between line 1 and line 2"
"S1: 90° between line 2 and line 3"
"S1: parallelism between line 1 and line32"
External Constraints
"S1 and S2"
"S1 and S3"
"S2 and S3"

Fig. 5. Library making process schema

- *coincident*: to indicate the geometrical coincidence (in a point) between two basic elements;
- *equalLength*: to indicate an equal length between two basic elements;
- *greaterLength*: to indicate the major length of a line than another line;
- *intersect*: to indicate the geometrical intersection (in one or more points) between two basic elements;
- *near*: to indicate the nearness between two basic elements;
- *parallel*: to indicate the geometrical parallelism between two lines;
- *acuteAngle*, *rightAngle* e *obtuseAngle*: to identify that two lines make up respectively an acute, right or obtuse angle.

Indeed, the previously introduced constrains make possible to infer several other geometrical properties, which are used to accomplish the description of the objects performed by the user. The external constraints are similar to the one used in the previous case, but they are applied on shapes and/or lines of the same object. In particular, a different additional constraint has to be take into account:

- *contain*: to indicate the geometrical containment of an object/shape in another one.

When the user has defined one or more libraries the recognition environment can be exploited to identify a user sketch. As for the library making also the

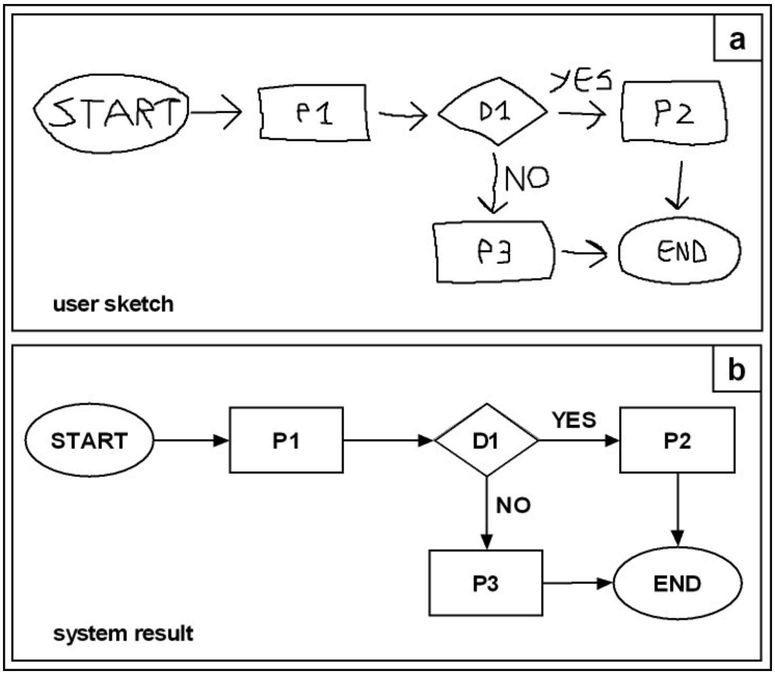


Fig. 6. An example of recognized sketch

recognition process can exploit the additional information (text and/or speech) during sketch activities. By this environment the user can have a suitable, vectorized and interpreted sketch, which can be suitably used to express concepts and to convey ideas. In Figure 6 a simple interpretation environment example is given.

In Figure 6a a sketch about a flow-chart diagram is performed. Inside the sketch several labels have been added to identify every element of the flowchart diagram. In Figure 6b the same recognized drawing is shown. Every sketched object/symbol in the Figure 6a is analyzed and substituted, Figure 6b, by a specific element of the related library.

It is important to observe that the set of the libraries building from the user are used to support the sketch recognition activities on heterogeneous domains. That is, every inserted library is considered, by the framework, as a element of a "abstract" macro library that contains the union of each object of each library. These elements, and the related application context, are jointly used to recognize the sketch performed by the user. In this way the framework makes easy the building of other kind of libraries, to define new sketch-based applications through these libraries and to augment the interpretation of the performed sketch.

It is important to observe that the SVG format used to represent each sketched symbols is a suitable way to perform advanced matching approach. This last step is commonly the focal task of the sketch-based recognition application.

4 Library Matching

The final purpose of matching between symbols sketched by the user in the interpretation environment and the symbols represented in a specific library is to choose the right element inside the library that provides the interpretation of the sketched symbols.

As stated in the previous sections. every symbol of every built library inside the framework is described through a SVG file. Moreover, also the user's sketch has been represented using SVG file. In this way the user's sketch contains information comparable to the symbols of the library.

The matching algorithm performs an exhaustive research of the particular pattern created by the user during composition of the sketched landscape, such as: complex DFD, architectural design, and so on. Every basic element, coming from library or from user's sketch, is represented by a set of nodes (the strokes) and a set of relationships (the constraints) between the nodes. Moreover, two or more related basic symbols (for example the containment constraint) are represented by two macro-nodes with a macro-relation between (in this case, the containment relation).

The matching approach starts from a generic node (of the sketched area). It performs a matching operation with every node of the same kind (i.e. line or ellipse) that are stored in the loaded library.

The next step is to get the second not examined node of the landscape sketch that has to be assigned to another node of the loaded library. To perform this second step it is necessary to exploit the relationships between the previous selected node and the actual selected node. In this way, with the increasing of the examined nodes decrease the disagreeing probability space.

The process ends when all nodes are examined. After the last node examination several possible configuration will be identified by the system. Each one will have a rank depending on the quality of the founded matching. Several configuration will have a too low rank, others will have a suitable ranking profitable to identify a right pattern. It is possible to define a threshold value in order to cut off the lowest ranking results.

Actually, the process has a complexity that strictly depends on the size of the given library and the number of sketched symbols. Therefore in the worst case the complexity is exponential.

5 Conclusions

The relevance of sketch-based interfaces has been advocated by many authors in the last decade. This kind of interaction is very useful to have a suitable modality to interact, in pervasive and ubiquitous way, with mobile information systems.

Sketch-based interaction becomes more and more important especially using devices, that enable users to interact with different modalities in different times and in different contexts.

However designing a sketch-based application is still a time-consuming work, because of the difficulty of the sketch interpretation in different application domains.

In this paper we provide a novel approach to solve well known issues in sketch recognition, in order to benefit of these approaches in pervasive environments. Moreover, we provide a framework for creating a library of symbols that can be used to develop effective sketch-based applications in different domains and can be effectively used for mobile devices.

The framework combines different modalities of interaction connected to the sketch modality in order to provide an eventual added value to the semantic meaning of each symbol belonging to the library.

Future works will regard a formal validation of our approach and an optimization of the developed framework. Moreover we are refining the matching process in order to decrease it's complexity.

References

1. Hammond, T., Davis, R.: Tahuti: a geometrical sketch recognition system for uml class diagrams. In: AAAI Spring Symposium on Sketch Understanding, pp. 59–68. AAAI Press, Stanford (2002)
2. Gross, M.D.: Stretch-a-sketch: a dynamic diagrammer. In: IEEE Symposium on Visual Languages, pp. 232–238. IEEE Press, Los Alamitos (1994)
3. Caetano, A., Goulart, N., Fonseca, M., Jorge, J.: Javasketchit: Issues in sketching the look of user interfaces. In: AAAI Spring Symposium on Sketch Understanding, AAAI Press, Stanford (2002)
4. Kavakli, M., Gero, J.S.: The structure of concurrent cognitive actions: A case study on novice and expert designers. Design Studies 23(1), 25–40 (2002)
5. Mahoney, J., Fromherz, M.: Three main concerns in sketch recognition and an approach to addressing them. In: AAAI Spring Symposium on Sketch Understanding, pp. 105–112. AAAI Press, Stanford (2002)
6. Ijiri, T., Owada, S., Igarashi, T.: The Sketch L-System: Global Control of Tree Modeling Using Free-Form Strokes. In: Butz, A., Fisher, B., Krüger, A., Olivier, P. (eds.) SG 2006. LNCS, vol. 4073, pp. 138–146. Springer, Heidelberg (2006)
7. Wang, S., Gao, M., Qi, L.: Freehand Sketching Interfaces: Early Processing for Sketch Recognition. In: Jacko, J.A. (ed.) HCI 2007. LNCS, vol. 4551, pp. 161–170. Springer, Heidelberg (2007)
8. Alvarado, C., Davis, R.: Sketchread: a multidomain sketch recognition engine. In: 17th annual ACM symposium on User interface software and technology, pp. 23–32. ACM Press, New York (2004)
9. QI, Y., Szummer, M., Minka, T.P.: T. P.: Diagram Structure Recognition by Bayesian Conditional Random Fields. In: 2005 IEEE International Conference on Computer Vision and Pattern Recognition, pp. 191–196. IEEE Press, Washington (2005)
10. Landay, J.A., Myers, B.A.: Sketching interfaces: Toward more human interface design. Computer 34(3), 56–64 (2001)
11. Stahovich, T.F., Davis, R., Shrobe, H.E.: Generating multiple new designs from a sketch. Artif. Intell. 104, 211–264 (1998)

12. Li, Y., Landay, J.A.: Informal prototyping of continuous graphical interactions by demonstration. In: 18th annual ACM symposium on user interface software and technology, pp. 221–230. ACM Press, New York (2005)
13. Stahovich, T.F., Davis, R., Shrobe, H.E.: Generating multiple new designs from a sketch. Artif. Intell. 2, 1022–1029 (1996)
14. Hammond, T.: LADDER: A Perceptually-based Language to Simplify Sketch Recognition User Interface Development. PhD Thesis, MIT (2007)
15. Fonseca, M.J., Jorge, J.A.: Using Fuzzy Logic to Recognize Geometric Shapes Interactively. In: 9th International Conference on Fuzzy Systems, pp. 291–296. IEEE Press, San Antonio (2000)

A Conceptual Framework for Handling Complex Administrative Processes in E-Government

Stefka Dzhumalieva and Markus Helfert

Dublin City University, School of Computing, Ireland
Stefka.Dzhumalieva2@mail.dcu.ie, Markus.Helfert@computing.dcu.ie

Abstract. Our research addresses one of the current challenges in e-government: the interoperability of ICT systems and business processes that they support. In order to achieve such interoperability, attention during the phase of analyzing processes is required. Combining the concepts of resource-based view and knowledge based view, we propose a conceptual framework that can be utilized for this task. We apply the proposed framework in selected organizational settings. In this article we summarize the application of our framework within public organizations in Bulgaria. Its usage reveals weak points and issues during process analyzing, in particular problems with processes that are spanning organizational boundaries. As a result of using our proposed framework, collaboration between organisations is facilitated and performance improved.

Keywords: Interoperability, business process management, resource-based view, knowledge-based view, e-government.

1 Introduction

In the the mid 80s and 90s a number of reform initiatives emerged in order to improve public management [1]. Influenced by the rapid advancement of information and communication technologies (ICT), these reforms in conjunction with ICT became the primary mean for achievements in public management. In order to modernize public management, the adoption of ICT is broadly referred to as electronic government (e-government) or digital government [2]. In the following we refer to e-government, comprising all ICT related initiatives in public administration.

E-government influences many areas of research. However the one directly oriented towards results and customers' satisfaction is concerned with an improvement in service delivery. This is the area where business process management (BPM) and e-government overlaps. Researchers early recognized that effective and efficient BPM is prerequisite for an effective and efficient service delivery. It requires a thorough understanding of the process behind as well as their analyzing and re-structuring [3].

Although most organisations and researchers emphasize the importance of redesigning processes, the current challenges include the interoperability of information and communication systems and the link to the business processes that they support [4]. Many complex processes, mostly spanning organizational boundaries, should be re-engineered towards ICT integration [5].

The concept of interoperability encompassed interactions at local, national and international level. It requires organizational, semantic and technical interoperability [6]. However, before achieving semantic and technical interoperability, organizational interoperability should be achieved.

Considering current trends in e-government research, BMP methodologies should be oriented towards integration and collaboration as well as oriented to the applied domain. Many BPM methodologies do not address interoperability directly and thus lacking of supporting collaboration. This is especially true for methodologies tailored to the public sector. There is a need to develop procedures, guidelines and conceptualizations, as this helps to deal with complex administrative processes.

Addressing these common shortfalls of current methodologies, our research aims to develop a conceptual framework for BPM in public administration. The framework aims to support analysis of processes in the public sector and therefore helps to improve performance. Our research builds on two observations: (1) analogies exist between administrative processes within and between different public organizations [3] and (2) appropriate analysis and evaluation of the processes would facilitate their maintenance and thus support the overall BPM endeavor [7].

In order to build an appropriate conceptualization for our framework, besides analysis of the public sector domain, our research reviews relations between: 1) The process performance and the overall organizational performance; 2) The charact-eristics of a public sector environment and the process performance

In order to derive these relations, the resource-based view (RBV) of the firm is utilized, as it represents a formal way to connect the organizational resources and capabilities with the processes performance and the overall organizational performance. It is supported by the knowledge-based view (KBV) of the firm as it represents a formal connection between knowledge and processes performance. Both RBV and KBV are applied to the public sector, while the characteristics of the environment are considered.

Our paper is structured as follows: After the introduction in Section 1, Section 2 gives some initial definitions, followed by discussion of the features of the public sector. The concept of value creation in public organization and the characteristics of administrative processes are outlined. Section 3 presents the research methodology. The concepts of RBV and KBV and their application to public organizations together with the actual framework are presented in Section 4. Initial evaluation of the proposed conceptualization is completed in Section 5. Section 6 summarizes and reflects on the work. It outlines the benefits, the limitation of the research and gives directions for further research.

2 Related Research

In order to outline characteristics of process management in public administration, in the following section we review related research. The review is structured along three main aspects:

1) The characteristics of the environment of the public sector and particularly the differences between public and private sector. This helps to facilitate the appropriate adoption of practices from the private sector to the public sector.

2) Selection of suitable performance indicators: we selected outcomes of processes in the public sector (i.e. the concept of value creation of a public organization).
3) The characteristics of processes within a public environment (i.e. the features of the administrative processes).

2.1 Differences between Public and Private Sector

The public sector is seen as a set of hierarchical structures responsive to the politicians, trough which it serves the society. It is inhabit by institutions of politics, government and bureau (administration), whereas the private sector is occupied by market institutions [8].

Three main distinctions between these two notions can be derived. (1) The main criteria for decision-making within the two sectors should be taken into account. Public sector is driven from public interest, while private sector is concerned with private interest [8]. (2) The public sector is stakeholder dependant, while private sector is shareholders dependant. Implementing policy or delivering services, governments should pay attention to the satisfaction of their stakeholders, whereas private firms should provide shareholders with their returns on investment. (3) Private sector is competition-based. In contrast, the public sector is more dependent on factors like service delivery, information provision, knowledge identification, sharing and utilization. Changes within the public sector are traditionally not driven by competitiveness [9].

Similar, but in more detail is the classification of differences provided by Rainey, Backoff and Levine [10], which consists of three categories: (1) **Environmental factors** including degree of market exposure, legal constraints and political influences; (2) **Organization-environment transactions** including coerciveness, breadth of impact, public scrutiny and unique public expectations; (3) **Internal structures and processes** including complexity of objectives, evaluation and decision criteria, authority relations, organizational performance, inventiveness and incentive structure and characteristics of governmental staff. Adopting this classification, issues relevant to the public sector can be identified and their influence can be considered, rather then imposing solutions that can lead to conflicts.

2.2 Value Creation in the Public Sector

Compared to the public sector, the conceptualization of "value" appears to be easier in the private sector. Value in the public sector is not the price of the services and it is neither the costs of performing it. The assessment of values could lead also to political debate. Eventually, even if "value" is defined within this specific environment, the public organization may have to continue to carry out procedures that do not result in any value [11]. These obstacles are directly explained by the differences between the two sectors, discussed in the previous section.

The predominant approach within the literature is to consider that the value is what is considered as value by stakeholders [12, 13]. Than the value within public organization is going to be multi-faced and must encompass many different elements [11]. It might not be possible and feasible to attempt to satisfy all of the stakeholders of a public organization, which evokes the need for identifying the key stakeholders for public organization.

Fulfilling the expectations of the key stakeholders is indirectly connected with the organization's performance [12]. It can be argued that the primary stakeholders of all public organizations are the citizens (businesses respectively). Thus, it can be concluded that the prevailing purpose of all public organizations is to create "public value" and their success strongly depends on the key stakeholders' satisfaction [14].

2.3 Characteristics of Administrative Processes

Diversity and the complexity of processes within the public organization are broadly recognized [15, 16, 17]. Nevertheless not much attention is paid on the specific characteristics of these processes. However, on reviewing selected literature the following characteristics can be identified:

- **Complexity** [15], [16], [17];
- **Information processing functions predominate**, therefore knowledge would be crucial for performance of public organization [3], [15], [17], [18];
- **Legal regulations** would impose constrains, but at the same time will leave points only to the on time interpretation [3], [17], [18];
- **Many points of interaction between customers (citizen and businesses)** on one side **and public officials and information systems within and between authorities** on the other stems for need for proper managing and integration of processes at different levels [3], [18];
- **Many individual cases of the customers** stems for dealing with big variety of cases, which could rarely appear [18]
- **Unpredictable decision making processes** is connected with dealing with individual cases and the complexity of administrative processes [18]
- **Limitations of cross-organizational collaboration** reveals need for analyzing the requirements and constraints that appear in the environment and hinder collaborative practices [18];
- **Need for support of the professional work of public official** [18];

The differences between the two sectors reveal the need for specifically adjusted solutions for the public sector. The discussed concept of value creation exposes the necessity for stakeholder satisfaction for good organizational performance in the domain on focus. However, the stakeholders' satisfaction goes trough an efficient and effective service delivery, which is based on efficient and effective BPM. Such BPM methodology needs to overcome the obstacles imposed from the administrative process characteristics and consider the current trends in the domain.

3 Research Methodology

Due to the nature of this work we adopted a design research approach. This method involves the analysis of the use and performance of designed constructs to understand, explain and to improve on the behavior or aspects of existing system [19]. Tekada in [20] distinguish between five stages of the design research cycle:

1) Awareness of problem, referring to the analysis of the public sector features
2) Suggestion for a problem solution derived from the theory, referring to exploration and application of RBV and KBV theory to a public organization
3) Development of solution (implementation of the suggestion), referring to deriving a conceptual framework for handling administrative processes
4) Evaluation of the partly or fully successful implementation of the suggestion, referring to the application of the framework.
5) Conclusions, representing the termination of the design process

It is important to outline that steps 2), 3) and 4) are interactively performed. The major limitation of the research methodology is that several iterations are needed in order to validate our outcome. Various ways of evaluation would scope the problematic area and bring more feedback to the suggestion and development stage.

4 A Conceptual Framework for Handling Complex Administrative Processes in E-Government

Above we summarized characteristics of the public sector and outlined the research methodology. In order to develop the conceptual framework, we explore the main concepts of RBV and KBV in the following section. Our aim is to relate the basic assumptions of the RBV to the processes and the process performance within public organization. This approach is combined and supported with the KBV, as it gives important insights of the most important resource of an organization – its knowledge [21]. Both the RBV and KBV are directly related to the organizations' performance.

4.1 Concepts Derived from the RBV and KBV Theory

The proposed conceptual framework is based on five main aspects derived from the theory. In following we outline these fine aspects in relation to the framework.

Barney [22] noted that a firm possesses certain resources (rare and inimitable), through which exploitation competitive advantage can be reached. Having competitive advantage good performance can be accomplished [23]. However in the public domain, resources and capabilities of an organization are valuable and rare [24]. Thus, competitive advantage within public sector is mainly achieved by the level of service provision. Thereby, if service deliverability, responsiveness, efficiency and effectiveness are capabilities possessed by public organization, they will provide competitive advantages to this organization [13]. From these researches, we conclude our first relation as **(1) public organization resources and capabilities are connected with its performance and processes performance.**

Reflecting on the work of [9, 21, 25, 26], another important relation is that **(2) knowledge is the main resource for performing processes in public organization.** Knowledge application might be crucial for the process performance due to the characteristics of the public sector and the administrative processes in particular.

Analyzing the concept of RBV in more detail, knowledge acquisition represents another linkage with the process performance [26] significant to the public sector. There is two main way for knowledge acquisition **(3) Internal learning (learning within the**

borders of the organization) and external learning (dealing with customers and/or other organizations) are the two main ways for knowledge acquisition [27].

Referring on the work of Moran and Ghoshal [28], **(4) combination and exchange of existing knowledge is the way of creation of new knowledge**. Once created, new knowledge can be acquired. It is the basis to build up the organizational knowledge. Combination and exchange of knowledge would cross organizational levels. That gives the relation to the organizational interoperability discussed earlier.

The combination and exchange of knowledge would not happen seamlessly. There will be **(5) specific requirements (or constraints) for combination and exchange of knowledge between public authorities**. They are discussed in four dimensions:

- **Knowledge requirements:** Transferability, Capacity of aggregation, Identification and appropriateness, Specialization [21];
- **Relation requirements:** Trust, Norms, Obligations [29];
- **Legal requirements:** Legal basis, Authorization, Authorization in specific case, No need for legal bases or authorization [30];
- **Semantic and technical interoperability requirements** [6];

However we do not claim to encompass all possible requirements and constraints that might appear when there is an attempt for combination and exchange of knowledge. We only consider the most critical once.

Finally, if the administrative processes are performed at an adequate level, the key stakeholders will be satisfied and good organizational performance will be accomplished. Although not part of the framework, the statement above gives the linkage to stakeholders' satisfaction and organizational performance. Figure 1 illustrates the relations between the KBV and RBV theory combined with the public organization perspective.

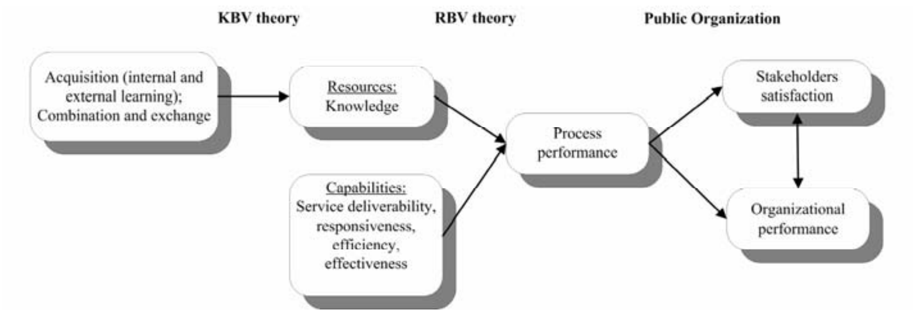

KBV theory
RBV theory
Public Organization
Acquisition (internal and external learning); Combination and exchange
Resources: Knowledge
Capabilities: Service deliverability, responsiveness, efficiency, effectiveness
Process performance
Stakeholders satisfaction
Organizational performance

Fig. 1. Relations between KBV, RBV theory combines with public organization perspective

4.2 Representation of the Conceptual Framework

Combining the main findings from the previous subsection, we can summarize following relations:

(1) resources and capabilities of public organization are connected to its process performance;
(2) knowledge is the main resource for performing processes in public organization;

(3) internal and external learning are the main ways for acquisition of knowledge;
(4) combination and exchange of knowledge (a way of integration or collaboration) is the way to resolve complex cases and to build new knowledge;
(5) the features of the public sector will impose certain requirement for combination and exchange of knowledge;

The complete conceptual framework for handling complex administrative processes is summarized and depicted in Figure 2. In following we explain the elements of the framework.

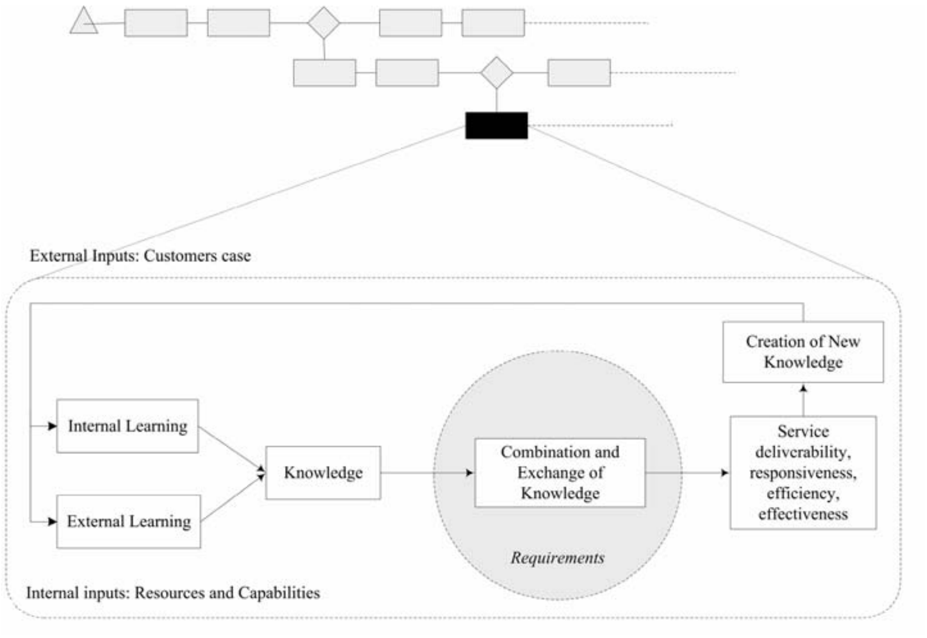

External Inputs: Customers case
Creation of New Knowledge
Internal Learning
Knowledge
Combination and Exchange of Knowledge
Service deliverability, responsiveness, efficiency, effectiveness
External Learning
Requirements
Internal inputs: Resources and Capabilities

Fig. 2. Conceptual framework for handling complex administrative processes

Performing certain services, public organization will face to manage complex administrative processes. Above we have already discussed the particular characteristics. In Figure 2 an administrative process is illustrated above the framework. The step of this process (illustrated in black box) could be either of the cases. This will invoke the usage of the proposed conceptualization.

- **Dealing with too much legal regulations;**
- **Need for investigation, research, etc. by the public official;**
- **Need for collaborative meeting (i.e. consultancy between colleagues);**
- **Need for dealing/bringing external actor (i.e. cross-organizational process, consultancy, validation of the taken decision);**
- **Serving specific individual case or case rarely appearing;**
- **Serving exception;**

Referring to Figure 2, the **external input** is represented by the case, which a public organization is dealing with, while the **internal input** relates to its resources and capabilities. There will be certain **knowledge** on which it could be relied for accurate decision making. This knowledge will be acquired trough **internal** and **external learning**. However, the knowledge possessed by the public organization will not be enough for serving all process steps. Thereby, the problem resolving would rely on **combination and exchange of existing knowledge** passing the organizational borders. However, specific **requirements** would appear. If successful, the process would have a positive outcome: **service delivery, responsiveness, efficiency, effectiveness**. Thus would refer to good process performances. Additionally, a **creation of new knowledge** can be achieved. This can be acquired again and eventually improve the organizational abilities. Overall if the case is managed properly and accurately, the outcome of the process will be positive, therefore the stakeholders (customers) will be satisfied and thus good organizational performance will be achieved.

The problematic process step, discussed in the beginning of the previous section will appear at organizational level. However the resolving of the case together with its most critical part (combination and exchange of knowledge) will have different levels. This represents the level of application of the conceptual framework:

- **Intra-organizational level**:
  a) Individual, when a public official needs to contact a colleague in order to resolve the case;
  b) Departmental, when for the resolving of the case departmental collaboration is needed;
- **Inter-organizational level**:
  a) Local, when authorities at local level should collaborate to resolve the case;
  b) National, when authorities at national level should collaborate to resolve the case;
  c) International, when authorities at international level should collaborate to resolve the case;

The conceptual framework proposed in the current section addresses the issues with specific process steps within the public domain. Once the model is applied, difficulties like constrains, enablers etc. can be revealed. Consequently the framework can contribute to overcome or at least identify these difficulties in order to achieve better performance. The next section will look at a possible application of the framework and will discuss its strengths and limitations.

5 Application of the Conceptual Framework

For an evaluation and validation of the conceptual framework, we applied it to an example. This provides insights for the benefits and limitations of the proposed framework. We examined a public organization based in Bulgaria (District Governor of Veliko Tarnovo District). We selected one of the services provided, analyzed it and applied the conceptual framework. It is followed by a discussion and an evaluation.

The District Governor is considered as territorial body of the executive power. Appearing as a middle level between central administration (the government) and local territorial administration (municipalities), the main responsibilities of the District Governor are concerned with coordination and control, (Law for the administration of Republic of Bulgaria, Art. 19 (3). Art. 29 (1), (3), Art. 57 (2) citied and translated in [31].

For our analysis we selected the process of the administrative service: "Approval of changes of district transport schedules". The District Governor is responsible for the coordination and approval of any changes to the transport schedules between the territories of two or more municipalities. The actual service of transportation of the citizens is outsourced. The legal bases for performing the service is according to decree № 2 from 15th of March 2002 for the terms and regulations for approval of transport schedules for carrying out of public transportation of passenger with buses and cars. The service is provided only for municipalities [32].

The performing of the service is initiated by an "argumentative proposal" for any changes of the transport schedules. The proposal is completed by the mayor of the correspondent municipality and brought to the attention of the District Governor. Within one month the District Governor should approve the requested changes or give justified refusal [32], [33].

Modeling and analysis of the current situation of a process promote transparency and support understanding, it reveals week points and thus ways of potential improvements [3]. In our research we identified week points and exposed at which steps the proposed conceptual framework can be applied.

The identified critical issues of the discussed administrative service are:

1) It is a rare service, thus difficulties concerning the necessary knowledge and professional experience for performing the service might occur
2) It is not possible to be familiar with the law regulation for each of the performed services, thus there will be need for exploring the law base before initiating performing of the service and there could be need for consultancy with senior co-worker(s) about the interpretation of the law bases.
3) There is a need for organizing committee meetings (consisting of representatives of other organizations concerned with the changes of the transport schedules). This evokes bringing external actors to the organization.

The first application of the framework can be completed at point 1 from the list presented. This application is at intra-organizational, individual level. The public official - dealing with the case- possesses certain amount of knowledge. This is a resource of the public organization and internal input to the problem resolving step, while the details of the provided service are the external input. The knowledge would be acquired through internal and external learning. If this is not enough, further consultation and collaboration for performing the task is required. The service delivery will be dependant on this collaborative practice. Considering the four dimensions of requirements for combination and exchange of knowledge:

- Knowledge requirements should be strongly considered (i.e. the ability of knowledge to be communicated)
- Relation requirements might not be as strong, due to the fact that the application is at intra-organizational, individual level. There would be certain amount of trust between colleagues, as well as obligation to collaborate.

- Legal requirements will not exist at intra-organizational level
- Semantic and technical requirements should be considered when the process step is partly or fully automated

Once the issue is resolved, it will feed back to the person's knowledge (and thus to the organizational as well) via the internal/external learning. Using the framework, issues when performing the process are revealed. That is the first important step for addressing the problem areas.

Reflection on this application is that when coping with such process step not always creation of new knowledge will appear. Simply sharing could be enough. This gives good insight for improvement of the proposed framework. Looking at point 2, possible application would lead to the same conclusion.

As this application is only at intra-organizational level, other possible applications could be for example for local social security agencies. Increasingly these agencies have to deal with cases in which citizens have worked in different countries and paid into various pensions funds. The local agency has to contact all other referred agencies, in order to calculate the citizen pension [6]. The local agency, in particular in small towns might not have dealt with such cases. Further, if interoperability for the service is achieved at European Union level, the service might be automated.

6 Summary and Conclusions

Our research addresses the current challenges of BPM in the public sector: the interoperability of the ICT systems and the processes that they support. In order to enable ICT system towards sharing of information and knowledge and facilitating collaboration, first the processes that they support should be analyzed and improved.

To approach the problematic field, the research explored the specification of the public sector. It makes rather broad analysis of the public sector, which is commonly missing in similar researches. This approach facilitates adoption of practices from the private sector and gives awareness of factors to be considered, when solutions for public organizations are provided.

To be able to build appropriate solution, the RBV and KBV theory were explored and applied to a public organization. These theories were used as a way to connect resources capabilities of public organization with process performance. The overall organizational performance was linked to the most important resource of public organization, its knowledge. Although approaches exist, the application of RBV, combined with KBV to public organization is a novel approach and relatively unexplored within the research community.

The main outcome of our work is conceptual framework for handling complex administrative processes in e-government. In comparison with other BPM methodologies (i.e. [34] and [35]) it handles only specific part of the whole method. However exactly this part brings the public sector environment adjustment and orientation towards interoperability and collaboration. Our framework will support the analysis of the case handling type of administrative processes, which is rarely covered by BPM conceptualizations. Its proper utilization will improve the process performance, which will give greater satisfaction to stakeholders and thus, better organizational performance will be achieved.

Despite of the benefits of the research, its limitations and directions for further research should be outlined as well. Firstly, the iteration of the research methodology was performed only once. More stages of the evaluation, where external view is involved, would give various feedbacks. The outcome of the first application of the model (sharing of knowledge), should expand the research. As mentioned, the proposed conceptualization covers only partly a BPM methodology and it is still at high abstract level. Research is needed in order to achieve model adjustment to the BPM methodologies used in practice and to bring the conceptualization from the abstract level to a technical layer. Tools and applications which can support its different sides should be explored.

Beyond the outcomes of the research, the practices in the public sector should be better analyzed. As Winter in [5] stated, only small part of the potential optimization in the field of e-government is achieved.

References

1. Beynon-Davies, P.: Models for e-government. Transforming Government: People, Process and Policy 1, 7–28 (2007)
2. Asgarkhani, M.: Digital government and its effectiveness in public management reform. Public Management Review 7, 465–487 (2005)
3. Becker, J., Algermissen, L., Niehaves, B.: Organizational Engineering in Public Administrations – A Method for process-oriented eGovernment projects. In: 19th Annual ACM Symposium on Applied, pp. 1385–1389. ACM Press, New York (2004)
4. Sarikas, O., Weerakkody, V.: Realising integrated e-government services: a UK local government perspective. Transforming Government: People, Process and Policy 1, 153–173 (2007)
5. Winter, A.: E-government: vision and implementation. In: Tranmuller, R., Makolm, J., Orthofer, G. (eds.) Eastern European E-gov Days 2007: Best practices and Innovation, pp. 3–9. Austrian Computer Society, Vienna (2007)
6. European Commission. European interoperability framework for pan-European eGovernment services, `http://ec.europa.eu/idabc/servlets/Doc?id=19529`
7. Aguilar, E.R., Ruiz, F., Garcia, F., Piattini, M.: Evaluation Measures for Business Process Models. In: 21th Annual ACM Symposium on Applied Computing, pp. 1567–1568. ACM Press, Dijon (2006)
8. Lane, J.-E.: The public sector: concepts, models and approaches. Sage, London (1995)
9. Cong, X., Pandya, V.: Issues of Knowledge Management in the Public Sector. Electronic Journal of Knowledge Management 1, 25–33 (2003)
10. Rainey, H., Backoff, R., Levine, C.: Comparing public and private organizations. Public Administration Review 36, 233–244 (1976)
11. Halachmi, A., Bovaird, T.: Process reengineering in the public sector: Learning some private sector lessons. Technovation 17, 227–235
12. Bryson, J.: What to do when stakeholders matter: Stakeholders identification and analysis techniques. Public Management Review 6, 21–53 (2004)
13. Bryson, J., Ackermann, F., Eden, C.: Putting the resource-based view of strategy and distinctive competencies to work in public organization. Public Administration Review 67, 702–717 (2007)
14. Moore, M.: Creating Public Value. Harvard University Press, Cambridge (1995)

15. McAdam, R., Donaghy, J.: Business process re-engineering in the public sector: A study of staff perceptions and critical success factors. Business Process Management Journal 5, 33–49 (1999)
16. Palkovits, S., Woitsch, R., Karagiannis, D.: Process-based knowledge management and modeling in e-government – an inevitable combination. In: 4th IFIP International Working Conference, pp. 614–621. Springer, Heidelberg (2003)
17. Olbrich, S., Simon, C.: Integration of legal constraints into business process models. Transforming Government: People, Process and Policy 1, 194–210 (2007)
18. Klischewski, R., Lenk, K.: Understanding and modeling flexibility in administrative processes. In: Traunmüller, R., Lenk, K. (eds.) EGOV 2002. LNCS, vol. 2456, pp. 129–136. Springer, Heidelberg (2002)
19. Association for Information Systems, `http://www.isworld.org`
20. Takeda, H., Veerkamp, P., Tomiyama, T., Yoshikawam, H.: Modeling Design Processes. AI Magazine Winter, 37–48 (1990)
21. Grant, M.R.: Toward a knowledge-based theory of the firm. Strategic management journal 17, 122–192 (1996)
22. Barney, J.: Firm resources and sustained competitive advantage. Journal of Management 17, 99–120 (1991)
23. Newbert, S.L.: Empirical research on the resource-based view of the firm: An assessment and suggestion for future research. Strategic Management Journal 28, 121–146 (2007)
24. Demsetz, H.: Industry Structure, Market Rivalry, and Public Policy. Journal of Law and Economics 16, 1–9 (1973)
25. Nonaka, I.: The Knowledge-Creating Company. Harvard Business Review 69, 96–104 (1991)
26. Teece, D.J., Pisano, G., Shuen, A.: Dynamic capabilities and strategic management. Strategic Management Journal 18, 509–533 (1997)
27. Schroeder, R.G., Bates, K.A., Junttila, M.A.: A resource-based view of manufacturing strategy and the relationship to manufacturing performance. Strategic Management Journal 23, 105–122 (2002)
28. Moran, P., Ghoshal, S.: Value creation by firms. In: Keys, J.B., Dosier, L.N. (eds.) Academy of Management Best Paper Proceedings, pp. 41–45 (1996)
29. Nahapiet, J., Ghoshal, S.: Social capital, intellectual capital, and the organizational advantage. Academy of Management Review 23, 242–266 (1998)
30. Otjacques, B., Hitzelberger, P., Feltz, F.: Interoperability of E-Government Information Systems: Issues of Identification and Data Sharing. Journal of Management Information Systems 24, 29–51 (2007)
31. OECD, http://www.oecd.org
32. Executive agency "Automobile administration", `http://www.rta.government.bg`
33. Tehnologichna karta za administrativna usluga "Utvarzhdavane oblastnata transportna shema" (Internal documentation of District Governor of Veliko Tarnovo)
34. Kragiannis, D.: Business Process Management Systems. SIGOIS Bulletin 16, 10–13 (1995)
35. Weske, M., van der Aalst, W.M.P., Verbeek, H.M.W.: Advances in Business Process Management. Data & Knowledge Engineering 50, 1–8 (2004)

Fuzzy Time Intervals for Simulating Actions

Vadim Ermolayev[1], Natalya Keberle[1], Wolf-Ekkehard Matzke[2], and Richard Sohnius[2]

[1] Department of IT, Zaporozhye National University, Zhukovskogo 66, 69063, Zaporozhye, Ukraine
vadim@ermolayev.com, kenga@zsu.zp.ua
[2] Cadence Design Systems, GmbH, Mozartstr. 2 D-85622 Feldkirchen, Germany
wolf@cadence.com, sohnius@cadence.com

Abstract. The paper presents time-related part of PSI[1] theoretical framework. In comparison to other theories of time based on interval logics our approach presents the advancement by introducing fuzziness of time intervals as transition periods at beginnings and endings. It is argued that, though quite simple (discrete, linear, and anisotropic), our theoretical model is expressive enough to be used as a logical formalism for reasoning about stochastic, unpredictable, weakly defined action and process flows. A metric and a rich set of axiomatic relationships among time intervals are introduced for that. Further on, a means for modeling and reasoning about singular, repeated, regular events and actions having phases and vague durations is elaborated. Presented theory of time is used for modeling and reasoning about events, environmental influences, happenings, and actions while planning and scheduling in our simulations of dynamic engineering design processes.

1 Introduction

A major trend in engineering design today is that a design system is flexible and responsive enough to be capable of meeting and compensating sudden changes. Changing factors may be: time-to-market constraints, design specification. Unpredicted distorting external influences like sudden reorganization of a design team, increased number of design activity iterations due to the changes in the quality requirements, or factual unavailability of a required resource may also influence a design system [1]. Another important feature of a design system is the increasing geographical and cultural distribution bringing up new challenges to performance management. Indeed, provided that the parts of a design system are spread globally, the proper time management may substantially increase "round-the-clock" performance.

Time modeling is a crucial feature in a vast variety of application domains dealing with change. Probably the first to perceive it was Aristotle [2]. The importance of time modeling is shown by the numerous works in the area of temporal databases [3] and temporal reasoning [4]. Our motivation for developing the model of time presented in this paper is the observation that accounting for mentioned specificities of changes in design processes requires a rich and expressive model of time.

[1] Performance Simulation Initiative (PSI) is the project of Cadence Design Systems, GmbH.

The rest of the paper is structured as follows. The context and the tasks of PSI project are briefly presented in Section 2. The requirements to the model of time are devised and outlined based on the tasks of the project. Section 3 presents modeling choices based on the requirements. Section 4 describes the basic crisp part of our general model of time and time interval calculus. Section 5 introduces fuzziness in the beginnings and endings of time intervals providing means for modeling vagueness, uncertainty, and subjectivity. The means for reasoning about irregularity, regularity, and recurrence of events and actions is introduced in Section 6. Section 7 reviews the related work in the field of fuzzy temporal reasoning and compares our results to the results found in the literature. Finally concluding remarks and our plans for future work are given.

2 Modeling Requirements

PSI project aims at developing models, methodologies, and software tools providing rigorous engineering treatment of performance and performance management. PSI performance modeling and management approach focuses on performance as a pro-active action. A fine-grained dynamic model of an Engineering Design Process is therefore developed. PSI approach considers that performance does not occur in vacuum, but is embodied in its environment and is controlled by the associated performance management process.

A Dynamic Engineering Design Process (DEDP) is a goal-directed process of transforming the representations of a design artifact in stateful nested environments. An environment comprises design artifact representations, resources, tools, and actors who perform actions to transform design artifacts using tools, consume resources. Actions are admissible in particular environmental states and may be atomic or compound, state-transitive or iterative, dependent or independent of other actions. The components of an environment may generate internal events or may be influenced by external events that are generated outside of this environment at run time. Events may have causal dependencies. A DEDP is considered a problem solving process which goals, partial goals, and environments may change dynamically. In PSI a decision taking procedure is associated with each state to allow environments adjust the course of a DEDP taking these changes into account. Decisions are taken by actors modeled by software agents.

PSI software tools are developed for assisting project managers to make robust planning, monitoring, and management of their design projects aiming at reaching best possible performance. Grounded decisions in planning are based on the knowledgebase of project logs accomplished in the past. These logs provide vast and finely grained records of the performance of the accomplished projects and may be used for simulating the behavior of the design system in response to different influences. At project execution phase PSI software may be used for predicting the behavior of the design system in the future based on the record of the partially accomplished DEDP, the knowledge about its environment(s), and performance simulations.

Mentioned functionalities may only be implemented if a rich and expressive model of time is used. This model should be capable of facilitating agents reasoning about environments, events, and actions employed in decision taking procedures enacted at environmental states. In particular, PSI bases its model of time on the following modeling requirements. As far as our objective in this development was providing the ontology of time for the use in agent-based software tool for DEDP simulation, the requirements were graded with respect to the use in the versions of the software as shown in Table 1. Minimal ontology of time is used in the current software of PSI software prototype. Crisp ontology of time is planned for the upcoming major version. Full ontology of time based on the fuzzy extension of the model of time will be used in the future versions. More details may be borrowed from [11].

Table 1. Required features of the model of time

Feature	Time Minimal	Time Crisp	Time Fuzzy
Absolute time points	X	X	X
Differently structured and grained time stamps: dates, times, time zones		X	X
Time intervals and their durations	x	x	X
Time intervals open or closed by beginning and ending instants			X
Finite and infinite time intervals		x	X
Time intervals with vague beginnings and endings			X
Means to analyze the overlaps of time intervals		x	X
Subjective treatment of beginnings and endings of actions by different actors			X
Sub-intervals of a time interval of an action (phases)		x	X
Regular intervals associated with working days, weekends and irregular but repeating intervals like that associated with vacations		x	X
Means for modeling and analyzing changing availability, non-availability, or consumption of resources with extrapolation to the future		x	X
Means for analyzing actor availability and occupancy in time: Who is doing what, when, and what is the capacity spent rated by the overall capacity?		x	X
Means for modeling and analyzing instant events or events with durations		x	X
Means for modeling and analyzing causal relationships among events in time: influences, pre-conditions, immediate-, and post-effects		x	X
Means for modeling and analyzing subjective perceptions of events: happenings			X
Means for modeling and analyzing atomic and compound events having different temporal parts with or without causal dependencies		x	X

Legend: x – partial coverage; X – full coverage dealing with vagueness and subjectivity.

3 Modeling Choices

Requirements analysis revealed that a simplified model of time is appropriate for simulation purposes in PSI. Time is represented by a single time line – a linear, anisotropic, discrete set of time instants as shown in Fig. 1. It is assumed in our model of time that the following properties hold.

Time is linear. Several branching time models exist. To mention just one of them, OMG adopts the model based on several time lines with different clocks and synchronization

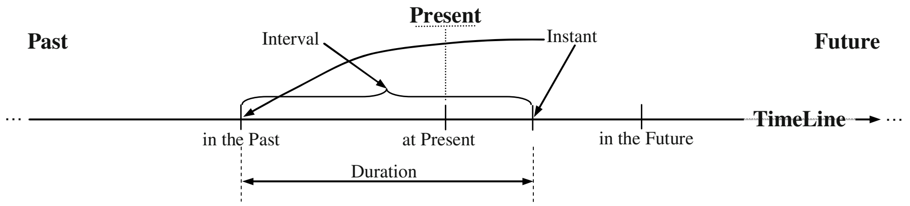


Fig. 1. An outline of the model of time

relationships for modeling and analysis of real-time and embedded systems [5]. However, for the purposes of PSI linearly anisotropic time model is sufficient. By stating that we also assume that there exists a single reference clock and all the clocks in a distributed system are synchronized with it, possibly with certain shifts representing time zones. By postulating the linearity of time we do not state that courses of action are linear. Their execution is deterministically linear in the past and at present, but may take alternative paths in the future.

Time is anisotropic. Indeed, processes are developed from the Past to the Future. If we need going back in the process we do not reverse time, but step back the subsequence of atomic actions already performed and take a repetition or a different process path. These repetitions or alternative paths are actions developed in the same temporal direction – from the Past to the Future. Events that have duration are developed in the direction from the Past to the Future. Causal relationships among Events are directed from the Past to the Future. A derivative of this assumption is that anisotropic binary relationships may be set among time instants and time intervals.

Time is discrete. Discrete model of time is used because its properties are sufficient for our simulation purposes. Computer clocks are used to measure the values of time points in simulations. These clocks have an upper bound of frequency constrained by hardware implementation. Hence, dense time, or continuous time models can not be feasibly used. The time line in PSI is used as a basic dimension in our simulation experiments, in discrete settings.

Time intervals are fuzzy. The introduction of fuzzy sets as the models of the beginnings and endings of time intervals allows accounting for uncertainty, vagueness and subjectivity in our treatment of events, happenings, actions, and their dependencies. Further on, fuzzy time intervals may be composed in interval sets denoted as periods and may contain sub-intervals denoted as phases.

4 Crisp Model of Time

A Time Line is denoted as an abstract axis describing abstract time as an Abelian group[2] with respect to addition operation $(T,+)$. The elements of a Time Line are time instants (or instants in short).

An instant *t* is a point in time having no duration. An instant has the value reflecting its position on the Time Line. The value of an instant may be:

[2] In abstract algebraic sense an Abelian group is a group in which the group operation holds commutativity.

– Negative: in this case the instant is said to be in the Past
– Zero: in this case the instant is at Present and is the Zero of the Time Line
– Positive: in this case the instant is said to be in the Future.

4.1 Linearity of Time

Time is assumed to be linear. If put formally the linearity of time means that a *total order* is set among the instants on the Time Line. Given two arbitrary instants t_1 and t_2, one of the following statements holds true:

$$Before(t_1, t_2) \text{ – means that } t_1 \text{ is before } t_2 \quad (1)$$
$$After(t_1, t_2) \text{ – means that } t_1 \text{ is after } t_2 \quad (2)$$
$$Same(t_1, t_2) \text{ – means that } t_2 \text{ and } t_1 \text{ are the same instant} \quad (3)$$

The following statements on the relationships among *Before*, *After*, and *Same* hold true:

$$Before(t_1,t_2) \equiv After(t_2,t_1) \quad (4)$$
$$Before(t_1,t_2) \wedge Before(t_2,t_1) \equiv Same(t_1,t_2) \quad (5)$$
$$After(t_1,t_2) \wedge After(t_2,t_1) \equiv Same(t_1,t_2) \quad (6)$$

4.2 Time Intervals

A time interval $I = [t^b, t^e]$ is a segment of time bounded by an instant t^b at the beginning, an instant t^e at the end, and $Before(t^b,t^e) \wedge \neg Same(t^b,t^e)$. The latter property hints about the anisotropy of time[3]. From a set theoretical perspective a time interval may be denoted as an ordered triple containing its beginning instant, the set of its inner instants, and its ending instant:

$$I = \{t^b, T^i, t^e\}, \quad (7)$$

where the order among the constituents is denoted as:

$$\forall t_j \in T^i (Before(t^b,t_j) \wedge \neg Same(t^b,t_j)) \wedge (After(t^e,t_j) \wedge \neg Same(t^e,t_j)) \quad (8)$$

The assumption about the discrete character of time allows us to denote a unit time interval. A unit time interval is a time interval $I_u = \{t^b, T_u^i, t^e\}$, where $T_u^i = \varnothing$. As the set of its inner instants is empty, it comprises only two instants: the beginning instant t^b and the ending instant t^e, therefore the following holds true:

$$\forall t : (Before(t^b,t) \vee After(t^e,t)) \rightarrow (Same(t,t^b) \vee Same(t,t^e)) \quad (9)$$

Binary relationships among time intervals are more complex than the ones over instants. Following [7], the set of binary relationships among intervals is defined below based on the binary relationships of their beginning and ending instants. Assume $I_1 = [t_1^b, t_1^e]$, $I_2 = [t_2^b, t_2^e]$. Then the following hold true:

[3] Otherwise we should have written $\neg Same(t^b,t^e)$.

I_1 is (distinctly) before I_2:

$$Before(t_1^e, t_2^b) \wedge \neg Same(t_1^e, t_2^b) \equiv Before(I_1, I_2) \quad (10)$$

I_1 meets I_2[4]:

$$Before(t_1^b, t_2^b) \wedge Same(t_1^e, t_2^b) \equiv Meets(I_1, I_2) \quad (11)$$

I_1 (distinctly) overlaps I_2:

$$\begin{aligned} &Before(t_1^b, t_2^b) \wedge \neg Same(t_1^b, t_2^b) \wedge Before(t_2^b, t_1^e) \wedge \neg Same(t_2^b, t_1^e) \\ &\wedge After(t_2^e, t_1^e) \wedge \neg Same(t_2^e, t_1^e) \equiv Overlaps(I_1, I_2) \end{aligned} \quad (12)$$

The relaxations of (12) result in the following:

I_1 contains I_2:

$$\begin{aligned} &Before(t_1^b, t_2^b) \wedge \neg Same(t_1^b, t_2^b) \wedge After(t_1^e, t_2^e) \\ &\wedge \neg Same(t_1^e, t_2^e) \equiv Contains(I_1, I_2) \end{aligned} \quad (13)$$

I_1 starts I_2:

$$Same(t_1^b, t_2^b) \wedge After(t_2^e, t_1^e) \wedge \neg Same(t_2^e, t_1^e) \equiv Starts(I_1, I_2) \quad (14)$$

I_1 finishes I_2:

$$After(t_1^b, t_2^b) \wedge \neg Same(t_1^b, t_2^b) \wedge Same(t_1^e, t_2^e) \equiv Finishes(I_1, I_2) \quad (15)$$

I_1 lies within I_2:

$$Starts(I_1, I_2) \vee Contains(I_2, I_1) \vee Finishes(I_1, I_2) \equiv Within(I_1, I_2) \quad (16)$$

I_1 is the same as I_2:

$$Same(t_1^b, t_2^b) \wedge Same(t_1^e, t_2^e) \equiv Same(I_1, I_2) \quad (17)$$

4.3 Infinite Time Intervals

An extension of the theory of time intervals is the account for their possible infinity at the beginning, at the end, or at both ends. We shall say that a time interval ${}^{\infty}I = [-\infty, t^e] = \{T^i, t^e\}$ is infinite at the beginning iff:

$$\forall t_j \in T^i \exists t_i : (t_i \in T^i) \wedge Before(t_i, t_j) \quad (18)$$

We shall say that an interval $I^{\infty} = [t^b, \infty] = \{t^b, T^i\}$ is infinite at the end iff:

$$\forall t_j \in T^i \exists t_i : (t_i \in T^i) \wedge After(t_i, t_j) \quad (19)$$

In frame of our understanding of time there exists only one interval ${}^{\infty}I^{\infty} = [-\infty, \infty] = \{T^i\}$ which is infinite at both ends – the Time Line. For ${}^{\infty}I^{\infty}$ it holds true that:

$$\forall t_j \in T^i \exists t_i, t_k : (t_i, t_k \in T^i) \wedge Before(t_i, t_j) \wedge After(t_k, t_j) \quad (20)$$

[4] Please note that I_2 does not meet I_1.

Please notice that *Past* and *Future* on the Time Line (Fig. 1) are the intervals which are infinite respectively at the beginning and at the end and bounded by the instant of *Present* at their finite end:

$$Past = [-\infty, Present] = \{T^i, Present\}; Future = [Present, \infty] = \{Present, T^i\} \quad (21)$$

Having defined Past and Future as time intervals and Present as an instant we may now reason about several temporal properties of arbitrary time intervals. We may say that an interval *I* continues if it overlaps with the Past:

$$Overlaps(I, Past) \equiv Continues(I) \quad (22)$$

We may say that an interval *I* ends now (at Present) if it finishes the Past (or meets the Future):

$$Finishes(I, Past) \equiv Ends(I) \text{ or } Meets(I, Future) \equiv Ends(I) \quad (23)$$

We may say that an interval is finished if the Past contains it:

$$Contains(Past, I) \equiv IsFinished(I) \quad (24)$$

We may say that an interval begins now (at Present) if it starts the Future (or is met by the Past):

$$Starts(I, Future) \equiv Begins(I) \text{ or } Meets(Past, I) \equiv Begins(I) \quad (25)$$

We may say that an interval has not yet begun if it lies within the Future:

$$Contains(Future, I) \equiv NotYetBegun(I)\,. \quad (26)$$

4.4 Metrics of Time

The following two aspects should be defined for associating meaningful values with the constituents of the Time Line: (i) the values of time instants; and (ii) the metric for time intervals.

Having assumed the linearity and the discreteness of Time we may denote the values of instants as mappings to integers[5]. We shall first define the basic mapping presuming that Present maps to zero of integers and the instants in the Past and the Future map to negative[6] and positive integers respectively. We shall then refine this mapping by introducing the instant in the Past, being the beginning of measured time.

Let us associate the value equal to zero with the instant t_0 (at *Present* – refer to Figure 1). Then an arbitrary instant t_p located in n unit time intervals to the *Past* of t_0, will have the value equal to $-n$ and an arbitrary instant t_f located in m unit time intervals to the *Future* of t_0, will have the value equal to m.

An alternative and, possibly, a more convenient way of defining instant values is to shift the zero point from the instant of Present to the instant which is commonly recognized as the zero reference point of time. For example, in Christian chronology

[5] The question about the clock units (nanoseconds, …, centuries) of these integers is intentionally left open because it depends on the frequency of the available clock.

[6] Mappings to negative values may be percept as artificial because human calendars and other ways of measuring time operate with positive values. However, there are timelines which are similar to the one proposed here. For example, a logarithmic timeline starts in the past (at Big Bang time) and lasts till Present: http://en.wikipedia.org/wiki/Logarithmic_timeline.

such a point is the instant implicitly associated with the happening[7] of the birth of Christ. It is topical to notice that year 0 does not exist in the chronology introduced by Dionysius Exiguus. Christian Era starts with Anno Domini (A.D.) which is year 1. A.D. is preceded by A.C.[8] Christian chronology is the most widely accepted time measurement system in the World today. We shall therefore align the basic mappings of time instant values to integers with it, when convenient.

Relative and absolute positions of time intervals on the Time Line may be measured and compared by referring to their beginning or ending instants. One more metric aspect of an interval is its duration. It is straightforward to assume that the duration $|I|$ of a finite time interval $I=[t^b,t^e]$ is the quantity of different[9] unit time intervals which lie within I in the sense of (16). It is easy to prove that $|I|$ can be computed as:

$$|I| = t^e - t^b \,. \tag{27}$$

4.5 Structure of Instant Values and Durations: Date and Time

Several finite time intervals and their durations are used in chronology. These intervals are associated with the phenomena occurring in the physical world. The following are trivial and very well known facts. They are discussed with a purpose of demonstrating the use of our means for modeling repeating and regular events (Section 6).

A year is a period associated with the event of revolution of the Earth around the Sun. A month is period which is the phase of a year and has the duration uncertainly equal to 1/12-th of the duration of a year. A day is a period associated with the revolution of the Earth around its own axis. An hour is the period with duration equal to 1/24-th of the duration of a day. A minute is a Period with duration equal to 1/60-th of the duration of an hour. A second is a Period with duration equal to 1/60-th of the duration of a minute.

A calendar is eventually a rule system which sets the relationships among the durations of a year, a month and a day. It organizes periods within a year using the concept of a date. Gregorian calendar, being the actual dating system today, states that: a year contains 12 months; a month comprises 28 to 31 days; a year starts on the first day of the first month (called January); the total order among the days in a year is set by dates – the values associated with days; a date contains the number of a month complemented by the number of a day within the month. The number of a year is added to a date for setting up the order among the days of different years. The number of a year is calculated as a successive number starting from A.D.

4.6 Shifting Instant Values: Time Zones

One of the important corollaries of the linearity of Time is that a time instant has one and only one value. Indeed, if supposed that it does not hold true, it is impossible to set the total order among the instants. An interesting question however is: how to reflect the existence of different time zones in such a theory.

[7] In our theory of events and happenings [11] a happening is the percept of the event of the occurrence of a real world phenomenon by an observer.

[8] Catholic Encyclopedia: http://www.newadvent.org/cathen/03738a.htm

[9] In the sense: $\mathit{Different}(I_1,I_2) \equiv \neg \mathit{Same}(I_1,I_2)$.

Based on the assumed linearity of time we stand on the assumption that an instant measured at different locations corresponding to different time zones has the same value. However, a relative shift[10] corresponding to the zone of measurement is associated with an instant. For example, the instant of the New Year 2007 has the only value (the quantity of unit time intervals passed from that instant to Present) irrespectively to the location where it has been percept. Time shifts are different at different locations.

5 Fuzzy Extension of the Crisp Model of Time

In reality time intervals may have fuzzy boundaries because human treatment of them is often vague and subjective. For example, when we say "I'll be back in a second", it normally (among humans) means that: (i) the action of returning back will take us quite a short time interval which duration is (desired to be) about one second; (ii) it is well possible that after one second passes we are still on our way; (iii) the probability of been back is greater than zero after 0.8 second time interval has passed and is definitely equal to one after 20 seconds.

Fuzzy time intervals may help modeling situations alike. We shall use our set theoretic definition of a time interval (7, 8) as the basic one and denote a fuzzy time interval as a fuzzy set with its membership function over the crisp set of time instants. Graphical representation is given in Fig. 2.

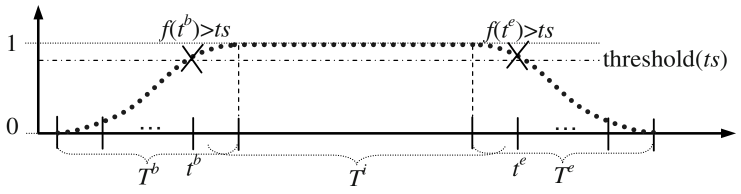


Fig. 2. A membership function of a fuzzy time interval

A fuzzy time interval I in a set theoretical sense is an ordered triple containing the fuzzy set of its beginning time instants, the (crisp) set of its inner time instants, and the fuzzy set of its ending time instants:

$$I = \{T^b, T^i, T^e, f\}, \tag{28}$$

where:

- T^b is the fuzzy set of instants which may be the beginning of I. If an instant $t_j^b \in T^b$ is the beginning t^b of I then $\forall t_j^b : t_j^b > t^b \rightarrow t_j^b \in T^i$
- T^i is the inner set of instants of I (also called the core [12])
- T^e is the fuzzy set of instants which may be the ending of I. If an instant $t_j^e \in T^e$ is the ending t^e of I then $\forall t_j^e : t_j^e < t^e \rightarrow t_j^e \in T^i$
- $f : \mathrm{Z} \rightarrow [0,1]$ is a discrete membership function

[10] Introduction of such a shift is natural. For example, a time zone in Italian is "fuso orario".

The order among the constituents is denoted similarly to (8):

$$\forall t_j \in T^b, t_k \in T^i, t_l \in T^e \, (Before(t_j, t_k) \wedge \neg Same(t_j, t_k)) \\ \wedge (After(t_l, t_k) \wedge \neg Same(t_l, t_k)) \tag{29}$$

A considerable difficulty in setting up relationships among the elements of fuzzy time intervals and fuzzy time intervals as wholes is that their beginning and ending instants can not be crisply determined. Therefore we have to denote these relationships having in mind that any member of T^b may appear to be t^b and any member of T^e may appear to be t^e. In that sense a fuzzy interval I may be also denoted as $I = [t^b, t^e]$ – a segment of time bounded by an instant t^b at the beginning, an instant t^e at the end, and $Before(t^b, t^e) \wedge \neg Same(t^b, t^e)$. In difference to a crisp interval the beginning and ending instants are located using the membership function *f*.

5.1 Binary Relationships among Fuzzy Time Intervals

We shall denote binary relationships (30-39) for fuzzy time intervals using (10-17) as the basics. An idea is that for fuzzy intervals their beginnings and ends are not crisp instants, but fuzzy sets of instants. Binary relationships among fuzzy intervals are more complex than the ones over intervals (10-17). The set of binary relationships among fuzzy intervals is defined below. Graphical illustration is given in Fig. 3. Assume, $I_1 = \{T_1^b, T_1^i, T_1^e, f_1\}$, $I_2 = \{T_2^b, T_2^i, T_2^e, f_2\}$. Then the following hold true:

I_1 is definitely before I_2:

$$Before(T_1^e, T_2^b) \equiv Before(I_1, I_2) \tag{30}$$

I_1 is likely before I_2:

$$\forall t \in T_1^e \cap T_2^b \, f_1(t) + f_2(t) < 1 \equiv LikelyBefore(I_1, I_2) \tag{31}$$

I_1 likely meets I_2:

$$\forall t \in T_1^e \cap T_2^b \, f_1(t) + f_2(t) \geq 1 \equiv LikelyMeets(I_1, I_2) \tag{32}$$

I_1 definitely meets I_2:

$$Starts(T_1^e, T_2^i) \vee Finishes(T_2^b, T_1^i) \equiv Meets(I_1, I_2) \tag{33}$$

I_1 overlaps I_2:

$$Meets(T_1^i, T_2^i) \vee Overlaps(T_1^i, T_2^i) \equiv Overlaps(I_1, I_2) \tag{34}$$

The variations of (34) result in the following:

I_1 contains I_2:

$$(Contains(T_1^i, T_2^i) \vee Same(T_1^i, T_2^i)) \wedge (f_1 > f_2) \equiv Contains(I_1, I_2) \tag{35}$$

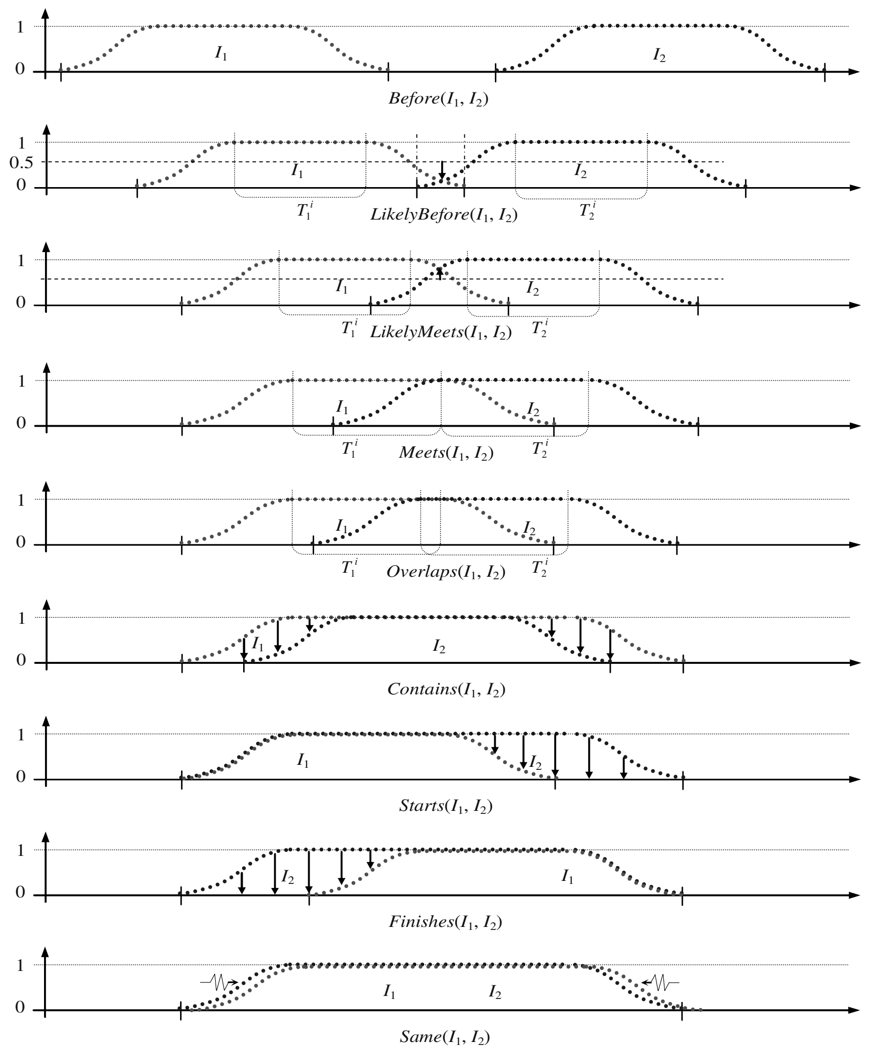


Fig. 3. Graphical representation of binary relationships among fuzzy time intervals

I_1 starts I_2:

$$Starts(T_1^i, T_2^i) \equiv Starts(I_1, I_2) \tag{36}$$

I_1 finishes I_2:

$$Finishes\ (T_1^i, T_2^i) \equiv Finishes\ (I_1, I_2) \tag{37}$$

I_1 lies within I_2:

$$Starts(I_1, I_2) \vee Contains(I_2, I_1) \vee Finishes(I_1, I_2) \equiv Within(I_1, I_2) \tag{38}$$

I_1 is the same as I_2:

$$Same(T_1^b, T_2^b) \wedge Same(T_1^i, T_2^i) \wedge Same(T_1^e, T_2^e) \equiv Same(I_1, I_2) \tag{39}$$

Please note that in (30) we do not require $f_1 \equiv f_2$. Therefore, I_1 and I_2 may in fact appear to be different. One of the interesting facts is that the theory of fuzzy intervals (28-39) is the proper extension of the theory of (crisp) intervals (7, 8, 10-17).

COROLLARY: If the constituents of I, I_1, and I_2 comply with the following:

$$T^b = \{t^b\}, T^e = \{t^e\}, T_1^b = \{t_1^b\}, T_1^e = \{t_1^e\}, T_2^b = \{t_2^b\}, T_2^e = \{t_2^e\}, \text{ and } f_1 \equiv f_2 \equiv 1, \tag{40}$$

then formulae (28-39) are logically equivalent to the formulae (7, 8, 10-17).

OUTLINE OF THE PROOF: We shall prove the statement of the corollary by transforming each of (28-39) according to the statement and proving its logical equivalence to the corresponding formula from (7, 8, 10-17). For example, to prove that (7) ≡ (28) we shall note that if (40) are applied to (28) it is transformed as follows: $(I = \{T^b, T^i, T^e, f\}) \equiv (I = \{\{t^b\}, T^i, \{t^e\}, 1\}) \equiv (I = \{t^b, T^i, t^e, 1\})$. $f \equiv 1$ means that t^b and t^e always belong to I. From the other hand, (7) has the equivalent meaning – t^b and t^e always belong to I. Hence, formula (28) is logically equivalent to (7) under the conditions of (40). Logical equivalence in the rest is proved analogously.

Duration of a fuzzy time interval is denoted similarly to that of a crisp interval. Duration $|I|$ of a fuzzy time interval $I = [t^b, t^e]$ is the quantity of different unit time intervals[11] which lie within I in the sense of (38). As for crisp intervals, it is easy to prove that the duration of a fuzzy interval can be computed as:

$$|I| = t^e - t^b. \tag{41}$$

5.2 Infinite Fuzzy Time Intervals

An extension of the theory of fuzzy time intervals is the account for their possible infinity at the beginning, at the end, or at both ends. We shall say that a fuzzy time interval ${}^{\infty}I = \{T^b, T^i, T^e, f\}$ is infinite at the beginning iff:

$$T^b = \varnothing, \forall t_j \in T^i \exists t_i : (t_i \in T^i) \wedge Before(t_i, t_j) \tag{42}$$

We shall say that a fuzzy time interval $I^{\infty} = \{T^b, T^i, T^e, f\}$ is infinite at the end iff:

$$T^e = \varnothing, \forall t_j \in T^i \exists t_i : (t_i \in T^i) \wedge After(t_i, t_j) \tag{43}$$

In frame of our model of time there exists only one interval ${}^{\infty}I^{\infty}$ which is infinite at both ends – the Time Line. For ${}^{\infty}I^{\infty}$ it holds true that:

$$T^b = \varnothing, T^e = \varnothing, \forall t_j \in T^i \exists t_i, t_k : (t_i, t_k \in T^i) \wedge Before(t_i, t_j) \wedge After(t_k, t_j) \tag{44}$$

[11] Please note that unit time intervals are crisp intervals and their difference is denoted as $Different(I_1, I_2) \equiv \neg Same(I_1, I_2)$.

6 Time Interval Sets: Phases and Periods

An interval or a fuzzy interval may contain sub-intervals (13, 35). Sub-intervals of an interval are often associated with the phases of an event or of a process. Examples of such sub-intervals are: (i) A working week contains a working days sub-interval and a weekend. Though it seems to be crisp at a first glance, some people have five working days, but the others have six, or prefer to work on Sundays and rest on Mondays. (ii) A year contains four seasons. Though one may argue that seasons as the phases of a year and time sub-intervals of seasons within the time interval of a year are crisply defined by a calendar. However, such a crisp definition is sometimes too rigid. For example, if spring is the season when flowers blossom and grass is green, then we can not guarantee that flowers and green grass can be found in the Northern Europe on March the 1-st. From the other hand, if a calendar winter is warm enough, this may well happen in February. More details are given in [11]. (iii) A quarter contains three months. This is an example of a crisp interval containing crisp sub-intervals, at least in academic, industrial, or business settings.

A reason to consider smaller interval chunks is that an interval is associated to a phenomenon which occurs in real World and characterizes how this phenomenon is manifested in time. A phenomenon may manifest itself differently in its different phases. Phases have different sub-intervals associated to them. An evident example is: we have low temperatures and snow (so far, somewhere) in winters, but thunderstorms and high temperatures in summers. One of the examples topical to the domain of engineering design is as follows.

When a designer is assigned to perform a particular design activity he first takes his time to prepare himself for it. This preparation phase is called a Ramp-up Phase and may be spent to learn: how to use a new version of a design tool; what are the specific features of the particular design; which design techniques may be most productive in this particular case, etc. Sometimes, when designers follow trial and test approach Ramp-up and Execution time sub-intervals of these phases may overlap. They of course are better represented by fuzzy intervals than by crisp intervals.

Assume that $S = \{s_1, ..., s_i, s_{i+1}, ...s_N\}$ is a finite set of fuzzy time sub-intervals of a fuzzy time interval *I*, then

$$\forall i = 1..N-1, j = i+1, (LikelyMeets(s_i, s_j) \vee Meets(s_i, s_j)) \tag{45}$$

and therefore

$$Same(\bigcup_{i=1}^{N} s_{1..N}, I)\,. \tag{46}$$

Please note that (36, 37) hold true for finite and infinite fuzzy time intervals.

Periods (π) are fuzzy time intervals associated with regular or repeating events [11]. One may consider finite and infinite sets of periods. A finite set of periods is associated with an event having a finite number of repetitions. For example, a particular person may have not more than two periods of presidency in many countries. On the contrary, the set of year periods is infinite because the Earth will hopefully continue its revolutions around the Sun forever.

Assume $\Pi = \{\pi_1 ... \pi_i, \pi_{i+1}, ..., \pi_N\}$ is the finite set of periods with cardinality N[12]. A total order may be set among the periods in the set:

$$\forall i = 1..N-1, j = i+1..N(Before(\pi_i, \pi_j) \vee Meets(\pi_i, \pi_j)) \tag{47}$$

Let us denote a set of periods infinite at the beginning as ${}^{\infty}\Pi = \{...\pi_i, \pi_{i+1}, ..., \pi_n\}$, a set of periods infinite at the end as $\Pi^{\infty} = \{\pi_1 ... \pi_i, \pi_{i+1}, ...\}$, a set of periods infinite at both ends as ${}^{\infty}\Pi^{\infty} = \{...\pi_i, \pi_{i+1}, ...\}$. Then the total ordering on these sets may be set as follows:

$${}^{\infty}\Pi^{\infty} : \forall i \exists j : (Before(\pi_i, \pi_j) \vee Meets(\pi_i, \pi_j)) \tag{48}$$

$$\Pi^{\infty} : \forall i \exists j > 1 : (Before(\pi_i, \pi_j) \vee Meets(\pi_i, \pi_j)) \tag{49}$$

$${}^{\infty}\Pi : \forall i < n \exists j : (Before(\pi_i, \pi_j) \vee Meets(\pi_i, \pi_j)). \tag{50}$$

7 Related Work

There is a vast amount of published results on representing temporal information in many domains and using a broad diversity of general theories of time [13]. However, most of them are limited to crisp settings and do not deal with any form of imprecisions. Similarly to [14], we take into account such imprecisions as uncertainty, vagueness and subjectivity. Research on uncertain temporal knowledge usually studies events and associated time intervals with precise boundaries, but state that our knowledge about them is uncertain or even unknown. This category of imprecision is often modeled using possibility theory [15]. Probabilistic or possibilistic constraints on the beginnings and endings of time intervals generalizing Allen's interval temporal algebra are also introduced in [16, 17]. Vagueness, in difference to uncertainty, means that it is impossible even to consider that an event or an action may have precise beginning or ending time instant – they have their "core" and transition periods at their ends. In this settings using probabilistic or possibilistic approaches for reasoning about the instant boundaries of an event is irrelevant. They should be substituted by time intervals as done in our work similarly to [14, 18, 19]. Speaking about subjectivity, most known approaches in temporal reasoning model subjective temporal knowledge using probability distributions expressed as fuzzy sets. However, that might be appropriate for representing uncertain subjective knowledge, but does not cope with representing subjective knowledge of vague transition periods. To the best of our knowledge, only two approaches [14, 18] are capable of representing subjective temporal knowledge relevantly to the requirements of PSI. The model presented in this paper is very close in its spirit to that of [14] and [18]. However, there are some differences. Both [14] and [18] use continuous temporal model – we use a discrete one. Having in mind that reasoning on fuzzy extension of Allen's temporal logic is an NP-complete problem [17], we may expect that our approach is more computationally efficient and still relevant for our project. Our approach, similarly to [18], is hybrid in

[12] N is the number of the repetitions of the phenomenon.

the sense that both use time points and time intervals, but [14] bases on time intervals only. We consider that having time instants is important because it allows building three nested models (minimal, crisp, and full fuzzy) having more expressive ones as proper extensions of less expressive ones. For PSI it is valuable because allows increasing model expressiveness in a natural way. In difference to [14, 18] the model presented in this paper is more expressive because it is capable of representing sets of time intervals. This allows us reasoning about singular, repeating, and regular events [11] as well as about iterative actions and their dependencies [10]. Finally, the technique we use in implementing our ontology of time and integrating it in PSI Suite of Ontologies [8] is similar to that of [14].

8 Concluding Remarks

We believe that enriching such a simplistic model of time as a linear and discrete one by fuzziness of time intervals may bring substantial expressivity benefit. Accounting for the fuzziness of beginnings, meetings, durations, overlaps, and endings of concurrent or sequential actions in time may facilitate making plans and schedules of business processes more flexible and better reflecting the reality. Effective reasoning on actions in processes may make simulations of these processes more grounded. Assessments and predictions of performance based on the results of these simulations may therefore give a better match to what happens or may happen in real business settings.

The paper presented a fuzzy extension of a discrete model of time intervals. In its crisp part the model follows Allen's interval temporal calculus [7]. In its fuzzy extension our model is close to [14, 18], but extends them by providing means for modeling singular, repeating, and regular events [11] as well as about iterative actions and their dependencies [10]. These features are very important for our application domain. Presented model is implemented in three ontologies having different expressive power: Time Minimal, Time Crisp, and Time Full. Time Minimal is already the part of the Core of PSI Suite of Ontologies [8] and is used in the current version of PSI software prototype. Our plans for future work anticipate gradual integration of Time Crisp and Time Full ontologies in software development.

Acknowledgements

This research is performed and funded in frame of PSI project of Cadence Design Systems, GmbH. The authors would also like to acknowledge the comments of the anonymous reviewers which substantially helped in the improvement of the paper.

References

1. Ermolayev, V., Matzke, W.-E.: Towards Industrial Strength Business Performance Management. In: Mařík, V., Vyatkin, V., Colombo, A.W. (eds.) HoloMAS 2007. LNCS (LNAI), vol. 4659, pp. 387–400. Springer, Heidelberg (2007)
2. Aristotle: Physics. Translated by R. P. Hardie and R. K. Gaye, eBooks@Adelaide (2007)

3. Etzion, O., Jajodia, S., Sripada, S.: Temporal Databases: Research and Practice. Springer, New York (1998)
4. Vila, L.: A survey on temporal reasoning in artificial intelligence. Artificial Intelligence Communications 7, 4–28 (1994)
5. A UML Profile for MARTE, Beta 1. OMG Adopted Specification, `http://www.omg.org/docs/ptc/07-08-04.pdf`
6. Masolo, C., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A.: WonderWeb Deliverable D18. Ontology Library (final). Technical Report. Laboratory for Applied Ontology, ISTC-CNR, Trento, Italy (2003)
7. Allen, J., Ferguson, G.: Actions and Events in Interval Temporal Logic. J. of Logic and Computation 4(5), 531–579 (1994)
8. Ermolayev, V., Jentzsch, E., Keberle, N., Sohnius, R.: Performance Simulation Initiative. The Suite of Ontologies v.2.1. Reference Specification. Technical Report PSI-ONTO-TR-2007-1, VCAD EMEA Cadence Design Systems, GmbH (2007)
9. Niles, I., Pease, A.: Towards a Standard Upper Ontology. In: Guarino, N., Smith, B., Welty, C. (eds.) Proc. Int. Conf. on Formal Ontologies in Information Systems (FOIS 2001), pp. 2–9. ACM Press, New York (2001)
10. Ermolayev, V., Jentzsch, E., Matzke, W.-E., Pěchouček, M., Sohnius, R.: Performance Simulation Initiative. Theoretical Framework v.2.0. Technical Report PSI-T-TR-2007-1, VCAD EMEA Cadence Design Systems GmbH, Feldkirchen, Germany (2007)
11. Ermolayev, V., Jentzsch, E., Keberle, N., Sohnius, R.: Performance Simulation Initiative. Theory of Time, Events, and Happenings. Technical Report PSI-T-TR-2007-2, VCAD EMEA Cadence Design Systems GmbH, Feldkirchen, Germany (2007)
12. Dubois, D., Prade, H.: Fundamentals of Fuzzy Sets. The Handbooks of Fuzzy Sets series, vol. 7. Springer, Heidelberg (2000)
13. Schreiber, F.A.: Is Time a Real Time? An Overview of Time Ontology in Informatics. In: Halang, W.A., Stoyenko, A.D. (eds.) Real Time Computing. Springer Verlag NATO ASI, vol. F 127, pp. 283–307. Springer, Berlin (1994)
14. Nagypál, G., Motik, B.: A Fuzzy Model for Representing Uncertain, Subjective and Vague Temporal Knowledge in Ontologies. In: Meersman, R., Tari, Z., Schmidt, D.C. (eds.) CoopIS 2003, DOA 2003, and ODBASE 2003. LNCS, vol. 2888, pp. 906–923. Springer, Heidelberg (2003)
15. Godo, L., Vila, L.: Possibilistic Temporal Reasoning Based on Fuzzy Temporal Constraints. In: 14th Int. Joint Conf. on Artificial Intelligence (IJCAI 1995), vol. 2, pp. 1916–1923. Morgan Kaufmann Publishers, Burlington (1995)
16. Dubois, D., Hadjali, A., Prade, H.: Fuzziness and Uncertainty in Temporal Reasoning. J. Universal Computer Science 9(9), 1168–1194 (2003)
17. Badaloni, S., Giacomin, M.: The Algebra IAfuz: a Framework for Qualitative Fuzzy Temporal Reasoning. Artificial Intelligence 170(10), 872–908 (2006)
18. Ohlbach, H.J.: Relations between Fuzzy Time Intervals. In: Combi, C., Ligozad, G. (eds.) 11th Int. Symposium on Temporal Representation and Reasoning (TIME 2004), Washington DC, USA, pp. 44–51. IEEE Computer Society Press, Los Alamitos (2004)
19. Schockaert, S., De Cock, M., Kerre, E.E.: Imprecise Temporal Interval relations. In: Bloch, I., Petrosino, A., Tettamanzi, A.G.B. (eds.) WILF 2005. LNCS (LNAI), vol. 3849, pp. 108–113. Springer, Heidelberg (2006)

Conceptual Design of a Method to Support IS Security Investment Decisions

Heinz Lothar Grob, Gereon Strauch, and Christian Buddendick

European Research Center for Information Systems, University of Muenster
Leonardo-Campus 3, 48149 Muenster, Germany
{grob,gereon.strauch,christian.buddendick}@ercis.de

Abstract. Information Systems are part and parcel of critical infrastructures. In order to safeguard compliance of information systems private enterprises and governmental organizations can implement a large variety of distinct measures, ranging from technical measures (e.g. the employment of a firewall) to organizational measures (e.g. the implementation of a security awareness management). The realization of such measures requires investments with an uncertain prospective return that can hardly be determined. An appropriate method for the profitability assessment of alternative IS security measures has not been developed so far. With this article we propose a conceptual design for a method that enables the determination of the success of alternative security investments on the basis of a process-oriented perspective. Within a design science approach we combine established artifacts of the field of IS security management with those of the field of process management and controlling. On that base we develop a concept that allows decision-makers to prioritize the investments for dedicated IS security measures.

Keywords: Security Metrics, ROSI, IT-Risk management, IT-Compliance.

1 Introduction

The necessity of information systems security measures is indisputable in the face of the increasing importance of information systems (IS) and the steady rise in security incidents. An evidence thereof is the in the past above-average increase of IS security budgets compared to overall IS budgets [1]. The planning and realisation of IS security measures typically hardly took into consideration questions of economic profitability [1]. In the meantime though, more recent works emphasis the elementary imperative of profitability analyses [2]—although available findings of this field of research are frequently characterised as being vague, unusable or without reference to concrete recommendations for a course of action [3-5]. The measurement of the profitability of IS security measures poses challenges similar to the field of general IT investments. Although the IT productivity paradoxon has been considered as outdated for years [6; 7], recent studies indicate continuing scepticism of executives that IT investments can provide for an adequate value proposition [8; 9]. The problem is additionally aggravated owing to the circumstance that the effects of successful measures are exclusively indirect, since they contribute to the reduction of future risks [10-12]. The statement: "IS security functions have been valuable whenever nothing has happened." [2] underlines this observation.

Moreover, the interdependencies among various IS security measures are to be considered, since oftentimes only the success of bundles of measures can be determined [13]. A possible approach to the necessary quantification hence is required to contain a differentiated composition parameter in order to give consideration to the inherent complexity. To complicate matters further, due to the technically dominated domain of security management and issues of profitability analyses of technical measures, reciprocal expert-layman relations exist, which require transparent accounts [14]. Particularly concerning a summarization toward a decision recommendation no elementary economic and scientific requirements may be disregarded.

In order to conceptualize a method for the decision support for IS security investments, these special challenges need to be considered. The chosen research approach can be characterized as design-oriented, where a conceptual-deductive research method has been applied [15]. An analysis of the state-of-the art in the field of IS security management will illustrate that the suggested methods are either theoretically inexact or practically improperly. Departing from the existing requirements, this work suggests a method which allows the profitability assessment of alternative IS security investments. At this it is taken into account that the implications of IT investments first of all can be observed on the process level [16]. The method both facilitate the calculation of payments and disbursements for all processes which are affected by IS security measures and aggregates those in form of a multi-period investment appraisal. In order to take uncertainty into account, the method incorporates aspects of Monte-Carlo simulation which enables to indicate uncertain parameters of the IS security investment decision by means of supposed probability distributions. The article concludes with a brief summary and an outlook on future research oportunities.

2 State-of-the-Art of Decision Support for IS Security Investments

In the following, different approaches identified within the literature are examined regarding the requirements explicated above. Since investments in IS security commonly do not exhibit an immediate return but rather contribute to averting undesired incidents and hence avoid loss, an established approach is to determine the reduction of expected loss. One of the earliest and most importance approaches hereunto is the Guideline for Automatic Data Processing Risk Analysis published by the National Bureau of Standards in 1979 [17]. The key performance indicator Annual Loss Expectancy (ALE) is composed of the sum of the expected annual loss which relate to security holes [18]. The ALE-concept has been increasingly employed in America in the 1980s in, primarily due to its promotion by the National Institute of Standards and Technology (NIST). With the phase-out of these projects however, the concept nearly entirely fell into oblivion [19]. Prime reasons for the failure of the ALE concept in practice are according to Soo Hoo its high level of detail and the high complexity of the model, the strong dependence on a complete data basis and the inherent assumption that all factors are deterministic and known in advance [13]. Besides, a lack of necessary empirical data has frequently been criticised [18].

After the ALE concept proved impracticable, further effort was put into the development of a stable approach to the assessment of the profitability of security investments. The results were the approaches of the second generation [13]. These are characterized by a reduced complexity in comparison to the ALE concept. Exemplarily the integrated business-risk management framework, solely value-oriented methodologies, scenario analyses and best practice approaches are to be mentioned. Concerning the integrated business-risk management framework, a non-technical model, the IS security risks are treated in analogy to common business risks (operative, financial, etc.) and hence inappropriately with regard to the specifics of the underlying problem. Value-oriented methodologies merely focus on the possible amount of loss of an incident, suppressing entry probabilities and in this vein not allowing for a complete quantification of the risk. Concerning scenario analyses the focus is on one or few threatening scenarios and hence do not allow for the assessment of extensive investments. Best practice approaches, taking the form of standardised recommendations for action do not allow for a consideration of individual specifications.

Since these approaches do not quantify the utility of an IS security investment and key figures are not taken into consideration, they are not appropriate for the purpose of decision support [20]. Many of the subsequently developed, so-called approaches of the third generation [13], e.g. by Gordon/Loeb, Cavusoglu/Mishra/Raghunathan or Cremoni/Martini [21-23], rather focus on the explanation of economic cause-effect relationships in the context of IS security measures. In this they provide a significant explanatory contribution, although due to the rather theoretic nature of the models which abstract from essential aspects or where required parameters cannot be determined with justifiable effort [13], these are not apt for decision support. Consequently the demand for an adequate and practicable method for assessing profitability remains acute [19].

However, a figure for profitability analyses which has achieved considerable diffusion in practice is the Return on Security Investment (ROSI), which is intended to assess the profitability of a security measure. The approach has caused increased interest for two reasons: First, the ROSI is said to provide a solid and recognized basis for investment decisions owing to a professed analogy to the Return on Investment (ROI) [24; 4]. Second, the apparently precise instruction and simplicity of the figure are emphasised [19] So far, the literature does not provide a consistent definition of the ROSI: besides the usage of both different variable denominations and different dependent parameters, the ROSI is described as an absolute value [25; 5] on the one hand and as quotient [24; 26] on the other hand [19]. Commonly however, the representation as an absolute value is favoured:

$$R - S + T = ALE \Leftrightarrow R - ALE = S - T = ROSI^{1} \tag{1}$$

Elsewhere, the ROSI is found in quotient notation [24; 26]:

$$ROSI = \frac{R - ALE}{T} = \frac{S - T}{T} \tag{2}$$

[1] ALE (Annual Loss Expectancy) denotes the annually expected loss according to the ALE concept; R (Revocery Cost) denotes the annual cost of recovering loss resulting from security incidents; S (Savings) denotes the sum of the annual measures to avoid losses; T (Tool Cost) denotes the cost for security measures, commonly calculated in accordance to the total cost of ownership concept.

Both variants of the ROSI concept in common combine the ALE concept with the classic TCO concept for IS security investments. However, therewith also the critical aspects of these concepts are inherited. On the one hand side these are the points of critique of the ALE concept as discussed above. On the other hand side, the principal drawbacks of the classic TCO method still apply, e.g. this approach does not consider the interest and tax payments required for a multi-period analysis [27]. Furthermore it is generally only focused on the direct variation of the expected loss and the pure costs of the measures—indirect payments are not taken into account.

Soo Hoo pursues a fundamental limitation of the usual representation of the ROSI approach, by considering investment decisions for which different security guidelines as a bundle of several security measures are compared regarding their “net benefit” [13]. By means of the “net benefit”, a quantitative assessment of a security investment in consideration of additional costs, benefits and the altered ALE related to the investment shall be achieved. All approaches explicated so far, as well as most alternative approaches take a single period perspective [23; 24; 13]. Thereby either average values are applied or the problematic assumption of parameters being constant is taken as a basis. Consequently it is impossible, to adequately represent the complexity of the decision situation by the means of such approaches. In case a multi-period view is taken nonetheless, this partly occurs without consideration of interest [5]. The assumption of such however pertains to one of the essential requirements that are posed with respect to instruments for the assessment of IT security measures [28].

Other approaches of the ‘third generation’ sure revert to classic methods of investment appraisal for a summarization of the data towards a decision recommendation. These approaches comprise for example the capitalized value method or the method of the internal interest rate as well as the net present value [29; 28; 30-32; 18]. There are several problems related to these approaches. From a theoretical point of view some approaches (e.g. internal interest rate) do not take different kind of financing options into account. Especially in the field of IT security these options are often combined, e.g. leasing of IT or outsourcing options are common in this field. Furthermore this approaches don not support period specific tax-rates, which are common in practice. Another important lack is the negligence of uncertainty aspects. Therefore it can be states that these approaches obvious may lead to wrong decisions. From a practical point of view the summarization of all cash-flows within different periods to one top indicator is often hard to understand for technical staff. There is no transparency of cash-flows, financing options and financial implications of tax payments in each period. Even more problematic in this context is the missing link to the derivation of cash-flows from processes. In the existing approaches payments are just predicted on experience and not on functions and related risks. Methods proposed to be applied comprise benefit value analysis, analytical hierarchy process, etc. [33; 29; 34; 35]. These methods summarize relevant data in a partially not transparent and methodically equivocal manner. The fundamental problem in this application context does not relate to the calculation of a financial value, but to the recording of all relevant cash flows, whereat, as already elaborated—particularly in this context of different domain knowledge—transparency has to prevail on every step of the decision making and aggregation.

Above all this implies the exhaustive recording of all relevant values in consideration of all potential direct, indirect and derivative values, which are particularly important in this context, such as tax or relevant own capital backing. In this vein e.g. the with regard to information systems essentially important decision of acquisition or leasing, make or buy respectively outsourcing without interest and tax is not to be assessed, since time effects and tax issues thereby play a significant role. The critique on such a proceeding, especially in the field of IS controlling, is nothing new [36]. A method is introduced in the following, which both facilitates an extensive and integrated recording of all relevant data and allows for a complexity adequate summarization towards a recommendation within the illustrated decision context.

3 Conceptual Design of a Method for Decision Support

3.1 Processes Models as Foundation of Decision Support

Due to the circumstance that the impacts of IS security investment measures, like all IT investments, firstly can be observed on the processes [16], the latter should—in analogy to business management—represent the focal point of security management [37-40]. Process orientation in the context of security management is most notably supported by the notion that the process view, as illustrated in Fig. 1, enables the integration of varied relevant perspectives of security management—such as organisational structure, room layouts, authorisation concepts, etc.—into the process models. Moreover, based on the processes, an analysis of the risk potential of incidents on value adding activities allows for the determination of the potential losses. For this purpose, the effects of the risks on the security quality of the processes with regard to confidentiality, availability and integrity, as well as possible counter measures, are to be mapped. Additionally, measures and damages, etc. can be stochastically incorporated into an appropriate combination of fault tree and event tree analysis—while the fault tree analysis maps loss occurrences, their respective impacts on the processes are modelled by means of an event tree analysis [41]. This proceeding is known from different contexts, e.g. within the scope of the failure modes and effects analysis (FMEA) or the hazard analysis critical control point method (HACCP) [42-44].In the context of business process management process models were used to analyze and evaluate risks and their impact on business processes. For that purpose Brabänder/Ochs, zur Mühlen/Rosemann and Rieke have developed different concepts to integrate risks into process models based on the event driven process chain (EPC) [45-47]. As exemplary shown in Fig. 1, it is possible to integrate IS-risks within process models based on the approach of zur Mühlen/Rosemann. They use so called *risk structure models* for specialisation and composition of risks and *risk state models* to show the consequences of different interdependent risks. The effects of the risks on the security goals (confidentiality, availability and integrity) as special security goals of the process can be mapped with risk goal models. The risk state model may be used– as exemplary shown in Fig. 1 – according to fault and event trees for modelling the interdependences between processes, information systems and related risks. For this purpose we suggest to combine risk structure and state models by adding probabilities and specialising different consequences resulting from common risks. We also suggest not only connecting risks to processes but also to resources like systems or users that belong to processes. Due to space restrictions however, the proceeding cannot be elaborated on in greater detail at this point, especially as there is further research demand with respect to the explicit design.

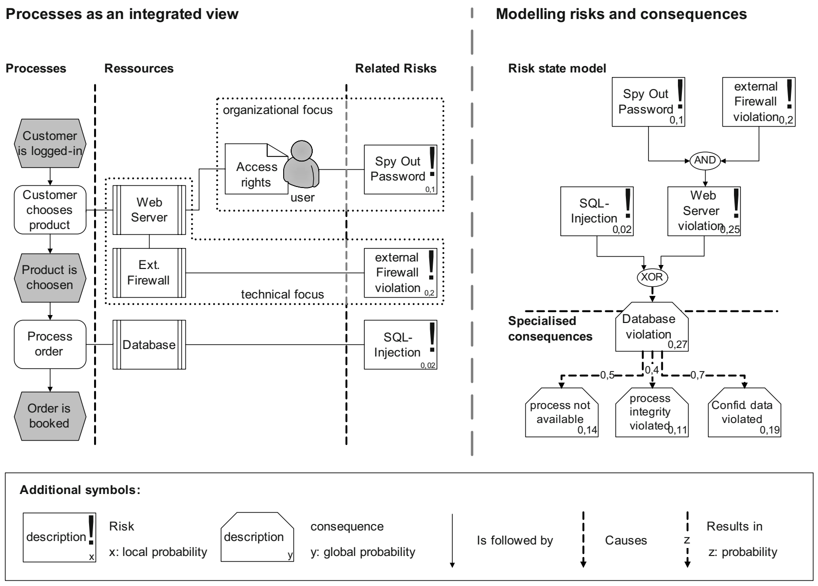


Fig. 1. EPC Process models and the risk extensions according to zur Mühlen and Rosemann as an instrument to integrate *processes, resources, information systems* and *risks*

Besides, the process models via the respective contributions for achievement and resource strain also allow for a mapping of the financial implications of the measures. This advance analogously takes place in the field of activity-based costing, although due to the investment character, here payments and disbursements instead of costs and activities as periodical values should be in the focus. Also it has already been described in other contexts in the form of a process-oriented investment appraisal [48-50]. The decision situation of an IS security investment is determined, in addition to direct payments and disbursements such as the alteration of the expected loss, by the need to consider all indirect payments which are caused by making the investment [51]. By means of security measures e.g. disbursements in the field of first level support can be reduced owing to the implementation of certificate based employee identification [24]. Besides, additional revenues should be considered also, for example resulting from an increase of prospects acquisition due to a visibly higher security level (e.g. an SSL encryption for an online shop). Similarly, different process designs cause different cash flows. In this vein, customers potentially can be detained from a purchase if ordering processes are cumbersomely. This notion is facilitated by the explicated proceeding in the sense that in addition to varying packages of measures, advanced implications such as changes in productivity may be analyzed bases on different process designs.

Based on the presented findings, a procedure model shall be introduced in the following, by means of which investment alternatives for IS security measures can be assessed. Rather than considering single measures or investments in an isolated manner, investment alternatives will be considered as different bundles of measures [13]. Consequently there are at least two alternatives that may be compared: the case with a security investment (with- scenario) and the case without (without-scenario). For these competing investments the respective complete cash flows are determined, i.e. also indirect consequences of the measures such as changes in productivity are incorporated, in order to subsequently aggregate the alternatives to target values (whilst deriving derivative payments such as interest and tax) which allow for a comparison. An approach shall be presented in the following, which enables to aggregate the extracted data for decision support purposes in such a manner, that the required transparency is not disregarded.

3.2 Key Figures for Decision Support

The fundamental concern of a further aggregation towards target values, which serves as decision support in the choice of a relevant alternative, should be that transparency and flexibility for a complexity adequate mapping of decision situations is guaranteed. For this purpose, the visualization of financial implications (VOFI) as an instrument of the dynamic investment appraisal is particularly promising [52]. This method is utilized as opposed to other approaches, because in addition to a consideration of interest and tax,

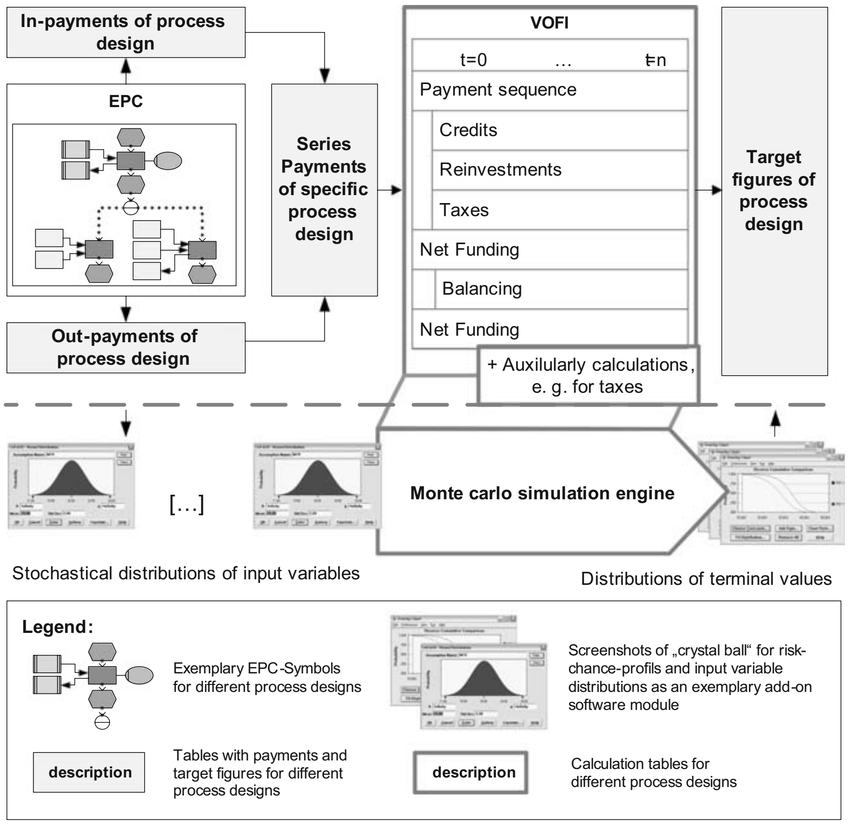

In-payments of process design
EPC
Out-payments of process design
Series Payments of specific process design
VOFI
t=0 … t=n
Payment sequence
Credits
Reinvestments
Taxes
Net Funding
Balancing
Net Funding
+ Auxilularly calculations, e. g. for taxes
Target figures of process design
[…]
Monte carlo simulation engine
Stochastical distributions of input variables
Distributions of terminal values
Legend:
Exemplary EPC-Symbols for different process designs
Screenshots of „crystal ball" for risk-chance-profils and input variable distributions as an exemplary add-on software module
description
Tables with payments and target figures for different process designs
description
Calculation tables for different process designs

Fig. 2. Using *EPC* and *VOFI* to evaluate different process designs and extension by *stochastic variables* (risks and chances analysis)

particularly the table-oriented layout accommodates the necessary transparency and expandability of a communication among the participating actors of different domains with respect to the risk analysis and assessment. Thus a mapping of varying relevant specifics of the individual alternatives becomes possible at small expenses.

Starting point of a VOFI is the payments sequence of the investment alternative. Drawing on this, varying loan and investment conditions with different interest rates, durations and amortisation modes can be considered for the calculation of a financing balance that is settled for every individual period. Within auxiliary calculations it can be paid consideration to the design of the legal form of a company that manifests e.g. in different disbursement and tax models. In this vein it is also possible to map e.g. the consequences of outsourcing decisions. For each period, the current portfolio of different forms of investment and financing are recorded in the bottom part and consolidated in a balance. Additionally this enables to represent different equity securitisation or allocation concepts, which are of particular interest in the context of risk management [28]. Within Fig. 2 the basic procedure of a combining EPC and VOFI for evaluating IS-security investments is illustrated.

The actual balance at the end of the last period represents the accumulated value of the investment, which coincides with the classic methods if assuming the restrictive premises [52]. This is to be contrasted to the accumulated values of the VOFIs of all beyond investment alternatives. If only one investment alternative is available, the accumulated value of the VOFI of the investment (with- scenario) is to be opposed to the accumulated value of the VOFI for the maintaining of the status quo (without- scenario). For this purpose a VOFI is compiled that represents the expected loss without influence of any measures as well as the alternate disposition of equity in the form of interest yield based on opportunity cost rate. The additional accumulated value of the investment results from the difference between the accumulated value of the investment and the accumulated value of the opportunity (the “second best” solution) [52]. In case of a positive additional accumulated value it appears sensible to make the investment. An example of use in the context of IS security investment has been proposed by vom Brocke et al. [53].

Complete finance plans can be realised by means of standard spread sheet software and hence are transparent, due to the explication of all original and derivative payments, for actors without economic education also. Furthermore, sequences of payments with regard to their composition can easily be programmed by heterogeneous user groups drawing on conventional office solutions. The process modelling and the calculations can be implemented with standard office software. Consequently adaptations are facile to make, e.g. by means of a conventional extension for Monte Carlo simulations it is possible to represent relevant stochastic values and to calculate a risks-chances profile of the investment [54]. Thus the uncertainty that is inherent to a risk consideration can easily and thoroughly be characterized. As illustrated within Fig. 2, the sequence of payments is connected with supposed probability distributions for this purpose. This can occur directly or via auxiliary calculations or result from initially executed process simulations. Founding on the simulation, distributions of the accumulated values can be determined, based on which such risks-chances profiles for the different alternatives can be established, that allow for a control of the disposition of the operative risks in the field of information systems in accordance with the desired risk disposition of the organization.

4 Conclusion

In the course of this paper, a method for the decision support of IS security investments has been introduced. The research follows a constructive research design, where a conceptual-deductive analysis was drawn upon as research method. The evaluation of the state-of-the-art in the field of IS security management has illustrated that existent approaches are either not practice oriented and hence not relevant for the practice or – as has been demonstrated with regard to the ROSI – lack the theoretic foundation and owing to an inadequate information summarization may lead to wrong decision recommendations. Since IS security investments primarily exhibit a direct impact on the organizational processes, the latter are in the focus of the suggested method. Starting from an integrated view on risks, security measures and benefits, payments and disbursements of all processes affected by a designated bundle of measures are determined. Owing to the long term nature of the investments, procedures of the dynamic investment appraisal are to be employed in order to represent the economic consequences of the decision alternatives. Additionally, the method provides for the possibility to consider uncertainty by applying principles of the Monte Carlo simulation. Therefore a complete decision support for the choice of alternative IS security measures can be guaranteed.

Future research should focus on the application of the method within corporate praxis. In this vein it is to be examined closely, to what extent a detailed modeling of single processes can be regarded as suitable in practice, or if an aggregation to primary processes should be effected. In addition it is to be explored which modeling languages prove to be particularly apt for the integrative representation of risks, security measures, benefits and costs of processes. Empirical studies may furthermore shed light on the distribution of the insecure parameters, such as e.g. the entry probability of a particular threat in the scenario of a particular bundle of IS security measures.

References

1. BSI: Kosten und Nutzen der IT-Sicherheit, Studie des BSI zur Technikfolgen-Abschätzung. SecuMedia, Ingelheim (2000)
2. Fettke, P.: State-of-the-Art des State-of-the-Art - Eine Untersuchung der Forschungsmethode “Review” innerhalb der Wirtschaftsinformatik. Wirtschaftsinformatik 48(4), 257–266 (2006)
3. Kütz, M.: IS Controlling für die Praxis. Konzeption und Methoden. dpunkt, Heidelberg (2005)
4. Peltier, T.R.: Information security risk analysis. Auerbach Publications, Boca Raton (2001)
5. Pohlmann, N.: Wie wirtschaftlich sind IT-Sicherheitsmaßnahmen? HMD 43(248), 26–34 (2006)
6. Brynjolfsson, E.: The productivity paradox of information technology. Communications of the ACM 36(12), 66–77 (1993)
7. Brynjolfsson, E., Hitt, L.: Paradox lost? Firm-level evidence on the returns to information systems spending. Management Science 42(4), 541–588 (1996)
8. Carr, N.: IT doesn’t matter. Harvard Business Review 81(5), 41–49 (2003)
9. Luftman, J.N.: Key issues for IT executives. MIS Quarterly Executive 3(2), 1–18 (2004)

10. McCumber, J.: Assessing and managing security risk in IT systems: A structured methodology. Auerbach Publications, Boca Raton (2005)
11. Rodewald, G.: Aligning information security investments with a firm's risk tolerance. In: Whitman, M.E. (ed.) 2nd annual conference on Information security curriculum development, pp. 139–141. ACM, Kennesaw, GA (2005)
12. Vossbein, R.: Datenschutz-Controlling – Den Wirtschaftsfaktor Datenschutz effizient planen, steuern und kontrollieren. SecuMedia, Ingelheim (2002)
13. Soo Hoo, K.J.: How much is enough? A risk management approach to computer security. Consortium for Research on Information Security and Policy (CRISP). Working Paper Stanford University, Stanford (2000)
14. Bromme, R., Jucks, R., Rambow, R.: Wissenskommunikation über Fächergrenzen: Ein Trainingsprogramm. Wirtschaftspsychologie 5(3), 94–102 (2003)
15. Wilde, T., Hess, T.: Forschungsmethoden der Wirtschaftsinformatik. Eine empirische Untersuchung. Wirtschaftsinformatik. 49(4), 280–287 (2007)
16. Tallon, P.: A Process-oriented Perspective on the Alignment of Information Technology and Business Strategy. Journal of Management Information Systems (JMIS) (forthcoming) (2008)
17. FIPS: Guideline for automatic data processing risk analysis. National Bureau of Standards, US Department of Commerce (1979)
18. Mercury, R.T.: Analysing security costs. Communications of the ACM 46(6), 15–18 (2003)
19. Nowey, T., Federrath, H., Klein, C., Plößl, K.: Ansätze zur Evaluierung von Sicherheitsinvestitionen. In: Federrath, H. (ed.) Sicherheit 2005. 2. Jahrestagung des GI-Fachbereichs Sicherheit. Lecture Notes in Informatics, vol. P-62, pp. 15–26. Köllen-Verlag, Bonn (2005)
20. Wang, A.J.A.: Information security models and metrics. In: Guimaraes, M. (ed.) Proceedings of the 43rd annual southeast regional conference, pp. 178–184. ACM, New York (2005)
21. Cavusoglu, H., Mishra, B., Raghunathan, P.: A model for evaluating IT security investments. Communications of the ACM 47(1), 87–92 (2004)
22. Cremonini, M., Martini, B.: Evaluating information security investments from attackers perspective: the return on attack.Fourth Workshop on the Economics of Information Security (WEIS 05), Boston (2005), http://infosecon.net/workshop/pdf/23.pdf
23. Gordon, L.A., Loeb, M.P.: The economics of information security investment. ACM Transactions on Information and System Security 5(4), 438–457 (2002)
24. Mayer, B.: Rosi - Return on Security Investment. Eine notwendige Rechnung (2006), http://www.it-daily.net
25. Berinato, S.: Finally, a real return on security spending (2002), http://www.cio.de/technik/806049/
26. Sonnenreich, W., Albanese, J., Stout, B.: Return on security investment (ROSI). A practical quantitative model. Journal of Research and Practice in Information Technology 38(1) (2006)
27. vom Brocke, J.: Service Portfolio Measurement (SPM), A Decision Support System for the Management of Service-Oriented Information Systems. In: Qiu, R. (ed.) Enterprise Service Computing from Concept to Deploymen, pp. 58–90. IGI Publishing, Hershey (2006)
28. Faisst, U., Prokein, O., Wegmann, N.: Ein Modell zur dynamischen Investitionsrechnung von IT-Sicherheitsmaßnahmen. Zeitschrift für Betriebswirtschaft 77(5), 511–538 (2007)

29. Butler, S.A.: Security attribute evaluation method: a cost-benefit approach. In: Tracz, W. (ed.) 24th International Conference on Software Engineering (ICESE2002), pp. 232–240. IEEE Press, Orlando (2002)
30. Gordon, L.A., Loeb, M.P.: Budgeting process for information security expenditures. Communications of the ACM 49(1), 121–125 (2005)
31. Gordon, L.A., Loeb, M.P.: Managing Cybersecurity Resources: A Cost-Benefit Analysis. McGraw-Hill, Maryland (2006)
32. Landoll, D.J.: The security risk assessment handbook: a complete guide for performing security risk assessments. Auerbach Publications, Boca Raton (2006)
33. Bodin, L.D., Gordon, L.A., Loeb, M.P.: Evaluating Information Security Investments Using the Analytic Hierarchy Process. Communications of the ACM 48(2), 79–83 (2005)
34. Conrad, J.R.: Analyzing the Risks of Information Security Investments with Monte Carlo Simulations. Fourth Workshop on the Economics of Information Security (WEIS 05), Boston (2005), `http://www.infosecon.net/workshop/pdf/13.pdf`
35. Longstaff, T.A., Chittister, C., Pethia, R., Haimes, Y.Y.: Are we forgetting the risks of information technology? IEEE Computer 33(12), 43–51 (2000)
36. Kaplan, R.P.: CIM-Investitionen sind keine Glaubensfragen. Harvard Manager 9(3), 78–85 (1986)
37. Jakoubi, S., Tjoa, S., Quirchmayr, G.: Rope: A Methodology for Enabling the Risk-Aware Modelling and Simulation of Business Processes. In: Österle, H., Schelp, J., Winter, R. (eds.) Proceedings of the Fifteenth European Conference on Information Systems (ECIS 2007), Universität St. Gallen, pp. 1596–1607 (2007)
38. Konrad, P.: Geschäftsprozeßorientierte Simulation der Informationssicherheit: Entwicklung und empirische Evaluation eines Systems zur Unterstützung des Sicherheitsmanagements. Dissertation, Köln, Josef Eul Verlag, Lohmar (1998)
39. Röhrig, S.: Using Process Models to Analyse IT Security Requirements. Dissertation, Zürich (2003)
40. Sitzberger, S., Nowey, T.: Lernen vom Business Engineering - Ansätze für ein systematisches, modellgestütztes Vorgehensmodell zum Sicherheitsmanagement. In: Lehner, F., Nösekabel, H., Kleinschmidt, P. (eds.) Multikonferenz Wirtschaftsinformatik 2006, Tagungsband 2, pp. 155–165. Gito, Berlin (2006)
41. Königs, H.-P.: IT-Risiko-Management mit System. Von den Grundlagen bis zur Realisierung – Ein praxisorientierter Leitfaden. Vieweg, Wiesbaden (2005)
42. Franke, W.D.: FMEA: Fehlermöglichkeits- und -einflussanalyse in der industriellen Praxis. Verl. Moderne Industrie, Landsberg/Lech (1989)
43. Pichardt, K.: Qualitätsmanagement Lebensmittel: vom Rohstoff bis zum Fertigprodukt. Springer, Heidelberg (1997)
44. Schneeweiss, W.: Die Fehlerbaum-Methode. Aus dem Themenkreis Zuverlässigkeits- und Sicherheits-Technik. LiLoLe-Verlag, Hagen (1999)
45. Brabander, E., Ochs, H.: Analyse und Gestaltung prozessorientierter Risikomanagementsysteme mit Ereignisgesteuerten Prozessketten. In: Nüttgens, M., Rump, F. (eds.) Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten (EPK 2002), pp. 17–35. Trier (2002)
46. zur Muehlen, M., Rosemann, M.: Integrating Risks in Business Process Models. In: Australasian Conference on Information Systems (ACIS 2005) Manly, Sydney (2005)
47. Rieke, T.: Prozessorientiertes Risikomanagement. Ein informationsmodellorientierter Ansatz. Dissertation, Wirtschaftswissenschaftliche Fakultät Westfälische Wilhelms-Universität Münster (2008)
48. Kesten, R., Schröder, H., Wozniak, A.: Konzept zur Nutzenbewertung von IT-Investitionen. Arbeitspapiere der Nordakademie Elmshorn Elmshorn (2006)

49. Küker, S., Haasis, H.-D.: Geschäftsprozeßmodellierung als Basis einer informationswirtschaftlichen Unterstützung für ein AQU-Management. In: Rautenstrauch, C., Schenk, M. (eds.) Umweltinformatik 99 - Umweltinformatik zwischen Theorie und Industrieanwendung, 13. Internationales Symposium "Informatik für den Umweltschutz", pp. 256–268. Metropolisverlag, Marburg (1999)
50. Müller, A., von Thienen, L., Schröder, H.: IT-Controlling: So messen Sie den Beitrag der Informationstechnologie zum Unternehmenserfolg. Arbeitspapiere der Nordakademie Elmshorn Elmshorn (2004)
51. Neubauer, T., Klemen, M., Biffl, S.: Business process-based valuation of IT-security. In: Sullivan, K. (ed.) Proceedings of the seventh international workshop on Economics-driven software engineering research, pp. 1–5. ACM Press, St. Louis (2005)
52. Grob, H.L.: Investitionsrechnung mit vollständigen Finanzplänen. Vahlen, München (1989)
53. vom Brocke, J., Grob, H. L., Buddendick, C. and Strauch, G.: Return on Security Investments. Towards a Methodological Foundation of Measurement Systems. Proceedings of the 13th Americas Conference on Information Systems (AMCIS 2007), Keystone (2007), forthcoming
54. Hertz, D.B.: Risk Analysis in Capital Investment. Harvard Business Review 42(1), 95–106 (1964)

A Query Language for Rational Tree Structures

Markus Kirchberg[1], Faizal Riaz-ud-Din[2], and Klaus-Dieter Schewe[3]

[1] Institute for Infocomm Research (I^2R), A*STAR, Singapore
MKirchberg@i2r.a-star.edu.sg
[2] Al Ain Men's College, Higher Colleges of Technology, Al Ain, United Arab Emirates
faizal.din@hct.ac.ae
[3] Information Science Research Centre, Palmerston North, New Zealand
k-d.schewe@xtra.co.nz

Abstract. Traditional query languages such as SQL have been highly successful in both the areas of academia and industry. One of the compelling reasons for this high level of success is the declarative and intuitive manner in which queries could be constructed using such query languages. Additionally, the generic nature of their update operations makes it much easier for the user to specify update queries. This works well for performing queries to retrieve and update data in databases based on the relational model. However, the demands for storing more complex data other than in the form of flat tables are increasing, as research continues into XML and object-oriented databases. Parallel to these demands is also the requirement for a more powerful query language that would enable the retrieval and update of these more complex databases in an intuitive and declarative manner. We introduce the idea of a higher-level intermediary language that allows the querying and update of rational-tree type data (as found in XML datastores and object-oriented databases). We present elements of a language that allows declarative query operations for both querying and update operations in a generic manner.

1 Introduction

One of the major strengths, and also one of the main reasons for the success of the relational data model is its query language. Whilst providing the ability to query and update data based on predefined underlying schemas, it is also intuitive and easy to use for both novice and advanced users. Arguably, the highest-impacting language thus far has been SQL [1].

However, there is now a greater demand to store more complex types of data than that allowed by the relational model. As a result, there is a resurgence in the search for object-oriented databases, and more so with XML databases. OQL [2] and, to some degree, SBQL [3] were introduced to allow the querying of complex data, namely object-oriented data; XQuery [4] was developed to allow the querying of complex data related to XML data. Consequently, we are currently witnessing more research, especially in the area of XML [5].

Yet, with the desire to have the ability to store more complex types of data, a requirement for querying the data also arises. This has resulted in the proposal of a number of languages for querying object-oriented databases, and has also resulted in languages being proposed for the querying of XML data. Whilst such languages have been proposed for the querying of these more complex data stores, they are still limited in what they can achieve, and are still lacking the ease-of-use as provided by relational query languages, such as SQL.

One area of research that has been undertaken for a while, is the area of defining complex data types, as allowed for in object-oriented databases and XML data stores, in a formal manner. One such proposal is that complex data structures such as these may be represented as rational trees, as described in [6].

In [6], a query algebra for rational trees is also proposed, which allows algebraic query operations to be performed over rational tree structures. This XML and object-oriented data are a representation of rational tree structures.

Our contribution proposes an intermediary query language that performs algebraic query operations over rational tree type structures. Object-oriented and XML query languages may be based upon this intermediary language that would be able to provide the ability to perform a number of different query operations on the underlying data. The lower-level operations shown in this paper are compatible with the operations defined for the run-time system introduced in [7,8]. Thus, it may be used in the context of distributed architectures.

The uniqueness of our language lies in its ability to perform query operations on data structures that have cycles in them, and also provide generic update operations which may be applied to any entity in the tree. We use the psi-terms that have been introduced in [9] to allow cyclic structures to be represented in the query language. In Section 2, we provide a quick overview of rational trees together with an example of a rational-tree schema. Section 3 provides examples of query operations, whilst Section 4 details how the examples would be translated into lower-level operations. Finally, Section 5 concludes the paper.

2 Rational Trees

A rational tree is an infinite tree that may be represented in a finite way through the use of cyclic references. The fact that a data structure may be defined with cycles implies the "infiniteness" of the values, but allowing the structure itself to use such cycles in a recursive manner makes it finite.

As an example, we could consider the case of a type Student, that comprises FirstName, LastName, and CourseTaken. FirstName and LastName are primitive types (strings in this case), whilst the CourseTaken component refers to a value of type Course. The type Course is comprised of a CourseName and a StudentRep, which refers to a value of type Student, representing the student who is the student representative for the course. If we then have as values for these two types, the values "John" as the FirstName of a student, and "Doe" as the LastName, and a course with "Programming 101" as the CourseName, where John Doe is also the class representative, we obtain a rational tree as follows: i_1 refers to the object-identifier for the Student object, which has the values John and Doe, and a reference to a Course object. The Course object, in turn, has a value Programming 101, and has a reference to a Student object, specifying the student representative for the course. The reference to the Student object, in this case, creates a cyclic reference between these two objects. By continuously following the references in this cycle, we get an “infinite tree” of cycles. However, this “infiniteness” is represented in a finite manner.

In [6], a query algebra has been proposed as a method of performing query on rational tree structures as described above. As a summary, we will consider the following query algebra operations for our query language: projection, selection, renaming, join, union, difference, nesting, and unnesting.

It has been argued that, in an object-oriented database environment, the concept of the object-identifier is something that is essential [10]. However, conversely, it has also been argued that the object-identifier is something that should remain internal to the database and should not be exposed to the user of the database. The concept of the object-identifier, in some ways is seen to be a foreign-concept when it comes to the object-oriented model, when considering its “real-world” applicability. That is, in the real world, objects to not have identifiers attached to them (in most cases).

So to consider that the majority of object-oriented query languages allow such identifiers to be used by users is considered far from ideal. It has been proposed in [11] and [12] that the user should ideally use the values of the object to uniquely identify them in query languages, and not through the use of object-identifiers.

It follows from this that in designing a query language that hides object-identifiers from the user, and only allows them to specify object values, one would also have to consider how object references would be specified in the language, and would also have to consider how the querying and update constructs would need to be modified to allow cyclic-references in the data to be resolved in an easy manner.

An additional consequence of this is also the design of generic update operations for each type defined in the schema. That is, all types have an associated insert, delete and update operation. This imposes a constraint that when identifying an object, all the values of its attributes and the attributes of its references must be specified. It also means that during the insertion of a new object, for example, all the values of the new object, plus the values of associated referenced objects must be specified.

Example 1. We introduce an object-oriented class schema (without methods to facilitate simplicity) that has cycles, as shown below. The schema outlines a model of a fictional town where there are relationships between people of different professions. There are two cyclic relationships in the model.

```
Shopkeeper (LastName:String, doctor:Doctor)
Doctor (LastName:String, barber:Barber)
Barber (LastName:String, plumber:Plumber)
Plumber (LastName:String, doctor:Doctor, mechanic:Mechanic)
Mechanic (LastName:String, barber:Barber, electrician:Electrician)
Electrician (LastName:String)
```

The database itself contains the following data:

Shopkeeper Instances	*Doctor Instances*	*Barber Instances*
s_1, "Jones", d_1	d_1, "Willis", b_1	b_1, "Locke", p_1
s_2, "Jones", d_2	d_2, "Willis", b_2	b_2, "Locke", p_2
s_3, "Davey", d_2		s_3, "Davey", d_2
s_4, "Jones", d_4	d_4, "Willis", b_4	b_4, "Locke", p_4
Mechanic Instances	*Electrician Instances*	*Plumber Instances*
m_1, "Tanner", b_1, e_1	e_1, "Wisker"	p_1, "Tapp", d_1, m_1
m_2, "Ellis", b_2, e_2	e_2, "Gillies"	p_2, "Tapp", d_2, m_2
		p_4, "Tapp", *null*, m_2

3 The Rational Tree Query Language (RTQL)

In this section, we provide examples of query operations, starting from basic operations found in current object-oriented languages, to advanced examples, illustrating the specification of cyclic structures using psi-terms [13]. It should also be noted that all queries return lists of objects, thereby ensuring the compositionality of the language. Additionally, queries based on single classes, that use projection, selection, renaming or any combination of those operations are updateable, thus facilitating updateable views.

Object-identifiers (or reference values in the case of non-object-oriented complex data) are hidden from the user. This implies that all references to objects when specifying objects are done through their values, and nothing else. Thus, in order to specify or refer to an object in the database, all of the values of the attributes of that object need to be specified. This, of course, implies that the combination of values for objects in a particular class need to be unique. Additionally, if the object references other objects, then the combination of those references are unique as well.

The query and update operations provided by the language work across all class types. This means that each operation is applied in a seamless way to any of the classes. It is also important to note that due to the fact that the language does not allow object-identifiers or keys to be utilised, and that such concepts are hidden from the user, the operations in the language must be subsequently modified to allow the identification of objects to be made in a more succinct manner. This is further illustrated in the examples to come.

The language introduces a syntax to allow the user to represent cyclic references in a simple manner using recursive syntax. This syntax is based on the psi-terms as proposed by [13]. The ability to represent cycles between multiple objects allow the user to construct queries that enable the manipulation and inspection of data with cyclic references.

RTQL (Rational Tree Query Language) may be described syntactically using the standard EBNF grammar. The following grammar productions provide an abstract syntax of the language:

```
QUERY = DATA_RETRIEVAL_QUERY | UPDATE_QUERY;
UPDATE_QUERY = DATA_INSERTION_STMT | DATA_DELETION_STMT |
    DATA_UPDATE_STMT;
DATA_INSERTION_STMT = "insert into" UPDATE_QUERY_DOMAIN
    "(" INSERT_ATTR_LIST ")" "values" "(" EXPR_SPEC_LIST ")";
DATA_DELETION_STMT = "delete from" UPDATE_QUERY_DOMAIN [WHERE_CLAUSE];
DATA_UPDATE_STMT = "update" UPDATE_QUERY_DOMAIN "set"
    UPDATE_EXPRESSION_LIST [WHERE_CLAUSE];
UPDATE_EXPRESSION_LIST = "(" UPDATE_EXPRESSION
    {"," UPDATE_EXPRESSION} ")";
UPDATE_EXPRESSION = DOMAIN_ATTR "=" ATTR_EXPR_SPEC;
UPDATE_QUERY_DOMAIN = CLASSNAME;
DATA_RETRIEVAL_QUERY = DATA_RETRIEVAL_STMT [WHERE_CLAUSE]
    [GROUPBY_CLAUSE] [ORDERBY_CLAUSE];
DATA_RETRIEVAL_STMT = (SELECT_STMT | UNION_STMT | JOIN_STMT |
    DIFF_STMT);
SELECT_STMT = "select" SELECT_ATTR_LIST "from" DOMAIN_LIST;
SELECT_ATTR_LIST = ATTR_EXPR_SPEC {"," ATTR_EXPR_SPEC} |
    "nest" "(" SELECT_ATTR_LIST ")" "as" ID |
    "unnest" "(" SELECT_ATTR_LIST ")" "as" ID;
DOMAIN_LIST = QUERY_DOMAIN {"," QUERY_DOMAIN};
QUERY_DOMAIN = EXPR_SPEC | CLASSNAME;
UNION_STMT = DATA_RETRIEVAL_QUERY {"union" DATA_RETRIEVAL_QUERY};
JOIN_STMT = SELECT_STMT {INNER_JOIN | LEFTOUTER_JOIN |
    RIGHTOUTER_JOIN | CROSS_JOIN};
INNER_JOIN = "inner join" JOIN_PREDICATES;
LEFTOUTER_JOIN = "left outer join" JOIN_PREDICATES;
RIGHTOUTER_JOIN = "right outer join" JOIN_PREDICATES;
CROSS_JOIN = "cross join" JOIN_PREDICATES;
JOIN_PREDICATES = (DATA_RETRIEVAL_QUERY | CLASSNAME) "on"
    JOIN_EXPRESSION_LIST;
JOIN_EXPRESSION_LIST = "(" JOIN_EXPRESSION {"," JOIN_EXPRESSION} ")";
JOIN_EXPRESSION = ATTR_EXPR "=" ATTR_EXPR | ATTR_EXPR "!=" ATTR_EXPR |
    ATTR_EXPR ">" ATTR_EXPR | ATTR_EXPR "<" ATTR_EXPR |
    ATTR_EXPR ">=" ATTR_EXPR | ATTR_EXPR "<=" ATTR_EXPR;
INSERT_ATTR_LIST = INSERT_ATTR {"," INSERT_ATTR};
INSERT_ATTR = CLASS_ATTR | PATH_EXPR;
ATTR_EXPR_SPEC_LIST = ATTR_EXPR {"," ATTR_EXPR};
ATTR_EXPR_SPEC = ATTR_EXPR | ATTR_AGGR_EXP ["as" ID];
ATTR_AGGR_EXP = "min" "(" ATTR_EXPR ")" | "max" "(" ATTR_EXPR ")" |
    "avg" "(" ATTR_EXPR ")" | "sum" "(" ATTR_EXPR ")" | "count"
    "(" ATTR_EXPR ")";
ATTR_EXPR = DOMAIN_ATTR_SPEC | EXPR_SPEC;
DOMAIN_ATTR_SPEC_LIST = DOMAIN_ATTR_SPEC {"," DOMAIN_ATTR_SPEC};
DOMAIN_ATTR_SPEC = DOMAIN_ATTR ["as" ID];
DOMAIN_ATTR = CLASS_ATTR | PATH_EXPR | CYCLIC_REF;
EXPR_SPEC_LIST = EXPR_SPEC {"," EXPR_SPEC};
EXPR_SPEC = EXPR ["as" ID];
EXPR = BOOLEAN | INT | FLOAT | STRING | "(" EXPR ")" | "-" EXPR |
    EXPR ( "+" | "-" | "*" | "/" | "^" ) EXPR |
    FUNCTION_NAME "(" EXPR_LIST ")" | RELATIONAL_EXPR |
    "(" DATA_RETRIEVAL_QUERY ")" | "(" DATA_UPDATE_STMT ")";
RELATIONAL_EXPR ::= EXPR "between" EXPR "and" EXPR | EXPR "in" EXPR |
    EXPR ( "<" | ">" | "<=" | ">=" | "<>" | "=" ) EXPR;
CYCLIC_REF ::= CYCLIC_TAG ":" CLASSNAME ":" "(" VALUE_EXPR ")";
VALUE_EXPR ::= CYCLIC_REF | CLASSNAME ":" "(" VALUE_EXPR ")" |
    EXPR ":" ATTRNAME {"," VALUE_EXPR} | CYCLIC_TAG;
```

The salient features of the language, as expressed via the abstract grammar shown above, include the ability to specify cyclic references, as defined by the `CYCLIC_REF` production. This allows the depiction of cyclic references in the data being queried. The language is also compositional, as the result of any query (apart from insertion and deletion operations) is a list of objects. As a consequence of this, any query may be input to another query, and may also form a complex output attribute of a query.

Additionally, the grammar does not permit the use of object identifiers to refer to objects. This feature allows the user to only refer to referenced objects via their values (as shown in the following section). In doing so, objects are uniquely identified via their values, and this allows the use of update operations in a generic manner [12].

Example 2. The query to return the last names of all shopkeepers is formulated as follows:

```
SELECT LastName FROM Shopkeeper;
```

This query projects the LastName attribute from all Shopkeeper objects. The result is a set of objects of an unnamed transient class, which has a single attribute called LastName. The result is a list of last names of Shopkeeper objects represented by object id's s_1 to s_4.

Example 3. The query to return all shopkeepers whose last names are 'Jones' is defined as follows:

```
SELECT LastName FROM Shopkeeper WHERE LastName = 'Jones';
```

This query returns a subset of objects of the Shopkeeper class. The result is a set of objects of the Shopkeeper class whose LastName attributes have the value 'Jones'. This corresponds to objects s_1, s_2, and s_4.

Example 4. The query to return all shopkeepers whose last names are 'Jones', that have a doctor whose last name is 'Willis' is defined as follows:

```
SELECT * FROM Shopkeeper WHERE LastName = 'Jones' AND
    doctor.LastName = 'Willis';
```

The result is a set of objects of the Shopkeeper class whose LastName attributes have the value 'Jones' and whose associated Doctor object has attributes whose last names are 'Willis'. This corresponds to objects s_1, s_2, and s_4.

Example 5. The query to return all shopkeepers whose last names are 'Jones', whose doctor's last name is 'Willis', whose barber's last name is 'Locke', whose plumber's last name is 'Tapp', whose doctor is the same doctor as that of the shopkeeper, is defined as follows:

```
SELECT * FROM Shopkeeper WHERE LastName = 'Jones' AND
    doctor = d:Doctor:('Willis':LastName,
    Barber:('Locke':LastName, Plumber:('Tapp':LastName, d)));
```

The result is a set of objects of the Shopkeeper class whose LastName attributes have the value 'Jones' and whose associated Doctor, Barber and Plumber objects are in a cycle with values 'Willis', 'Locke' and 'Tapp' respectively. This corresponds to objects s_1 and s_2. s_4 is not selected because it does not have a cyclic value (see the instances table in Section 2 above).

Example 6. The query to return the last names of all shopkeepers is formulated as follows:

```
SELECT (LastName AS KnownAs) AS AllShopkeepers FROM Shopkeeper;
```

This query projects and renames the LastName attribute from all Shopkeeper objects. The result is a set of objects of a transient class called AllShopkeepers, which has a single attribute called KnownAs.

Example 7. The query to return the last names of all shopkeepers and barbers is formulated as follows:

```
SELECT LastName FROM Shopkeeper UNION SELECT LastName FROM Barber;
```

This query returns a list of objects of an unnamed class, that has a single attribute - LastName, which are taken from the combined result of all Shopkeeper and Barber objects.

Example 8. The query to return the last names of all shopkeepers where the last name of the shopkeeper is not the same as the last name of a barber is formulated as follows:

```
SELECT LastName FROM Shopkeeper DIFFERENCE
    SELECT LastName FROM Electrician;
```

This query returns a list of objects of an unnamed class, that has a single attribute - LastName, which are taken from those names in Shopkeeper objects, which are not in Electrician objects.

Example 9. The query to return the last names of all shopkeepers and plumbers joined by their doctor (reference) attributes is formulated as follows:

```
SELECT LastName AS SName FROM Shopkeeper JOIN
    SELECT LastName AS PName FROM Plumber
    ON Shopkeeper.doctor = Plumber.doctor;
```

This query returns a list of objects of an unnamed class, that has two attributes - SName and PName, which are taken from the combined result of all Shopkeeper and Plumber objects where the shopkeeper and plumber objects have references to the same doctor object.

Example 10. The query to return the last names of all shopkeepers and plumbers where their doctor-barber-plumber references are cyclic and are the same is formulated as follows:

```
SELECT LastName AS SName FROM Shopkeeper JOIN
    SELECT LastName AS PName FROM Plumber
    ON sd:Doctor:(Shopkeeper.doctor.LastName:LastName,Barber:
       (Shopkeeper.doctor.barber.LastName:LastName, Plumber:
       (Shopkeeper.doctor.barber.plumber.LastName:LastName,sd))) =
       pd:Doctor:(Plumber.doctor.LastName:LastName,Barber:
       (Plumber.doctor.barber.LastName:LastName, Plumber:
       (Plumber.doctor.barber.plumber.LastName:LastName,pd)));
```

This query returns a list of objects of an unnamed class, that has two attributes - SName and PName, which are taken from the combined result of all Shopkeeper and Plumber objects where the doctor, barber, and plumber objects have the same last name and are cyclic.

Example 11. The query to return the last names of all shopkeepers with a nesting of different doctors for each set of shopkeepers with the same last name is formulated as follows:

```
SELECT LastName, NEST(doctor.LastName) AS DoctorsName FROM Shopkeeper;
```

This query returns a list of objects of an unnamed class, that has two attributes - LastName and DoctorsName, which is a list of strings.

Example 12. The operation to insert a new shopkeeper is formulated as follows:

```
INSERT INTO Shopkeeper VALUES ('Thompson', d:Doctor
    ('Willis':LastName, b:Barber:('Locke', Plumber:('Tapp', d,
    Mechanic:('Tanner', b, Electrician:('Wisker'))))));
```

or as,

```
Shopkeeper.Insert ('Thompson', d:Doctor ('Willis':LastName,
    b:Barber:('Locke', Plumber:('Tapp', d, Mechanic:('Tanner', b,
    Electrician:('Wisker'))))));
```

This insert operation inserts a new shopkeeper into the database, which uses an existing doctor instance to associate it with. If the doctor instance was not found, it would first be inserted into the database as well.

4 Translation to Lower-Level Language

In Section 3, we saw numerous examples of how high-level operations could be used to take advantage of psi-terms [9] to allow cyclic data structures to be referenced. Now, we will show the translation of those higher-level operations into a lower-level language, that forms the interface to the runtime system. This language supports typical oo constructs, multiple inheritance, parametric polymorphism, and other commonly-found features. It also includes static methods on all classes allowing all instances of that particular class to be retrieved. Additionally, it has access to library functions that allow constraint checking to be turned on or off, and support update / delete propagation along references.

Example 13. In Example 2 we saw how projection would be used in the query language. The translation of that operation would be as follows:

```
transient Class UniqueRandomClassName {
    private Shopkeeper sh = constructor;
    String get LastName() {return sh.LastName};
    void set LastName(String newval) {sh.LastName = newval;}}
foreach o in ShopKeeper.GetInstances() {
    new UniqueRandomClassName(o);}
return UniqueRandomClassName.GetInstances();
```

Example 14. We would translate the first selection from Example 3 as follows:

```
List resultlist;
foreach o in ShopKeeper.GetInstances() {
    if (o.LastName == 'Jones') {resultlist.Add(o);}}
return resultlist;
```

Example 15. The translation of the second selection from Example 4 is as follows:

```
List resultlist;
foreach o in ShopKeeper.GetInstances() {
    if (o.LastName == 'Jones' && o.doctor.LastName == 'Willis') {
        resultlist.Add(o);}}
return resultlist;
```

Example 16. The third selection from Example 5 is translated as follows:

```
List resultlist;
foreach o in ShopKeeper.GetInstances() {
    if (o.LastName == 'Jones' && o.doctor.LastName == 'Willis' &&
        o.doctor.barber.LastName == 'Locke' &&
        o.doctor.barber.plumber.LastName == 'Tapp' &&
        o.doctor.barber.plumber.doctor == o.doctor) {
        resultlist.Add(o);}}
return resultlist;
```

Example 17. We would translate the renaming from Example 6 as follows:

```
transient Class AllShopkeepers {
    private Shopkeeper sh = constructor;
    String get KnownAs() {return sh.LastName;}
    void set KnownAs(String newval) {sh.LastName = newval;}}
foreach o in ShopKeeper.GetInstances() {
    new AllShopkeepers(o);}
return AllShopkeepers.GetInstances();
```

Example 18. We would translate the union from Example 7 as follows:

```
transient Class UniqueRandomClassName {String LastName;}
foreach o in ShopKeeper.GetInstances() {
    new UniqueRandomClassName(o.LastName);}
foreach o in Barber.GetInstances() {
    new UniqueRandomClassName(o.LastName);}
return UniqueRandomClassName.GetInstances();
```

Example 19. We would translate the difference from Example 8 as follows:

```
transient Class UniqueRandomClassName {String LastName;}
foreach o in ShopKeeper.GetInstances() {
    new UniqueRandomClassName(o.LastName);}
foreach o in UniqueRandomClassName.GetInstances() {
    foreach p in Barber.GetInstances() {
        if (o.LastName == p.LastName) {delete o; break;}}}
return UniqueRandomClassName.GetInstances();
```

Example 20. We would translate the first natural join from Example 9 as follows:

```
transient Class UniqueRandomClassName {String SName; String PName;}
foreach o in ShopKeeper.GetInstances() {
    foreach p in Plumber.GetInstances() {
        if (o.doctor == p.doctor) {
            new UniqueRandomClassName (o.LastName, p.LastName);}}}
return UniqueRandomClassName.GetInstances();
```

Example 21. The second natural join from Example 10 would be translated as:

```
transient Class UniqueRandomClassName {String SName; String PName;}
foreach o in ShopKeeper.GetInstances() {
    foreach p in Plumber.GetInstances() {
      if (o.doctor.LastName == p.doctor.LastName &&
          o.doctor.barber.LastName == p.doctor.barber.LastName &&
          o.doctor.barber.plumber.LastName ==
            p.doctor.barber.plumber.LastName &&
          o.doctor.barber.plumber.doctor = o.doctor &&
          p.doctor.barber.plumber.doctor = p.doctor) {
          new UniqueRandomClassName(o.LastName, p.LastName);}}}
return UniqueRandomClassName.GetInstances();
```

Example 22. We would translate the nest operation from Example 11 as follows:

```
transient Class UniqueRandomClassName { String LastName;
    List DoctorsNames;}
foreach o in ShopKeeper.GetInstances() {
    flag = false;
    foreach (p in UniqueRandomClassName.GetInstances()) {
        if (o.LastName == p.LastName) {flag = true; break;}}
    if (flag) {p.DoctorsNames.Add (o.doctor.LastName);} else {
        q = new UniqueRandomClassName (o.LastName);
        q.DoctorsNames.Add(o.doctor.LastName);}}
return UniqueRandomClassName.GetInstances();
```

Example 23. We would translate the insert operation from Example 12 as follows:

```
foreach (o in Shopkeeper.GetInstances()) {
    if (o.LastName == 'Thompson' &&
        o.doctor.LastName == 'Willis' &&
        o.doctor.barber.LastName == 'Locke' &&
        o.doctor.barber.plumber == 'Tapp' &&
        o.doctor.barber.plumber == o.doctor &&
        o.doctor.barber.plumber.mechanic.LastName == 'Tanner' &&
        o.doctor.barber.plumber.mechanic.barber == o.doctor.barber &&
        o.doctor.barber.plumber.mechanic.electrican.LastName ==
          'Wisker') {flag = true; break;}}
Run as transaction {
    if (flag) return o;
    s = new Shopkeeper('Thompson', null);    // adding new shopkeeper
    foreach (o in Doctor.GetInstances()) {
        if (o.LastName == 'Willis' &&
            o.barber.LastName == 'Locke' &&
            o.barber.plumber == 'Tapp' &&
            o.barber.plumber.doctor == o &&
            o.barber.plumber.mechanic.LastName == 'Tanner' &&
            o.barber.plumber.mechanic.barber == o.barber &&
            o.barber.plumber.mechanic.electrican.LastName ==
              'Wisker') {flag = true; break;}}
    if (flag) {s.doctor = o; return;}
    // since a new doctor needs to be added, and is the start of a
    // cyclic relationship, the others must not exist, so add them all
    d = new Doctor('Willis', null);
    b = new Barber('Locke', null);
    p = new Plumber('Tapp', null, null);
    m = new Mechanic('Tanner', null, null);
    e = new Electrician('Wisker');
    d.barber = b;
    b.plumber = p;
    p.doctor = d;
    p.mechanic = m;
    m.barber = b;
    m.electrician = e;}
return UniqueRandomClassName.GetInstances();
```

5 Conclusion

In this paper, we have provided a definition for a query language that is able to query and update complex data types with cyclic references, and provides a way for the user to depict cyclic relationships. We have also shown the translations that are possible from user queries based on the language. The translations map to a lower-level language that allows object-oriented constructs, along with constructs enabling basic querying of persistent objects.

Additionally, the query language does not permit the user to specify object-identifers in the query operations, as outlined in [12]. This allows other query languages that are based on complex types, such as object-oriented databases, and XML databases to be based on the presented language, without needing to use object-identifiers as an identity-primitive [10].

References

1. Melton, J., Simon, A.R.: Understanding the New SQL: A Complete Guide. Morgan Kaufmann, San Francisco (1993)
2. Catell, R.G.G., Barry, D.K., Berler, M., Eastman, J., Jordan, D., Russell, C., Schadow, O., Stanienda, T., Velez, F.: The Object Data Standard: ODMG 3.0. Morgan Kaufmann, San Francisco (2000)
3. Subieta, K., Beeri, C., Matthes, F., Schmidt, J.W.: A stack-based approach to query languages. In: Eder, J., Kalinichenko, L.A. (eds.) Proc of the 2nd International East/West Database Workshop, pp. 159–180. Springer, Heidelberg (1994)
4. DeRose, S.J.: XQuery: A unified syntax for linking and querying general XML documents. In: Proc of the Query Languages Workshop (QL) (1998)
5. Meijer, E., Beckman, B., Bierman, G.: LINQ: Reconciling object, relations and XML in the .NET framework. In: Proc of the ACM SIGMOD International Conference on Management of Data, pp. 706–706. ACM Press, New York (2006)
6. Schewe, K.-D.: On the unification of query algebras and their extension to rational tree structures. In: Proc of the 12th Australasian Database Conference, pp. 52–59. IEEE Computer Society Press, Los Alamitos (2001)
7. Kirchberg, M., Schewe, K.-D., Tretiakov, A., Wang, B(R.): A multi-level architecture for distributed object bases. Data & Knowledge Engineering 60(1), 150–184 (2007)
8. Kirchberg, M.: Integration of Database Programming and Query Languages for Distributed Object Bases. PhD thesis, Massey University, New Zealand (2007)
9. Aït-Kaci, H.: An overview of LIFE. In: Stogny, A.A., Schmidt, J.W. (eds.) EWDW 1990. LNCS, vol. 504, pp. 42–58. Springer, Heidelberg (1991)
10. Abiteboul, S., Kanellakis, P.C.: Object identity as a query language primitive. Journal of the ACM 45(5), 798–842 (1998)
11. Schewe, K.-D., Schmidt, J.W., Wetzel, I.: Identification, genericity and consistency in object-oriented databases. In: Hull, R., Biskup, J. (eds.) ICDT 1992. LNCS, vol. 646, pp. 341–356. Springer, Heidelberg (1992)
12. Schewe, K.-D., Thalheim, B.: Fundamental concepts of object oriented databases. Acta Cybernetica 11(1-2), 49–84 (1993)
13. Aït-Kaci, H.: An introduction to LIFE-programming with logic, inheritance, functions, and equations. In: Proc of the International Symposium on Logic Programming, pp. 52–68. MIT Press, Cambridge (1993)

A Learning Object Composition Model

Victor H. Menendez[1] and Manuel E. Prieto[2]

[1] Universidad Autónoma de Yucatán, Facultad de Matemáticas, Periférico Norte Tablaje 13615, Cordemex, CP 97110, Mérida, Yucatán, México
[2] Universidad de Castilla-La Mancha, Escuela Superior de Informática, Paseo de la Universidad 4, CP 13071, Ciudad Real, España
mdoming@uady.mx, manuel.prieto@inf-cr.uclm.es

Abstract. The composition of Learning Objects can be defined as the necessary process for its construction from a set of eventually simpler Learning Objects (assets, atoms). Traditionally, this process is done manually, but it can also be automated, which constitutes a motivation of our current research. In this paper, a model for this process is presented. The main phases of the model are described and also a possible way to apply them into an architecture that offers diverse characteristics and benefits useful for the composition: interoperability, reuse and adaptation. For the model, Learning Objects composition begins by defining the instructional needs and then, locating and selecting the best resources conformed. Finally, it is necessary to transform them into some e-learning standard (as LOM), and sequencing and packing them for their distribution.

Keywords: Learning object, composition, model, reusability, interoperability.

1 Introduction

The use of Learning Objects is a new trend in Computer-Based Learning [1]. Numerous educational institutions have initiated a change process of its educational resources and courses to structures based on Learning Objects. In Internet, many repositories of Learning Objects have proliferated. A query in some searching service will give us an idea about the impact of Learning Objects into research projects developed lately in the instructional computing area.

A clear definition of what a Learning Object is, does not exist. Many authors define a Learning Object in different forms [2]. Generally, all of them consider a Learning Object as a recyclable multimedia-content element that has an educational intention. This demonstrates the dual nature of the object of learning: educational and computer related. It is clear that there are multiple advantages that grant the Learning Objects for e-Learning, specially reusability and interoperability [3] [4].

Reusability consists in to assemble and to use Learning Objects in different contexts of learning. Thus, Learning Objects must be sufficiently simple in terms of instructional aims to be used in different situations. In another words, they must have a small granularity. Many authors call this type of Learning Objects assets [5], atoms [2], etc. The reutilization process focuses on the instructional intentions of Learning Objects.

Interoperability refers to take a Learning Object created into some computational environment and use it in other one, preserving its functional characteristics. The e-Learning standards deal with this situation. Interoperability focuses on the computational aspects of Learning Objects.

Consequently, it is very important to take the best advantage of the time and the cost invested in the construction of Learning Objects because it is not an easy process. It is necessary to consider factors like intention, style of learning, instructional design, interactivity, interface, format, descriptors, etc. Generally, this process is done by a teacher in a manual and empirical way. The automation of this process is subject of many researches [2] [6].

This paper presents a model and architecture for that process. The activities required in the construction process of Learning Objects are defined, and an architecture that implements it, is described as well. The solution defines one highly flexible interface that allows composing Learning Objects from digital resources. In the following sections the parts are detailed, its functionalities and its relationships.

2 Conceptual Framework

The Learning Object concept is analogous to the Component concept, which is used in Component-based Software Engineering (CBSE) [7]: every component does a specific functionality, which can be used in different contexts. The same component can be employed for different solutions doing the same work.

Therefore, the process of Learning Object´s composition is similar to the process of the composition that follows Component Based Software Engineering (CBSE). In CBSE, the process starts defining a set of requirements and searching the more adequated components to it. The selected components are adapted and assembled in a defined architecture [8] [9] (fig. 1). On the other hand, the result of Learning Object composition is a higher-level instructional unit, built as a collection of simpler units.

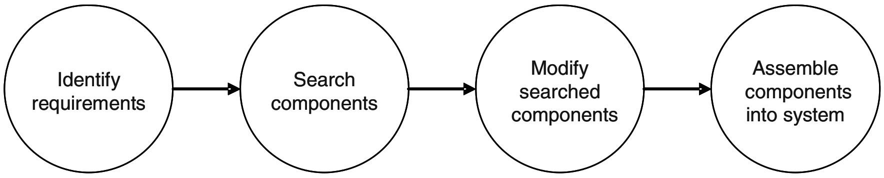


Fig. 1. The Composition in CBSE

Therefore, we can assume than the composition of Learning Objects is the necessary process for its construction from a set of digital resources or other Learning Objects, eventually simpler, and that in addition, satisfies certain specific aims of learning for a concrete context.

In order to achieve this, it is necessary to be able to locate, identify, transform, organize and pack resources.

The composition consists of the phases of location, identification, transformation, organization and distribution (fig. 2). The teacher or developer does a search to get a set of Learning Objects or digital resources than can be used for an instructional purpose. The search is done in the web, repositories or directories.

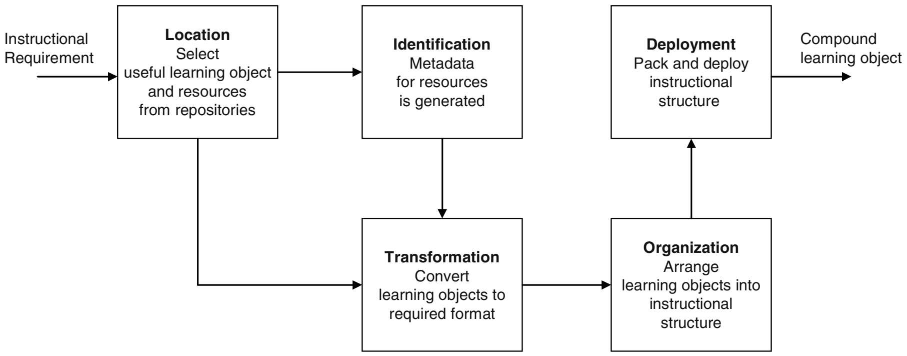

Instructional Requirement
Location
Select useful learning object and resources from repositories
Identification
Metadata for resources is generated
Deployment
Pack and deploy instructional structure
Compound learning object
Transformation
Convert learning objects to required format
Organization
Arrange learning objects into instructional structure

Fig. 2. The main phases of Learning Object composition

If the selection included digital resources, then the generation of metadata is required. A transformation process of Learning Objects is necessary because many of them are saved into repositories in a different format than the target for use. All Learning Objects are sequenced to match the instructional design. The last step is to pack and deploy the compound Learning Object.

3 Model and Proposed Architecture

From the process described before, a model and an architecture that implements it, are described next. The model is constituted by a multilayer structure, where each layer is specialized for a part of the process described previously (fig. 3).

The application layer is formed by editors, searching tools, Learning Management Systems, and any other applications that produce elements to be incorporated in the compound Learning Objects.

The service layer that constitutes the core of the model is shaped by a set of basic services that do the whole process of the composition. The services can be executed in a sequential form or according to a specific need, like just the search of resources.

The information layer includes repositories from which, digital resources could be extracted. Repositories will be able to be real (a database) or virtual (servers, directories). It also includes any database that contains useful information for the model, like specifications, rules, etc.

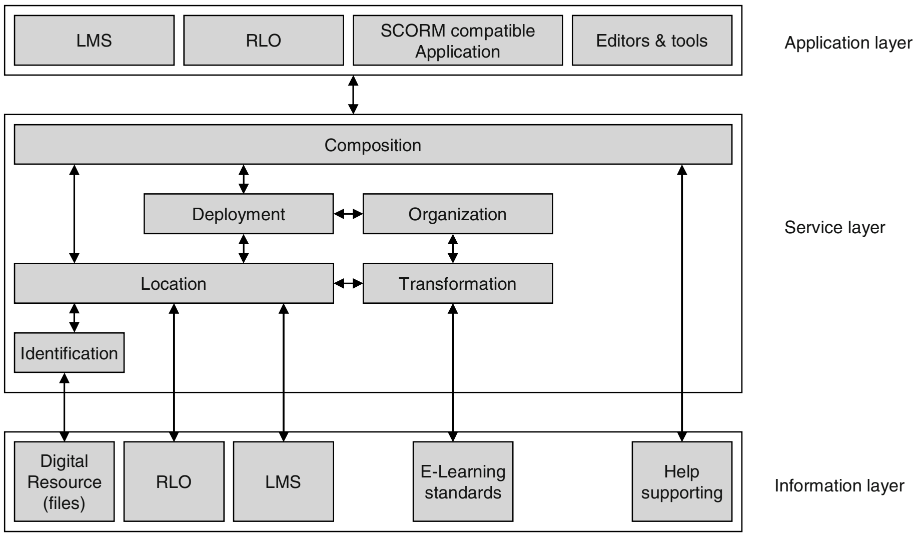


Fig. 3. Model for the composition of Learning Objects

The architecture for the model is shaped by a collection of services that are exposed by a Web Services-based interface (fig. 4). The resulting architecture is highly flexible. The Web Services [10] can be distributed allowing to improve the performance of the whole set. The basic Web Services can be used as part of others, increasing the functionality of the architecture. The execution of the services can be done regardless of the technology, the platform or the used programming language.

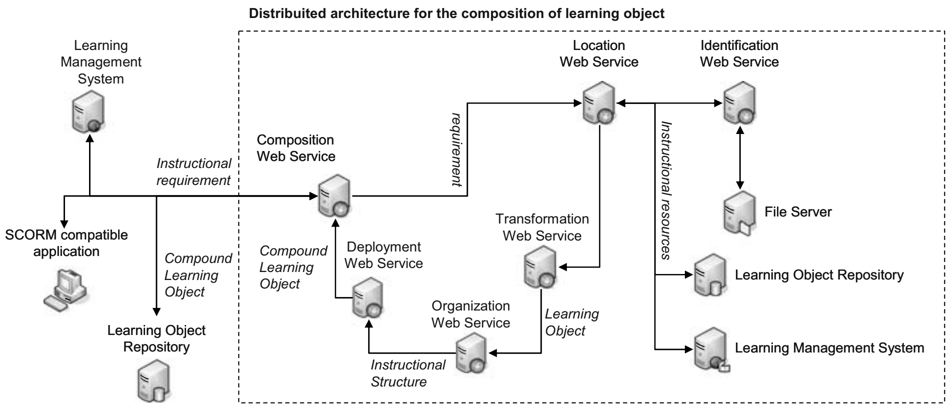

Distribuited architecture for the composition of learning object
Learning Management System
SCORM compatible application
Learning Object Repository
Instructional requirement
Compound Learning Object
Composition Web Service
requirement
Location Web Service
Identification Web Service
Instructional resources
File Server
Learning Object Repository
Learning Management System
Transformation Web Service
Learning Object
Organization Web Service
Instructional Structure
Deployment Web Service
Compound Learning Object

Fig. 4. The Web Services-based Architecture for the of Learning Objects

The location service does the search and recovery of digital resources using a collection of distributed and heterogeneous repositories. The identification service analyzes and generates describers for a basic multimedia resource, using for it a set of rules. The transformation service generates Learning Objects according to a defined standard. The organization service arranges the objects in an instructional structure and the distribution service packs, and delivers the compound learning object.

4 Conclusions

The composition of Learning Object is a process that is done generally in a manual and empirical form. Many resources are consumed, so there's a need for taking the most advantage of them. The automation of the whole or a part of the process is a motivation of numerous researches. Therefore, it is important to describe the whole process.

We developed a model for the composition based on a CBSE methodology. Modifications have been done to adapt it to Learning Objects composition.

The presented model considers a series of steps that are necessary to compose a learning object required for an instructional need. This model is implemented in an architecture based on Web Services.

The architecture can be integrated to other applications regardless of the platform or the used programming language. Just a web access is required. Each of the services are modeled as single components, so they can be used independently to solve concrete problems. The Web Services can be separated and joined to create other ones, like distributed components.

A characteristic of the proposed architecture is the aptitude to use any repository of resources as a source of Learning Objects as well as the flexibility to generate and to compose them according to specifications or standards and customizable formats.

Acknowledgments. This work is partially supported by PAC06-0059 SCAIWEB project, JCCM, Spain; TIN2007-67494 F-META project, MEC-FEDER, Spain, A/8172/07 project AECI, Spain and YUC 2006-C05-65811 project FOMIX CONACYT, México.

References

1. Farrell, R., Liburd, S., Thomas, J.: Dynamic assembly of learning objects. In: 13th International World Wide Web Conference on Alternate Track Papers & Posters, pp. 162–169. ACM, New York (2004)
2. Wiley, D.: The Instructional Use of Learning Objects. In: Wiley, D.A. (ed.), Online Version (2000), http://reusability.org/read/chapters/wiley.doc
3. Pitkänen, S., Silander, P.: Criteria for Pedagogical Reusability of Learning Objects Enabling Adaptation and Individualized Learning Processes. In: 4th IEEE International Conference on Advanced Learning Technologies, pp. 246–250 (2004)
4. Sicilia, M., Garcia, E.: On the Concepts of Usability and Reusability of Learning Objects. In: International Review of Research in Open and Distance Learning, vol. 4(2) (2003), http://www.irrodl.org/content/v4.2/sicilia-garcia.html/
5. Advanced Distributed Learning SCORM Specification, http://www.adlnet.org

6. Torres, J., Dodero, J., Aedo, I., Zarraonandia, T.: An Architectural Framework for Composition and Execution of Complex Learning Processes. In: 5th IEEE International Conference on Advanced Learning Technologies, pp. 143–147 (2005)
7. Pressman, R.: Ingeniería del software, un enfoque práctico, 6th edn. McGraw-Hill, New York (2006)
8. Sommerville, I.: Ingeniería de software, 6th edn. Adisson Wesley, México (2002)
9. Douglas, I.: Instructional design based on reusable learning objects: Applying lessons of object-oriented software engineering to learning systems design. In: Frontiers in Education Conference. Frontiers in Education Conference, vol. 3(31), pp. 1–5. IEEE Computer Society Press, Los Alamitos (2001)
10. Vossen, G., Westerkamp, P.: E-Learning as a Web Service. In: 7th International Database Engineering and Applications Symposium, p. 242 (2003)

Gathering Preference Data from Restricted Natural Language

Johann Mitlöhner

Institute for Information Business
Vienna University of Economics and Business Administration
Augasse 2–6, A-1090 Vienna, Austria
mitloehn@wu-wien.ac.at

Abstract. The Resource Description Framework provides elaborate methods of specifying semantic information for automated aggregation and other types of processing; however, RDF is too complicated for most casual computer and internet users who prefer instead to communicate in natural language text form. Recently, much progress has been made in the area of extracting semantic data from unrestricted natural language, such as automatic document classification; however, much of the content of natural language texts still remains inaccessible for automated semantic processing. In this work a restricted natural language approach to the expression of preferences is proposed that sacrifices freedom of expression for accurateness of semantic translation. A limited number of grammatical constructs for the expression of preferences is described, and an application to the expression of cinema and movie related preferences is outlined. It is shown that with a moderate amount of restriction on natural language at least in the domain of preference specification a large amount of semantic information can be extracted while maintaining a level of user-friendliness suitable for more wide-spread application beyond the limited number of RDF experts.

Keywords: Natural language parsing, RDF, preference aggregation.

1 Introduction

While the automated syntactical analysis of natural language texts has been developed to a high degree decades ago [1], the advancement in automated semantic processing remains noticeably slower. Term-based statistical methods allow for moderately reliable automated classification of natural language documents; however, most of the content of these documents is still only accessible to human readers. At the same time, simple semantic markup methods such as Microformats [2] are slowly adopted outside the semantic web community, while more powerful but complicated approaches such as RDF [3] still are only used by a limited number of people, and the vision of the Semantic Web [4] is realised only slowly.

The approach presented in this text is to identify a set of phrases and constructs that constitute a restricted natural language in the sense that people using this subset of English (or other natural languages) can easily communicate information on selected topics such as preferences in a manner that is accessible to people and computers, i.e. it is human- and machine-readable.

While formal languages and XML-based markup such as Microformats and the Resource Description Format (RDF) can go a long way towards making available information burried in free text, it is still difficult for casual and non-expert users to understand and apply these types of approaches. On the other hand, using a simple subset of natural language poses much less learning effort for online authors such as web content creators and contributors in discussion forums. This paper explores ways of using limited sets of phrases and simple sentence types to provide easy access to automatic semantic processing.

2 Human-Readable Text and Automatic Semantic Processing

The World Wide Web today consists of a very large number of natural language documents which are aimed at human readers. Keyword indexing and search can make them accessible to search and retrieval, but information extraction still needs to be done by human readers. This quickly becomes impractical with increasing number of documents. Search engines may return many thousands of relevant documents, but it is impractical for human readers to digest such amounts of information. Only automatic processing can deal with such magnitudes. Unfortunately, unrestricted natural language presents numerous problems for automatic content translation into a format suitable for automated processing with semantic web tools. The RDF Resource Description Framework encouraged the development of XML-based description languages for document content, such as the Dublin Core set; the following RDF statements use elements from the Dublin Core that describe a particular web page.

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:dc="http://purl.org/dc/elements/1.1/">

   <rdf:Description rdf:about="http://deepblue.net/rush2.html">

     <dc:creator>Rush Randle</dc:creator>
     <dc:title>Advanced Windsurfing Tips</dc:title>
     <dc:date>2007-07-18</dc:date>
     <dc:subject>windsurfing</dc:subject>

   </rdf:Description>
</rdf:RDF>
```

The data structure behind this RDF description is a directed graph where nodes stand for resources such web pages, which are connected by edges that stand for properties. In this way sentences of the type Subject/Property/Value can be formed. The main statements concerning the web page rush2.html can also be written in the simplyfied N3 notation as

```
:rush2_html :creator "Rush Randle".
:rush2_html :title   "Advanced Windsurfing Tips".
:rush2_html :date    "2007-07-18".
:rush2_html :subject "windsurfing".
```

Even for its limited scope of making available information on creator, title, date, and subject for this page, the RDF syntax is quite elaborate and unsuitable for the casual user. Other vocabularies describe more elaborate sets of elements and their meaningful application, but at an even greater cost in complexity and therefore limitation in use.

The Microformats approach is easier to understand and apply, since it basically just extends the originaly idea of semantic markup already present in HTML from the beginning. While HTML in its development turned more into the direction of layout specification instead of futher content markup the new Microformat approach aims at embedding semantic markup into the HTML code by using existing HTML tags in a new way. The following HTML code shows how the tag `<div>` can be used in this way by adding the option `class` to determine the function of the corresponding part of text within the vcard entry:

```
<div class="vcard">
  <div class="fn">Long John Silver</div>
  <div class="org">Treasure Island Diving Expeditions</div>
  <div class="tel">604-555-1234</div>
</div>
```

The full name, organization and telephone number are not only displayed in the browser, but also made accessible for automated data extraction tools, such as Firefox plugins. Instead of separating the page text and the metadata on the page the Microformat integrates them with the free text that is also addressed at the human reader. This method is based on the fact that undefined options in HTML tags are simply ignored by the browser, allowing for the overloading of the `<div>` tag with semantic markup.

While the Microformat approach is much more straightforward than full RDF it still requires a learning curve that typical casual authors will not be willing to accept. For this type of user, and therefore for a large part of Internet content creators we would wish for a solution that satisfies the following points:

– Simplicity: casual and non-expert users are not able or willing to incur significant learning cost. Any solution that aims at more than minimal adoption must be simple to understand and easy to apply.

- Extension: although simple the approach must be extensible to other domains and more elaborate types of statements, if the users so desire.
- Standard: existing building blocks and standards should be used to the highest possible degree; there is no point in re-inventing the wheel since in the RDF community a lot of methods and tools have been in development for years, such as the OWL Web Ontology Language [8].

2.1 Restricted Languages and the Expression of Preferences

Restricted natural languages such as Formalized English, Attempto Controlled English [7] and others have been formulated for a variety of domains; however, despite being designed for breadth of application practical usability often does not live up to expectations, i.e. it turns out that the expression of more complicated assertions is riddled with all sorts of problems [5]. It still remains a good idea to apply proven heuristics in restricted domains instead of aiming at generality in the design of a language that is meant to be processed by humans and machines. Pattern matching can achieve good results once the domain of a text, such as advertisements for used cars, is understood [6], since in specific domains certain types of formats are used anyway in written natural language, which can be exploited for automated processing.

In this work a bottom-up approach for the definition of restricted languages in the domain of preference statements is suggested that originates at the target RDF notation in its simplyfied N3 format. Preferences concern alternatives which are put in a relation to each other. The preference statement “A is preferred to B” is formally written as $A \succ B$. Preferences are typically stated by a number of individuals, in the context of Social Choice often called “voters”, over a set of “candidates” or alternatives. A number of requirements can be placed on preferences, such as completeness (voters must put all available candidates in relation), weak or strong preference (whether or not to allow indifference), as well as transitivity ($A \succ B$ and $B \succ C$ must mean $A \succ C$) and exclusion of circles (such as $A \succ B,\ B \succ C,\ C \succ A$).

Once preferences have been gathered an aggregate relation and if possible an aggregate ranking over the available alternatives can be constructed be means of various aggregation rules. These aggregate rankings can then be used for various purposes, such as political decision making, investment decisions, or in the case of preference data gathered from web pages and discussion forums, opinion polls. Therefore, there is potential in more than one respect for using preference data that is already available on the web, albeit in a format not directly usable for preference modeling. Some set of RDF assertions need to be assembled in order to be able to express preferences in the semantic web, and such a process will be outlined here in the simpler N3 notation that allows for easy reception by the human reader while retaining all formal rigor of the full RDF language. From this ontology corresponding natural language sentences will be formed, and patterns will be identified for automated processing of text entered by contributors of web-based discussion forums.

3 An Ontology for Preference Modeling

In this section a small ontology for the modeling of preferences is developed. An ontology describes the terms in a vocabulary and their proper use. RDF and RDF Schema offer the necessary predicates, which are prefixed with `rdf:` and `rdfs:` to distinguish them from other terms. The N3 notation is quite intuitive, once some namespaces have been defined; see [9] for further details.

```
@prefix rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix : <#>.
```

These statements define the prefixes rdf: and rdfs: in order to avoid possible confusion with other terms; e.g. there is a property “range” in RDFS which is now written as “rdfs:range”, since other ontologies may define a property “range” as well, but with different meaning.

```
:Alternative rdf:type rdfs:Class.
:better      rdf:type rdf:Property.
:better rdfs:domain :Alternative.
:better rdfs:range  :Alternative.
```

These statements determine that :Alternative is a class i.e. it can have instances, such as A and B in the previous examples. The RDF property rdf:type can be abbreviated as simply “a” since it is so frequently used in ontologies.

The relation :better is defined as a property with domain and range :Alternative, i.e. all statements :A :better :B are valid if :A and :B are of type :Alternative. RDF allows for multi-classing, which means that :A and :B can be something else besides alternatives, such as movies, projects, or books; in the context of preference modelling they are referred to as alternatives. With these definitions, the following statements are meaningful for preference modeling:

```
:A a :Alternative.
:B a :Alternative.
:C a :Alternative.
:A :better :B.
:C :better :A.
```

This corresponds to the expression $C \succ A \succ B$. However, for preference aggregation there is a need to model more than one voter, i.e. a mechanism is needed to state that certain preferences are held by certain voters. In the RDF framework this can be done by making statements about statements, which is known as “reification”. Without reification different preferences stated consecutively would look like this:

```
:A :better :B.
:B :better :A.
```

This is only meaningful when statement one and two refer to different voters. In N3 statements about statements can be made in various ways, such as curly braces:

```
:Jim :says { :A :better :B }.
:John :says { :B :better :A }.
:Sue :says { :B :better :A }.
```

Now it is obvious that :A is not inherently better than :B, but only where :Jim is concerned. Once these statements are available some aggregation rule can be applied to arrive at an aggregate ranking, such as :B :better :A.

4 Search Patterns for Preference Gathering

In order to arrive at a simple yet extensible method of gathering preference data from written natural language several approaches are possible:

- Parse each sentence using an English language grammar and identify sentences expressing preferences.
- Use pattern matching to identify one of several known constructs for preferences.

While natural language parsing is feasible and works reasonably well for text typically found in dicussion forums and similar sources of text where overly complex sentences are very rare it is not trivial to use the syntax tree provided by the parser for semantic processing. Since the approach described here is only aimed at identifying a limited number of language constructs that express preferences a simple pattern matching approach is proposed for processing. It turns out that practical problems tend not to involve language parsing into syntactical units but assigning proper meaning to those constructs once they are identified.

Search patterns are needed to gather preference expressions from large amount of natural language texts. These patterns are compiled by listing alternative ways of formulation for the N3 statements which are the target format. The concept :A :better :B can expressed in a number of ways:

```
Cats are better than dogs.
I prefer cats to dogs.
I like cats more than dogs.
```

When a list of occurences in actual discussion contributions has been compiled the corresponding patterns can be defined, such as

```
?a are better than ?b
?a is better than ?b
I prefer ?a to ?b
I like ?a more than ?b
```

The trouble with this approach applied to full natural language is that the list would grow very large. However, language used in internet discussion forums and similar places is not very complex, since people tend to communicate in an informal manner; elaborate sentences and sophisticated grammatical constructs are unlikely to occur. In this respect, the language is restricted by the medium; it is further restricted by the patterns that are recognized by the automatic semantic processing programs.

The author of sentences is usually easy to identify, since discussion forums assign user names, and the authors of different contributions can be distinguished with little cause for error.

Since the language to be processed semantically is simple and the domain is restricted to preference expressions a pattern matching approach appears to be feasible. The patterns described above can therefore be augmented with translation rules in the following format, given here in pseudo code:

```
IF   contribution by user ?u matches "I like ?a more than ?b"
THEN :u :says { :a :better :b }.
```

While user names can be identified rather easily, the more difficult part is to identify alternatives. However, since within a discussion forum there usually evolves a certain vocabulary simply from continued use by the participants, who usually do not invent new terms and tend to refer to things discussed previously in the same manner as already established, a dictionary-based approach can be applied. In much the same way as shown for the preference expressions a list can be compiled for the references used in previously posted contributions; based on this list alternatives can then be matched.

```
:lotr :pattern "Lord of the Rings"
:lotr :pattern "LOTR"
```

Here, two ways of referring to the work by Tolkien are presented, and numerous other similar entries will describe other works of fantasy as they are discussed in various forums. Naturally, some basic list has to be provided as a starting point for the application, but only by continous extension from user input can it ever approach comprehensiveness. Therefore, an interactive approach to ontology definition is suggested that is based on user satisfaction with the benefits provided by such a system.

5 Interactive Ontology Extension

If it is possible to engage a sufficient number of users in participation in the project the ontology can be expanded and fitted to the actual language use in the discussion forum by allowing users to define new entries in a simple and straightforward fashion. The most obvious point for doing this is the definition of additional patterns which identify alternatives:

```
:lotr_tt :pattern "The Two Towers".
:lotr_tt :pattern "LOTR Two Towers".
:lotr_tt :pattern "Lord of the Rings - The Two Towers".
:lotr_tt :pattern "LOTR TT".
:lotr_tt :pattern "TT".
```

The last entry in this dictionary shows that within the context of talking about fantasy by Tolkien the abbreviation “TT” is easily understood as refering to the Two Towers, the second part of the Lord of the Rings; however, outside of this context it may refer to quite different things. Fortunately, RDF and its easier notation N3 provide the namespace mechanism described above to resolve conflicts of this type, therefore an appropriate prefix can provide the required context:

```
@prefix f:  <http://www.fantasy.org/2007/fantasy-terms#>.
...
f:lotr_tt :pattern "TT".
```

By simply typing some statements about new terms and their patters the forum users can be enabled to interactively extend the ontology. Depending on the subject matter, the type of users and their level of expertise and commitment an additional mechanism such as approbation by other users before integration into the system ontology may be applied to ensure compliance with the already existing definitions. Ultimately, for each term defined in the ontology that has relevance for the automatic semantic processing some information has to supplied describing the corresponding implementation, and this information as far as the ontology is concerned can only be in natural language, since it is aimed at the programmers who, e.g, develop an implementation for the aggregation rule that uses the data expressed with the :says and :better predicates.

However, even without further semantic processing immediate benefits from using the approach of restricted natural language are apparent for all users which may incite them to apply the method in their own contributions: once statements are put into N3 format queries can be issued that do not rely on the use of often misleading keywords, but instead on precisely defined N3 predicates and subjects, e.g., with the use of the cwm Closed World Model program the N3 statements can be queried in various ways [10], such as who made statements about certain terms, who used a certain predicate, and rules such as if :x is used as the range of :better than :x must be an instance of :Alternative.

6 Future Work and Conclusions

A number of students at the Vienna University of Economics are taking part in the development of a basic ontology for the expression of preferences for work of fiction, such as fantasy books and movies. A manually compiled list of common expressions in this domain is being prepared, together with natural language comments for each definition in the ontology.

The interface for the discussion forum that allows for definition of additional vocabulary and patterns is currently only envisioned as N3 notation input, since the participating students all have at least a rudimentary knowledge of RDF and ontology concepts, alleviating at least for the moment the need for more user-friendly types of interfaces. Obviously, for any approach that hopes to achieve more wide-spread acceptance a simpler interface for additional ontology definitions is required, possibly again applying the ideas explored in this work and using restricted language for the definition of additional terms in the ontology.

This work has explored the use of restricted language for the gathering of preferences in online media such as web discussion forums. Given the narrowly defined domain of expressions concerning preferences for certain types of alternatives, such as fantasy fiction books and movies, and the comparatively simple grammatical properties of the natural language used in discussions of this sort, a pattern matching approach has been described for the automatic analysis and extraction of preference data from free text contributions. These data can then be used in semantic processing such as application of aggregation procedures for the construction of aggregate rankings and various reports on the usage of predicates and statements concerning certain objects. The semantic data also allow for content analysis such as opinion polls and therefore may encourage further participation in the use of the restricted language developed interactively by showing immediate benefits to the users.

References

1. Winograd, T.: Understanding Natural Language. Academic Press, London (1972)
2. Microformats, `http://microformats.org`
3. RDF Primer, W3C Recommendation (February 10, 2004), `http://www.w3.org/TR/rdf-primer/`
4. W3C Semantic Web Activity, `http://www.w3.org/2001/sw/`
5. Pool, J.: Can Controlled Languages Scale to the Web? In: CLAW 2006. 5th International Workshop on Controlled Language Applications. Cambridge, Mass (2006)
6. Liddle, S.W., Hewett, K.A., Embley, D.W.: An Integrated Ontology Development Environment for Data Extraction. In: Godlevsky, M., et al. (eds.) 2nd Internation Conference on Information Systems Technology and its Applications. GI Lecture Notes in Informatics, pp. 21–34. Gesellschaft für Informatik, Bonn (2003)
7. The Attempto Project, University of Zurich, `http://attempto.ifi.unizh.ch`
8. W3C Semantic Web Activity, Ontology Web Language, `http://www.w3.org/2004/OWL`
9. Getting into RDF & Semantic Web using N3, `http://www.w3.org/2000/10/swap/Primer`
10. CWM, A General-purpose Data Processor for the Semantic Web, `http://www.w3.org/2000/10/swap/doc/cwm.html`

Patterns for Screenography

René Noack and Bernhard Thalheim

Christian Albrechts University Kiel, Department of Computer Science, Kiel, Germany
{noack,thalheim}@is.informatik.uni-kiel.de

Abstract. The visual design development of Web Information Systems is a complex task. At present, the process is mainly based on experiences and seems to be an immovable part of art, because there is no systematic concept to develop the layout from intentional towards presentational level. Typically, there is a late consideration of graphical issues that results in inflexibility and cause problems for extension and change management. Therefore, we propose an early involvement of layout and playout. Further, we follow a pattern-based approach to be able to reuse concepts. This paper introduces such patterns and try to clarify their structure and task for the whole development. Because the WIS development process is based on six dimensions, we initially introduce development dimensions.

1 Introduction

If we take a look at websites, we perceive sites as good sites and bad sites in terms of impression, composition, usability, and utility. In general, there are no methodologies to define overall good sites, because the perception depends, for example, on users recognition resp. their profile. Nevertheless, we sense well designed sites as well structured and well orchestrated, which isn't a result of coincidence. At present, the development process is often based on experiences of a developer because the absence of global layout development concepts. The reasons for development decisions are as far as possible unknown and seems to be an immovable part of art. Because we try to turn the development process from an art into a craft, we have to detect the basic principles of good sites, as far as possible. Therefore, we analyse existing sites, searching for multiple used patterns, and try to understand their applicability. Moreover, we consider other approaches like task models, e.g. CTT [11], and website description languages, e.g. SiteLang [16]. The main objective of this direction is the development process generalisation. We aim in handling development decisions to derive an adequate layout from generic descriptions.

However, we missing a full story integration at present. Also the important part of user adaptation exists only rudimentary in most cases. Another deficiency is the development from the scratch, piece by piece each time. In general, the recycling of parts is used on a very low level, because the layout is often examined as a big black box developed by a designer. Conceptual approaches only rudimentary exist. In the field of art seems to exist suitable concepts to solve this development problems. Particularly, the concepts to initiate classical dramas point out some interesting approaches that we try to use. Currently, graphical issues will late considered while the website development. It results in inflexibility and cause problems for extension and change management. Our aim is to support the systematic and early involvement of layout and playout for that we develop an approach to *screenography* [10], which adopts and generalises dramaturgy and scenography. Scenography has its roots in theatre, film and television, i.e. outside the web area, and contains the composition of action space, plot and dramaturgy. The dramaturgy controls the sequence of scenes and determines the composition of information. Our claim is to show that WIS layout also requires scenography and dramaturgy to facilitate the understanding and memorisation of content and to support orientation within the WIS.

Engineering must be based on a variety of aspects, languages and tools for construction and methodologies for work. We describe the first two elements of *website engineering* in this paper: Aspects are founded by dimensions of websites and construction is based on pattern approaches.

The following chapter introduces development dimensions we have to consider for screenography. We try to clarify the aims of these dimensions and demonstrate some different weightings we have noticed in practice. Further, we propose a possible composition of dimensions for websites in future. Chapter 3 presents a pattern approach regarding the layout development. The patterns will help us to derive possible solutions from an intentional description. Therefore, we demonstrate possible types of patterns and introduce one type of pattern in detail. After this, we depict the mapping of patterns to grids. Finally we conclude in section 5 and give an short outlook of further research directions.

2 Dimensions to Consider for Screenography

Layout development is affected by a huge set of parameters and conditions. Because it's possible to generalise this aspects, we distinguish six dimensions of layout development. In the following we introduce this dimensions and discuss their composition. Moreover, we give some examples how different the weighting of each dimension can be because the influence of the dimensions depend on application area. It clarifies, too, that it isn't necessary to consider all dimensions in each situation. From this, we derive a solution for websites we prefer in future.

2.1 Development Dimensions

According to [12] the development process is based on 6 dimensions, the intention, context, storyboard, content, functionality, and presentation. They are useful for high-level development, and we have to define their task and scope.

Intention
The intention dimension specifies the type of an application we try to develop at a very general level. Therefore, we analyse aims, goals, and visions and assign the application to a well-known and adequate application type. The classification is

useful because the difference of main objectives and their different weightings. For example, the major aim of information sites is the information and explanation while it could be a minor aim to sell some products. By contrast, the major aim of electronic commerce sites is to sell as much as possible products. A minor aim of such sites could be to give a detailed view about the offer to support the selling.

The differences, regarding the objectives, influence the way of representation implicit. Explicit, we are able to define the atmosphere resp. the ambience of an application. The ambience definition, as part of intentional dimension, will only be done conceptional and helps to define colour-schemas. The dimension doesn't specify concrete colours because the intention does not consider cultural or religious preferences.

Another part of the intention dimension is the adaptation of layout and playout that is influenced by profile and portfolio of the users resp. the provider. The profile ascertainment is important for the preferences and behaviour patterns of the users. For instance we need it in the field of e-learning. As mentioned in [2] there can be users prefer formal thinking and systematic working and others who are more example oriented. Thus, the profile of users influences the way of representation resp. the layout and playout in dependence on their preferences. The same is true for the portfolio [13] of the users and the provider, which defines the tasks, involvement and collaboration. So, the type of a task or a bundle of tasks determines the way of representation. Moreover, the playout is influenced by individual task collections, too, because they can be responsible for story changes.

Story
As users doesn't act in the same way and their profile and portfolio is very heterogeneous, we need an individualised representation resp. several runs through the site. The story is able to provide only the needed content to avoid that the orientation of the users get lost in hyperspace. Mainly there are several specified stories within the story space. Thus, it is possible to handle different portfolios and demands, and so we are able to switch to other stories if they are more appropriate, in dependence on the availability resp. the rights and roles of a user. If we specify a concrete run through the story, we call it scenario. Typically, the task of a scenario is to guide the users.

Context
The environment is an important part of the context dimension. It isn't easy to grasp but often a problem while using a system. A part of context is the application context, e.g. the attention a user have or need to interact with the system. Further, we have to consider the equipment, e.g. hardware and software resp. its abilities. Particularly, the usable bandwidth and the performance of a client strongly determines the way of representation. Moreover, the rights a user have and roles are assigned to him influence the presentation resp. possible interactions. Besides, the quality of service (QoS) is a part of context dimension and can provoke that some parts have to be restricted to ensure the demands.

For example a very fast response time could be ensured by removing details or support information.

Content
Content is the most important part of information systems, because it is the reason for build-up such systems. Of course, regarding the layout development the content dimension is very important too and concerns all presentable data as well as their types. However, it is often difficult to design an adequate database, handling the content in an appropriate way, because, for example, the structure of presented data typically differs from the organisation within the database.

Functionality
The functionality dimension is mainly represented by navigation and interaction aspects. Moreover, integrity constraints and usable functions are a part of this dimension. Functions can have static or dynamic effects. We differ functions act local resp. cannot change the system from functions that are able to change the system or itself. Further, we can distinguish internal and external functions. Internal functions realise, for example, the adaptation of the content to users preferences while external functions have to be perceivable by the users.

Presentation
The presentation dimension is influenced by the others and strongly determines the result of a development process as illustrated in figure 1 (development pentagon). The presentation dimension has the aim to concretise the definitions and restrictions of the other 5 dimensions. It completely defines the look and feel on the basis of these definitions. The representation is determined by the weight of each dimension and depends on the application area.

2.2 Dimension Composition

The specific composition of the dimensions plays an important role while the development process, because the influence of each dimension is determined by the application area. Therefore, we analyse some typical processes and try to find an appropriate solution for website development.

Website Development – State of the Art
The classical website development considers the content, functionality, and presentation dimension. Mainly it starts with content and functionality specification and therefrom the presentation dimension will derived. In the case of predefined presentation templates it is necessary to integrate content and functionality, maybe realised by a downgrade of them. Choosing an adequate presentation template can avoid such downgrades, but typically, a template is developed for a specific application. Thus, if we start with the presentation dimension, it is essential to check content and functionality for adequacy regarding the chosen template, and if required, we have to inform the developer about integration problems. The remaining three development dimensions play a minor role while

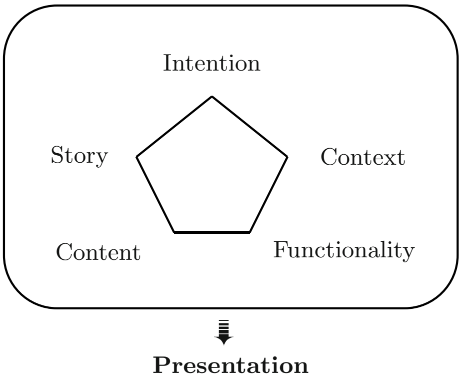


Fig. 1. Development dimensions pentagon

current website development. Often they will completely neglected. For instance, the intention isn't an explicit modelled part of a current development process. Often the involved persons hope that they all have the same imagination of the aims until the concrete realisation. Mainly well structured high-level commitments are missed but would be useful to find a real conformable main direction. Also the story and the context dimension will rarely considered. Ignoring the context can result in an inappropriate playout regarding the equipment of the target group. Further, without a story specification, we only clarify what should be a part of output, but it disregards how to move within the website.

Intention	Story	**Content**	**Presentation**
	Context	**Functionality**	

Fig. 2. Website development dimensions - state of the art

Architectural Development
Development in the field of architecture is hard affected by context, because typically, buildings won't planned without consideration of it's neighbourhood. In general a good contextual embedding is a claim of inhabitants. Inside a building the context is important too. For instance a good orientation requires to satisfy the expectations of the main target group. We notice that architectural conditions can be demands of websites, too. Mainly the intention dimension has the top priority of architectural development. The intention represents the starting point and determines the main direction. Other dimensions, e.g. the context, are able to overwhelm these definitions, but they shouldn't ignore intentional aspects. Only carefully selected parameters should differ from users expectations to avoid perception and cognition process impairments. However, sometimes it is helpful to disappoint expectations and arising the attention of the users. In the

past, it has often been realised by animated-gifs. The third important dimension concerns the utility of the development. Therefore, there is a functionality dimension that is able to influence the intention and context dimension.

As a result of intention, context and functionality determinations, other minor dimensions can influence the overall result. Mainly the content and story are parts of a second step. The story is important to ensure usability, e.g. museum tours should avoid crossing ways. The content dimension can have placement requirements and have to check the compatibility with decisions of the other dimensions. A third step of architectural development concerns the presentation. The presentation is a result of the definitions and restrictions of the other five. Nevertheless it is an important dimension responsible for the realisation in the form of an instance. Figure 3 highlights the main dimensions of architectural development.

Intention	Story	Content	Presentation
	Context	**Functionality**	

Fig. 3. Architectural development dimensions

Art Development – Classical Drama

A third way of development we can adapt from the art, particularly while looking to classical dramas. Main dimensions of art developments are content, story, and presentation. So, it is the opposite approach of architectural development. The content of theatricals often is a given part. Also the story exists in the case of classical dramas. Hence, the presentation dimension is mainly determined by content and story definitions, but not exclusive. Often there isn't a requirement to realise the presentation in a strict original way. So, the directors often try to combine classical themes with current events, wherefore the presentation differs from the original and reflects the directors view. In this case the presentation will influenced by secondary dimensions of art development, the intention, context, and functionality. So, if the director has the intention to combine classic and modern aspects, it will necessarily influence the presentation dimension. In the same way changes regarding the context or functionality can have effects on presentation.

This weighting of dimensions we detect regarding the arrangement within museums. Presentation, story, and content are the most important dimensions, while the other three dimensions are responsible for unique impressions resp. for individualisation of the supply. Online museums nearly have the same demands.

Intention	**Story**	**Content**	**Presentation**
	Context	Functionality	

Fig. 4. Art development dimensions

Website Development – State of Science
We noticed that there are bindings between the abstraction layer model (ALM) [15] and the development dimensions because of a hierarchy of the development dimensions and the possibility to assign the dimensions to the layers of the ALM. Because our main goal to generalise the layout development process, we aim in start at intentional level. From this, we try to derive the story and contextual issues. Particularly, the context strongly determines the subsequent representation because big differences regarding the equipment. The story specification help us to get a good screen partitioning in dependence on story progress. With these high-level definitions we derive appropriate functionalities and specify the properties of content in consideration of the upper level definitions and restrictions. A last development step can be the presentation dimension that determines the layout in detail. At best the other dimensions lead to one possible result. Figure 5 illustrates the weighting we prefer for a general step by step website development allows e.g. flexibility as well as adaptivity.

Intention	**Story**	Content	Presentation
	Context	Functionality	

Fig. 5. Website development dimensions (possible future scenario)

The next section shows how these different dimensions influence the development and application of pattern during website construction.

3 Pattern Development

We often have similar problems while development processes and prefer to find a universal solution for a whole problem field. Therefore, Christopher Alexander had introduced *design patterns* [1] to solve some architectural problems by reusing concepts. Gamma [5] picked up this approach to solve object-oriented software programming issues. Further, he said that patterns are useful for other application areas, too. We try to find patterns for layout development or in other words we are searching for presentation patterns.

3.1 Types of Pattern for Websites

In [10] we distinguished between three principles taken from cognitive psychology, principle of visual communication, principle of visual perception, and principle of visual design. Visual communication deals with navigational and interaction aspects and aims in understanding as prerequisite of the communication partners. Visual perception resp. visual cognition concerns the orientation within the story and the screen. The objective of composition resp. the visual design is to realise a sufficient placement of media objects so that users have a good orientation within the arrangement and there are no problems to perceive the needed content.

We generalise these three principles as main parts of layout pattern development and enrich the classification in figure 6 by work progress and kind. The kind distinguishes between detected pattern types characterising a specific solution, e.g. there are many different types of evolution pattern underlie the same global restrictions. The work progress is important, because patterns are not only represented by static states. So, it can be helpful to define the behaviour of some steps within the story and to derive a composed pattern.

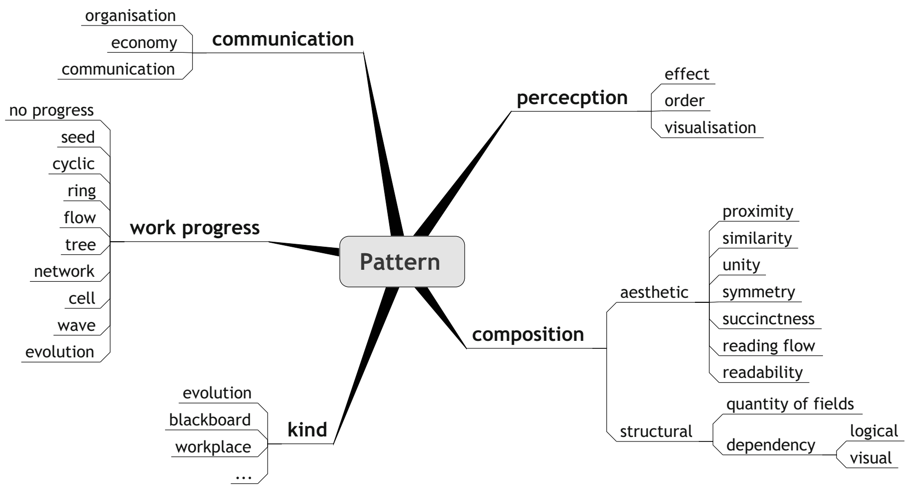


Fig. 6. Pattern classification

3.2 Work Progress Pattern

A simple step of searching for an adequate presentation only concerns the composition of content within the presentation space. It's a good solution in case of common arrangements, but in general, we need an extended view, that considers the communication and perception, too. Moreover, it is useful to distinguish the progress, we will introduce within this section.

Work progress patterns are able to consider all development dimensions that were introduced in chapter 2.1. The weighting of each dimension depends on the application and specified development targets, again. Figure 7 depicts the development dimensions regarding the work progress. Within this paper, we are mainly interested in presentational aspects resp. progress patterns regarding the presentation dimension. However, we have to consider the other dimensions and benefit from their results.

A prerequisite of progress perception is the visualisation and sometimes the ability to keep it in mind. How much we have to keep in mind mainly depends

Progress	Evolution Story	Evolving Content	Work Progress
Intention	Evolution Context	Evolution Functionalities	Pattern

Fig. 7. Work progress dimensions

on the application, e.g. the set of changes between progress steps as well as the overall length of the process. It is urgently necessary to perceive all specified steps of a progress unit as related. For example it can be realised by a similar way of representation within a page or between pages. If the coherence of such units get lost or it isn't clear the users will have problems to understand the progress and maybe come to wrong decisions. We are able to avoid such situations if we analyse the needed type of progress and derive therefrom an appropriate way of representation. Consequently it is necessary to distinguish possible types of progress, at first. Therefore, we pick up the segmentation of Brown [3] and complete it. Thus, we are able to differ the progress types *no progress*, *seed*, *cyclic*, *ring*, *flow*, *tree*, *network*, *cell*, *wave*, and *evolution*.

The simplest type of work progress is *no progress*. It isn't a hypothetical type, because sometimes there isn't a need for interactions as well as changes regarding the presentation. Also there are possibly long-time changes so that a progress exists but the users cannot perceive it because of the temporal distance. For instance, in the case of electronic billboards, typically, there is a wish to separate the adverts so that it isn't possible to perceive a relation between these. Otherwise the advert-firm relation could be affected.

Sometimes there are no restrictions except that there have to be a progress. We call this progress type *network*. It is the first real and most flexible progress type, which is hard to overlook for users and is appropriated for general parts of presentation to improve the accessibility by cross-links.

If we detect some decompositions in addition, we speak of the *cell* type. We need it for refinement, because it can be necessary to differ the way of presentation at several levels.

Iterative additions to the network type result in the *ring* type and if it is directed, we call it *cyclic*. If cyclic types are further affected by advancements, it is an *evolution* type. For example, the cyclic type is useful in the case of offering online museum tours, because typically, a round trip is closed and directed.

If a work progress with interactions is directed and moreover affected by advancements, we are able to describe a *flow* as another progress type. If there are duplications in addition, we call it the *seed* type. This type of progress occurs in the case of collaborations. By contrast, if there are decompositions in addition to the properties of a flow, we call this type *tree*, which could be helpful regarding retrieval. Besides, a flow with decompositions and duplications means the *wave* type.

Evolution Pattern

In the following we introduce the *evolution pattern* to demonstrate how the progress pattern work. In [4] *lifestreams* were introduced as time-ordered streams of documents, considering the aim of organising and ease the finding. We generalise this approach as *evolution pattern* and try to organise general output units, not only documents. Because it is close to the lifestreams, the evolution pattern describes a perceivable progress, too. So, we need parts of history and future or at least one of them, because it's very important to perceive more than one progress step. Sometimes it is impossible to present more than one progress step at the same time. Then the users have to keep in mind the presentation of previous steps while the presentation of the following is hidden. Subsequently, we prefer examples with a visible progress, because it is easier to grasp.

Evolution patterns must not be hard time-based in opposition to the lifestream approach. For instance, in the case of a task list, the ordering is much more important than the time. Further, it's urgent necessary that the users perceive the progress direction, because otherwise, problems occur as illustrated in Figure 8. The arrows within the illustrations point out that the users can have two different lines of vision, wherefore this example representation (grid) is inappropriate to induce an evolutional impression.

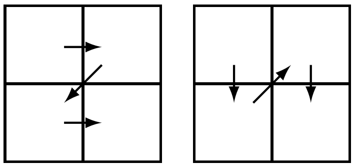


Fig. 8. Lines of vision

So, we have to consider the presentation hierarchy between the output units, e.g. ordering by size, alignment, or colouration. In general there is a hierarchy, too, regarding all of the other dimensions, e.g. hierarchic dependencies resp. the hierarchic level in the case of functionality. To support the perception of evolution, it is helpful to avoid hard and abrupt direction changes regarding the line of vision. Therefore, we analyse in general the line of progress, e.g. the hops within in a tree of navigation.

Typically one output unit of such evolutional arrangements acts as present part. It can and should dominate the presentation to avoid perception problems. It's possible that we don't need it only in the case of overview representations.

4 Related Work

Website development has been the subject of several studies. As a result, some approaches and tools were developed. The presentation development is an important part of the website development and should typically result in an adequate layout and playout of content in consideration of user and provider aims.

[14] discussed basic elements of presentation development, e.g. dots, lines, and shapes. Because each presentation is a composition of these elements, they are

a prerequisite of presentation development. The perception of these elements is influenced by characteristics, e.g. size, colour, and contrast. So, the general impression of a user is influenced by both elements and characteristics, and it depends on user preferences, capabilities, and cultural as well as religious views. Regarding the impression of colours, Itten [8] introduced an approach how to use colours. [9] discussed impressions and effects regarding the style of letters, words, and images.

The composition of elements needs some rules, e.g. placement rules to order the content on the screen. The grid-based approach in [6] tries to solve the placement problem by combinatorial analysis. The approach is possible because there are not too much sizes of basic elements and an overlapping isn't allowed.

At present, some general methodologies for WIS design exist, e.g. [7,11,16]. All of them should be able to interact with aspects of presentation development. For example, within SiteLang we can specify some types of stories, which should be supported by an adequate layout as well as vice versa. Because we detected general and reusable parts as *work progress* and *composition*, we prefer an pattern-based approach, e.g. to interact with the specified story. Presentation patterns ease the development process by reusing concepts and are able to influence other parts of the WIS development process.

5 Conclusion

Layout development is a complex task because of the large variety of conditions and multiple choices to compose these. Development of coherent pages must be based on website style in the same fashion while maintaining the specifics of each story step. Therefore, we need a systematic approach to development of suites of web pages that can be easily used by a wider auditory. Software Engineering (SWE) has an understandable description of requirements to the SW that can be discussed with any stakeholder. SWE of presentation systems must also provide a way for description of the interfaces that should be developed. A pattern-based approach seems to be most suitable to accomplish the demands. Therefore, the paper has presented *screenography pattern* as an pattern-based approach of layout development.

Next steps in future will be the specification of another patterns and dimensions. Further, we will demonstrate in detail how to compose several patterns and how to benefit from the pattern interplay. Moreover, we have to specify the grids in detail developing an universal description.

References

1. Alexander, C.: A Pattern Language: Towns - Buildings - Construction. Oxford University Press, New York (1977)
2. Binemann-Zdanowicz, A.: Towards generative engineering of content-intensive applications. In: Maibaum, T., Frias, M. (eds.) Proc. of Intl. Conf. on Principles of Software Engineering (PRISE 2004), pp. 41–49. Buenos Aires (2004)

3. Brown, L.: Integration Models: Templates for Business Transformation - The Authoritative Solution. Sams Publishing, Indianapolis (2000)
4. Fertig, S., Freeman, E., Gelernter, D.: Lifestreams: An alternative to the desktop metaphor. In: Tauber, M.J. (ed.) ACM SIGCHI Conference on Human Factors in Computing Systems (CHI 1996), pp. 410–411. ACM Press, New York (1996)
5. Gamma, E.: Design patterns. Addison-Wesley, Amsterdam (1996)
6. Graf, W.H.: Constraint-based graphical layout of multimodal presentations. In: Maybury, M.T., Wahlster, W. (eds.) Readings in intelligent user interfaces, pp. 263–285. Morgan Kaufmann Publishers Inc., San Francisco (1998)
7. Houben, G.-J., Barna, P., Frasincar, F., Vdovjak, R.: HERA: Development of semantic web information systems. In: Cueva Lovelle, J.M., Rodríguez, B.M.G., Gayo, J.E.L., del Pueto Paule de Ruiz, M., Aguilar, L.J. (eds.) ICWE 2003. LNCS, vol. 2722, pp. 529–538. Springer, Heidelberg (2003)
8. Itten, J.: Kunst der Farbe. O. Maier, Ravensburg (1961)
9. Meggs, P.B.: Type and Image: The Language of graphic design. John Wiley & Sons, New York (1992)
10. Moritz, T., Noack, R., Schewe, K.-D., Thalheim, B.: Intention-driven Screenography. In: Mayr, H.C., Karagiannis, D. (eds.) Proc. of Intl. Conf. on Information Systems Technology and its Applications (ISTA 2007). LNI, vol. 107, pp. 128–139. Köllen Verlag, Bonn (2007)
11. Paterno, F.: Model-Based Design and Evaluation of Interactive Applications (Applied Computing). Springer, London (2000)
12. Schewe, K.-D., Thalheim, B.: Conceptual modelling of web information systems. Data & Knowledge Engineering 54, 147–188 (2005)
13. Schewe, K.-D., Thalheim, B.: User models: A contribution to pragmatics of web information systems design. In: Aberer, K., Peng, Z., Rundensteiner, E.A., Zhang, Y., Li, X. (eds.) WISE 2006. LNCS, vol. 4255, pp. 512–523. Springer, Heidelberg (2006)
14. Skopec, D.: Digital Layout for the Internet and other Media. Ava Publishing SA, Lausanne (2003)
15. Thalheim, B.: Entity-Relationship Modeling - Foundations of Database Technology. Springer, Berlin (2000)
16. Thalheim, B., Düsterhöft, A.: Sitelang: Conceptual modeling of internet sites. In: Kunii, H.S., Jajodia, S., Sølvberg, A. (eds.) ER 2001. LNCS, vol. 2224, pp. 179–192. Springer, Heidelberg (2001)

A Model of Self-organized Criticality in Emergent Web Systems

Christian Russ and Mathias Lux

Klagenfurt University, Universitätsstrasse 65-67, 9020 Klagenfurt, Austria
{christian.russ,mathias.lux}@uni-klu.ac.at

Abstract. Self-organized criticality (SOC) is a measure to identify if complex systems have the potential to build out emergent behavior. This phenomenon is known in many different areas of physics, biology, ecology, economy and social systems. Its core assertion is that an over critical energy input can lead to spontaneous, self-enforcing and unpredictable self-organization. In such a process new structures and orders emerge. “Online Crowds” are masses of internet users who behave and act collectively on the web. They tend to follow specific online trends and can generate tremendous online traffic and attention (cp. Social software like YouTube.com, MySpace.com, Friendster.com, etc.). “Online Crowds” are observed rarely, are hard to predict and even harder to generate artificially. So it is essential for online business ideas to foster and facilitate these “Online Crowds” to reach a critical mass of online users to grow continuously and to produce profit. The paper in hand introduces the “Online Crowds” model (OCM) in context of web information systems employing the concepts of self organization theory and self organization criticality. Furthermore a statistical approach for detection of indicators of self organized criticality is presented along with experiments and preliminary results showing the applicability of the approach.

1 Motivation

Web information systems are massively changing the last years on the Web. They started similar to mass media as typical push concepts with a clear separation of content creator and consumer. This has changed since the start of the world wide web: Nowadays also users are taking the lead and content consumer becomes co-creator and multiplier of information on the Web [8], [26]. This has also tremendous implications on the existing and new upcoming business models on the Web and innovative approaches with social software and Web 2.0 technologies are emerging everywhere on the internet [20], [24]. User-centric and do-it-yourself online services offer huge opportunities for their owners, because they just have to provide the online platform with the appropriate “social architecture of participation” and the online masses (Online Crowds) are meant to populate it [27].

Hence it is not easy to attract or gather these “Online Crowds” as a hand full of big players take most of the market share as they follow the laws of “path dependence” [1] and network effects [29]. It says the “rich get richer” meaning the more users are gathering on a web site the more attractive and valuable it is to follow, because of the network externalities which arise [4], [17]. Consequently just offering a social software platform is not enough to prevail and to obtain market share.

To build up a sustainable online community for a web information system is a challenging task and can get costly, if the initiator is not able to utilize the mechanisms of the "Online Crowds". Consequently there are two important questions to be answered for successful web information systems: (i) How can the critical mass of online users be reached within minimal effort? (ii) When is it possible to predict whether the online venture has the high potential of a blockbuster or if it is just another flop?

The scope of this paper is to explain the "Online Crowds" model (OCM) in context of self-organization and to present an approach to identify over criticality with statistical means. First we present the OCM and discuss its relation to the self organization theory and especially to self organized criticality (SOC) in complex systems. The statistical means and experiments are described in the second part of the contribution. Finally the findings are summarized and concluded and open questions as well as future research are pointed out.

2 "Online Crowds" and Self-organization

Successful online communities are hard to initiate and even harder to maintain. There are no physical relationships like in brick-and-mortar shops and their businesses and business competitors always try to offer smarter web sites or more attractive services. Additionally online businesses always have to struggle against the natural drain of inactive users and the attrition rate in interest and involvement of users [22], [23].

One way out is to make use of the psychological concept which is called *social contagion*. In this highly collective process the spread of ideas, attitudes, or behavioral patterns in a group are emerging through imitation and conformity of the affected individuals [10]. Examples of these social mass phenomena of humans are trends, fads, as well as panics and hypes. The outcome of such a social contagion processes can have positive or negative effects on the involved attendees and the environment. Sometimes it even leads to extraordinary results in a spontaneous, rapid and unpredictable way ([7], [12], [26]).

We assume that on the internet humans show similar behavior as in the physical world. The difference is mainly the socio-technological changes which are provided with the internet, e.g. missing time and space limitations and anonymity and virtuality. People are using web technologies to facilitate and enjoy their social life in a highly interactive, dynamic and self-referential way. Hence these general conditions can lead to collective actions of online users which sometimes result in social online behaviors, which we call the "Online Crowds" [27].

For sure not all internet phenomena are "Online Crowds" and even many online communities do not behave like "Online Crowds". However many of the few successful online ideas and business have aspects of the "Online Crowds" embedded, which helped to foster the growth and diffusion process.

Self organization theory (SOT) is a promising approach to explain emergent behavior in social processes and helps to provide a theoretical framework supporting a

"Online Crowds" model (OCM), which we detail in the subsequent chapter. SOT systems tend to build up new structures or patters as a response to external circumstances and are able to influence elements of their system without explicit control [15] [18]. Moreover these kind of open systems are dynamic, non-deterministic, self-maintaining that exist far from equilibrium and sometimes employ highly emergent phenomena of reactions. Based on [2] SOT has been discovered and used in various contexts like thermodynamics, biology, cybernetics, information theory, virtual communities [11] and others [15] to explain new evolving structures and order.

On the other side, collective behavior cannot be explained easily by the attributes and relations of a system: It has to be assumed that the whole is more than the sum of the respective parts. To understand social mass processes we can use the characteristics of SOT which are supporting this concept. According to [3], [18], [15], [2] they offer following properties: Complexity, global order from local interactions, autonomy, redundancy and resilience, self-referentiality, non linearity and feedback loops, organizational closure, bifurcations and symmetry breaking and finally far from equilibrium dynamics (dissipative structures).

In [25] dissipative structures are introduced as non equilibrium thermodynamic systems which generate spontaneously order with new and stable structures. This is performed by absorbing (dissipating) energy from their external environments until it reaches a critical point of energy input and a bifurcation is happening. In [27] we generalize this approach and present the stationary states of dissipative structures which follow with some modifications the concept of [19] p. 254.

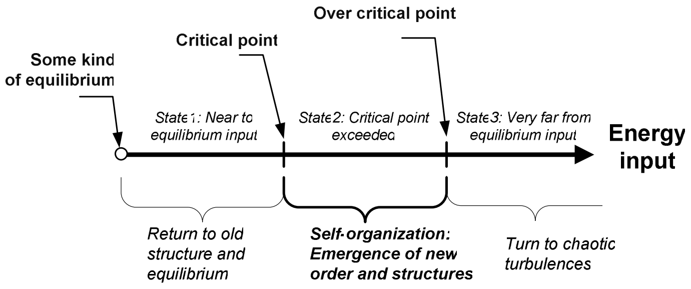

Over critical point
Critical point
Some kind of equilibrium
State1: Near to equilibrium input
State2: Critical point exceeded
State3: Very far from equilibrium input
Energy input
Return to old structure and equilibrium
Self-organization: Emergence of new order and structures
Turn to chaotic turbulences

Fig. 1. Stationary states of dissipative structures

The left side equilibrium is a kind of stable state where a system resides and the entropy is very low. The absence of entropy production means that no energy is dissipated. In the context of this contribution energy is seen in a broader term and also includes information, money or beliefs. Without the supply of external energy, it will remain in this minimizing energy state. However, if there is an asymmetric input of new energy into the system, it dissipates this energy and starts to move away from the equilibrium. Depending on the degree of energy it can pass several stages of behavior:

- **State 1 – Near to the equilibrium:** If the energy contribution is not high or ending up early the system state will move right back to the prior equilibrium. The energy input is *under critical.*

- **State 2 – Critical point exceeded:** But, if a constant energy influx is provided then the system can reach the critical point of self organization. If this threshold is exceeded the system can emerge and build up a new order and different stable structures. Hence small instabilities in the system can lead to irreversible bifurcations and new stable system states [15]. The energy input is *over critical* and has to persist to keep the established structures alive.
- **State 3 – Very far from the equilibrium:** In some cases the energy input continues to grow, then the system moves to the last possible state: chaotic turbulences. At this time the energy input is highly asymmetric and *beyond over critical* and can destroy build up structures.

[3] and [2] suggested that this pattern can be seen as a kind of self-organized criticality (SOC) of the system. It states that large dissipative systems can drive themselves to a critical state with a wide range of length and time scales. These dissipative structures with SOC are applied to a diverse set of phenomena, e.g. ecological disasters, economic dynamics, biological evolution and emergent social systems. If those conditions are fulfilled, the stationary equilibrium becomes unstable and self-organization is building up a kind of new order of structures. So we assume that in collective behaviors [5] and "Online Crowds" the same basic principles can be applied.

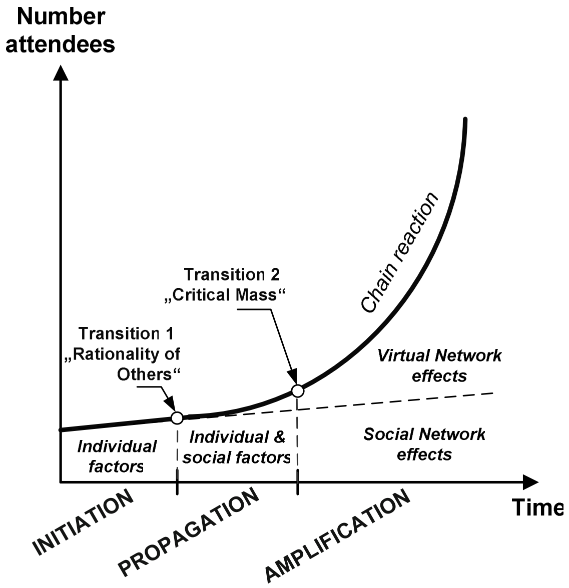


Fig. 2. Development phases of an Online Crowd process ([27])

2.1 Using "Online Crowds" to Reach SOC on the Web

The "Online Crowd" development model (OCM) based on [27] is visualized in Figure 2. It describes the most important phases and some reasons how such online phenomena can arise. First of all it has to be noted that every "Online Crowd" need a suitable and anticipating environment. Depending on the situation these are states like over information, under information, regulative or social changes, some disruption in the society through, new innovations, breaking values and norms and so on. From the SOC point of view, this represents some kind of critical energy input in the affected system. In such a favorable digital biotope there are additional *psychological attractors* necessary to focus the attention and the interest of some enclosed audience. Typical attractors are sensational news, shocking events or intriguing messages. If the attractors are strong enough to obtain a minimal set of attendees, the core followers are formed and the process begins to start and with these phases [27], [7], [12]:

- **Initiation phase:** Mainly facts and individual opinions of attendees are dominant. Basically these are early adaptors, trend setters and geeks who search for new challenges, tend to try everything and have a low risk aversion on failing. Like in dissipative structures, if the energy input is not enough the process will die off because of missing new attendees and psychological attractors.
- **Propagation phase:** Users begin to look at the behavioral signals of other users instead of using their own experience and cognition. So the new attendees tend to follow the real and virtual paths of earlier users. In the transition from the individual to social decision making the *rationality of others* is more and more dominating the process. The concept of the imitation is known in various areas like in observational and social learning [9], information cascades [6], as well as with path dependence [1] and network externalities [17], to name only a few of them.
- **Amplification phase:** Users are attracting more and more new followers until a threshold is reached: the critical mass of involved users. The *critical mass* [28] turns the social online process into a chain reaction of self-enforcing and self-accelerating online re-actors and the transition from propagation to amplification phase happens. At this stage the critical point of dissipative structures is reached and the system builds up a new structure of collective and self-synchronized online users. As mentioned before small changes can lead to bifurcations in unexpected and extraordinary results of collective re-actions. The self-organization of the "Online Crowd" is activated and can grow very fast. Practically saying, in this phase the social network effects [29] are amplified with the virtual network effects of the Internet [4]. They represent the inexhaustible energy pool based on euphoria and over-enthusiasm which is necessary to reach the self organized criticality of a web system.

A practical example of a successful accomplished "Online Crowd" can be illustrated with the social bookmarking tool del.icio.us[1]. The system offers a so called tagging mechanism which admits the users to assign a number of freely chosen keywords (tags) to the bookmarks. This non-hierarchical keyword categorization is also known as folksonomy [21]. From the "Online Crowd" point of view del.icio.us perfectly matches the development phases described before. The facilitating **environment** of del.icio.us was present, because a lot of online users intended to share bookmarks and URIs with online friends instead of saving them locally. So at the beginning one of the *initial attractors* was definitely the novel and unique internet domain name

[1] http://del.icio.us

combining wordplay with the US ending domain. Last but not least the web 2.0 technologies delivered the fundamental technical background for building up an easy to use and scalable platform for online masses.

In the **"initiation phase"** the early adopters simply found it convenient to store their individual bookmarks on the web to have it accessible from everywhere. It was free to use and even the aggregating data was not lost or locked in; simple programming APIs allow data aggregation and re-use. So primary it made sense and it was comfortable to have a central web store for all thousand web links cumulated over a web surfers life. Furthermore it was easy to share some web links with friends and fun to explore the links and favorites of others.

So the service grew and one very important factor was already present: the others. With del.icio.us it was possible to "see" the visibility of others through their published web links. Hence the initiation phase transformed slowly to the next state the **"propagation phase"**. From this time on people started to look more what others are doing on this service than asking their own rational why "joining the club". The *rationality of others* [27] provided the urge to publish a lot of web links, getting hopefully noticed by others and in parallel, observing the most popular web links of others and the absolute numbers of bookmarks. Additionally it was very important that just surfing and discovering of bookmarked web links was free of charge and easily accessible. Features like "most poplar", "tag of the day", "tags to watch" and the tag cloud concept helped to foster the acquisition of new users. With this vehicle positive feedback loops were appearing, which pushed the online service to the most important point: the threshold of critical mass of attendees. Today del.icio.us has reached the **"amplification phase"** and the emergence of an "Online Crowd" is fully developed. More than a million users are populating the favored service today and charging the service with millions of web links. The service is not a stand alone concept any more, several other web sites, blogs, mash-ups and syndication mechanisms are re-using del.icio.us content and features. Moreover the social bookmarking task is easy so every internet user can join the trend. A lot of web site providers, online news channels and bloggers have included simple "bookmark this" buttons to their articles to foster and spread their content into del.icio.us. Critics about the business value, copyright, privacy issues [14] and the potential of misuse of the social bookmarking concept for fads, rumors and manipulation are mostly ignored [24]. The risk awareness and the carefulness of new joining users is low, which is a very good indicator that del.icio.us is residing in an "Online Crowds" process.

With this multi dimensional concept of social and virtual network effects del.icio.us designed a smart and successful online architecture of self-organized criticality (SOC). The "Online Crowd" readiness factor for this kind of approach helped the service to grow and prosper adequately in a reasonable short time.

3 Measuring Self Organized Criticality in Web Systems

Unfortunately "Online Crowds" are hard to predict and follow their own rules. In general they have some similar patterns which are unique, but one never knows where they start to emerge. Several effects similar to the concept of "Online Crowds" have been reported in the area of social software and large online platforms: The actual emerging of such a crowd can – in a first step – only be supported with empirical evidence afterwards. Statistics provide a good method to support such theories, but the outcome of each statistical test heavily depends on the data used for the test. As can be seen in Fig. 3 the data needed is the number of users *u* at a given time *t*. While this is an intuitive approach for communication networks, where online or offline users are registered for web communities the actual number of users is hard to obtain. For instance regular visitors, who do not contribute but only consume, might not be registered but can be counted as users. Several different alternative measures are available:

- Views – How many pages of a web site were viewed in a given time period
- Unique visits – How many visitors with different IP addresses have accessed the site in a given time period?
- Registered users – How many users are registered at the website?
- Active participations – How many contributions (posts, uploads, etc.) have been made in a given time period?

Each of those measures has its flaws: While the views can also be initiated by crawlers or spider, unique visits do not reflect the time spent on the web site: A unique visit may consist of one single up to many views. The same with registered users: They include also inactive users and those long having abandoned the site. Active participation might be a measure coping with this inability, but it does not include lurkers (people just consuming without contribution) [22], [23].

An obvious way would be to investigate all of these variables and find out for which of them the model might apply and in how far it applies, but most of this data remains hidden as it is not only issue of research but also a valuable business asset.

3.1 Methodology

As there is currently no mathematical model given in the context of the "Online Crowds" approach we intended to research if the model follows certain functions $u=f(t)$ in different phases. Generally it is assumed that popularity, which is certainly plays a role in the OCM, can be described by a sum of exponential functions. Finding parameters defining the exponential functions is a hard task. However we assume that in the initiation phase the increase in users can be modeled linearly. Our intention was to find the *critical point*, where the increase of the user count exceeds linear behavior, to find the time point where a system gets critical and enters the next phase in the "Online Crowds" model. The main idea is that the number of users *u* depends on the time *t*. Intuitively in the initiation phase *a linear increase* of users should be observed: $f(t)=c_0+c_1t$.

For our approach we used linear regression: We tried to find a best fitting regression for the initiation (beginning of the sample) for 25 and more samples under the assumption that the initiation phase is at least 25 samples (or respective users) long. As soon as the coefficient of determination R^2, which measures the goodness of fit of the linear regression, gets too small the initiation phase is assumed to be over and the propagation phase begins. Therefore we observe a local maximum of R^2 around the critical point. Unfortunately this maximum can not be assumed to be a global maximum: As the number of users and therefore the number of samples increases dramatically in the propagation and in the amplification phase a linear regression modeling the steep rise in the amplification phase is likely to give a good fit too.

As we search for a very flat rise and a respective low gradient in u we weighted R^2 with the regression coefficient denoting the gradient of the regression line c_1. So our function for determining the phase transition is

$$f(t) = R^2 \cdot (1 - c_1) \tag{1}$$

The weight in this function decreases with the increase of the gradient. Therefore a global maximum is more likely at the beginning. In the following the undertaken experiments to support this model are described.

3.2 Preliminary Results

As companies in general don’t publish their web access logs or user stats real world data sets are hard to obtain. Most obvious possibility is the acquisition of a full test data set for research purposes. Such a test data set is available from the social bookmarking portal BibSonomy [16], which is run by the University of Kassel. This set features data on 1212 users describing also when they interacted with the system. Another opportunity is offered by the social bookmarking system del.icio.us [13]. For each URL bookmarked the system offers a XML based data file describing who bookmarked the resource at which time. Why this data might be appropriate to show the emergence of “Online Crowds” as well as tests based on both data sets and results are described in the subsequent subchapter.

3.2.1 Bibsonomy

The Bibsonomy data set consists of several files resembling a relational database, where all bookmarked links and publications of a user are stored. Along with description, link and user id also the time of the bookmarking action is given. As the very first bookmark of a user indicates that the user has registered, the data was cleaned in the following way: For each user the first bookmarking action was chosen and the respective date and time were put it into a list. The minimum of time points was subtracted from each subsequent time point, so the time of the very first bookmark was time point 0. Therefore we got 1212 relative time points starting from 0, which denotes a first use of a user of the system. From the 1212 time points we had to remove the first 6 as they stretched over 8 years, which has to be – semantically speaking – an error in the data as the system has not been in use for such a long time. In our experiment the number of users is understood as the number of previous time points given at any time point in the data. This gives the cumulative number of users who have interacted with the system up to a given time. While this does not include the possibility of *one time use* we assume that this gives an upper border and approximates the number of actual users.

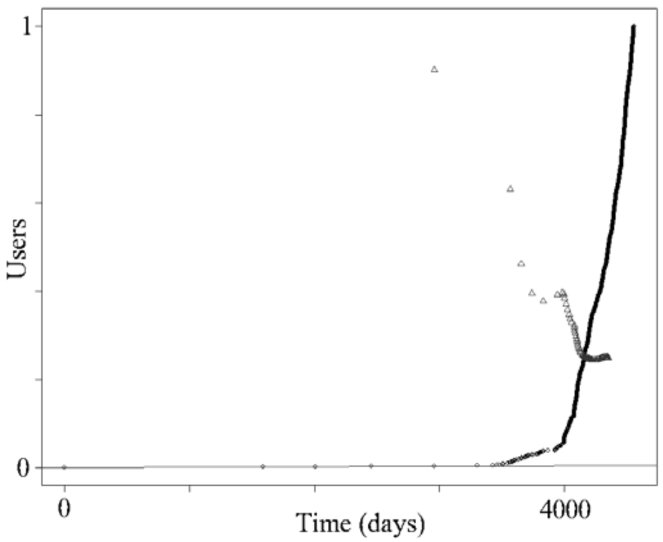


Fig. 3. Regression of the Bibsonomy data set. The x axis gives the time in days and the y axis the relative number of users.

Fig. 3 visualizes the results of the regression: For the found optimal regression line (according to above given function for maximization) and the data samples (black circles) the x-axis gives the time relative from the first bookmarking action recorded in the system in days while the y-axis gives the normalized cumulative number of users. The weighted R^2 for the regression using the samples is visualized with triangles. As can be seen easily the maximum occurs for the weighted at the very first samples on day 519 after the 49th user.

3.2.2 Del.icio.us

While the Bibsonomy data-set originates directly from the bibsonomy web site, del.icio.us is a secondary data source. We assume that the number of users that bookmarked a specific site on del.icio.us is an estimate for the overall quantity of potential users of a website. So while it does not reflect the actual number of users it estimates the potential users out of the huge del.icio.us user community. Data aggregated from the del.icio.us portal includes 6 different resources bookmarked by the del.icio.us users. The resources have been bookmarked a different number of times. All 6 data sets have been cleaned in the same way as described for the Bibsonomy data set.

Table 1. Found values for the 6 data sets aggregated from del.icio.us

Resource	**Sample size**	**Critical *t***	**Critical *u***
YouTube.com	27,200	50.7	26
MyVideo.de	481	129.8	44
W3schools.com	14,402	699.0	88
Last.fm	22,262	364.7	49
Google Video	5,831	154.6	481
ColorSchemeGenerator	14,565	237.0	26

In Table 1 the data sets of the analyzed resources and their sample size with approximated critical time and users are given.

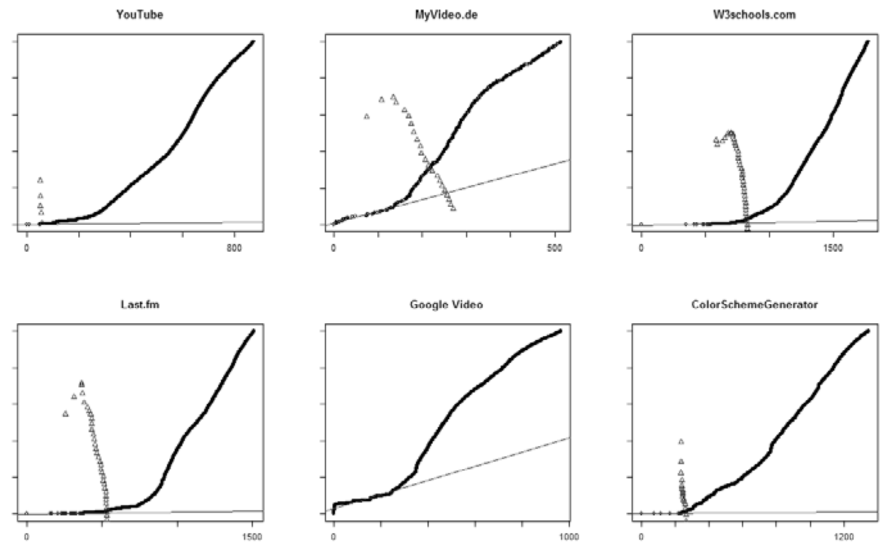


Fig. 4. Regression on the 6 selected resources from del.icio.us. The graphs show the data from YouTube, MyVideo.de, w3schools.com, Last.fm, Google Video and ColorSchemeGenerator (from left to right). The x axis gives the time in days and the y axis the relative number of users (from 0 to 1).

3.3 Conclusions from Experiments

The experiments with the Bibsonomy data set as well as the del.icio.us data sets show promising results: With linear regression a first 'linear' phase can be identified and a critical point can be found. While the visualization of the results indicates a good fit the number of users at the critical time point is surprisingly small. Either the number of users needed to enter the propagation phase is rather low or the assumption that the aggregated data sets from del.icio.us approximate the number of users is wrong. The Bibsonomy data set however supports the first theory. The only outlier with a rather big number of users at the critical point – Google Video – can be explained easily: Google Video was announced surprisingly with a big echo from the media. Therefore a lot of users bookmarked this site at the very beginning resulting in the step at the start of the sample. Note also that for the first and the last resource of the del.icio.us data a maximum R^2 was found at user 26, which is right at the start. In case of the del.icio.us data this indicates that the heuristic to start with 25 has a disadvantage. While only a few del.icio.us users were interested at the beginning the actual site might have had a much bigger user community. The sparseness of del.icio.us bookmarks at the beginning of the sample is thus responsible for the early criticality.

4 Overall Conclusions and Future Work

We have described the "Online Crowds" model in context of self organized systems. Based on the model a mathematical approach for finding self organized criticality (SOC) and the start of the propagation phase was presented. For the presented experiments the approach worked fine although the number of users in the initiation phase was surprisingly small. This supports the SOT approach that small changes can have big effects, respectively only a few additional users can change growth dramatically. Hence an interesting research question would be if there is such a minimum quantity of critical mass and how the value of this actual minimum user count could be defined.

However we want to point out that OCM is a simplified approach to describe the main phases of some of the collective Internet phenomena we have seen in the last years. It does not cover all aspects of online success stories, but it helps to understand the combination of social, psychological and non-technical factors of such social online contagion processes.

For future work we intend to concentrate on real world data sets and on the robustness of the approach, as well as its applicability for prediction of phase transitions. Furthermore we want improve the statistical model to better predict "Online Crowd"-readiness of emerging web information systems and will help to better understand unexpected user growth and how arising online mass phenomena can be supported or even created artificially.

References

1. Arthur, W.B.: Increasing Returns and Path Dependence in the Economy. University of Michigan Press, Ann Arbor (1994)
2. Banzhaf, W.: Overview of self-organzing systems in Science. In: Encyclopedia of Physical Science and Technology, 14th edn., pp. 589–598. Academic Press, New York (2002)
3. Bak, P., Tang, C., Wiesenfeld, K.: Self-Organized Criticality: An Explanation of 1/f Noise. Physical Review Letters 59, 381–384 (1987)
4. Barabási, A.L., Bonabeau, E.: Scale-free networks. Scientific American 288, 60–69 (2003)
5. Beckenkamp, M.: The Herd Moves? MPI Collective Goods Preprint No. 14 (2006)
6. Bikhchandani, S., Hirshleifer, D., Welch, I.: Learning from the Behavior of Others: Conformity, Fads, and Information Cascades. Journal of Economic Perspectives 12(3), 151–170 (1998)
7. Bonabeau, E.: The Perils of the Imitation Age. Harvard Business Review, 49–54 (2004)
8. Bughin, J.R.: How the companies can make the most of user-generated content. The McKinsey Quarterly (2007)
9. Celen, B., Kariv, S.: Observational learning under imperfect information. Games and Economic Behavior 47(1), 72–86 (2004)
10. Colman, A.: A Dictionary of Psychology. Oxford University Press, Oxford (2001)
11. Fuchs, C.: The Self-Organization of Virtual Communities. Journal of New Communications Research 1(1), 29–68 (2006)
12. Gladwell, M.: The Tipping Point: How Little Things Can Make a Big Difference. Back Bay Books, Boston (2002)

13. Golder, S.A., Huberman, B.A.: The Structure of Collaborative Tagging Systems. Journal of Information Science 32(2), 198–208 (2006)
14. Gross, R., Acquisti, A., Heinz, I.H.J.: Information revelation and privacy in online social networks. In: ACM Workshop on Privacy in the Electronic Society 2005, pp. 71–80. ACM Press, New York (2005)
15. Heylighen, F.: The Science of Self-organization and Adaptivity. In: Kiel, L.D. (ed.) Knowledge Management, Organizational Intelligence and Learning, and Complexity, Eolss Publishers, Oxford (2001)
16. Knowledge & Data Engineering Group, University of Kassel: Benchmark Folksonomy Data from BibSonomy. Kassel (2006)
17. Katz, M.L., Shapiro, C.: Network Externalities, Competition, and Compatibility. The American Economic Review 75(3), 424–440 (1985)
18. Kurihara, S., Sugawara, T., Onai, R.: Self-organization based on nonlinear nonequilibrium. Dynamics of autonomous agents. Artificial Life and Robotics 2(3), 102–107 (1998)
19. Ossimitz, G., Lapp, C.: Das Metanoia Prinzip – Eine Einführung in systemgerechtes Denken und Handeln. Franzbecker Verlag, Hildesheim (2006)
20. O'Reilly, T.: What Is Web 2.0. Design Patterns and Business Models for the Next Generation of Software (2005), `http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html`
21. Mika, P.: Ontologies Are Us: A Unified Model of Social Networks and Semantics. In: International Semantic Web Conference, pp. 522–536. Springer, Berlin (2005)
22. Nielsen, J.: Participation Inequality: Encouraging More Users to Contribute. Jakob Nielsen's Alertbox,
`http://www.useit.com/alertbox/participation_inequality.html`
23. Nonnecke, B., Preece, J.: Silent participants: Getting to know lurkers better. In: Fisher, D., Lueg, C. (eds.) From Usenet to Co Webs: Interacting with social information spaces, pp. 110–132. Springer, Berlin (2003)
24. Pressley, L.: Using Social Software for Business Communication. Technical Report, Lauren Pressley (2006)
25. Prigogine, I.: Time, Structure and Fluctuations. Science 201(4358), 777–785 (1977)
26. Rheingold, H.: Smart Mobs: The Next Social Revolution. Perseus Books, New York (2003)
27. Russ, C.: Online Crowds - Extraordinary mass behavior on the Internet. In: Tochtermann, K., Maurer, H. (eds.) Proceedings of i-Media 2007, Graz, pp. 65–76 (2007)
28. Schelling, T.C.: Micromotives and Macrobehavior. W. W. Norton, New York (1978)
29. Watts, D.J.: The "New" Science of Networks. Annual Review of Sociology 30, 243–270 (2004)

Towards Prototyping-Based Technology for Adaptive Software Development

Mykola Tkachuk, Andriy Zemlyanoy, and Rustam Gamzayev

Computed-Aided Management Systems Department,
National Technical University "Kharkiv Polytechnic Institute",
21 Frunze str., Kharkiv, Ukraine
tka@kpi.kharkov.ua, {andrey_zemlyanoy,gamzaev}@mail.ru

Abstract. An adaptive process of component-based software development implies permanent changes on underlying components and system architecture with respect to appropriate changes on system requirements. Efficiency of a development process depends on how close the changed software meets new requirement. In the paper we present a new design approach for adaptive software development which is based on prototyping technology. This approach integrates models and technologies which provide the processes of system prototypes building, configuring, execution and estimation, with respect to main software design goal: to reduce development costs and needed resources.

Keywords: Adaptive approach, prototyping, component-based software, simulation, development costs.

1 Introduction: Research Actuality and Aims

Recent trends in the domain of software systems development show the advent of agile-oriented design methods and component-based technologies. Besides that in many cases the software system has to be developed in context of some existing legacy applications, external information resources, and communication infrastructures, which definitely are the subjects of constant change. Therefore these systems must be able to cope with constant changes in system requirements (SR) during their design and operation. In other words, an advanced software system has to be adaptable and adaptive. Within this paper, we present a framework to solve this problem. This framework is based on our experiences collected and analyzed as a result of performing some projects for Ukrainian gas-and-oil transport and production enterprises [1].

Basing on our own collected project experience, and as a result of analyzing some modern trends in software engineering, we have elaborated the adaptive prototyping framework (APF) for multi-level distributed software systems which we would like to present in this paper.

The paper is structured as follows: *Section 2* outlines very briefly some modern trends in this research domain, in *Section 3* our vision for APF is outlined, which is based on the information meta-space metaphor. The short overview of the data models, methods and software tools elaborated for APF-toolkit is given in *Section 4*, and some experimental results are also discussed. Finally *Section 5* concludes with a short outlook on future work to be done within that field.

2 Adaptive Prototyping Design: Some Modern Trends

Prototyping approach is already well-known technology used in the software industry to agree requirement with the customer and to receive vision of the future system, and to receive feedback from the customers as well, especially it is good to use prototyping when SR are uncertain. There are few main types of prototyping technology: *throwaway* or *rapid prototyping*, *evolutionary prototyping* and *incremental prototyping*. Throwaway prototyping is used to reduce requirements risk; model will be discarded rather then become a part of the final system. The main goal to use evolutionary prototyping is to build a very robust prototype in a structured manner and constantly refine it. When developing a system using evolutionary prototyping, an appropriate software is continually refined and rebuilt. This technique allows development team to add new system features, or make changes that couldn't be conceived during the SR engineering and design phase. Evolutionary prototyping have an advantage over throwaway prototyping in that its prototypes are functional systems. Although they may not have all the features the users have planned, they may be used on an interim basis until the final system is delivered.

There are some tools to define prototypes available. One of the first of them was the Computer-Aided Prototyping System (CAPS) has been developed at the Naval Postgraduate School [2] for rapid prototyping of hard real-time embedded software systems, such as missile guidance systems, space shuttle avionics systems, etc. It used evolutionary approach of software prototyping and was based on Prototype System Description Language (PSDL). It is used for requirements analysis, feasibility studies, and the design of large embedded systems. PSDL has facilities for recording and enforcing timing constraints, and for modeling the control aspects of real-time systems using nonprocedural control constraints, operator abstractions, and data abstractions.

The modern SLAM system [3] allows for an effective use of formal methods in rapid application development (RAD) and other prototyping processes. This system includes an expressive object oriented specification language and development environment that, among other features, is able to generate efficient and readable code in a high level object oriented language (Java, C++, etc.). SLAM is able to generate prototypes that can be used to validate the requirements with the user. The additional advantage is that the prototype is not throwaway because most part of the generated code can be directly used and the other part can be optimized with the additional help of assertions automatically included.

Ripple system [4] has separate multilevel abstractions, including process-oriented level: the objects of the language are processes and operators that link processes to achieve some form of control, function-oriented level, and implementation-oriented level. Ripple is formally defined using algebraic semantics. The formal definition of the three abstraction levels allows transformation among the different levels. There are three different sub-languages corresponding to the three abstractions levels and are all textual languages.

Axure RP Pro [5] is software product that enables application designers to create wire-frames, flow diagrams, prototypes, and specifications for applications and web sites; especially it allows developing user interface rapidly without writing source code.

iRise Studio [6] is an application authoring tool used by business analysts, usability professionals, and project managers to quickly define and visualize business software. iRise Studio can be used to create fully interactive, high definition replicas of the target application that stakeholders actually delight in using. iRise simulations are easy to assemble for non-programmers.

All these systems are mostly used for prototyping of functional SR. CAPS is used for real-time systems prototyping, and Axure RP Pro and iRise Studio used for Web Application workflow and UI prototyping, allowing to build UI in a short time.

3 Adaptive Prototyping Framework Metaphor

Our concept considers all APF-activities as the adaptive control scheme including some metrics and estimation models in its feedback loop. All these models are integrated in the form of the multi-dimensional information environment, which is a part of the multi-dimensional information space consisting of 4 local information spaces [7]: 1) *P1-space* for system architecture modeling; 2) *P2-space* for SR traceability; 3) *P3-space* for system design methods and project solutions representation; 4) *P4 – space* for software evaluation models and metrics. All these spaces *P1-P4* are connected via some *system trajectory*, which represents the appropriate designing variant for PCS to be elaborated. APF is based on spaces P2, P3, P4.

The space *P2* is built basing on 3 criteria for SR estimation, which are invariant regarding any application domain, namely: *Fullness of SR specifications* - it defines the degree of requirements completeness in some project, *Degree of SR formalization* - it indicates the formalizing degree of given SR*, Measure of agreement* - it shows to which level the stakeholders (domain experts, analysts, programmers, etc.) are agreed from their points of view to SR in project considered. Using fuzzy logic approach the new method for SR evaluation is proposed that improves efficiency their analyzing [8]. The space *P3* to be a space for decision making for *Project Solutions* which have to be established taking into account some *System Property* to be implemented using an appropriate *Design Method*. In particular, *P3* includes the catalog of design patterns for scalable system networking architecture. Finally the space *P4* provides a collection of modeling technologies and metrics for system performance and reliability evaluation.

Taking into account the information meta-space metaphor [7] the APF has the following workflow (Fig.1). In *P2*-space each system conflict can be represented as fuzzy description of an appropriate SR and using the methods elaborated the alternative description for SR has to be met, which belongs to so-called acceptable area. Basing on the alternative fuzzy specification of SR its non-fuzzy value (e.g., using Mamdany method) is generated, and it constitutes as some system property to be met. Basing on this one the appropriate design method in *P3*-space can be chosen, which produces a collection of possible project solutions. In *P4*-space for each project solution an estimation procedure for such system characteristics as performance, reliability, etc. is provided: If testing results are not satisfied, we return to the previous step and generate corrected project solutions. The achieved system design version is documented and if other design criteria (financial, temporal, etc.) are satisfied, then designing process can to be finished. Otherwise, we should return to the first step of the workflow.

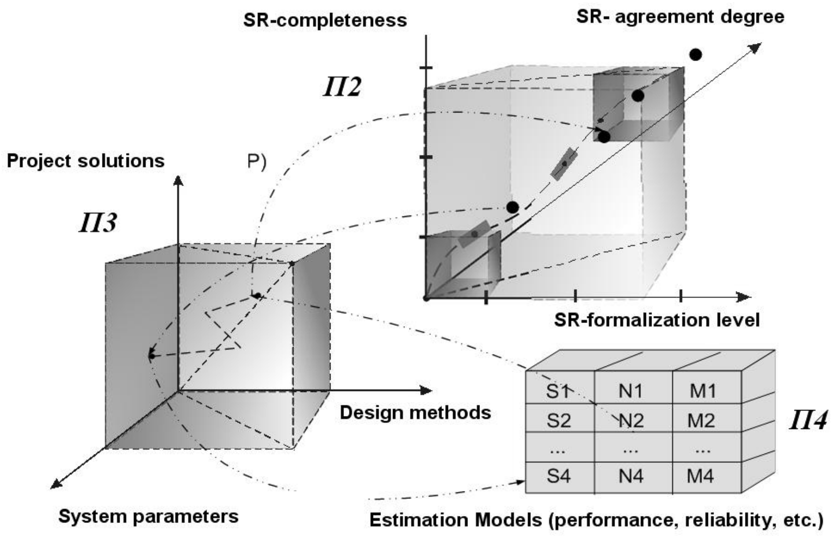


Fig. 1. APF's metaphor

In order to support APF-approach the appropriate data models, methods and software tools have to be elaborated.

4 Models and Tools for the Prototyping-Based Technology for Adaptive Software Development

The local spaces *P2-P4* from the technological point of view are represented in APF-approach as a collection of appropriate data models, algorithms and software tools.

The *P3*-space provides a collection of project solutions for PCS architecture, which can be implemented as reusable software components. On Fig. 2 one of such component-based software solution (CBSS) is presented, which realized distributed multi-level PCS for real-time data processing w.r.t. to some programmable logical controllers (PLC) configuration. There are the following typical software components elaborated and running together in this CBSS, namely: *Data Exchange Server* (DES) including 2 sub-components *XProtocol* and *TechXObject*; *Data Visualizing Service* (DVS), *Data Archiving Service* (DAS), *Integrated Database* (IDB). For their implementation such technologies and programming tools as MS COM/DCOM, C++, MS IIS, MS IE and MS SQL Server were used.

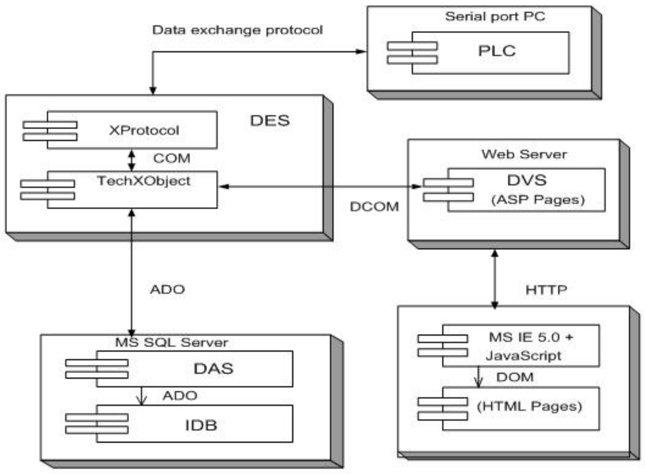


Fig. 2. CBSS for distributed Web-based PCS

The *P4*-space includes models and algorithms supposed to prove the CBSS w.r.t. their non-functional SR. This toolkit consists of a modeling environment and a set of mod-els which can be executed within the environment. The environment is designed to support various types of models: metric models, simulation model and prototyping models.

Simulation models play considerable role for the investigation of CBSS functioning, first of all, for their performance and reliability (see in [9, 10] for more details). Simulation modeling approach was chosen as the most powerful tool especially for systems with complex structure and dynamic behavior where strong mathematical methods are not adequate. The elaborated toolkit can be presented as the integrated software package, which includes both the real PCS components to be tested and the CBSS-models with the appropriate simulation CASE-environment as well. Simulation modeling of CBSS allows us to investigate and to adapt the appropriate target PCS - features for new conditions to be fulfilled by their real-life operating. Alongside with the advantages of metrics and simulation models, they have a number of shortcomings as this type of modeling does not approach system implementation and under certain circumstances may be not accurate enough.

As a decision for such cases we introduce an integrated technology of evolutional prototyping (ITEP) for CBSS. The technology is designed for rapid investigation of non-functional properties of CBSS and their agreement with system's non-functional requirements. ITEP uses existing and ready-to-assemble components to implement the designed system after successful prototyping. ITEP introduces the following concept. Non-functional requirements are mapped to target system properties, or non-functional constraints. The system must comply with the constraints to be acceptable by customer. The structure of a prototype is built using one of existing architectural patterns so that its components comply with provided functional requirement. This is an iteration process which finishes when a prototype with the target properties is found. This prototype serves as a base for the real system implementation. Structurally ITEP consists of a number of logical blocks. Each of them is designed to provide specific functionality on every iteration of the process. The blocks are PREPARE, BUILD, CONFIG, EXECUTE, ANALYSE.

The first block PREPARE is designed to perform conversion from non-functional requirements to a target region in the property space. In most cases this is an informal operation and can be performed with the help of requirements engineering methods. This step is quite informal as in the most cases the requirements are described using natural languages, which are ambiguous, context-dependent and informal.

Block BUILD is proposed to build the structure of the prototype. The structure is selected from a set of designed patterns with respect to the system properties generated on the previous step. This process can be partially formalized with the help of heuristic search. Each component has properties of different types: constant properties (that depend on implementation and cannot be changed), configurable properties and unknown properties. The problem of selecting initial components is that constant properties of every component must already meet the non-functional requirements and that the whole system must be able to meet the initial functional requirement.

As each component of the structure has configurable properties, block CONFIG is designed to apply a desired configuration to the structure found on the previous step. More specifically, every configurable property must get its value. The acceptable value domains are defined by the target properties. As there may be a number of different possible configurations there is a need of applying methods of parametric search.

Once the prototype is configured it is installed into an executing environment provided by the EXECUTE block. It includes a number of profiling and data acquisition tool supposed to collect and aggregate information required finding unknown properties of the system components. During execution this block uses methods of statistic analysis to find unknown properties of the executed prototype.

Unknown properties are the key point in the iteration process. As soon as unknown properties are found and belong to the region of target properties the process finishes and we can state that the solution is found. The comparison is performed by ANALYSE block. It uses internal criteria for the comparison. Due to complexity of comparison criteria formalization it uses methods of fuzzy logic for the comparison.

Due to limitations of the paper size we will present a simplified mathematical model of the processes taking place in the EXECUTE and ANALYSE blocks. Let us suppose $U = \{u_i\}$ is a set of unknown properties available for testing. $P = \{p_k\}$ is a set of selected properties for a certain testing session ($P \subset U$). During prototype execution we gather point values v_i^k for each parameter k: $\forall p_k \in P : \exists V^k = \{v_i^k\}$. Point values of all parameters can be represented as a matrix V:

$$V = \begin{bmatrix} v_1^1 & v_2^1 & \ldots \\ v_1^2 & v_2^2 & \ldots \\ \ldots & \ldots & \ldots \end{bmatrix}$$

For property analysis we have a set of analysis methods $M = \{m_i\}$. Matrix $MP = \{mp_k^i\}$ defines which methods can be applied for an unknown property analysis. mp_k^i is 1 method k is applicable for parameter i and 0 if not. After calculations we will obtain a matrix of results $R = \{r_k^i\}$, where r_k^i is a result of calculations for property i using method k: $r_k^i = F(V^k, mp_k^i, m_i)$. The complete workflow is shown on Fig.3.

The general scheme of ITEP blocks is shown on Fig.4.

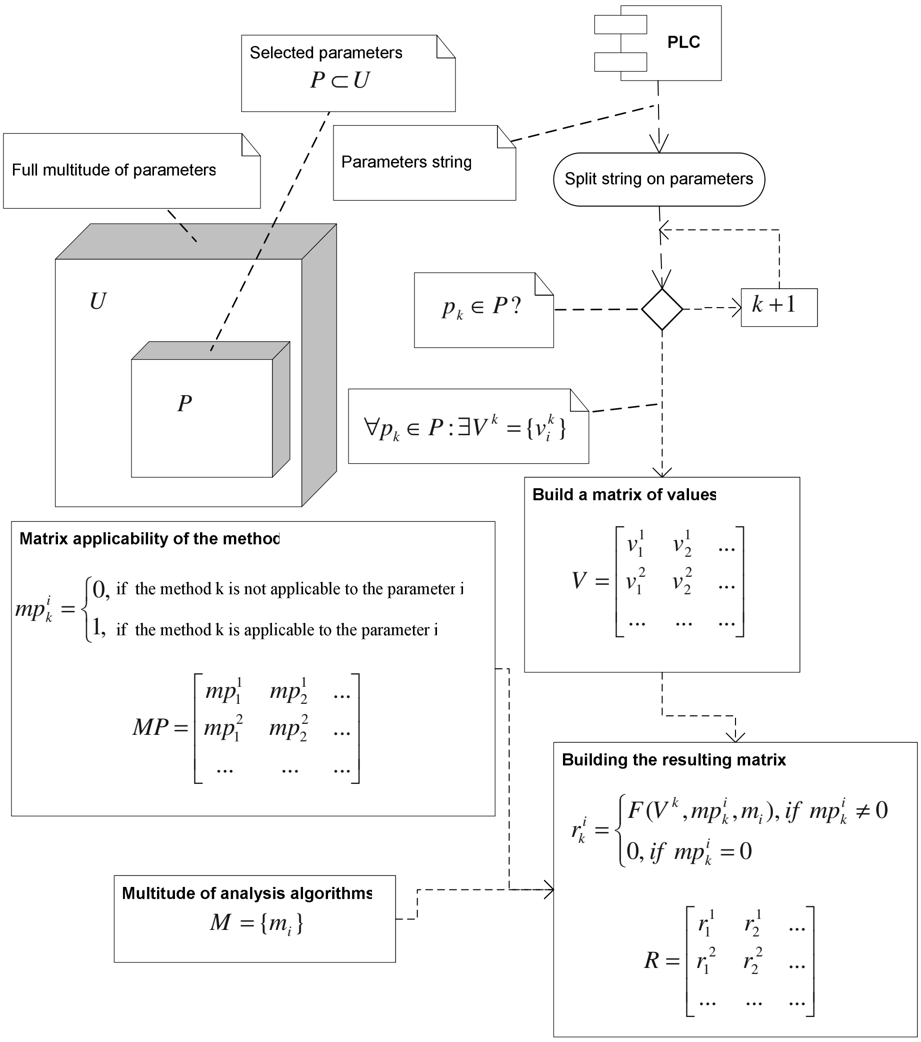

PLC
Selected parameters
$P \subset U$
Parameters string
Split string on parameters
Full multitude of parameters
U
P
$p_k \in P?$
$k+1$
$\forall p_k \in P : \exists V^k = \{v_i^k\}$
Build a matrix of values
$V = \begin{bmatrix} v_1^1 & v_2^1 & \dots \\ v_1^2 & v_2^2 & \dots \\ \dots & \dots & \dots \end{bmatrix}$
Matrix applicability of the methoc
$mp_k^i = \begin{cases} 0, & \text{if the method k is not applicable to the parameter i} \\ 1, & \text{if the method k is applicable to the parameter i} \end{cases}$
$MP = \begin{bmatrix} mp_1^1 & mp_2^1 & \dots \\ mp_1^2 & mp_2^2 & \dots \\ \dots & \dots & \dots \end{bmatrix}$
Building the resulting matrix
$r_k^i = \begin{cases} F(V^k, mp_k^i, m_i), if\ mp_k^i \neq 0 \\ 0, if\ mp_k^i = 0 \end{cases}$
$R = \begin{bmatrix} r_1^1 & r_2^1 & \dots \\ r_1^2 & r_2^2 & \dots \\ \dots & \dots & \dots \end{bmatrix}$
Multitude of analysis algorithms
$M = \{m_i\}$

Fig. 3. Formal model for EXECUTE and ANALYSE blocks

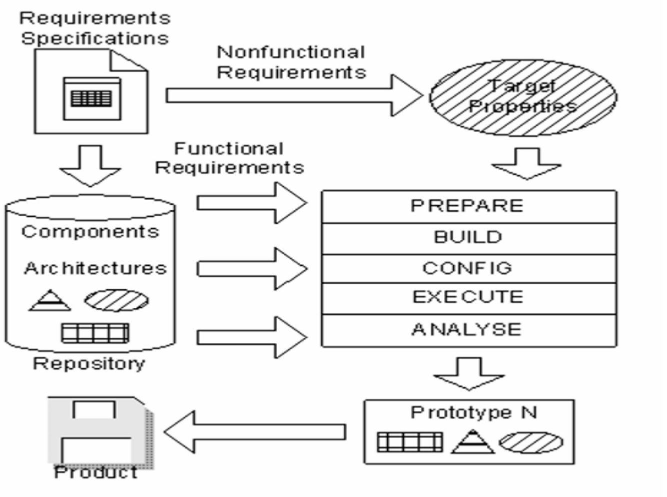
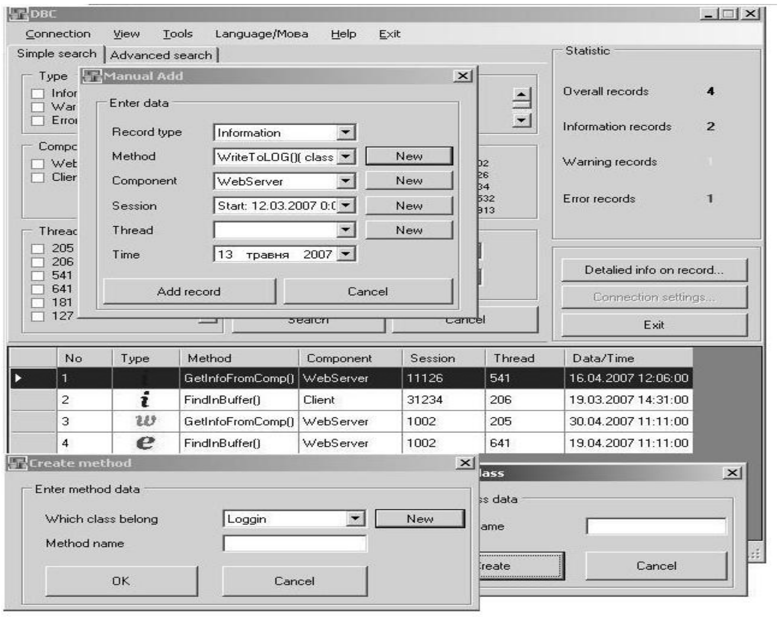


Fig. 4. General scheme of ITEP

As an implementation of the prototyping environment there was developed anumber of software tools. Each tool is designed to support certain blocks from the prototyping environment - PREPARE, BUILD, CONFIG, EXECUTE, ANALYSE. Sample visual forms of a tool for EXECUTE and ANALYSE blocks are shown on the Fig. 5.


DBC
Connection View Tools Language/Мова Help Exit
Simple search
Advanced search
Detalied info on record...
Connection settings...
Exit

Manual Add

Enter data

Record type	Information
Method	WriteToLOG()(class
Component	WebServer
Session	Start: 12.03.2007 0:(
Thread	
Time	13 травня 2007

Add record · Cancel

Statistic

Overall records	4
Information records	2
Warning records	1
Error records	1

No	Type	Method	Component	Session	Thread	Data/Time
1	i	GetInfoFromComp()	WebServer	11126	541	16.04.2007 12:06:00
2	i	FindInBuffer()	Client	31234	206	19.03.2007 14:31:00
3	w	GetInfoFromComp()	WebServer	1002	205	30.04.2007 11:11:00
4	e	FindInBuffer()	WebServer	1002	641	19.04.2007 11:11:00

Create method

Enter method data

Which class belong	Loggin
Method name	

OK · Cancel

Fig. 5. Software tool for EXECUTE block

The resulting prototype complies with initial system requirements and can be used as a base for implementation of the system. This makes the development process more efficient because of reuse of existing components with known properties. However, the degree of reuse often depends on the number of modifications that should be applied to components during the real system development. The modifications may change the properties of the system so it does not meet the given requirements. This technology allows to use existing structure templates and does not generate new templates in case if none of existing ones meets the system requirement. These shortcomings define the way of technology development and refinement.

Described technologies allow us to define and execute different kinds of CBSS models in order to investigate various properties or operating characteristics of CBSS without need to develop the solution itself. For example, we can investigate configurations with various numbers of client components in an appropriate CBSS, and find estimates for such their operating characteristics as the average response time and the request / response rate by client-server interaction in CBSS. As simulation is a type of statistical modeling the results ate stochastic in a certain degree and their quality should be estimated with the help of statistical analysis. The same operating characteristics of CBSS could be found using prototyping as well. The results in this case would also have stochastic properties and should have estimates for their quality. In both cases the quality of results depends not only on the investigation technology, it depends on the system being investigated as well. The results quality may differ depending on properties of the system components, number of the components, connections among them and so on. The results quality shows how good the selected modeling technology is for a specific investigated system and thus it should be considered as a criterion of modeling technology efficiency.

Another efficiency criterion for modeling technologies is based on costs required for the full cycle of system investigation. The costs include such components as costs for the technology development or acquisition, costs for elaboration of an appropriate model using the technology, costs for the modeling process and result analysis: $S_i\,(c) = S_i^{(1)}(c) + S_i^{(2)}(c) + S_i^{(3)}(c) + S_i^{(4)}(c)$. Each of the technologies assumes different complete costs depending on investigated system complexity and the dependencies are generally not linear. It means that for a given system complexity (C) it is possible to find the most efficient investigation technology (i) which implies the lowest costs (S) for the full cycle of system investigation. The cost trends example is shown on the Fig.6.

There is no common approach to define all cost components in the say way because they are significantly different by the physical nature. However, we can define the first cost component, i.e. the costs for technology development using the well-known COCOMO II model. The model allows us to find costs for technology development at two levels. The first level represents the early design model and allows to find person-month (PM) costs as $PM = (NOP * (1 - \mathrm{Re}use)/100)/\mathrm{PROD}$, where NOP is the number of object points, Reuse is the degree of code reuse, PROD is productivity. The next level represents the post-architecture model and allows to refine the estimates as $\mathrm{PM} = \prod_{i=1}^{7} P_i * A * Size^{0{,}91+0{,}01\sum_{j=1}^{5} SF_j}$, where P_i are cost attributes, A – a constant, Size – the size of the software in object points, SF_i – scalability attributes.

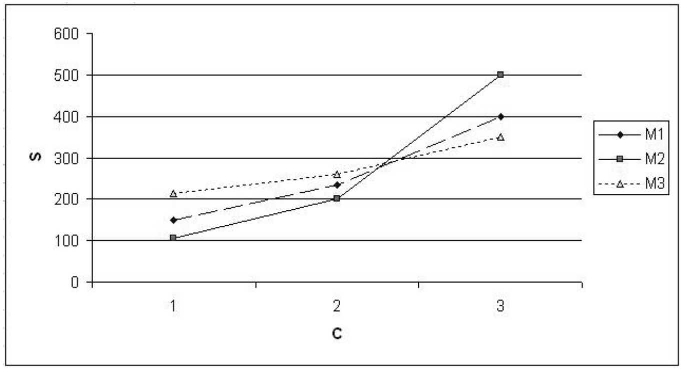


Fig. 6. Cost trends for investigation technologies depending on investigated system complexity

As long as the technologies may have different efficiency for different systems being investigated we need compare different technologies using the defined criteria and choose one of them for investigation. In every particular case the criteria values will vary depending on technology and complexity of the system being investigated. Below is shown the updated modeling process workflow with regards to the step of choosing an appropriate modeling technology.

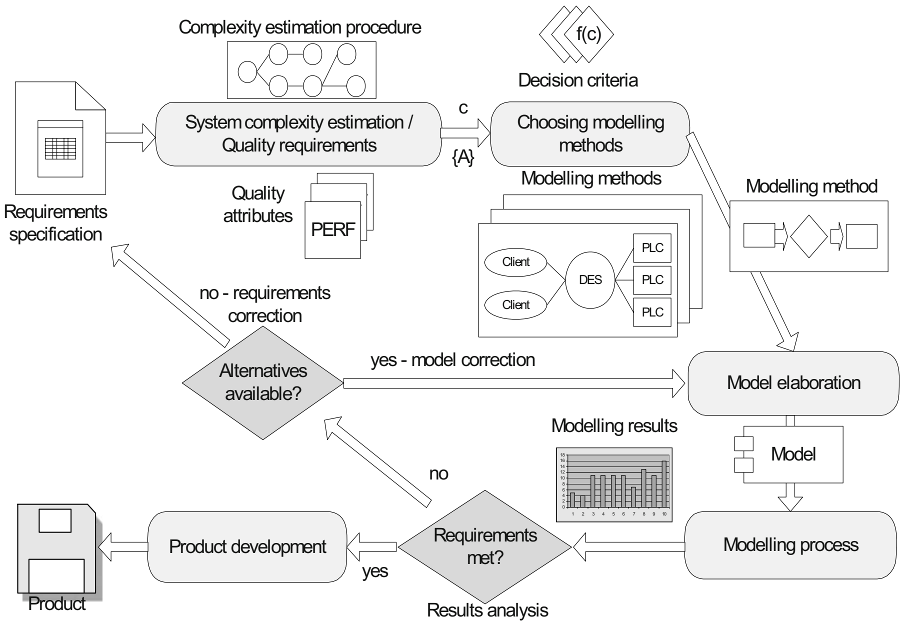


Fig. 7. The general scheme of software modeling within the APF approach

The scheme includes two important steps which precede the system investigation: 1) system estimation from the point of view of its configuration and complexity, 2) choosing modeling technology depending on selected criteria and estimated system complexity. This allows to use efficient technologies for system investigation taking into account the process costs and the quality of results.

5 Conclusions and Future Work

We have presented the framework for adaptive software prototyping in multi-level distributed PCS, which can help to reduce development costs and ensure required software quality by providing reusable, scalable and efficient software solutions. We are going to extend the adaptive prototyping framework in the way of refining the prototyping technology. Finally, it is important to elaborate a methodic for development costs estimation by different system complexity types which can be used for comparison of prototyping and simulation alternative approaches.

Acknowledgments. We would like to thank our students Olga Oliynyk and Dmitry Tertyschnyk for their efforts by software implementation in this project.

References

1. Tkachuk, N., Mayr, H.C., Kuklenko, D., Godlevsky, M.: Web-based Process Control Systems: Architectural patterns, Data Models, and Services. In: Shafazand, H., Tjoa, A.M. (eds.) EurAsia-ICT 2002. LNCS, vol. 2510, pp. 721–729. Springer, Heidelberg (2002)
2. Luqi, R., Steigerwald, G., Hughes, V., Berzin, A.: CAPS as a Requirements Engineering Tool. In: Proc. Conference on TRI-Ada, San Jose, CA, pp. 75–83 (1991)
3. Herranz, A., Moreno-Navarro, J.: Rapid Prototyping and Incremental Evolution Using SLAM. In: Proc. 14th IEEE International Workshop on Rapid Systems Prototyping, pp. 201–209. IEEE Computer Society, San Diego (2003)
4. Zhang, K., Song, G., Kong, J.: Rapid Software Prototyping Using Visual Language Techniques. In: 15th IEEE International Workshop on Rapid System Prototyping, pp. 119–126. EEE Computer Society, Geneva (2004)
5. Axure Software Solutions, `http://www.axure.com/products.aspx`
6. iRise Company, `http://www.irise.com`
7. Tkachuk, M.V., Zemlyaniy, A.O., Al-Hassanie, Z., Polkovnikov, S.V.: Adaptive Macro-Designing Framework for Distributed Multi-level Process Control Systems. In: Advanced Computer Systems and Frameworks, Proceedings of 3rd International Conference ACSN-2007, pp. 20–24. NTU "Lvivska Politechnika", Lviv (2007)
8. Tkachuk, M., Godlevsky, M., Zemlyanoy, A., Gamzayev, R.: Fuzzy-Logic Modeling Approach and Software Solutions for System Requirements Management. In: Proceedings, Series of the German Informatics Society (GI). LNI, vol. P-84, pp. 185–188, Printed in Bonn (2006)
9. Tkachuk, N., Goreliy, A., Zemlyaniy, A.: Complex of Simulation Models for Component-based Software Solutions. In: J. Computerized Control Systems and Devices, National University of Radio-Electronic, Kharkiv, vol. 18, pp. 145–152 (2004)
10. Tkachuk, N., Polkovnikov, S., Al-Hassanie, Z.: Multi-level Composed Framework for Simula tion and Estimation of Component-based Software Solutions. J. Radio-electronic and Computerized Systems, National Aerospace University, Kharkiv, vol. 4, pp. 93–99 (2006)

Management of Multi-services Structures Through an Access Control Framework

Luigi Ubezio[1], Enrico Valle[1], and Claudia Raibulet[2]

[1] ET.TS Srl,
Via Orio al Serio, 29, 24050 Grassobbio, Bergamo, Italy
{ubezio,valle}@etts.it
[2] Universita' degli Studi di Milano-Bicocca, DISCo – Dipartimento di Informatica Sistemistica e Comunicazione, Viale Sarca 336, Edificio 14, 20126 Milan, Italy
raibulet@disco.unimib.it

Abstract. This paper aims to present an architectural model designed to manage the information related to a multi-functional structure offering various types of services through an access control framework. It describes several implementation details of our prototype focusing on the emerging technologies based on radio frequency identification approaches exploited to reveal the presence of customers inside a multi-functional area.

1 Introduction

This paper presents a case study proposed and developed by ET.TS Company in cooperation with the Computer Science Department of the University of Milano-Bicocca in Italy. The two main motivations behind this case study are of two different natures: technological and social. The technological one is related to new emerging paradigms for identification and localization of objects through wireless approaches exploiting radio frequencies. These technologies have overcome the prototyping phase, being used today for enhanced software solutions. Actually, hardware based on RFID (Radio Frequency Identification) [3, 4, 8, 9] or WI-FI [2, 5] technologies is now able to identify and trace with good accuracy the presence of objects or humans inside relatively large areas or buildings.

From the social point of view, multi-functional areas such as sports complexes, fitness centers, malls, multi-cinemas, museums, are changing the way they are doing business. Two opposite aspects operating at different levels may characterize their behavior and lead their development and evolution:

- *unification*, which regards the geographical location of the multi-functional areas; for obvious reasons (i.e., nearby or commercial) different providers offering different types of services choose to operate in the same location, or even inside an existing multi-functional center to gain advantages of an already established and known business area; the location concept becomes of relevant importance for these systems being straightly related to the access rights of the customers to the common and restricted areas and, to the billing policies;

- *division*, which regards the types of services offered by the multi-functional areas; there are different providers offering different services in the same area, each of them managing her/his own services independently of the others.

In such a context characterized by independent service providers and shared location areas, new solutions are required to efficiently address issues related to the physical and logical organization of common and independent areas, owners of areas, customers and their related access rights information and/or billing applications.

The solution we propose in this paper is also characterized by unification and division: domain entities (i.e., customers, access rights, areas organization) are unified (because there is a common practice for customers to chose two or more services in the same multi-service area or for providers to make discounts for packages of services), while applications exploiting this information are independent of each other managing the common information based on their own interests.

The rest of the paper is organized as following. Section 2 introduces a motivating example for our case study. Section 3 describes the main components of the software architecture of our solution. The emerging technologies used in our solution together with the main implementation aspects are presented in Section 4. Conclusions, open issues and further work are dealt within Section 5.

2 Motivating Example

This motivating example may be common in the context of a company or a public structure where customers may enter various areas offering various services based on their access rights or the tickets paid. In this paper we consider a sports complex organized as in Fig. 1.

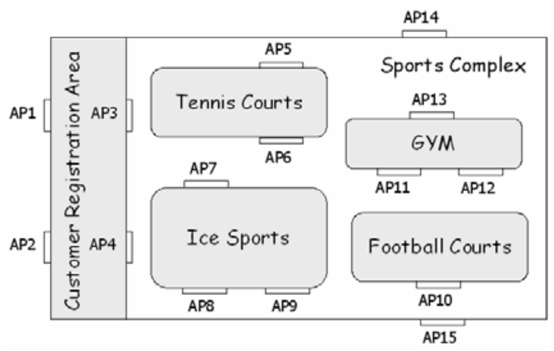


Fig. 1. Sports Complex Structure

The physical organization of the sports complex consists of five main areas: customer registration, tennis courts, gym, ice sports, and football. Each of these areas may be managed independently of each other. There are four access points from

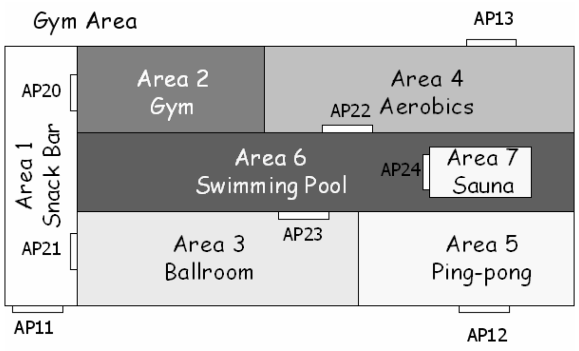


Fig. 2. Structure of the GYM Area

outside, two of them for the customer area where any person may enter, and two of them directly to the sports areas where access is restricted to the persons who are allowed to enter one of the internal areas. Each of the internal areas may have a complex structure. For example, the gym area is subdivided as shown in Fig. 2.

The gym area has three main access points and seven different areas. Each area may be managed independently of the others and has one owner. Two or more physical areas may be grouped together and form a logical area (i.e., Area6 and Area7 may form one logical area if all the customers having access to one of them have implicitly access to the other). Areas can be divided in various categories based on various criteria. For example, there are areas which have a direct access point from outside (e.g., Area1, Area4, and Area5), while there are areas which may be accessed only by passing through another area (e.g., Area2, Area3, Area6, and Area7). Or, there are areas where access is free (e.g., Area1, for all the persons allowed to enter the internal areas of the sports complex) and areas where access is restricted.

In this case study we have considered customers have an access tag through which their presence is revealed. The system notifies also the presence of persons with no tag.

The issues raised by such a system are related to the management of the following aspects:

- various access points for the same area (e.g., three access points in Area1, two access points in Area4);
- areas which are physically located in other areas (e.g., Area7 is inside Area6);
- the passage of the customers in the areas where they are not allowed to stay, but only to pass to reach their target area (e.g., to enter Area6, a customer has to cross Area4 or, Area1 and Area3);
- areas are managed independently; each area (logical or physical) has only one owner;
- customers rights, which may change every hour every day; for example, customers may be allowed to enter the complex in certain periods of time in certain days of the week (e.g., customers attending courses);
- customers who may enter the complex a given number of times (e.g., customers who have paid for ten entrances to the swimming pool);

- unusual situations: persons who are not allowed to enter an area, persons who does not have a tag, persons entering the area not in the scheduled timetable, and so on;.

3 Architectural Model

The architectural model we propose for multi-functional systems offering various types of services consists of five main components (see Fig. 3).

- *domain entities*, which define all the information about the multi-functional system such as the physical areas and their related access points, customers and their access rights; furthermore, we have introduced a particular entity called *application-dependent observable events*, which contains the events in which applications are particularly interested and which should be notified as soon as they are revealed; the implementation of this module exploits the publisher-subscriber pattern;
- a *presence detector* mechanism, which reveals the passage of a person from an outside to an inside area and vice-versa, as well as the passage from one area to another;
- an *access manager*, which acts as a middleware between applications and the underlying system; actually, it implements the business logic of the system; it receives the presence notifications from the presence detector and it stores them into a log module; based on the current configuration of the system and the current events in which applications are interested in, it notifies the last about the occurred events;
- a *configuration interface*, which enables an initial static configuration of a system (i.e., areas, customers) and a dynamic configuration of a system (i.e., especially access rights of customers);
- *applications*, which may be interested in various aspects of the system ranging from statistical information (i.e., how many people are currently in an area) to payment-related applications (i.e., payment of the services provided to the customers).

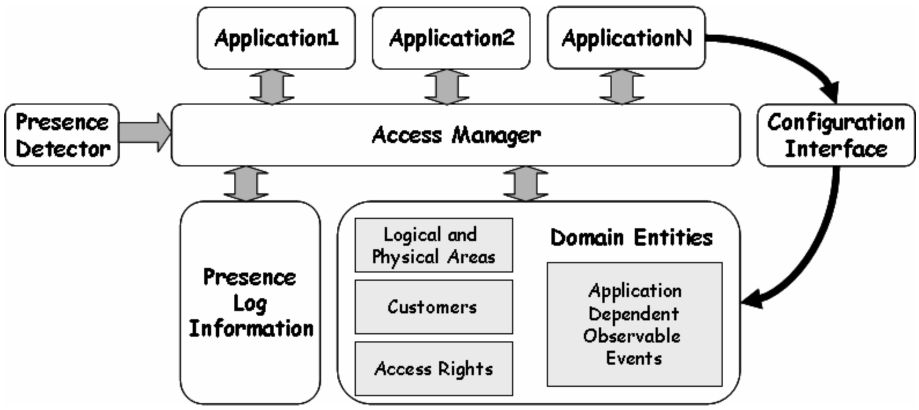


Fig. 3. Architectural Components

The system is able to detect the presence of any person who has or not a tag. For the persons who enter the structure without having a tag the system generates an error code specifying the areas which have been entered and/or left.

The system supports two types of configuration: static and dynamic. Static configuration is performed during the initialization of the system when areas, activities (i.e., courses) and customers are defined. Dynamic configuration may be done at any moment and regards the modification of the customers and their access rights.

3.1 Modeling Domain Entities

The domain entities and the relationship among them are shown in the UML class diagram in Fig. 4.

An area may be composed of other areas or it may be part of another one. Each area may have one or more logical entrance and exit access points. For example, in Fig. 2 the logical entrance and the logical exit points for Area2 are associated to the same physical unique access point. Or, Area1 has one logical entrance point composed of two physical access points AP20 and AP21, and one logical exit point associated to the physical AP11 access point. Another possible solution considers that Area1 has two different logical entrance points (AP20 and AP21) and one logical exit point (AP11). Note that a logical access connects two areas and it has associated also a crossing direction. A particular type of area is the external world: there are access points which connect the "outside" area to an area belonging to the sports complex or vice-versa.

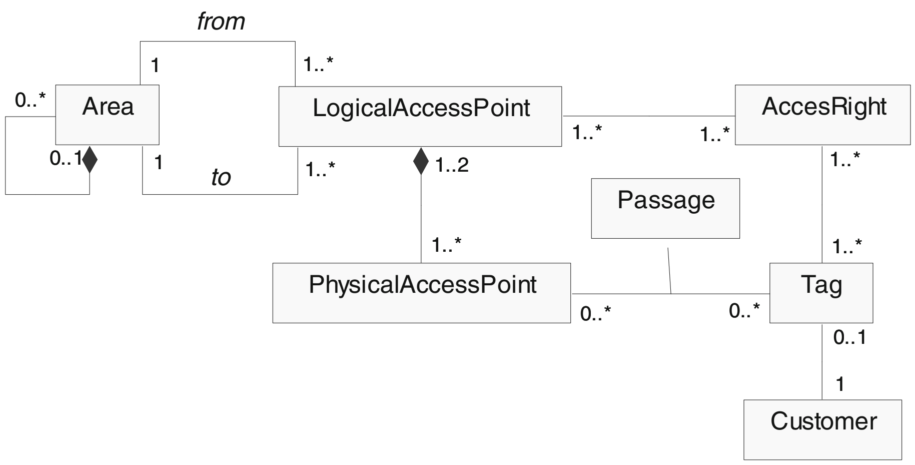


Fig. 4. Domain Entities

Registered customers have a tag through which their access rights for various logical access points are established. The Passage association class registers an actual passage of a customer through a physical access point.

3.2 The Business Logic

When a person passes through an access point, the presence detector creates an event which is sent to the access manager. If the connection between the presence detector and the access manager is corrupted, an occurred event is stored locally and sent to the access manager as soon as the connection becomes available.

An event contains the following information:

- the tag code of the person who has passed through the access point or an error code in case the person has no tag associated or the access point does not work correctly;
- the identification code of the physical access point;
- the passing-through direction: which represents the identification of two areas connected by the physical access point: a *from* area and a *to* area as show in Fig. 4;
- the recorded timestamp.

When the event information is received by the access manager it is stored in the log module by a dedicated thread called Recorder. In this way the information becomes immediately available for further computations. An autonomous thread within the access manager, called Checker, compares the information stored in the log module with the customers access rights stored in the domain entities. Considering that access rights can change dynamically, the Checker identifies for each event stored in the presence log which are the current access rights and, consequently, tags an event as following:

- *allowed access* if the user is recorded in the system and he/she has the right to pass through the related access point;
- *unknown user* if the user is not recorded in the system;
- *not allowed user* if the user is recorded in the system, but he/she does not have the rights to access the area;
- *elaboration error* if the system is not able to perform any elaboration on the event (e.g. the code of the physical access point is not associated to any area).

The applications can exploit the information stored within the presence log in two ways:

- by queries: when an application needs information related to the access history, it can directly interrogate the presence log with different criteria (e.g. an application can ask for a list of all the *not allowed user* tagged events in the last six hours);
- by a publish/subscribe mechanism: when an application is interested in monitoring continuously particular events, it asks the access manager for an immediate notification of the subscribed event (e.g. an application can subscribe for all *unknown user* and *not allowed user* events occurred for one or more access points).

In the first case the access manager answers to the application requests when they are issued, while in the second case an independent thread, called Notifier, notifies the applications of the occurred events.

The thread based implementation of the main functionalities of the access manager can be further exploited by additional dedicated strategies, which establish further policies in managing the information exchange and in providing applications with the information they are interested in. Currently, the implemented strategy works as following: the Recorder works on demand reacting to the events raised by the presence detector, while the Checker performs its operation every *t* minutes. When the Checker has finished, the Notifier looks in the presence log to notify the interested applications of the occurred events.

3.3 Static and Dynamic Configuration

Besides asking information about the occurred events, applications are also interested to perform modifications of the domain entities such as the change of the users' access rights or insertion/deletion of users.

Conceptually, domain entities can be changed either statically, when the systems is off-line, or dynamically, when the system is running. In the first case, modifications can be easily done. Any modification: of users and their access rights or of logical areas and access points is possible.

More challenging is the runtime configuration of the domain entities. To maintain consistency among the domain entities a Controller has been designed to check the system integrity after the application of any modification: if the data integrity cannot be maintained the requested changes are rejected.

It is possible to perform modifications for the following types of entities:

- *observable events*: in this case an application subscribes itself for the observation of one or more types of events that can occur in the system. The check that must be performed is related to the existence of the areas which should be observed;
- *access rights*: applications can require the modification of customers' access rights. In this case, the Controller checks the applicability of these rights. For example, if application A wants to assign a right to customer C to enter the area B, the Controller must check that:
    - the entities A (application), B (area), and C (customer) are defined in the system;
    - there is a set of access rights which allows the customer C to reach the access point to the area B, or more precisely that there is a path in the structure which allows customer C to reach the area B.
- *customers*: applications can require to add and remove a customer. In the case of removing a customer, the Controller must verify that the customer is not currently inside the multi-functional structure.

The modification of observable events can be requested by any application and every application has its list of observable events: in this way there is no overlapping of responsibilities. When the request to change the entities is related to access rights and/or customers a problem of coordination among applications arise. It is possible to identify two main cases:

- only one application is responsible to perform changes related to one area of the system. In this case other applications can only read the information related to access management. The applications can be notified of changes occurred in the structure leveraging the same mechanisms used for observable events;
- two or more applications are responsible of overlapped areas. This case could be faced by the introduction of moderation mechanisms among applications. The development of such mechanisms would be object of further investigation.

Currently, modifications related to physical and/or logical access points are statically performed.

3.4 Applications

Currently, there are two types of applications we have considered: billing and business intelligence.

Billing applications may exploit the entrances number of the customers in the structure. For example, customers pre-pay twenty entrances to aerobics. In this case, the access rights of the customer are restricted to the aerobics area and the billing application has to look into the Passage objects to count how many entrances has a customer to the aerobics area.

More complex applications may consider courses and timetables, and the access rights of customers may change every day every hour. For example, a customer has paid an aerobics course every Friday for six months. The access rights of this customer are related only to the aerobics and in addition have associated time constraints.

There are also cases in which accesses to an area are not considered for billing. They are mostly related to transit areas to reach the target one (customers of Area7 in Fig. 2).

Business intelligence applications are interested in unusual situations and/or statistical information. For example, it is useful to know if there are cases in which a tag enters consecutively two times an area without exiting it.

Statistical information are mostly related to how many people are inside an area, which are the areas with more people than their regular capacity, which are the most used access points, or how long do customer remain in a certain area. Such applications exploit all domain entities.

3.5 Implementation Details

The communication between the access manager and the presence detector is performed via a WI-FI communication protocol. The access manager gains information from the presence detector using virtual COMs. Domain entities, presence log information and the application-dependent observable events are stored in an open source MySQL database.

The communication between applications and access manager has been implemented via a proprietary protocol which uses an XML Schema to identify documents exchanged via XML-RPC. From this point of view, leveraging on XML-RPC, applications can be implemented in any language.

A critical point of this approach is related to the processing of events. From a formal point of view the way of processing can be considered an implementation issue (for example, events are processed one by one). To cope with this problem a process-grouping strategy has been inserted. The access manager has a buffer in which events, collected from the virtual COMs, are stored before they are processed. The strategy states the number of events which can be processed together by the access manager through two parameters:

- MAX, the maximum number of events which can be processed together. When this number is reached the access manager processes the occurred events;
- TIME, the interval of time in which, even if MAX is not reached, the available events must be processed

The aim of this strategy is to decrease the number of insert operations in the database. Note that parameters can be set in such a way that events can be processed singularly.

4 Technologies

Before choosing the suitable technology to develop the presence detection mechanism, it was necessary to define what actually was intended to be monitored. In this application, the basic expectation was to identify the presence of people passing through a chock point in a public structure where doors are 2.5 meters (8.20 ft) tall and 2 meters (6,56 ft) wide, while gates can be maximum 4 m (13.12 ft) tall and up to 6 meters (19,69 ft) wide. An important issue is that no physical controls could be associated to the chock point like magnetic door-locks or electrical bars and the presence identification should require no action other than passing through the supervised area.

The benchmark to select the best technology to match these requirements has considered the following possible candidates: Passive RFID [7], Active RFID [1], Active Wi-Fi [2], TDOA (Time Difference of Arrival) [10] localization, RSSI (Radio Signal Strength Identifier) [6] localization and Active Zig Bee [11] devices. The solution which suited best to our requests is a combination of Active and Passive RFID. The deep analysis of the feature of each technology is demanded to foreseeable works.

4.1 Detection System Architecture

Every authorized person receives a tag, a small device similar to a credit card, which is able to be excited by radio signals and to send back its own identifier. On each monitored gate is placed an antenna that works on the same radio frequency of the tag (see Fig. 5). The information coming from the tag is sent through the RS 485 serial protocol to the electronic control where it is decoded and either stored in the local memory or forward to the final application using a variety of possible protocols: RS 232 serial protocol, IEEE 802.11.b - Wi.Fi, or Ethernet.

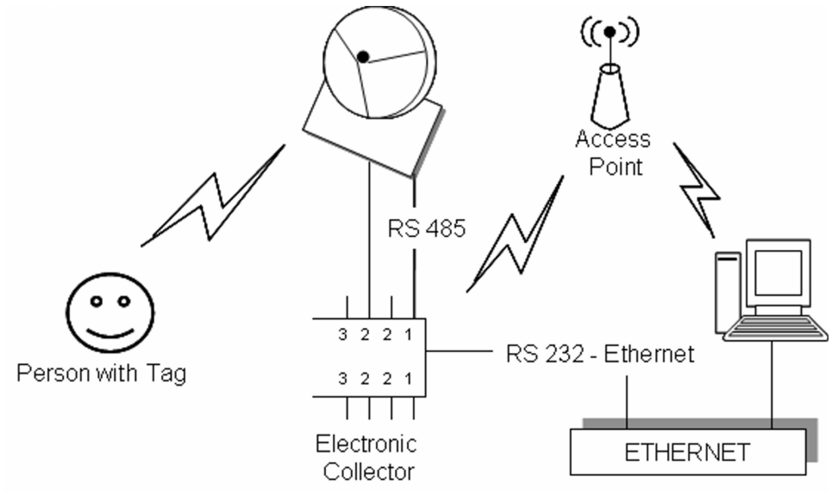


Fig. 5. Chock detection-basic architecture

The antenna works on the non-regulated radio frequency of 2.4 GHz, the same of the Wi.Fi. and ZigBee but it implements its own communication protocol. It is able to detect tags in a range of 6 meters and with a polling of 80 ms.

4.1.1 Tags

Tags are small devices containing sophisticated micro circuits able to receive the radio signal coming from the antenna and to send it back adding its own identifier. Tags can be classified in accordance to their power supply as following:

- *Active*: which integrate batteries and transceivers. The power provided by a battery helps to power the transceiver which properly modules the radio signal to be sent back to the reader. The active tag can broadcast radio signals up to 100 meters, much more than passive or semi-passive one. This kind of technology has two main limits: the cost of each single device is significant and because of its battery dependency it requires a continuous maintenance process:
- *Passive*: which reflect the radio signal coming from an antenna or a reader and send it back adding information by modulating the reflected signal. The efficiency of this device is strongly limited to a range between 2 and 20 cm due to the current European legislation;
- *Semi-Passive*: which represent a hybrid solution between the previous two. When the device is excited by the antenna radio beam, a tag sends back a signal powered by the internal battery. The efficiency of this device is up to 10 meters from the antenna position and a passage can be perceived at a speed of 400 km/h.

4.1.2 Multi-services Structure Tag Analysis

The decision of which tag to adopt for our system has been driven by the following two issues:

1. *choke point detection:* people should move independently everywhere inside the controlled areas without dealing with badge readers or other physical devices, and in the meanwhile the system should be able to trace their position and the authorization level. This problem can be easily solved placing an adequate number of antennas on every choke point between two service areas and providing every customer with a semi-passive tag;
2. *services activation:* in every area there are several services like: lockers, hairdryers, showers, etc. which should be activated by a personal control provided to the end user. This kind of services should be provided on demand.

In order to fulfill the requirements briefly listed above the best choice is a combination between a semi-passive and a pure passive tag. This kind of solution is commonly known as a *combo tag*, i.e. in a single device are located both the semi-passive circuit and the passive component.

5 Conclusions and Further Work

This paper has presented an architectural model for managing the information related to a complex structure offering various types of services through an access control framework. Furthermore, it has described several implementation details of our prototype focusing on the emerging technologies exploited to reveal the presence of customers inside a multi-functional area.

The two main advantages of our solution are: *separation of concerns* and *modularity*. The first one is related to the fact that the *information offered* (presence information) by the location and access control framework is separated from the *information used* (domain entities) by applications due to an efficient interaction mechanism among the various elements of our architecture. The second one refers to the separation of the responsibilities inside the system among the presence detector, the access manager, domain entities and applications, fact which enables the substitution or modification of one of them without influencing the others.

Furthermore, the proper design of the domain entities and the related mechanisms for their manipulation has allowed us to successfully address all the issues specified at the end of Section 2.

There are also open issues that should be further addressed:

- **a collaborative definition of the access rules: t**he current implementation of the multi-functional structure is based on a central authority which determines the various areas and the customers' access rights. In the real case this is not true. Every area should have a manager which describes its internal structure, its related activity and the customers' rights. A collaborative strategy should supervise the configuration of the structures, to notify inconsistencies and, ideally, to suggest an appropriate solution.
- **improvement of processing strategy:** the processing of events is currently optimized to reduce the interactions with the database containing the domain entities. It does not consider that there could be a relatively high number of transmitters (proportionally to the number of physical access points). This aspect would cause a critical processing point in the system.

- **implementation of long term statistical analysis:** currently, data collected in the domain entities database contains information related to one single day. Consequently, the statistical analysis offered to applications is based only on this short period of time. Usually, statistical analysis is performed on large number of data. Further work includes the design of a data warehouse system where every day data should be stored.

We kindly thank our colleagues from *ET.TS S.r.l* and *Roma Ricerche Consortium* for their valuable support during the development of this work.

References

1. Cho, H., Baek, Y.: Design and Implementation of an Active RFID System Platform. In: Proceedings of the International Symposium on Applications and the Internet Workshops, Phoenix, Arizona, USA, pp. 1–4. IEEE Computer Society, Los Alamitos (2006)
2. Ferro, E., Potorli, F.: Bluetooth and WI-FI wireless protocols: A survey and a comparison. IEEE Wireless Communications 12(1), 12–26 (2005)
3. Finkenzeller, K.: The RFID Handbook – Fundamentals and Applications in Contactless Smart Cards and Identification. Wiley & Sons LTD, Swadlincote (2003)
4. Jeong, D., Kim, Y.-G., In, H.P.: SA-RFID: Situation-Aware RFID Architecture Analysis in Ubiquitous Computing. In: Proceedings of the 11th Asia-Pacific Software Engineering Conference, pp. 738–739. IEEE Computer Society Press, Busan, Korea (2004)s
5. Lansford, J., Stephens, A., Nevo, R.: Wi-Fi (802.11b) and Bluetooth: enabling coexistence. In: IEEE Network, vol. 15 (5), pp. 20–27. IEEE Press, Los Alamitos (2001)
6. Lymberopoulos, D., Lindsey, Q., Savvides, A.: An Empirical Analysis of Radio Signal Strength Variability in IEEE 802.15.4 Networks Using Monopole Antennas. ENALAB Technical Report 050501, Yale University (2005)
7. Penttila, K., Sydanheimo, L., Kivikovski, M.: Performance development of a high-speed automatic object identification using passive RFID technology. In: Proceedings of the IEEE International Conference on Robotics and Automation, vol. 5, pp. 4864–4868. IEEE Press, New Orleans, USA (2004)
8. RFID Journal, `http://www.rfidjournal.com`
9. Song, J., Kim, H.: The RFID Middleware System Supporting Context-Aware Access Control Service. In: Proceedings of the 8th International Conference on Advances Communication Technology, vol. 1, pp. 863–867. IEEE Press, Phoenix Park, Korea (2006)
10. Zhenjie, X., Changjia, C.: A Localization Scheme with Mobile Beacon for Wireless Sensor Networks. In: Proceedings of the 6th International Conference on ITS Telecommunications, pp. 1017–1020. IEEE Press, Chengdu, China (2006)
11. ZigBee, `http://www.zigbee.org`

Contract-Based Exception Handling Process Patterns

Jelena Zdravkovic and Vandana Kabilan

Department of Computer and Systems Sciences
Stockholm University and Royal Institute of Technology
Forum 100, SE-164 40 Kista, Sweden
{jelenaz,vandana}@dsv.su.se

Abstract. Use of ontologies in enterprise modeling is extensively increasing. Two emerging uses are (a) making implicit domain explicit, and (b) facilitating interoperability between information system applications. For the first case, conceptual models play a key role. The domain of our interest is contractual obligations which are to be realized by enterprise business processes of the involved parties. A problem is that processes are not typically capable to handle diverse non-successful paths of contract executions. The main reason for this lies in the lack of a knowledge base for identifying possible exceptions and the procedures for handling them. To enable acquiring of this knowledge base, in this paper we propose the use of ontology for exception handling in business processes, where the exceptions are conceptualized to match to possible contract violations. The ontology is utilized for forming of autonomous process patterns for exception handlings, which extend the core processes to enable full realizations of vast contract requirements.

1 Introduction

In philosophy, *ontology* is a description of the world *as-is*. In the information systems domain, this is taken to be synonymous with the world *as-it-should-be*. Ontology within the computational science is being used for various purposes, from shared understanding, interoperability, reusable knowledge base etc. We see an increasing use of ontology for enterprise modeling [1], knowledge representation [2], information integration [3], and so on. However, the most common use of ontology in information systems has been for domain knowledge modeling – to make implicit knowledge explicit [4]. By doing so, a consensus and shared visualization of a specific domain of discourse is obtained. Informal domain ontologies such as Enterprise Ontology [5] can be used for knowledge sharing and understanding, whereas rigorously formal domain ontologies such as TOVE 5**.** are used for advanced features like reasoning and inference support systems. Most enterprise- and business process- ontologies model static concepts, i.e. *endurants*. Dynamic concepts, i.e. *perdurants*, such as change or process, need to be conceptualized in the way to enable modeling of automated business processes. Aligned with this, Smith [7] emphasizes a fundamental difference in approach and design of ontology in information systems, which he terms as “Descriptive Ontology” and “Procedural Ontology”. This has been further strengthened by Poli [8] in distinctions of “descriptive”, “formal” and “formalized” ontologies. In this paper, we address this dilemma of ontology design and modeling. In particular, we consider conceptual modeling in the realm of information systems. The fundamental objective behind conceptual modeling and that of ontology is the same; *to conceptualize the domain of interest.* According to Johannesson et al. [9], conceptual modeling has three aspects, (a) semantics (b) syntax and (c) pragmatics. UML has become widely used for conceptual modeling [10]. However, there still remains the issue of modeling the dynamics or behavior of a domain, in an *integrated, reusable* and *generic* conceptual model.

The research problem for this paper is, therefore, to propose an approach for modeling a set of coherent, generic and reusable conceptual models, by combining inter-disciplinary ontology modeling approaches and conceptual modeling methods. Our application interest concerns business contracts and their alignment to business process models, with a specific orientation towards exception handling of business processes. Business processes are not typically capable to handle diverse non-successful paths of contract executions, mainly due to a lack of a knowledge base for identifying possible exceptions.

We present two sets of conceptual models, each belonging to a different category of ontology. The first, Multi-Tiered Contract Ontology (MTCO) [11] is a *generic domain ontology* [12], aimed primarily for making explicit the intrinsic domain knowledge of legal business contracts understandable. As such, the targeted users have been human decision makers and business process modelers. The second set of conceptual models that we introduce in this paper, pertain to Exception Handling Process Ontology (EHPO), *task ontology* [12]. We use the EHPO further as a basis for modeling Exception Handling Process Patterns (EHPP) that are primarily intended as design templates for generating executable process models for resolving exceptions that occur due to contract violations. EHPP belong to the *application ontology* categorization, according Guarino’s proposal [12].

In this paper, we propose an approach for merging the “descriptive” and “procedural” perspectives, in order to capture the “dynamics” of a domain of interest. In this process, we set efforts to reuse existing domain and task ontologies (semantics), standards and recommended languages (syntactic) to make explicit the pragmatics of the domain being modeled. To illustrate the approach, we have considered contractual trade violations and their realizations in the form of business processes.

The rest of the paper is structured as follows - we begin with a short description of the business contracting domain and the relevant related research. In Section 3 we discuss our design approach and its application in the context of our domain of interest, i.e. contract violations and their remedies. After briefly describing the major elements of the MTCO (Section 4), in Section 5 we present a case scenario with possible contract violations, which provides a ground for outlining possible behavioral patterns for exception handling. In Section 6, we propose the EHPO ontology, and in Section 7 we describe its mapping within executable error-handling process models. We conclude the paper and outline a future work in Section 8.

2 Related Research

Standard contract models such as ICC Model for International Commercial Sale of Goods [13] and delivery terms INCOTERMS [14] formulate the acceptable modalities for trade relationships between enterprises. A business contract elucidates the coordination mechanism for the actual occurrence of a trade transaction. It also outlines the *duties* of the parties involved and their *rights* for remedy in case the execution does not occur as expected.

In the MTCO, the main "static" concepts of a business contract have been described (Section 4). Constraints and conditions of a contract are represented as static properties of the main concepts, or as description logic axioms. However, this does not explain how the agreed coordination of business actions is to be modeled. The *dynamics* i.e. *flow aspects, policies and rules* of the business transactions need to be captured as well. As Grosof et al. in SWEETDEAL [15] suggest, one alternative would be to model the logic aspects of contractual terms and conditions, as *rules,* residing on top of an ontology using, for instance, the W3C recommended Semantic Web Rule Language[1]. Tan [16] has used deontic logic for modeling the directed obligations, whereas Milosevic [17] has used subjective logic. Even though all the outlined alternatives represent viable approaches, there is still lack of coherent recommendations for representing the "behavior" using UML class diagrams.

In the context of business transactions and contract executions, the behavioral, i.e. procedural perspective is significant and as such, must be considered. Some procedural aspects have been captured in the MIT process handbook [18]. However, contractual violations and the corresponding exception handling mechanisms have not been addressed. Exception handling at the technical level has been addressed to an extent in B2B protocol standards such as ebXML [19], and also by Klein in [20].

3 Approach for Designing Descriptive and Procedural Ontology

Pentland [21] supports the definition of processes as "grammatical models". He compares syntax of organizational processes to a "grammar" that describes a finite set of patterns in terms of a finite lexicon, and a finite set of rules or constraints that specify allowable combinations of the elements in the lexicon.

For our domain of interest, it must be understood (a) what the contract expects the enterprise to do, (b) how the actual transaction will occur as agreed (c) if something goes wrong, what procedures are to be followed to attend the problem. For (a), we proposed the MTCO; for (b), we have proposed a methodology for deducing contract compliant workflow models [22], and for (c) we propose the set of Exception Handling Process Patterns as the "grammar" of process execution handling "lexicon".

To achieve the outlined, we first design "Descriptive Ontology":

– We capture the "static" concepts of the domain, both explicit and implicit, following current ontology design guidelines such as that of Gruninger [1] or METHONTOLOGY [23]. These domain concepts are modeled as concepts in UML class diagram. Relationships between these concepts are drawn as UML

[1] Semantic Web Rule Language http://www.w3.org/Submission/SWRL/, 08-05-2007.

associations. The associations capture the semantic relationship and not structural composition.

– For capturing the dynamics or behavioral characteristics we propose to semantically embed these characteristics within the UML class diagram using stereotypes mainly for associations. For example, we use association stereotype *activates* to semantically indicate a natural flow of objects or events between identified target and source concepts. Further more, we propose to add further characteristics to these stereotype associations.
– For facilitating automatic translation into machine readable form like OWL, we adopt the recommendation of ODM [24] for the syntactic representation of the concepts in UML class diagram. That is, we use the ODM as a meta-model for defining the conceptual model. For example, the UML generalization association ("is-a") is mapped to OWL:SubClassOf property, a normal association is OWL:ObjectProperty. Details maybe referred to in the ODM specification. This has allowed us a richer notational capability to model the conceptual model. For example, we have been able to depict semantically a behavioral choice by using OWL:OneOf stereotype on associations.
– For capturing the pragmatics of the domain, we propose the use of another meta-model based on the linguistics approach of Speech Act, specially focusing on the performative and declarative aspects. However, discussion of these mappings is out of the scope of the current study.

The above approach is applicable for designing the "procedural ontology" as well, as it will be demonstrated in Section 5 for Exception Handling Process Ontology. The procedural ontology is assumed to be "action-centric" as against an "object-centric" focus of a descriptive ontology. That is, the central concept of interest is an activity or change or process, and we start our knowledge capture and modeling approach with the action as the centre of focus. The action object is thereafter considered similarly to a static concept in the case of descriptive ontology and the other steps are essentially the same.

The summarization of our design approach:

- To make domain knowledge explicit, we propose the use of UML class diagrams as conceptual models. We propose the above mentioned design approach to ensure a rich semantic knowledge representation that can be translated in to machine readable form.
- By adopting an ontology design architecture as that of Guarino, we propose the segregation of the generic, static, declarative concepts in to a generic domain ontology (conceptual model), and the dynamic, procedural domain concepts as a separate task ontology.
- According to Guarino [12] - "two systems A and B using the same language L can communicate only if the set of intended models $I_A(L)$ and $I_B(L)$ associated with their conceptualizations overlap" The overlap between our design approach for a domain, task and application conceptual models is on the common 'concepts' and 'relationships' mappings which can be traced on all three conceptualizations.

In the following sections, we shall illustrate the above approach by using the proposed conceptual models for the business contracts and their related exception handling business processes.

4 Overview of the MTCO

As stated in Introduction, we have proposed the Multi-tiered Contract Ontology (MTCO) as a set of structured ontologies in various levels of specializations. In Figure 1 below, we depict only the top-level concepts described in MTCO using the UML classes. In addition, we represent existing ontologies which may/are used to extend the semantics of MTCO. Some of the possible extensions to the MTCO may be the standard recommendation for delivery as the INCOTERMS.

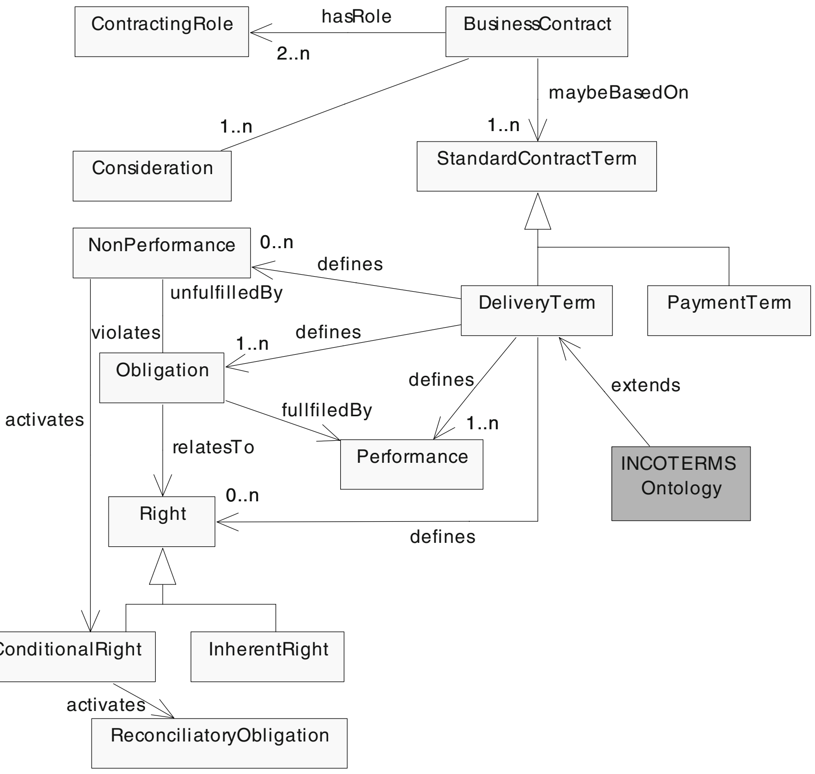


Fig. 1. The top-level elements of the MTCO

The MTCO defines contract related concepts such as *Obligations* which are the promises made by the *ContractingRole(s)*, for the *Consideration* (the objects of interested exchange for the business transactions, such as goods and money) by carrying out the expected *Performance.*

Contract violations, or *NonPerformance,* give rise to 'business exceptions' which on the contract perspective, activates *ConditionalRight(s)* on the part of the wronged-party. A contract usually stipulates a set of remedial options, for resolving the contract violations and the final decision and choice is left to the business enterprise to enforce the options. Once the choice is made then the *ConditionalRight* is transferred as a reconciliatory obligation to the other party who is now expected to fulfill it. We shall discuss specific cases for these in the next section.

5 Contractual Violation Scenarios for Exception Handling

Using the concepts of the MTCO as described briefly in the previous section, in this section we conceptualize the trading case scenarios, which will be used as the basis for identifying the Exception Handling Process Patterns (Section 7). In a typical business trade contract, the Seller agrees to "sell" something of value or interest to the "Buyer" who in turn agrees to "pay". They agree upon on *what* are the acceptable conditions for satisfaction, as also some contingency plans and options on *how* they will resolve their differences, in case the transaction does not go according to agreement. There are explicit statements as well as implicit understandings and best practices embedded in a physical contract. There are obviously elements of *procedural aspects.*

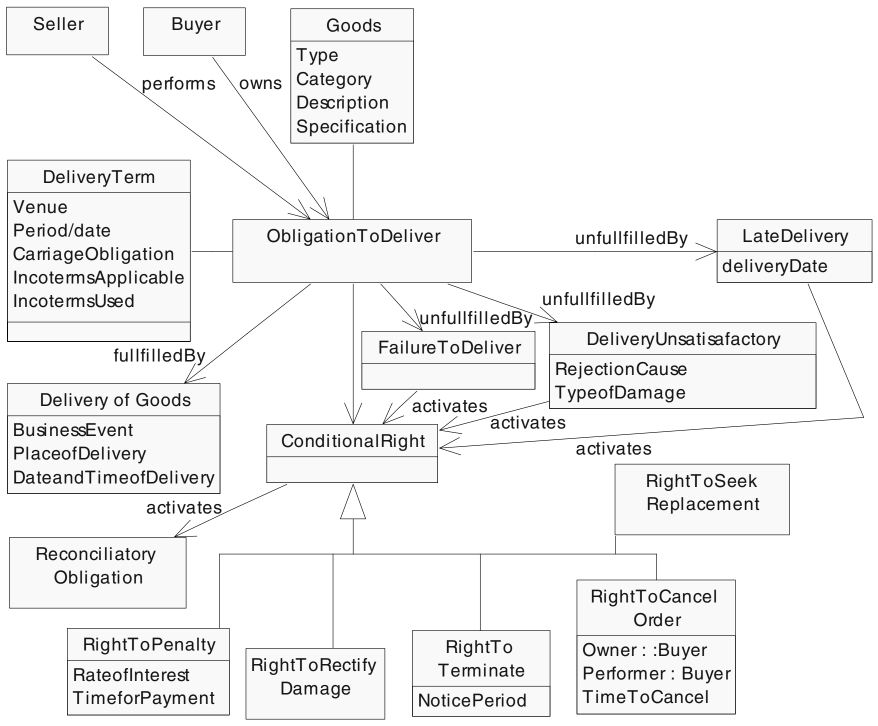


Fig. 2. Obligation to Deliver, conceptual model

We choose the ICC Model for International Sale of Commercial Goods as the model for the specific contract domain model. The contract specifies a number of obligations and their expected fulfillment criteria. One such primary obligation, on the part of the Seller, is the *ObligationToDeliver*. (Figure 2). The seller's obligation to delivery is guided by the set of *DeliveryTerms* agreed to. These *DeliveryTerms* generally specify the venue, and a delivery date. If INCOTERMS are used, then more details like who is responsible for arranging carrier, division of costs and risks, export and import clearance duty are also specified.

The expected *Performance* (event) for fulfilling the obligation to deliver is that of *Delivery of Goods*. This performance is deemed to be violated, that is *unfullfilledBy* an occurrence of a *NonPerformance* event under a set of circumstances such as (a) the delivery is late (*LateDelivery*) (b) the delivery is on time but the goods are not as per the specification (*DeliveryUnsatisfactory*) (c) the seller does not deliver any goods at all (*FailureToDeliver*). Each of these contract violations may be resolved via a number of options as identified by the *Conditional Rights*, such as:

- *RightToSeekReplacement*: the buyer may choose to seek replacement of the faulty goods, for instance, in the case of unsatisfactory delivery.
- *RightToPenalty*: Buyer may impose a monetary fine for a late or an unsatisfactory delivery.
- *RightToCancelOrder*: The buyer may cancel the specific order instance.
- *RightToRectifyDamage*: The buyer may require compensation in the case of a late or unsatisfactory delivery, etc.
- *RightToTerminate*: In extreme cases (such as a big latency in delivery) the buyer may choose to terminate all trading relationships, that is, to cancel the contract in itself.

A combination of these rights is also possible, like the Buyer may choose both redelivery and a penalty.

Note that we model flow sequencing through the use of stereotype association "activates" which connects a non-performance event like *LateDelivery* to the concept of *ConditionalRight*. The conditions for this activation are the actual "business logic" or "rules" to be implemented by an executable process models. However, these are generic and abstract to the degree that they allow us to identify the concepts involved but not the detailed business rules or logic in itself. As described earlier, we model, such "constraint" like conditions described in the contract; they may be modeled using Object Constraint Language [25] in the conceptual models. Syntactically these may be represented using Common Language (as recommended by the ODM) or even Semantic Web Rule Language. An example of such a constraint may be given as follows:

If a delivery made by the seller is late or not can be determined by examining the *DeliveryTerms* to see the agreed *DeliveryTerms.Period* (for instance, 7 days for example). Then, assuming that every single contract execution will have an order date and/or an expected delivery date (*Order.OrderDate*), it is implicit that the actual delivery date for that particular order should be within the agreed upon delivery period. If the actual delivery date (*DeliveryofGoods.DateAndTimeOfDelivery*) is more than agreed upon delivery date then it is a case of *LateDelivery*, which activates the conditional rights of the buyer as described earlier (each conditional right has a set of *states* just like the obligation states of inactive, active, triggered, fulfilled) This enabling condition is semantically represented as "*activates"* in Figure 2 and can be represented in OCL(s) as:

```
IF DeliveryofGoods.DateAndTimeOfDelivery
> Order.OrderDate + DeliveryTerms.Period THEN LateDelivery=True
```

6 Modeling Exception Handling Process Ontology (EHPO)

Once signed, a contract is to be realized by a business process. As we explained in the previous section, a contract specifies possible obligation violations and how they are to be resolved. A business process must have capability to realize contractual violations by realizing chosen remedial procedures.

A problem is that business process models typically specify only the "successful" flow of events. Protocols for exceptional flows are usually not designed, due to two main problems:

1. Exception flows cannot be anticipated;
2. Specification of too many exceptional flows greatly increase complexity of a process and decrease its readability and maintainability.

However, one of the main considerations in design of effective processes is supporting broad exceptions originating from contractual violations.

In order to enable handling of undesired contract occurrences by business processes, to solve the first outlined problem, it is necessary to systematically anticipate possible exceptions, and to select appropriate exception handling procedures. For solving the second problem, we propose defining every exception procedure (i.e. exception handler according to the MIT Process Handbook [18]) in the form of a process pattern. Every pattern defines a self-standing process template intended to be executed when a particular exception occur.

For classifying process exceptions and the corresponding handlers, we propose ontology depicted in Figure 3 below.

In the chosen Sale of Commercial Goods contract model domain, obligations are concerned to transfers of documents, goods and money between the buyer and seller. In Figure 3 below, these concepts are classified as *informational*, *physical* and *financial objects*, and transfers are modeled as *flows* acting on particular objects. In Figure 2, the illustrated concept of *ObligationToDeliver* (see Figure 2) is concerned to a physical object, i.e. to *Goods*.

A flow of an object is prone to a set of *exception conditions*. For instance, the buyer may deposit money with a delay, or he might transfer a wrong amount, and so on. As contract violations are typically alarmed on the side of the party that expects to obtain an object, we classify the exception conditions as object *NotAvailable*, *AvailableWrongPlace*, *AvailableWrongTime* and *AvaliableWrongContent*. Here, the goal is to ensure mapping of obligation non-performances as found in the MTCO trading domain from Figure 2, to the exception conditions in Exception Handling Process Ontology (Figure 3). For instance, LateDelivery, GoodsUnsatisfactory, FailureToDeliver non-performances are mapped to exceptions AvailableWrongTime, AvailableWrongContent and NotAvailable, respectively (Figure 3).

As explained in the previous section, from the contract perspective, non-performance events give rise to activation of a set of conditional rights. For instance, LateDelivery non-performance (see Figure 2) may activate RightToPenalty or RightToCancelOrder, or even both these conditional rights. In the EHPO this is solved by relating an exception condition to an *exception handling protocol*, which consists of one or more ordered *exception handlers*. As explained in the beginning of the section, an exception handler instructs a procedure to be taken in order to resolve an occurred exception.

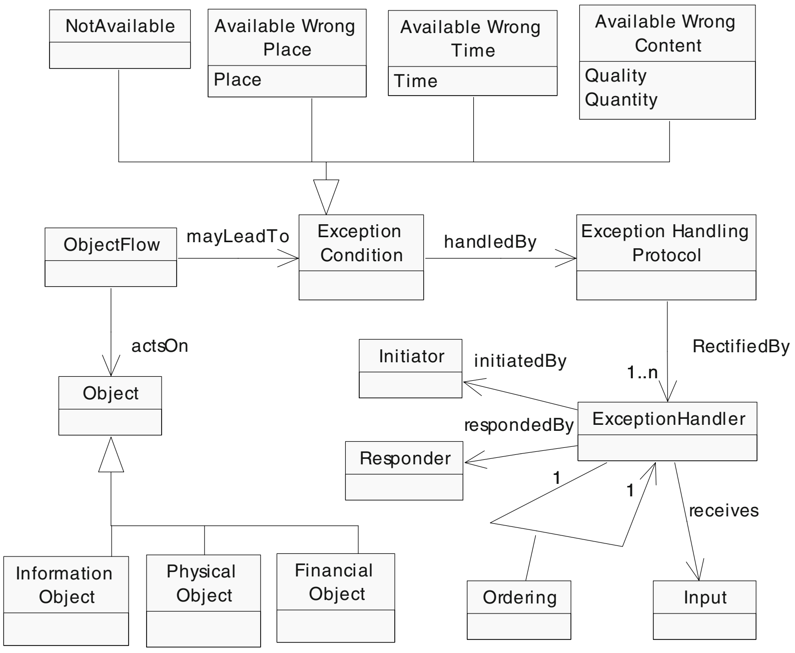


Fig. 3. Exception Handling Process Patterns Ontology

Every exception handler is derived from a conditional right, such as for instance, Penalty from RightToPenalty, CancelOrder from RightToCancelOrder, and so on. A handler is activated by an *Initiator*, such as the Buyer in case of the penalty procedure, and countered by a *Responder*, such as the Seller accepting the request for a penalty. Through the *Input*, a handler receives the total set of constraints of a chosen conditional right, such as RateOfInterest and TimeForPayment in case of RightToPenalty (see Figure 2). It also receives the information on what object flow the exception has occurred.

To summarize, the purpose of the EHPO is to provide a self-standing knowledge base about business process exceptions, by interpreting and classifying the information originating from the contract domain. The ontology enables understanding such as – what was the object of an exception, what condition has caused the exception, and what are the procedures that need to be performed in order to resolve the conflicting situation. In this way modeled, the ontology provides a bridge for enabling modeling of contract violations with the concepts of business processes.

7 Realizing Exception Handling Process Patterns (EHPP)

Exception handlers, as conceptualized in the EHPO, need to be realized in the form of executable business processes. Those processes need therefore to be modeled in the way to resolve detected exception conditions.

In this study, we propose modeling of exception handling processes in the form of patterns. Patterns can be described from different points of view. From a model designer perspective, they are off-the-shelf business processes that can be used as modular components pluggable to the "main" process, as well as linked to build more complex exception procedures. The UN/CEFACT Unified Modeling Methodology [26] states that the use of patterns optimizes business process and information model reusability.

The notation we will use for process pattern models is BPMN [27], a standard developed by the Business Process Management Initiative (BPMI). The BPMN is an easily comprehensible notation that can be converted to executable XML languages for process specifications such as BPEL4WS [28].

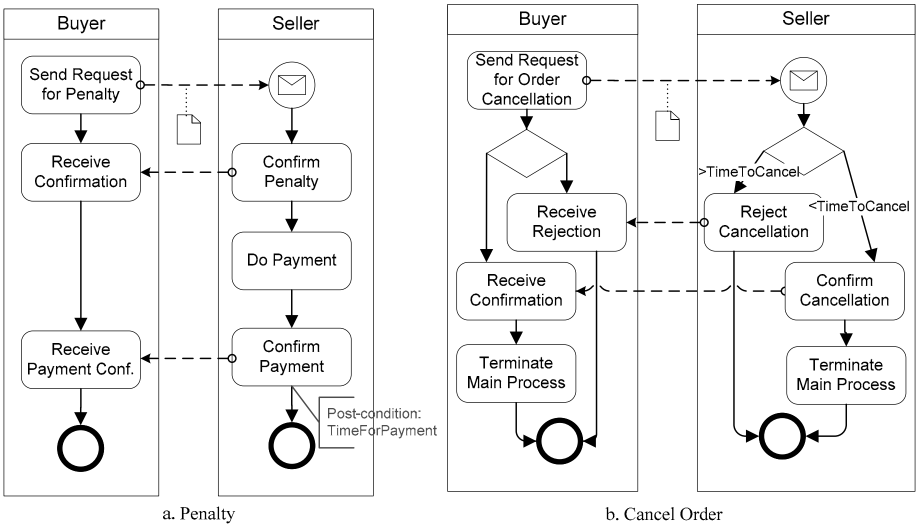


Fig. 4. a. Penalty and b. Cancel Order BPMN process patterns

Every pattern defines coordination of activities of the collaborating parties, as well as their interactions. A process pattern is matched with an Exception Handler from the EHPO in the following elements:

- *Names* of handlers and patterns must be identical.
- The *Initiator* of a handler is the role that triggers a process pattern instance. The *Responder* is the role that accepts the incoming initiator's request for a remedy.
- The *Input* of a handler containing a set of constraint parameters set on chosen conditional rights is mapped to pre-conditions, process variables, post-conditions or to elements of the documents that are to be exchanged within an exception handling process. In addition to this, the information on object that has come to an exceptional state (see Figure 3), is also passed from Input of the handler.

In the case study of our interest, when goods are not available on time, the exemplified exception handlers are Penalty and Cancel Order. The Buyer is the initiator of the Penalty exception handler (Figure 4.a), having inputs representing a rate of interest to be paid, and a timeframe available for that. The two parameters are in the Penalty process used as the elements of the request for penalty document which Buyer sends to the Seller. In addition, the Seller might specify a time-based post-condition for its process in order to ensure on time payment of penalty. In the case when Cancel Order exception handler is chosen, the initiator and respondent roles are distributed similarly to the Penalty pattern. The input parameter, defining the deadline for an order cancellation, is mapped to a corresponding process variable of the Seller's process The Buyer initiates the exception handling process by sending a request for canceling the order to the Seller. If the request is sent before the TimeToCancel (see Figure 2 also), the Seller confirms the request and both sides issues termination of the main processes; if the deadline for the cancellation has passed, the Seller rejects the request and proceeds with the main process. Finally, in the case when a late delivery is agreed to be handled by a Penalty followed by the Cancel Order process, the two patterns are linked in a ordered set as instructed by the *Ordering* element from the EHPO.

In this section we have described the concept of error-handling process pattern as a means for realizing defined contract violations and corresponding remedial options. In the following section, we conclude our work.

8 Conclusion

In this paper, we have addressed the problem of handling vast contract violations, which are typically not captured in the technical realizations, such as in executable business processes.

Our approach aimed on grasping of different domain perspectives into a consolidated and traceable conceptual model. To achieve this, we strived to reconcile between a "static", contract-level ontology (MTCO) and a "dynamic", business process-based ontology (EHPO). Using the MTCO ontology, we modeled the intrinsic domain knowledge of legal contract violations, illustrated on the Sale of Commercial Goods case. With the EHPO, we structured the concepts of business exceptions concerned to flow of informational, physical and financial objects. When modeling the EHPO, we aimed for grasping and mapping of the contract-level concepts, such as non-performance events and remedial options, to the business process-level concepts, such as exception conditions and exception handlers. Having the knowledge on business exceptions structured and available, we used it as the basis for designing executable error-handling processes, in a modular, pattern based way.

We believe that our approach, with its concepts, subsumes the following contributions:

- We described a conceptual modeling approach for designing ontologies in contract- and business process- management domain for (a) making explicit domain knowledge (b) conceptualizing the dynamics of the domain using UML conceptual models (c) using the proposed conceptual models for deriving machine readable applications.
- On a more specific level, we have introduced the concepts needed for designing of reusable, machine readable patterns for handling exceptions in business processes, with an orientation towards business contract compliancy and with a minimum need for human interference.

As a summary, in this study we aimed to propose the concepts needed for enabling traceability from contract to process models. We set the focus on handling exceptional business situations, as there is still lack in methodologies for creating executable business processes capable for realizing effectively contracts where violations occur. Regarding the future work, in addition for structuring occurrences of exceptions in object flows, we aim for extending the EHPO ontology to support detecting and preventing of exceptions.

References

1. Uschold, M., Gruninger, M.: Ontologies:Principles, Methods and Applications. The Knowledge Engineering Review 11(2), 93–136 (1996)
2. Artala, A., Franconi, E., Guarino, N., Pazzi, L.: Part-Whole Relations in Object- Cantered Systems: an Overview. Journal of Data and Knowledge Engineering 20(3), 347–383 (1996)
3. Bergamaschi, S., Castano, S., Vimercati, D., Vincini, M.: An Intelligent Approach to Information Integration. In: Int. Conference on Formal Ontology in Information Systems (FOIS 1998), Trento, Italy, pp. 72–88. IOS Press, Amsterdam (1998)
4. Noy, N.F., McGuinness, D.L.: Ontology Development 101: A Guide to Creating Your First Ontology. Stanford, CA, 94305, Stanford University (2001), Available at http://www.ksl.stanford.edu/people/dlm/papers/ontology-tutorial-noy-mcguinness-abstract.html
5. Uschold, M., King, M., Moralee S., Zorgios, Y.: The Enterprise Ontology, http://www.aiai.ed.ac.uk/ entprise/enterprise/ontology.html
6. TOVE Enterprise Ontology Project, http://www.eil.utoronto.ca/enterprise-modelling/tove/
7. Smith, B.: Ontology and Information Systems. In: Foridi, L. (ed.) Blackwell Guide to the Philosophy of Computing and Information, pp. 155–166. Blackwell, Oxford (2003), http://ontology.buffalo.edu/ontology(PIC).pdf
8. Poli, R.: Formal and Formalized Ontologies. In: Fisette, D. (ed.) Husserl's Logical Investigations reconsidered, pp. 183–210. Kluwer Academic Publishers, Dordrecht (2003), http://www.formalontology.it/
9. Johannesson, P., Boman, M., Bubenko, J., Wangler, B.: Conceptual Modeling. Prentice-Hall, Englewood Cliffs (1996)

10. Baclawski, K., Kokar, M., Kogut, P., Hart, L., Smith, J., Holmes, W., Letkowski, J., Aronson, M.: Extending UML to Support Ontology Engineering for the Semantic Web. In: Gogolla, M., Kobryn, C. (eds.) UML 2001. LNCS, vol. 2185, pp. 342–360. Springer, Heidelberg (2001)
11. Kabilan, V., Johannesson, P.: Semantic Representation of Contract Knowledge using Multi-Tier Contract Ontology. In: Proceedings of Semantic Web and Databases workshop (SWDB 2003), co-located with VLDB 2003, Berlin, Germany, pp. 395–414 (2003)
12. Guarino, N.: Formal Ontology and Information Systems. In: Proceedings of the First International Conference on Formal Ontologies in Information Systems, Trento, Italy, pp. 3–15. IOS Press, Amsterdam (1998)
13. qInternational Chamber of Commerce. ICC International contract for sale of goods, `http://www.iccwbo.org/incoterms/id3045/index.html`
14. Ramberg, J.: ICC Guide to INCOTERMS, Understanding and Practical Use. International Chamber of Commerce, http://www.iccwbo.org/incoterms/id3045/index.html
15. Grosof, B., Poon, T.: SweetDeal: Representing Agent Contracts with exception using XML rules, Ontologies and process descriptions. In: Proceedings of the 12th International World Wide Web Conference (WWW 2003), Budapest, Hungary, pp. 124–146. ACM Press, New York (2003)
16. Tan, Y.H., Thoen, W.: A Logical Model of Directed Obligations and Permissions to Support Electronic Contracting. Int. Journal of Electronic Markets 10(1), 78–92 (2000)
17. Milosevic, Z., Jøsang, A., Patton, M.A., Dimitrakos, T.: Discretionary enforcement of Electronic Contracts. In: Proceedings of the 6th International Enterprise Distributed Object Computing Conference (EDOC 2002), Lausanne, Switzerland, pp. 39–50. IEEE Computer Society, Los Alamitos (2002), ISBN 0769517420
18. The MIT Process Handbook, ISBN 0262134292, MIT Press, Cambridge (2003)
19. ebXML, `http://www.ebXML.org`
20. Dellarocas, C., Klein, M.: A Knowledge Based approach for handling exceptions in Business Processes. In: Int. Journal of Information Technology and Management, vol. 1(3), pp. 155–169. Baltzer Science Publishers (2000)
21. Pentland, B.: Grammatical Models of Organizational processes. The MIT Process Handbook, pp. 191–214. MIT Press, Cambridge (2003), ISBN 0262134292
22. Zdravkovic, J., Kabilan, V.: Enabling Business Process Interoperability Using Contract Workflow Models. In: Meersman, R., Tari, Z. (eds.) OTM 2005. LNCS, vol. 3760, pp. 77–93. Springer, Heidelberg (2005)
23. Fernandez, M., Gomez-Perez, A., Juristo, N.: METHONTOLOGY: From Ontological Art Towards Ontological Engineering Workshop on Ontological Engineering, Stanford, California (1997)
24. Object Management Group (OMG): Ontology Definition MetaModel, `http://www.omg.org/docs/ad/05-08-01.pdf`
25. Akehurst, D., Linington, P., Patrascoiu, O.: Object Constraint Language 2.0: Implementing the standard. Technical Report no. 12-03, University of Kent, UK (2003)
26. UN/CEFACT Modeling Methodology (UMM), User Guide, `http://www.unece.org/cefact/umm/UMM_userguide_220606.pdf`
27. White, S.: Business Process Modeling Notation 1.0 (BPMN). Business Management Initiative, `http://www.bpmi.org`
28. Microsoft, B.I., Siebel, S.A.P.: Business Process Execution Language for Web Services, `http://www.ibm.com/developerworks/library/ws-bpel`

Mobile Applications for the Academic Environment

Răzvan Daniel Zota[1], Ileana Stănescu[2], Emil Stănescu[2], Raluca Stănculescu[3], and Laura Stănescu[3]

[1] Academy of Economic Studies Bucharest, Piata Romana 6, sector 1, Bucharest, Romania
zota@ase.ro
[2] National Institute for Research and Development in Informatics, Av. Mareşal Al. Averescu 8-10, sector 1, Bucharest, Romania
{ileanas,stanescu}@ici.ro
[3] Academy of Economic Studies, Faculty of Cybernetics, Statistics and Business Informatics, Calea Dorobantilor 15-17, sector 1, Bucharest, Romania
raluca.stanculescu@gmail.com, aural@yahoo.com

Abstract. We present in this paper some of the various usages of wireless technology, through web services and WAP. The importance of this paper is emphasized by the permanent interest of web developers to add WAP functionality and support to their web sites.

As for the practical part of the paper, we provide here two applications, both developed for the academic environment. The first one uses web services and let users consult their schedule from mobile devices. The second one leverages the WAP protocol to provide students with the opportunity to check their schedule and have access to information about professors and classes.

Keywords: Web services, WAP, WML, mobile phones, PDA.

1 Introduction

In the past years mobile technologies became strategic for a lot of businesses, organizations, disciplines and activities [1]-[3]. In this context education is no exception and, as anyone may notice, in the past years the number of students from the academic environment which are using mobile devices was increasing a lot. Since even from school children use a lot the mobile phones [4], nowadays a lot of students use different communication related devices, such the mobile/smart phones or Personal Digital Assistants (PDAs). Students are always on the move and in permanent interaction, so there is great demand for message interchange. They ask also to have access to information about their schedule or about the availability of their professors for consulting hours. Following up, a few models of information are analyzed pertaining to a user from the academic environment, that has at his disposal mobile terminals with some communication and processing facilities, such as a mobile phone or a PDA. Further on, the architecture of a student and professor information system over the schedule and other events is described. Respective of the mobile device that the user is using, he/she can use the browser of the mobile phone to access WML files, using the WAP protocol, or he/she can upload a user application that can be of the JavaME type or Microsoft Compact .NET framework type, application that accesses Web services that reside on the application server. Because of memory and computation power constraints, the implementation of client applications for web services can relay access to much bigger computation resources.

Wireless applications accessed from a mobile phone, provide the user with the information that he wouldn't otherwise have access to on the move. Let's assume that the user (a student or professor) is a hurry, and does not have the time to swing by to check his faculty's notice board, or he/she is just plain ignorant about his/hers professors' email, office number and location, etc. A mobile phone could go a long way in any such situations. The user could access via WAP, his/hers personal page application and retrieve the needed information.

Concerning web services, interoperability is a solution for the correlation of information from different information systems, implemented independently on different platforms, and for unitary implementation of some services that can be utilized by different clients independently of the platform that these clients' applications have been developed on.

2 Technologies for Communication

The Wireless Application Protocol (WAP) is an open international standard for applications that use wireless communication. Its principal application is to enable access to the Internet from a mobile phone or PDA [5], [6].

2.1 WAP, WML, XHTML Technologies

WAP sites are websites written in, or dynamically converted to, WML (Wireless Markup Language) and accessed via the WAP browser. The latest WAP Forum's specifications are known as WAP 2.0. The most important architectural components of WAP 2.0 are the following [7]:

- Protocol Stack Support – WAP 2.0 adds to WAP 1.* stack support for TCP, TLS and HTTP, providing a connectivity model on a broader range of networks and wireless bearers.
- WAP Application Environment – includes markup languages viewed by "WAP Browsers". WAP 1.* application environment uses WML (Wireless Markup Language) as a conformant XML language. WAP Browser, for the WAP 2.0 Application Environment has evolved to embrace developing standards for Internet browser markup language that permits wireless devices to utilize existing Internet technologies. XHTML MP (XHTML Mobile Profile) is the markup language of WAP 2.0. XHTML MP is a subset of XHTML, which is the combination of HTML and XML. eXtensible HyperText Markup Language (XHTML) has been developed by the W3C to replace and enhance the currently used HTML language.
- Additional Services and Capabilities –With WAP 2.0, there is a considerable increase in the number of features available to developers, operators and users.

2.2 Web Service Technologies

Web Services are based on XML, classified as an extended language, because it aims to facilitate the common use of data between different systems.

Web services standards provide a façade type interface for the development environment of mobile applications that is based on a server component and a client one. This integration of web service technologies with object oriented programming techniques, and with the structure and communication based on the web raises special development problems for mobile devices such as Smart Phones and PDAs.

3 The Architecture and Functions of the Student Information System

3.1 The Architecture

The architecture of the software system frames into a multi-level architecture type, with the following levels: data administration level, business level, client level.

The information system is formed of two base software applications. One is based on the client level use of some *JavaME* or *Microsoft Compact Framework* modules, corresponding to the equipment type. At the application level, on the server, there are web services that deliver useful content to the user. These services are accesed by the client applications fore-cited.

The second application uses, on the client side, only WAP browser facilities. On the server side the PHP, WML, WMLScript files are developed in correlation with an Apache web server. Another possibility would be that the server side is developed on a Microsoft platform using the IIS Web server and ASP.NET for the new generation of user content. The two applications basically, have the data level formed of shared data bases, developed using Microsoft SQL Server. Nevertheless, for experimentation reasons, the MySQL data server was used for the WAP software application.

The information system also provides transmission support for SMS messages, that permits user notification about certain events or it gives the possibility to navigate on WAP type Internet links to obtain the neccesary information. We show the schema of the information system architecture in figure 1.

3.2 System's Functions

The main functions of the system are the following:

- Handling information about the users: specific to the profile (the group for students, department, title for professors) or common to all profiles (first name, last name, CNP – Personal Number Code, etc.);
- Handling information regarding the educational institution: faculties, departments, classes within the departments, groups within the faculties;
- Handling elements of the schedule: week-day, hour and minute to start (begining time), hour and minute to stop (ending time), room, professors, class, group, subgroup, type (course, seminar, lab), scheduling (weekly, even week, odd week);

- Authentication using the mobile phone or PDA;
- Visualisation of the user's profile(s): student profile and/or professor profile;
- Examine the schedule using different criteria;
- Information about the academic situation (grades obtained during the academic year, credit count);
- Sending messages to users.

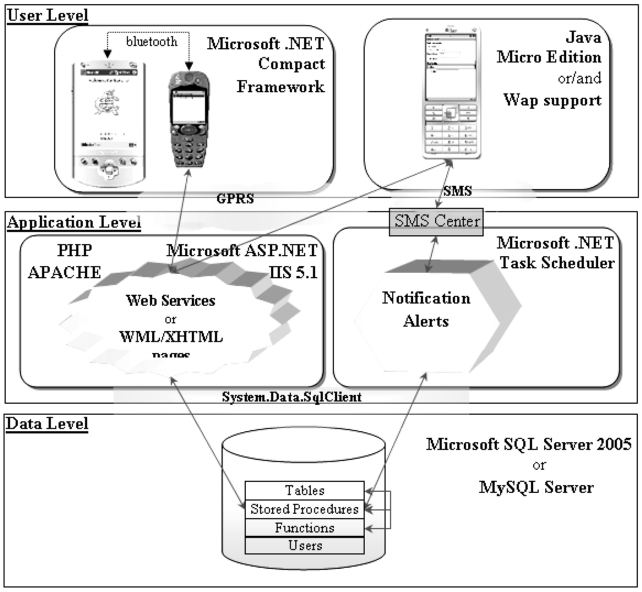

User Level
bluetooth
Microsoft .NET Compact Framework
Java Micro Edition or/and Wap support
GPRS
SMS
Application Level
SMS Center
PHP APACHE
Microsoft ASP.NET IIS 5.1
Web Services or WML/XHTML pages
Microsoft .NET Task Scheduler
Notification Alerts
System.Data.SqlClient
Data Level
Tables
Stored Procedures
Functions
Users
Microsoft SQL Server 2005 or MySQL Server

Fig. 1. System information architecture

4 Web Services Based Information Software Application

The first application example uses Web services and enables users to consult their schedule from mobile devices, on many different criteria. The example describes how the application was created on a simple PC (the application architecture, technologies used in implementation), the software used for its development and the upload to the mobile phone.

The application contains four modules which are assigned on the levels of the architecture in the following way:

- at the data level: a module which is implemented as a data base of the *Microsoft SQL server, 2005 Express Edition* version type.
- at the business level: a module which contains two web services with similar functionalities, but with different access possibilities. This module is implemented using a Microsoft *Internet Information Server 5.1* web server.

- at the client level: two modules which implement the functions of the client interface for two types of mobile devices: *Pocket-PC 2003* (PK-PC) and mobile phone (MP).

4.1 The Description of the Web Services

Two Web services have been developed into Microsoft Visual Studio 2005 Web Service project types, using C#. This medium permits great automatization of the defining and development of a Web service, specific to the common infrastructure described by the used XML standards.

Microsoft technologies offer the *DataSet class* for the data interchange, structured corresponding to a relational data base, class that facilitates very much the information handling, once received by the Web service client. The only problem is that the *DataSet* structures (dynamically created by the adaptors that read data from other data sources) are not so easily accessible from client applications developed on other platforms. That is the reason why we developed two web services: one with methods that return only *DataSet* objects and that will be referred only from the application for the *PocketPC*, and another service that returns simple data types, that can be referred from the *PocketPC* application, as well as from the application for the mobile phone developed in *J2ME*.

4.2 The Schedule Querying Application From the Pocket-PC Using a Web Service

The module intended for the PK-PC client was developed in the Microsoft .NET Compact Framework 2.0 technology, in the Visual Studio 2005 IDE environment, using C# language. This client application will consume the created web service.

The client application will send a request to the Web service, that will return a result to the application from the mobile device, using a XML/SOAP interface over the HTTP protocol.

The mobile application offers the following functions:

– *User's authentication* after inserting the account and password information. There can be two categories of users – students and professors – and corresponding to the category to which the user pertains, they can have access to certain menu features.
– *Visualization of the user's profile* and utilization of the data corresponding to this profile for viewing the schedule. For example, a user can go to two faculties, and from the form in which is described the profile can choose the group for which can view the schedule.
– *Visualization of the schedule.* The user can choose to visualize the schedule depending on several parameters. Thus, the user can choose one or more parameters between the following:

- the starting point from which begins the visualization of the schedule,
- the faculty at which the user studies/teaches,
- the group no., which belongs to the faculty which was filled out previously,

- the professor's name, if the student wants to find out the schedule of a certain teacher,
- the class name,
- the room number in which the class takes place.

4.2.1 The Consumption of the Web Service from the Pocket-PC

The application includes two web services as was mentioned above.

After the creation of a web reference, the referred web service is available to the client through a class named *Service* (the name of the class from the ASP.NET project that has implemented the web service) from the namespace *denumire_referintaWeb* (in our case *serviciu_web_ASE* and *serviciu_web_ASEms*).

4.2.2 Running the Application

The first step in running the application is the user authentication. (See figure 2).

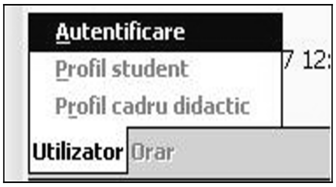


Fig. 2. Users' options

The menu options will be activated corresponding to the user type (student or professor). The user can be student and teacher at the same time, in the case of master post-graduates and doctorate post-graduates that teach in the academic institution.
After authentication, the user can view his profile and consult his schedule using certain parameters.

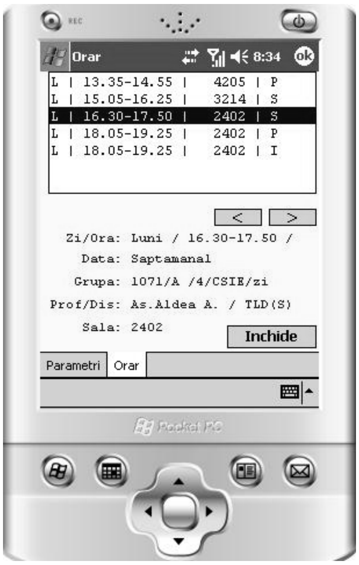
By choosing the Orar (schedule) option from the menu, a form will open which will contain two *TabPage elements* (with the displayed text *Parametri*, respectivelly *Orar*) of an object of *TabControl* type from the form. When the form is displayed the active page is the one with *Parameters*.

In this form, different parameters can be introduced which will help generate the schedule. If the *Profesor* parameter is introduced with the name (or part of it), then the professor's schedule can be visualised, or professors with the respective name.

4.2.3 Presentation of the Schedule Consulting Operation on the PDA

With the page change from *Parametri* to *Orar* (by touching the text *Orar*) a series of events ensue. Two events are treated in the application: the *Validating* event attached to the page with the text *Parametri* and the *SelectedIndexChanged* event attached to the control *tabControl1*. In the procedure that treats the first event the validation of the faculty field is achieved. (Other validations can also be implemented). In the second procedure that treats the second event, a test is performed to see if the activated page is *Orar* (schedule), in which case the request for the schedule is called. In Figure 3 we present a schedule from Monday, with classes that start later than 13:03, for group no. 1071 of the faculty with the CSIE abbreviation.

The schedule is listed in a list (a *ListBox* type control) with simple selection. At the selection of a list element, the details of the element schedule from the selected position will be displayed in the lower side of the screen. Using the buttons "<" and ">", one can navigate from a day to the other so that the schedule can be visualised.



```
L | 13.35-14.55 |   4205 | P
L | 15.05-16.25 |   3214 | S
L | 16.30-17.50 |   2402 | S
L | 18.05-19.25 |   2402 | P
L | 18.05-19.25 |   2402 | I
```

```
 Zi/Ora: Luni / 16.30-17.50 /
   Data: Saptamanal
  Grupa: 1071/A /4/CSIE/zi
Prof/Dis: As.Aldea A. / TLD(S)
   Sala: 2402
```

Fig. 3. Student's schedule details

4.3 The Application for Schedule Consulting from a Mobile Phone Using a Web Service

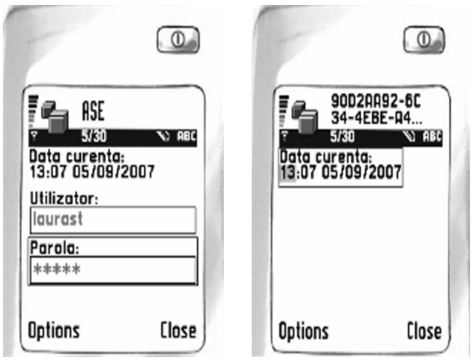
The *Java 2 Micro Edition* (*J2ME*) web service specification is a standardization attempt of the programmatic access at web services from the client applications *J2ME*. Developed in the *Java Community Process* under the name of *JSR 172*, the *J2ME Web Services API* (*WSA*) extends *Java 2 Micro Edition* platform in order to support web services [8].

The *J2ME* web service implementation offers two optional packages based on XML that can be used to develop Java web services for mobile devices, such as cellular phones or PDAs: one for remote call services (*Java API for RPC* based on XML or *JAX-RPC*) and one for XML parsing (*Java API* for XML processing or *JAXP*). The packages are independent one of the other. The APIs interact in order to provide access to the web services through profiles from *Connected Device Configuration* (*CDC*), or from *Connected Limited Device Configurations* (*CLDC*) [9].

The module destined for the *TM* client was developed in *NetBeans 5.5* for the *SUN Java* (*TM*) *Wireless Toolkit* platform 2.5.1, in the *Java* language.

Next, the most important screens of the application are presented:

– Figure 4.a presents the authentication screen (web service call, authentication)
– Figure 4.b presents the correct authentication screen (with the userid in the title).

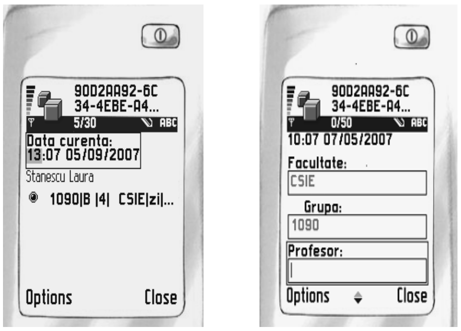

ASE
5/30 ABC
Data curenta:
13:07 05/09/2007
Utilizator:
laurast
Parola:
*****
Options Close

90D2AA92-6C
34-4EBE-A4...
5/30 ABC
Data curenta:
13:07 05/09/2007
Options Close

a) b)

Fig. 4. User authentication

– Figure 5.a presents the student profile information screen (web service call, the *getStudent* operation)

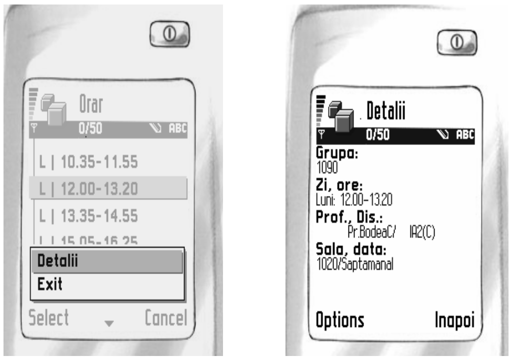

90D2AA92-6C
34-4EBE-A4...
5/30 ABC
Data curenta:
13:07 05/09/2007
Stanescu Laura
◉ 1090|B |4| CSIE|zi|...
Options Close

90D2AA92-6C
34-4EBE-A4...
0/50 ABC
10:07 07/05/2007
Facultate:
CSIE
Grupa:
1090
Profesor:
Options Close

a) b)

Fig. 5. Student profile and interrogation

– Figure 5.b presents the schedule interrogation parameter screen (Data (date), Facultate (faculty), Grupa(group) fields – automatically filled).
– Figure 6.a represents the selection screen of a schedule element for the visualization of details)



L | 10.35-11.55
L | 12.00-13.20
L | 13.35-14.55
L | 15.05-16.25

Detalii

Grupa:
1090
Zi, ore:
Luni: 12.00-13.20
Prof., Dis.:
Pr.BodeaC/ IA2(C)
Sala, data:
1020/Saptamanal

Options Inapoi

a) b)

Fig. 6. Schedule details

– Figure 6.b represents the Schedule details screen (title, professor's name and the subject are in the abridged form).

5 WAP Based Software Information Application

The second application of the information system has the purpose of making good use of the WAP protocol and providing both students and professors with essential information. In this section we describe how the application was developed on a PC and how it was set up on a server. A WAP enabled phone is necessary, that implements the WAP 1.x or WAP 2.0 standard.

5.1 Developing the Application

We have followed a client-server architecture supported by standard Internet protocols. This technology allows building dynamic Web pages on the server from a non-static information stored in a database. This information is previously supplied by the system administrator or student/professor input (in the case of messages). The application was developed using an *Apache* server with client-side support for *WML* content and server-side support for *PHP*. Using the *WML* standard we can develop pages that can be dynamically generated with *PHP* and served by *Apache*. A *MySQL* database is used to obtain information about students, professors, subjects, etc. The bundle of files with the *.wml* extension is uploaded onto the server configured to serve *WML* content. The *Opera* browser was used to test the application, being one of the best *WML* renderers. For the purpose of this presentation the *Openware* phone simulator is used. Its phone-like shape creates the best image possible of how the application will look on a mobile phone. Also, *Nokia Mobile Toolkit* was used to write the code, although it does not help with error alerts at all.

5.2 Setting Up the Application Environment

The installation of the *Apache* server is needed on the server (the 2.2.4 version was used) and of the *PHP* module (*PHP* version 5). Also installation of *MySql Server 4.1* is needed. This choice of development software is very popular amongst web developers, and it can be successfully applied to wireless *WML*-programming as well.

5.3 Running the Application

The application caters for two types of users: a professor and a student. Both of them have different options and possibilities on the application menu. Access is granted after authentication. The accounts are predefined and are outside of the application scope. In case of wrong authentication, for whatever reason, the application warns the user of the mistake made and requests the correction of the entered data. Once the user has accessed his account, the are several options provided to the student. A similar menu is available to the professor. The used images presented in the application are *WBMP* pictures that can be integrated in the *WML* page.

The application provides support for a correct authentication. In addition, the password can be modified as a security measure. The message options provide *INBOX*, message writing, message deletion possibilities. Also, the professor can send “spam” messages to a whole year, series or group of students. This way, a professor can advertise oncoming classes for the next semester or year, for example. A student can check for grades from the current year or the previous ones and can send or receive messages. Both type of users can check their schedule, an option that can prove very useful if they are on the move.

6 Conclusions

The paper presents two mobile applications designed for the academic environment. These two approaches have been considered for designing, developing and testing the applications in relation with the different features of devices and environments of the mobile user. The first application uses Web services to supply the information to Web services consuming mobile clients such as Pocket PCs and mobile phones. The second software application of the system uses WAP and WML technologies to make a point about the simple implementation of a very useful program that leverages the wireless web.

As a future development of the system we will integrate the mentioned two applications and we will develop a notification/alerts module that will incorporate push technology to initiate notification messages from the server which can send SMS messages to users so that they to connect to the information server.

Further on, the presented applications will be developed and put together into a “Integrated System for the Mobile User” that will be tested, first of all, in the academic environment. The main reason for developing these mobile applications is to simplify students’ lives in a permanent dynamic and changing academic world.

Acknowledgments. This paper was supported by the Romanian Ministry of Education and Research, under the research program CEEX no. 80/10.10.2005 SIUM.

References

1. Siau, K., Nah, F.: Mobile Technology in Education. In: IEEE Transactions on Education, pp. 181–182. IEEE Press, New York (2006)
2. Nah, F., Siau, K., Sheng, H.: The value of mobile applications: A utility company study. Communications ACM 48(2), 85–90 (2005)
3. Sheng, H., Nah, F., Siau, K.: Strategic implications of mobile technology: A case study using value-focused thinking. Journal of Strategic Information Systems 14(3), 269–290 (2005)
4. Haddon, L.: More Than a Phone: Emerging Practices in Mobile Phones Use amongst Children. The Mobile Information Society, Budapest (2007)
5. Houghl, D., Zafar, K., Brown, M.: Essential WAP for Web Professionals. Prentice-Hall, Englewood Cliffs (2001)
6. WAP Protocol, http://en.wikipedia.org/wiki/Wireless_Application_Protocol
7. WAP 2.0 Technical White Paper, http://www.wapforum.org
8. Ortiz, C. E.: Introduction to J2ME Web Services (2004), http://developers.sun.com/mobility/apis/articles/wsa/
9. White, J.: Turn Your J2ME Mobile Devices into Web Services Clients (2005), http://www.devx.com/wireless/Article/28046

Modeling Trust Relationships in Collaborative Engineering Projects

Carlo Argiolas[1], Nicoletta Dessì[2], and Mariagrazia Fugini[3]

[1] Università degli Studi di Cagliari, Dipartimento di Architettura, Piazza D'Armi, 09124 Cagliari, Italy
argiocar@unica.it

[2] Università degli Studi di Cagliari, Dipartimento di Matematica e Informatica, Via Ospedale 72, 09124 Cagliari, Italy
dessi@unica.it

[3] Politecnico di Milano, Dipartimento di Elettronica e Informazione, Piazza da Vinci 32, 20133 Milano, Italy
fugini@elet.polimi.it

Abstract. Engineering projects may involve the efforts of specific partners experts in various disciplines. Grid and Web service technologies enable engineering project teams to perform resource discovery, selection, negotiation, and composition in a networked environment. However, the need arises for trust relationships, allowing team entities to participate in the project under security guarantees of privacy, of authorized disclosure of project data, and of correct users' credential management. This paper proposes a Web service based framework to support this process. Giving prominence to the needs of a secure and dynamic collaboration in the style of a Virtual Organization (*VO*), the paper identifies and models the trust aspects featuring actors and objects involved in engineering team collaborations. An architecture is illustrated, centered around a Security Manager component supporting security in the composition and execution of distributed engineering services. A prototype implementation is presented.

Keywords: Security, trust, Web services, engineering projects, grid services.

1 Introduction

The area of engineering projects, such as, for example, projects in the building construction industry, are changing rapidly towards environments of networked virtual collaborative type, where collaboration rules define the cooperation policies [1]. This requires newly defined responsibilities and cooperation rules. Processes, models, and tools allowing for a better use of computing and networked resources could offer a feasible solution for interconnection, integration, and large information source sharing during project management processes. Recent ICT improvements accelerate the trend towards collaboration, which can be enhanced by network services, such as email, chat, file sharing, the web, mobile agents, or the grid. Although important for collaboration, these services seem not enough for engineering project teams, whose components participate in several projects and consortia at the same time, and can join or leave a consortium depending on their assigned responsibility. Moreover, they need to share large data quantities, and business processes, services, and markets, in mutual competition or under strategic alliances or outsourcing. In such a business environment, the process models need to take security into consideration, since totally open cooperation approaches are not always feasible. The philosophy of engineering teams is based on a collection of economically motivated partners (e.g., structural designers, engineers, technicians) that cooperate in carrying out loosely coupled co-operations to meet the varied construction process management needs. Inherent in this philosophy is the shift from the concept of "computing resource" to the concept of *Virtual Organization* (*VO*), which combines the resources of many computers, so that they are presented to the team participants as a single, wider, and networked accessible resource pool. This approach is especially effective in conjunction with the use of more widely deployed infrastructures, such as Web service applications [2] and Grid computing [3]. These infrastructures enable engineering project teams to perform resource discovery, selection, negotiation, and composition. However, a growing need arises for the constitution of *trust relationships*, allowing team entities to participate in an engineering process obtaining efficient project information while preserving security and privacy of some data sets, as well as local security policies according to the local ICT platforms. In fact, team protagonists usually work with different software systems, data models and formats, and operating systems, and, deal with information at different security levels. For example, some operative data about construction computation are public, while costs of some project components need to be kept private to each team.

Generally, the implementation of security mechanisms (i.e., certification, authorization, or cryptography in communications) appears as an overlay with respect to the policies in place for single teams. For example, Grid computing research has produced security mechanisms and policies [4,5] that require resource providers to outsource certain policy controls to a thirty party, which acts as a coordinator. It has been stressed [6] that this approach is not suitable for a *VO*, whose dynamic nature introduces security issues, such as remote authentication and access control, federated identity management, and secure data exchange among differently cleared partners and resources at different security levels. Additionally, the resource access policies can be heterogeneous, since each team administers its computational resources on the basis of local policies while the need for *federated access policies* arises [7]. Furthermore, participating teams may adopt different security models that need to interoperate based on different levels of trust in order to support short-lived collaborations as well as long-lived projects.

The described scenario requires secure access to data and secure interaction between partners, and drives the need for integrated security models and architectures. Although ad hoc solutions are possible, this paper aims at exploring the potential for a general approach dealing with multiple security facets of a trusted environment for the collaborative management of engineering construction projects.

Analogously to the approach presented in [8,9], we adopt a process oriented view in order to facilitate the cooperation of the distributed teams: each collaboration activity is modeled as a process that is described by an abstract process, or workflow.

Each workflow is made individually executable as a set of granular services [10]. In particular, the environment for engineering projects is based on *orchestrated* Web services, that is, all the available resources are modeled as Web services, or as compositions thereof, within the context of a process describing the project and fixed at project start time. Accordingly, local project tools, resources, and data repositories are modeled as Web services cooperating within the overall process. They can be automatically accessed in a remote way by the team components, according to the orchestration and the security rules defined for them in the process definition [11]. In particular, we define a security framework for such environment, allowing the teams to access and share the environment resources in a controlled way, namely, according to their level of trust, to their personal credentials, and to their role in the project. The novelty is that security dimensions are associated to agents and resources in a dynamic way, respecting the virtual collaborative environment, and that a uniform modeling and architectural solution is given. Such security associations is coded in the process description and handled at run time by a security manager component.

The paper is organized as follows. In Section 2, we overview related work. In Section 3, we present the basic aspects of a secure collaboration among engineering teams. In Section 4, we define the elements of our trust framework that models trust in collaborative projects, namely actors, objects and trust dimensions in an engineering team. Sections 5 and 6 illustrate a reference architecture for one site of the collaborative team and present the basic security features of the prototype. In Section 7 we give the concluding remarks.

2 Related Work

Related work regards research in *VO* environments, and in distributed authorizations for federated environments, especially in Web and Service Oriented Systems.

Security is often equated with access control, which consists of authentication and authorization, and it is realized in such a way that unauthorized operations are identified in advance and subsequently forbidden. However, predefined access rights contradict with the openness and adaptation requirements of dynamic systems [12], of the type of *VO* systems.

In the general setting of a *VO*, principals are distributed in different trust and management domains which can span governmental, industrial and academic organizations, and whose security requirements need deep analysis [13]. These principals are also ad hoc related to one another. This is because: i) a *VO* usually does not have a reliable control over a principal as a real organization does over its employees and assets, (ii) these principals need not maintain a responsible relationship to one another as ones should in a real organization, and (iii) a *VO* is dynamic: it usually comes up into being, growing, diminishing or terminating, in a un-predetermined manner. Despite the ad hoc and dynamic properties, grid computing needs strong security services. In addition to usual security services for conventional distributed computing to protect mainly owned or organizationally controlled assets against external adversaries, a principal in grid computing also has interest on a platform which is out of the principal's ownership or organizational control, and the needed protection is often against the very owner of the platform.

Among typical grid and *VO security problems*, we mention security for users, security for resource providers, and *VO* security policies. For example, a *VO* policy may be that any participant must not be able to disseminate certain data owned by the *VO* environment outside the *VO*. The difficulty is the conformity of the policy to be maintained despite the ad hoc nature of the *VO*. For example, even with little control over its members, a *VO* must still be able to remove a member without letting *VO* data be taken away. Concerning audit, any misuse of resource by users, and compromise to users' data and/or computations possibly by a privileged entity at a resource provider, must be detected in a undeniable manner. Recently, in the context of self-healing services, that is, capable of self-management actions, such as repair from a fault or reaction to an attack, security frameworks based on pools of trusted Web services is receiving attention (see [14] for an overview of issues and a proposal). For access control, the abstraction of *roles* has been previously used in the Web and distributed environments [15]. In [16] a framework is proposed for regulating service access and release of private information in Web services, while [17] discuss in depth privacy, authorization and authentication of Web users and services. Various approaches to security of Web systems and applications, and in general to technical and organizational security are reviewed in [7]. The various proposals are a sound basis to construct an integrated approach to manage trust and security in a *VO,* although they need to be tailored to our dynamic scenario of cooperative engineering.

3 Secure Collaboration in Engineering Teams

The management of an engineering project is a very lengthy and expensive process that usually involves not only geographically distributed teams but also teams of specialists in different areas, working with different software tools. The interoperability of those tools is still an issue in the industry and is a mandatory requisite for any viable collaborative solution. Due to their huge complexity, projects are divided into smaller interrelated subprojects, each dealing with an abstract representation of the others. Each subproject can be modeled by a process, or workflow, that expresses a sequence of cooperative tasks. When the cooperation occurs among geographically distributed teams, collaborative project management can be carried out by activating a number of services which instantiate the workflow and its tasks, independently of the physical distribution of the resources. Building design, structural analysis, engineering project *scheduling* and *risk analysis* are just some examples of possible project tasks that can be carried out in a network of cooperative e-nodes interacting to offer or to ask for specialized services. Each task can be decomposed and be made available as a Web service individually; Web services are then composed into an orchestrated process, where individual tasks cooperate as Web service invocations. By providing a service based approach to support distributed collaborative engineering, we can greatly reduce the difficulty and costs in engineering projects. The logical process followed by the project manager when he designs the workflow is based on a "structured" vision of the project: he is well aware that the selection of the cooperative partners and services has to be fixed at design time choosing among a number of available teams and services and according to an a-priori composition pattern, which is supported by a proper infrastructure.

From this viewpoint, the workflow can be regarded as the enactment plan of a collaborative process, consisting of a number tasks and potentially resources (data, files, applications, resources, and so on), which are spread across multiple distributed teams and basically out of control of the project manager. The workflow execution has to support different process phases, such as discovery, selection, and monitoring of services. Basics of Web service technologies (e.g. WSDL, SOAP, and UDDI) provide ways to automate and streamline distributed business processes. The combination of Web service technologies can facilitate the execution of autonomous and heterogeneous applications, using data and services residing in distributed environments. Security aspects in cooperative engineering projects consist in keeping important information in the hands of authorized users and basically deal with ***authentication*** – being able to verify identities; and ***authorization*** – limiting access to resources to selected users or programs.

Currently, Web browsers and Web servers support authentication and authorization by the creation of a limited lifetime private key and a certificate pair, known as a proxy, which can be used to authenticate access to Web resources. Users are authenticated and authorized to use the resources of the application against their proxy certificate. Different trust levels exist in a collaborative project: some activities involve with close collaborators, while others involve people with known qualifications but total unverified identities. Hence, the distributed project approach brings about new challenges, including how to evaluate the trustworthiness of a service provider. There are several aspects to be considered in a trust relationship:

integrity – being able to verify that the content of the communication is not changed during transmission

confidentiality – ensuring that only the parties involved can understand the communication;

awareness – allowing users to see each other's identity and observe each other's actions;

reputation – reflecting a collective measure of the effective Quality of Service (QoS) of each team as seen by the whole community of people who collaborate more intensely.

reliability – evaluating if a subject is able to deliver a service with an acceptable certainty.

Moreover, each participant in a project team is assigned to one (or more) role(s) which have an associated set of *access rights* and *obligations*. These are defined not only for the participant but also for each task in the context of a specific project according to the service level agreements and the security policies stated by the collaborating service providers. Consequently, the rights and obligations of a service class or instance are not fixed but rather specific to the context of each collaborative task where the service operates. All these security properties need to be associated to agents and resources dynamically, in a *VO* style, so that the security properties of a Web service executing, say, a structural analysis computation task on a remote site, where powerful computation CAD resources are available, can be assigned dynamically to the Web service (e.g., based on the security level of the executing engineer), and revoked when the Web service terminates. In this way, the overall process is described as a set of Web services containing a security attribute in their WSDL definition, to which a value is assigned dynamically upon Web service invocation through a mechanism of delegation of agent credential and then revocation. One challenge in building a secure environment made of dynamic resources is the integration of various authentication and identification mechanisms in place at each site, maintaining an overall coherence among user credentials, resource access control lists, and other mechanisms and policies.

4 Modeling Trust in Collaborative Projects

We now model the application context by defining as an *actor* each relevant subject capable of executing actions in a cooperative project carried out on networked *resources*, which we consider as objects. Processes, services, data, and tools are resources of the environment. Possible actors are the architect, the structural engineer, the contractors, the product suppliers, and each single team. The selection of a specific resource from an actor depends on his needs and preferences among the various candidate objects available in the network.

Table 1. Actor trust dimensions

Authentication - checking if the actor is the subject he/she claims he is. Depending on the network infrastructure, the authentication can be based on username-password or on X.509 certificates (which store all user information: accounts, rights, groups, and roles, e.g., in RBAC style).
Authorization - allowing the resource owners to define and enforce a given security policy about who (actor) has privileges on which object, and under which conditions.
User Capability Level (UCL) - expressing a user authorization in terms of Security Levels.
Qualification - describing the technical specialization of each actor. For example, an actor (or team) can be specialized in providing structural engineering services or architectural design.
Reputation – aiming at checking whether a team is able to perform specific services while ensuring an effective Quality of Service (QoS) at run time.

Table 2. Resource trust dimensions

Top Secret - Resources that, if made public, can create a high damage and risk to their owner. For example, strategic data about a project, project costs.
Secret: Resources that, if made public, might cause an economic damage to their owner. For example the design of a new industrial building component, studied along years, should be kept secret: if copied, it could make long research efforts void.
Confidential -Information which, if made public, can damage the image of an organization, or could leak operational information that one would not want to share, such as an estimate and resource-based schedule for completing the project.
Restricted - Material whose diffusion is unadvisable. It can include both operational data of low interest and yet usable in a wider context and procedures internal to the organization, not to be made available outside.
Unclassified - public information.

All these resources are instances of a single class that we denote as *Resources*. For *actors and resources* we define suitable security properties called *trust dimensions*.

The *actor trust dimensions* are defined as in Table 1. Analogously, resources are characterized by a single trust dimension, i.e., the *Resource Security Level (RSL)* that is specified as shown in Table 2.

Fig. 1 presents a general trust framework that addresses the challenges posed by trust in Web services. The dashed ovals include usual software components of a workflow-based system: grey boxes depict, respectively, the *process manager* (supervising the workflow execution), the *task manager* (in charge of executing the workflow instances), and the *service manager* (that supervises the successful completion of each task). The ***Security Manager*** component manages and updates the qualification of each actor and the reputation of the service it exposes. Since qualification and reputation change over time, this component performs both long-term and short time management activities.

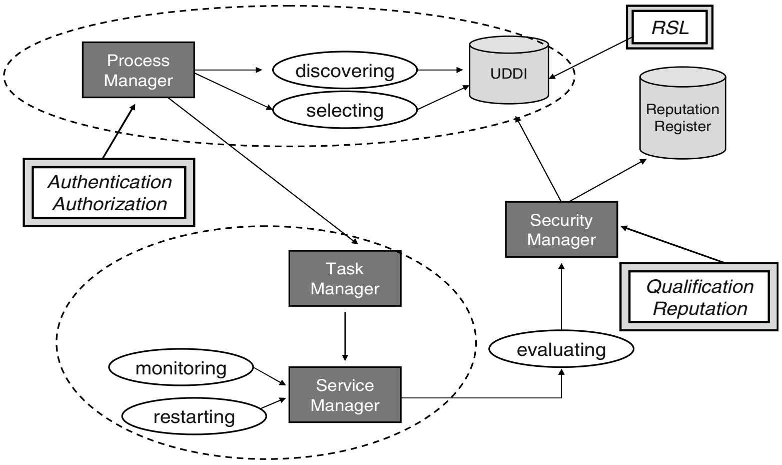


Fig. 1. Components of the security framework

Short time activities are carried on during workflow execution and include updating the reputation of the single services invoked by the workflow on the basis of current information stored in the *Reputation Register*. Long-term activities update the reputation and qualification of each actor, by aggregating and evaluating information related to the evaluation of the offered services. This allows the level of trust between cooperating teams to be *dynamically established* and adjusted to reflect current collaboration. The evaluation is based on QoS annotations to the service description containing information about the service state, collected by the workflow components operating on individual services. Possible annotations are about response time, failure rates, cost of each service, availability, and security. Such information is stored in tables located in the Reputation Register, which is accessed and maintained by the Security Manager. Teams or actors that are less likely to violate their Service Level Agreement (*SLA)* have a good reputation and are more likely to be chosen in the future, when new planning processes are composed. Conversely, when a team with a bad reputation and a low level of qualification starts to behave correctly and/or to expose new specialized and qualified services, the Security Manager will increase the consideration of such a pool. Methods for automatic refinement of reputation are outside the aims of this paper and can be found in [14].

5 A Reference Security Architecture for a Team Site

We present now a reference architecture for a collaborative team site (see Fig. 2). We have defined and experimented its components in order to test its overall behavior for engineering project workflows [11]. The prototype implements one site security. Its basic components are:

- A Security Server
- A Front-end
- A Back-end
- A Workflow system.

The *Security Server* handles security-specific functionalities aimed at data privacy, trust, access control by implementing and managing Authentication and Authorizations, *RSL* and cryptographic functions. These functions are implemented by *Security Web services* at the Site Trust Level (see right-hand side of Fig. 2). These services access local databases storing the security levels of both local and remote services, users and resources. *Security Contracts* [14] model the negotiation of service provisioning. These store the agreed security level; if such level is violated, *Security Contracts* are the basis to revise the trust level of a project server and have to be re-negotiated for subsequent cooperation.

The *Front-End* is composed of a set of applicative *Engineering Web services* that can be invoked by the workflow editor, after negotiation of security performed by the Security Server. Engineering Web services are managed by the Security Server, from which they require all information related to security locally, and during interaction with remote servers.

The *Back-End* is constituted by the local resources of the site Server, (Web services, databases, tools, computation procedures, or project data. Not all the resources in the Back-End are exposed as Web services and can be invoked by remote Security Servers or by a singe remote Web service. In fact, each resource has its own *RSL* assigned by an administrator; the applied policy is "no read up, no write down". Invocations of the Back-End services are protected via SSL or other site-defined cryptographic functions. A workflow does not have an associated global Security Level, but rather each service in the workflow has its own security level that depends on the clearance of the users who develop and manage the Engineering Web services, in the style of a *setuserid* operating system control function.

Authentication is executed through insertion of a secret code by the user requesting the execution of a protected workflow. If the code is correct, the system checks the validity of the security level of the actor w.r.t. the required service. For example, a user with Clearance = Top Secret can enable a strong authentication function; the Security Server will respond to a challenge sent by the client. As an implementation mechanism for access rights, the Security Server uses Access Control Lists, generated each time an Engineering Web service, executing locally, needs to access local or remote resources. Actors and security elements interact as follows.

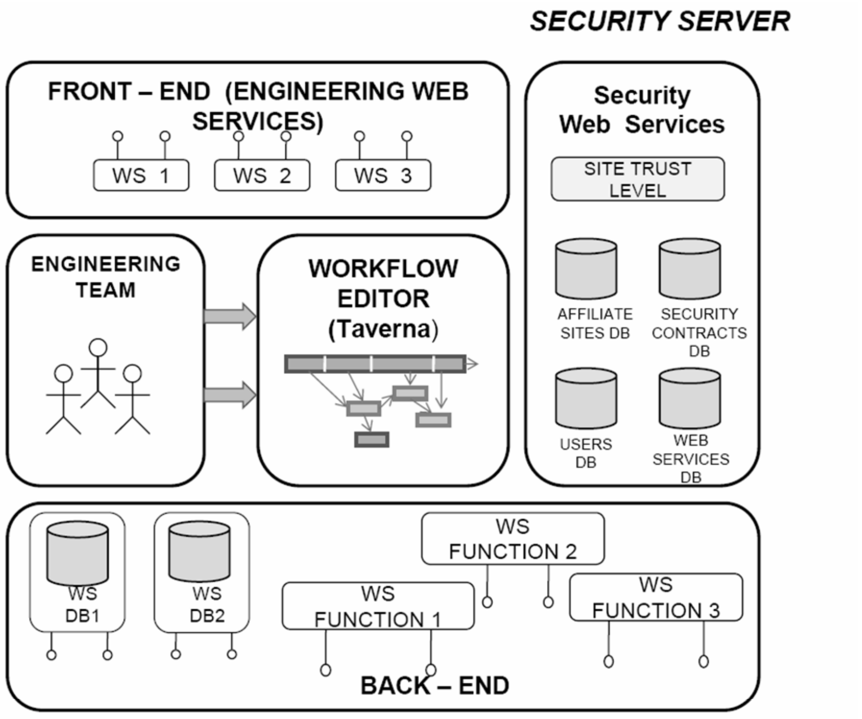


Fig. 2. Collaborative Team Site: one site's architecture

1. Collaborative teams join in a *VO*. This means that an engineering team (represented as a operating within the site of Fig. 2) decides to group for a given time duration in order to carry out a collaborative project. Each team decides which services should be available to the other project partners, by adding such services to the Back-End Web service databases. Each service uses resources to perform operations, and each resource has a *RSL* assigned by the server administrator, according to the policies stated by the Project Manager. Each site has a trust level received from a Certification Authority.

2. Managing a project task requires the creation of its workflow, that co-ordinates the execution of different Web services. After authentication, each actor has his own clearance and his access privileges to Web services and to resources that can be added to the workflow, depending on stated access policies. A service can be added to a workflow only if the *UCL* of the actor is greater or equal than the *RSL* of the resources used by the service. The access rules to resources are based on *RSL* w.r.t. *UCL* comparison and conform to the "no read up, no write down" multilevel security principle.

3. The *RSL* of the required resource is matched against the clearance of the actor requesting to execute a remote Web service; if the security rules are satisfied, access is granted. In local use of resources, i.e., resources belonging to the server that has assigned a clearance C_1 to the user, a secure interaction can take place, again following the "no read up / no write down" rules.

4. When the experiment terminates, the Security Server summarizes security aspects in a log file, to analyze if the Security Contract has been fulfilled, and possibly updates trust levels and reputations of the engineering servers that have been involved in the project.

6 A Prototype Implementation

To evaluate our proposals, we have considered security in developing a collaborative environment. We tested secure cooperation from the perspective of a single team, for which a *Security Server* has been implemented with security functions deployed as *Security Web services*.

The prototype is built on top of Taverna[1], a workflow editor that allows designers to map the initial abstract workflow into a detailed plan. Each Taverna workflow consists of a set of components, called Processors, each with a name, a set of inputs and a set of outputs. The aim of a Processor is to define an inputs-to-outputs transformation. Local security agents have been be installed as processors by adding to Taverna new plug-in processors that can operate alone or can be connected with data and with workflow through control links. When a workflow is executed and the execution reaches a security Processor, an associated task is started that invokes a specific Security Web service. The Scufl workbench [19] provides a view for composition and execution of processors.

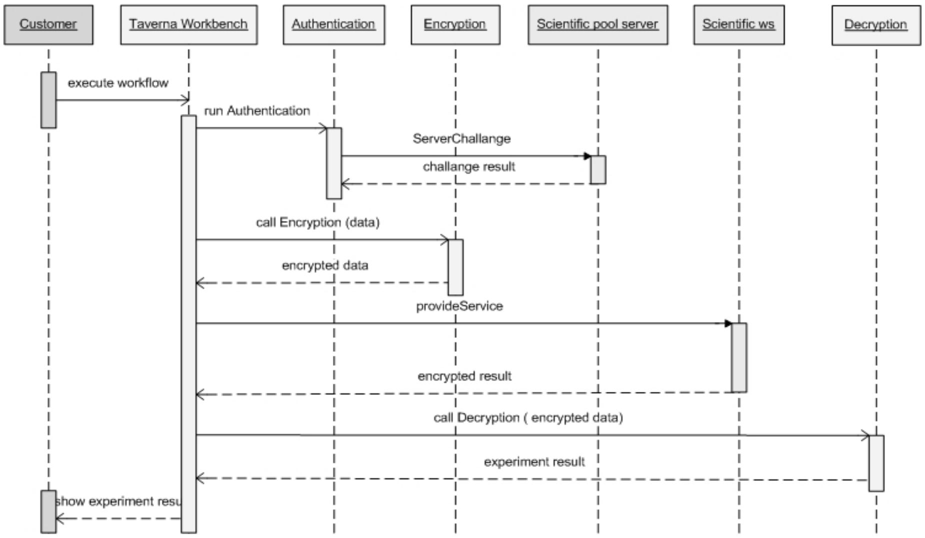


Fig. 3. Sample sequence of interactions for accessing remote services/resources under security requirements

[1] Taverna is available in the myGrid open source E-science environment http://www.mygrid.org.uk/

Fig. 3 shows a sample sequence of interactions among the architecture components. Taverna invokes the services as well as the security Processors, which retrieve from the Security Server a list of resources having a trust level equal or greater than the trust level of the service executor. To perform this retrieval, the Security Server asks Processors to each remote site a list of the available resources (in our implementation the list is available just for the local site). Then it requires the Trust Level of each site contained in the list. Only if the trust level of the remote site is greater or equal than the clearance of the executor, the remote resource will be included in the Trusted Resource List. The Project Web services invoke resources in the list, and then use the obtained result data to perform further computations; it finally returns the global results to the Taverna client.

7 Concluding Remarks

This paper has presented aspects related to secure cooperation in executing engineering workflows. We have highlighted the needs for a secure and dynamic collaboration in a *VO*-style environment, and have illustrated the fundamental components involved in the execution of a project workflow. By focusing on security and trusts aspects, we have defined actors and objects and identified trust dimensions. We have also described a reference architecture, as it has been implemented in a prototype built using some of the *myGrid* environment tools. A major contribution of this work is also the introduction of a framework that supports security aspects in the composition of distributed services in an engineering workflow. The framework separates concerns related to modeling the secure execution environment from workflow tasks execution aspects, related to the execution of services on different distributed engineering sites.

Further research will aim at investigating how security dimensions possibly influence agent-based negotiations and interactions in carrying on specific activities assigned to agents (as proposed in [11]).

Acknowledgments. Part of this work has been developed under the Italian TEKNE Project.

References

1. Argiolas, C., Dessì, N., Meloni, R.: A Service Based Framework Enabling Collaborative Construction Project Management. Joint CIB W65/W55/W86 Symposium, Rome (2006)
2. Proc. IEEE International Conference on Web Services (ICWS 2006), Chicago, Illinois, USA, IEEE Computer Society Publisher, Los Alamitos (2006)
3. Fox, G.C., Gannon, D.: Workflow in Grid Systems. Concurrency and Computation: Practice and Experience 18(10), 1009–1019 (2006)
4. Shirasuna, S., Slominski, S., Fang, L., Gannon, D.: Performance Comparison of Security Mechanisms for Grid Services. In: 5th IEEE/ACM International Workshop on Grid Computing, Pittsburgh (2004)

5. Barton, T., Basney, J., Freeman, T., Scavo, T., Siebenlist, F., Ananthakrishnan, R., Baker, B., Goode, M., Keahey, K.: Identity Federation and Attribute-based Authorization through the Globus Toolkit, Shibboleth, Gridshib, and MyProxy. In: 5th Annual PKI R&D Workshop (2006)
6. Alfieri, R., Cecchini, R., Ciaschini, V., dell'Agnello, L., Frohner, A., Lorentey, K., Spataro, F.: From Gridmap-file to VOMS: Managing Authorization in a Grid Environment. Future Generation Computer Systems 21, 549–558 (2005)
7. Bellettini, C., Fugini, M. (eds.): Security in Distributed Information Systems: Trends in Methods, Tools, and Social Engineering. IDEA Book Publishing (2004)
8. Console, L., WS-Diamond Team: WS-Diamond: an Approach to Web Services – DIAgnosability, MONitoring and Diagnosis. In: Proc. Int. e-Challenges Conference. The Hague (2007)
9. Special Issue on Privacy and Security in Highly Dynamic Systems, Comm. of the ACM, 49(9) (2006)
10. Bosin, A., Dessì, N., Fugini, M., Liberati, D., Pes, B.: The Future of Portals in E-Science. Encyclopedia of Portal Technologies and Applications. Information Science Reference Publisher, New York (2007)
11. Amigoni, F., Fugini, M., Liberati, D.: A Virtual Laboratory For Web And Grid Enabled Scientific Experiments. In: Proc. ICEIS 2007 Conf., Madeira (2007)
12. Sackmann, S., Strueker, J., Accorsi, R.: Personalization in Privacy-Aware-Highly Dynamic Systems. Comm. of the ACM 49(9) (2006)
13. Herrmann, P., Herrmann, G.: Security Requirement Analysis of Business Processes. Electronic Commerce Research 6(3-4), 305–335 (2006)
14. Fugini, M., Pernici, B.: A Security Framework for Self-Healing Services. In: Proc. UMICS 2007 CAiSE Workshop, Trondheim (2007)
15. Barkley, J., Cincotta, A., Ferraiolo, D., Gavrila, S., Kuhn, D.: Role Based Access Control for the World Wide Web. In: Proc. 20th National Computer Security Conference, Baltimore, MD (1997)
16. Bonatti, P.A., Samarati, P.: A Uniform Framework for Regulating Service Access and Information Release on the Web. Journal of Computer Security 10(3), 241–272 (2002)
17. Yao, D., Frikken, K.B., Atallah, M.J., Tamassia, R.: Point-based Trust: Define How Much Privacy is Worth. In: Ning, P., Qing, S., Li, N. (eds.) ICICS 2006. LNCS, vol. 4307, Springer, Heidelberg (2006)
18. Jøsang, A., Ismail b, R., Boyd, C.: A Survey of Trust and Reputation Systems for Online Service Provision. Decision Support Systems 43, 618–644 (2007)
19. Oinn, T.: XScufl Language Reference. European Bioinformatics Institute, http://www.ebi.ac.uk/tmo/mygrid/XSculfSpecification.html

Loosening the Hierarchy of Cross-Company Electronic Collaboration

Christoph Schroth[1,2]

[1] University of St. Gallen, MCM Institute, 9000 St. Gallen, Switzerland
[2] SAP Research CEC St. Gallen, 9000 St. Gallen, Switzerland
christoph.schroth@sap.com

Abstract. Today, the organization and implementation of cross-organizational business relationships on the basis of electronic services still implies high costs for the seamless communication and coordination of those services. These costs often lead to centralized and inflexible service structures. To investigate major enablers for decentralized "service markets" which allow for increased flexibility and productivity, we draw an analogy to both the societal and the business context. The hypothesis underlying this article is that the organizational evolution of services shares strong similarities with the development of societies and businesses which have evolved from unconnected groups of people over large hierarchies to more decentralized forms. On the basis of case studies, we show that this hypothesis holds and provide recommendations for improved electronic cross-organizational collaboration. We particularly propose a reference architecture for electronic collaboration which is currently being applied in the context of Swiss public administration and show its potential for performance amendment.

Keywords: Cross-Organizational Collaboration, Reference Architecture, SOA.

1 Introduction

Seamless cross-organizational collaboration is gaining significant momentum and facilitates the emergence of a globally networked service economy [1]. However, the organization and technical implementation of such business relationships still shows considerable weaknesses with respect to productivity, quality and flexibility. Business processes are often unstructured, unclear terminology prevents from a common understanding, functional as well as non-functional parameters are rarely formalized and standardized, and the frequently manual execution lead to a huge variability and hard manageability of results. Emerging technologies today support the setup of seamless connections between electronic information-intensive services (EIIS) across corporate boundaries [2, 3]. However, involved stakeholders still face high costs for the seamless communication and coordination of electronic services, often leading to highly centralized service structures where few influential stakeholders prescribe standards and protocols which other stakeholders have to comply with.

In this work, after defining relevant terms and research methodologies applied, we briefly revisit the organizational evolution of societies and businesses from unconnected groups of people over large hierarchies to more decentral forms (*section two*). We investigate major enablers and driving forces of the respective developments to derive a generalized model which we apply to the field of cross-company EIIS. The hypothesis underlying this work is that the organizational changes of human co-existence (societies), human work (businesses) and electronic services share significant similarities. On the basis of case studies, we prove that this hypothesis is true: From unconnected services which follow a self-supply philosophy, lowered service communication and coordination costs allowed for the setup of (hub-based) service hierarchies (*section three*). Today, even more diminished costs enable the loose and decentralized coupling of EIIS, leading to "service markets" with a high degree of individual freedom, flexibility and specialization. In *section four*, we present a reference architecture for electronic business media which support a wide-spread decentralization of service organization. To prove its real-world applicability and to show its performance improvement potential, we apply it to a specific case in the context of Swiss public administration. We conclude this work with a brief summary and outlook on future work (*section five*).

2 The Decentralization of Societies and Businesses

2.1 Definition of Terms and Research Methodology

Numerous scholars [4, 5, 6] have worked on differentiating services from products, resulting in various kinds of characteristics which they consider as unique to the world of services. The service definition underlying this work is as follows: A service is considered as activity that is performed to create value for its consumer by inducing a change of the consumer himself, his intangible assets or his physical properties. In specific, information-intensive services can be described as activities that are conducted by either machines or humans and deal with the "manipulation of symbols, i.e. collection, processing and dissemination of symbols – data, information and decisions" [5, p.492]. **Electronic information-intensive services** (EIIS) finally represent automated manipulation of symbols, i.e. applications which electronically store, process or issue information on request and to provide value for the respective service consumer. The focus of this study lies on information intensive services which are provided and consumed across corporate boundaries (e.g., the collaborative creation of documents or order and invoice processes).

Studies of cross-organizational collaboration can be traced back several centuries: In the 18th century, Smith [7] argued that the division of labor among agents facilitates economic wealth. Through their specialization on limited scopes of duties, productivity can be improved as the repeated execution of the same activity helps tapping existing potential for optimization. However, the collaboration between different stakeholders implies certain effort which is often referred to as **transaction costs**. The theory of transaction costs was mainly founded by Coase [8] who suggested that the exchange of goods (physical as well as informational goods) or services via markets leads to costs which may result in hierarchical organizations as preferable forms of organization (rather than markets): „The main reason why it is profitable to establish a firm would seem to be that there is a cost of using the prize mechanism" [8, p. 390]. Mainly three drivers for the existence of transaction costs have been identified by North [9]:

Uncertainty through the ***unpredictability of future events.*** The transaction of goods and services underlies uncertainty, for example with respect to the future value of transferred goods and thus induces costs for estimating prices.

Bounded rationality inherent to human agents. Even if all future events were predictable, human agents have a limited capacity in deciphering their environments and thus induce further transaction costs.

Asset specificity indicates the degree to which investments are required for the execution of a transaction which are solely useful for this specific transaction and loose their value in the context of other transactions.

2.2 The Evolution of Societies and Businesses, Impacted by Decreasing Transaction Costs

In the following sections, we briefly revisit the evolution of societies and businesses (thereby strongly adhering to Malone's findings [10] as well as transaction cost theory as theoretical framework) in order to derive essential developments, responsible forces and milestones.

Societies have evolved from isolated tribes of gather-hunters [10], which lived in small groups and followed a comprehensive self-supply strategy and thus enjoyed a high degree of freedom. As hunting resources were scarce, tribes lived in large distances from each other, which made communication difficult (leading to high transaction costs). After people switched from hunting to agriculture, they lived more closely together, thereby reducing the distances which needed to be covered to transfer a message; also, novel technologies such as writing emerged and facilitated information exchange even more. Through lowered costs for communication, the principle of division of labour could be introduced, allowing for economies of scale and individual specialization. To organize this collaboration which also strove along with military advantages, rigid hierarchies were installed which mostly comprised one king which gathered all knowledge necessary to efficiently manage the large number of subordinates. Disadvantages of such forms of organization included that leaders frequently kept large share of commonly created values for themselves, while the individuals lost the autonomy they had enjoyed before joining (or being forced to join) hierarchical kingdoms. As soon as information communication costs (and consequently transaction costs) declined further through novel technologies such as the Gutenberg machine [10], large groups of people had knowledge which was previously only available to kings. It was not further necessary to have one single decision maker: Instead, democratic forms of societal organization became possible which allowed people to retain economic and military benefits of large organizations, while regaining some of the freedom and flexibility they had given up long before [10].

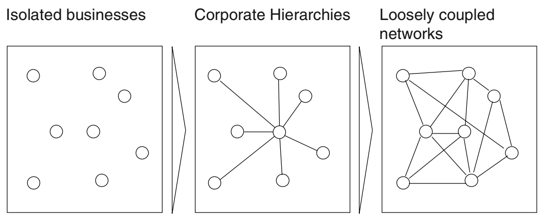


Fig. 1. Organizational evolution of businesses [10]

Very similarly to societies, **business organizations** have evolved during the past centuries: In the first place, small and unconnected family businesses with a high amount of personal freedom existed which mostly focused on agriculture. As costs for communication and physical transportation declined significantly through technologies such as the emerging railroad and the telegraph, businesses had the possibility to change their way of organizing themselves. Markets for trading their products became bigger and richer as well as faster means of communication allowed firm executives to coordinate large-scale operations. Through increased and simplified information exchange, the above discussed drivers of transaction costs such as uncertainty and the limited human rationality were reduced heavily. While technologies made the emergence of large corporate hierarchies possible, several factors made it also desirable (Malone, 2004): Large corporations could, similar to kingdoms in the societal context, leverage economies of scale through division of labor and tight coordination of the different involved stakeholders. Technologies such as the moving assembly line allowed for unprecedented productivity through mass production, but required centralized companies, controlled by a hierarchy of executives. Possibilities to decentralize work and still benefit from close collaboration and economies of scale were not available at that time. Throughout the 18th century and the early 1900s, business communication mostly occurred through one-to-one media (face-to-face communication, letters, memos, telephone calls). Businesses didn't truly begin to make significant use of one-to-many or many-to-many means of communication until the second half of the 20th century: Fax, e-mail and the Internet finally allowed the wide mass of people to gather, create and disseminate a greater wealth of information on many subjects than was available to even the most elite decision makers at the tops of huge organizations like IBM, etc. a few decades ago [10]. Similar to the emergence of democracies, the reduction of communication costs (and subsequently transaction costs) allowed the establishment of loosely coupled business networks (see Figure 1), where decisions could be made on a peer level rather than by a few executives controlling a huge hierarchy. Technologies allowed for the decentralized organization of work, and several factors made its desirable in certain cases: Economic benefits of such organizations include the increased individuals' motivation (due to increased responsibility), creativity and also flexibility, while non-economic benefits comprise individual freedom (which employees had to give up in large corporate hierarchies) and general individualization.

Summing up, we have seen a basic three-stage pattern in both societies and business organizations. In stage one, agents operate in small, unconnected groups. In stage two, large groups are formed and decision making becomes more centralized. In the third stage, the large groups remain, but decision making becomes more decentralized. For both societies and businesses, lowered costs for communication and coordination were the key enablers for a switch from one stage to the other. The central hypothesis underlying this work is that the same is true for the management of EIIS: In a first stage, they are managed in an isolated fashion and also follow a "self-supply" philosophy. Then, with declining communication and coordination costs, the benefits of tight collaboration (division of labour, specialization and economies of scale) motivate service operators to setup centralized service hierarchies where one stakeholder determines and controls protocols as well as standards. Not until simplified communication and coordination heavily lowers electronic transactions costs, decentrally organized service networks (also referred to as Mash-ups [11]) emerge as a third and final step of an EIIS decentralization continuum.

3 The Decentralization Continuum of EIIS

Similar to the ancestors who chose to give up their freedom to receive the benefits of bigness in their political lives as well as the workers who gave up the freedom and informality of family farms for the unprecedented prosperity of corporate hierarchies, the organization of information services has recently moved towards service hierarchies (see Figure 2): The costs for the communication and coordination between EIIS have declined significantly through the emergence of new technologies. Standardized message formats such as UN/EDIFACT[1] or later XML-based formats or Internet-related transmission protocols such as HTTP heavily simplified structuring, correctly interpreting and sending information form one service to the other. In this way, the uncertainty (and thus transaction costs) of stakeholders with respect to data representation formats and transmission protocols was significantly reduced. Such technologies also simplified the work of programmers who - like all human beings according to Simon [12] - suffer from a limited capability in deciphering their environments. Last, through worldwide standardization efforts (e.g. the EDI standard promoted by the United Nations, UN/EDIFACT), asset specificity of required solutions was reduced as well, reducing transaction costs even more.

Inspired through simplified service communication and coordination, different stakeholders operating their so far proprietary and freely designed EIIS gave up the freedom of design of their technologies for the unprecedented operational efficiency inherent to the seamless interconnection - and the increased, central control - of their services. Benefits of connecting EIIS tightly are manifold, but basically again resemble those identified in the societal as well as organizational context: "When a supplier and a procurer use information technology to create joint, interpenetrating processes, at the interface between value-added stages, they are taking advantage of the electronic integration effect" [13, p.488]. Services exchange information such that each of them has up-to date data available. By linking "the supplier's and procurer's inventory management processes so that the supplier can ship the products "just in time"" [13, p.488], for example, inventory costs for both involved companies can be reduced. Other business benefits include real-time data transparency between companies and data consistency as well as prevention of redundancy. Also, a large

[1] http://www.unece.org/trade/untdid/welcome.htm

"ecosystem" of EIIS allow the involved companies to share the costs for setting up certain technology, agreeing on certain information transaction standards regarding process patterns, data formats and semantics (data mapping efforts between the proprietary services usually represent a major transaction cost driver), again lowering the overall effort required to interlink different EIIS. Benefits of tight collaboration such as individual specialization (information services rely on automatically delivered information from other remote services rather than following a self-supply philosophy), economies of scale (services can be reused by a number of other services) and the resulting division of labour between EIIS generally leads to increased productivity and quality of information products which are collaboratively created across company (and system) borders.

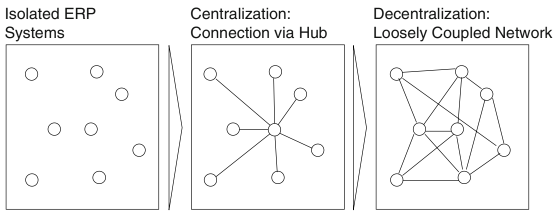

Isolated ERP Systems
Centralization: Connection via Hub
Decentralization: Loosely Coupled Network

Fig. 2. Decentralization Continuum of EIIS

As identified in international studies [14], the early forms of service interconnections were mostly organized centrally: One or a few experts are in charge of setting up necessary infrastructure, driving the process of defining a commonly acceptable data exchange format, building interfaces and more. These experts are also in a position to finally prescribe a technical standard which all connected members have to follow. Huge retail companies or automotive companies, for example, are able to convince (and partly force) their suppliers to adopt certain interface standards and transmission protocols; message exchange patterns are defined and executed centrally. Individual service managers loose their design freedom (are forced to adhere to centrally prescribed technical standards and even message exchange choreographies), while the leader (the player coordinating the electronic collaboration) often keeps a lot of the collaboratively created value (the economic benefits of tight and seamless service collaboration) for himself. Apart from that, individual operators of EIIS may experience a technology lock-in effect. After adopting the solution of one central service coordinator, it becomes expensive to switch to another one, thereby even extending the leader's power and thus the economic disadvantages of the participants. A more decentral, "democratic" organization of service interconnections was and still is not possible in many cases as technologies are not mature enough to sufficiently decrease service communication and coordination costs: Especially small and medium sized companies are still not able to conceptualize, collaboratively agree on certain standards and eventually deploy inter-organizational service connections on their own as transaction cost drivers are high. They face significant uncertainty (e.g. with respect to standards which vary with respect to scope, industry focus, granularity as well as adaptability and may become obsolete after a short time), still have to deal with high technical complexity (which is counter-productive as EIIS operators only have a limited capability in deciphering their environments [9]) and would have to invest in asset specific technologies which cannot be simply reused in many different kinds of transactions.

Table 1. B2B Software and Services Market Overview (Selection) [15, 16]

Vendor	Software			Service	
	Multi-enterprise SW	MFT	EDI Transl.	IaaS	B2BPO
Axway	x	x		x	
Crossgate	x			x	
E2Open				x	x
eZCom	x			x	x
Hubspan				x	x
Inovis	x	x	x	x	x
Seeburger	x		x	x	
Sterling Com.	x	x	x	x	x
TietoEnator				x	x

Table 1 provides a brief overview of emerging software products and services and their respective vendors in the field of business-to-business (B2B) integration [15, 16]. **Multienterprise software** (rich software for coupling of EIIS across corporate boundaries), Managed File Transfer (**MFT**) Suites (focused software allowing for secure and reliable exchange of documents between organizations) and traditional **EDI** Translators (for interpreting and transforming EDI-based information) represent the major classes of available software products today. A novel market has been established in the field of Integration-as-a-Service (**IaaS**): Vendors offer integration capabilities (services in the fields of communication, trading partner management, technical integration (towards internal systems) and different application services) as a service, mostly in multitenant environments over the Web. Such integration services (also referred to as B2B Hubs) which are also available for small-and medium sized enterprises herald a new era of more decentralized EIIS organization. Readily available adapters for the mapping between different existing formats (which may reflect individual requirements best), support for the modelling and automation of collaborative processes and the provision of a common communication infrastructure clearly reduce the costs for communication and coordination between services and thus allow pushing decision making about service design and execution patterns down the hierarchy. It is not further necessary to have one central instance to fully coordinate service interfaces, related processes, data formats and semantics, message patterns and relationships/ governance in place. Instead, with the help of B2B platforms, decision making processes with respect to design and operation can be conducted in a more collaborative fashion by a greater number of stakeholders. We refer to this type of EIIS organization as "l**oose hierarchy**" [10]. B2B Project Outsourcing (B2BPO) finally describes the outsourcing of integration projects (not only the integration infrastructure, but also the people and their organization).

However, these offerings still exhibit major shortcomings regarding their support for decentrally organized service markets. Many “B2B communities” are still being setup as stand-alone island solutions for specific purposes. However, the frustration of organizations in establishing and supporting multiple, single-purpose portals and partner communities grows. “These communities and their underlying networks are not configured to support heterogeneous processes or applications.” [17, p.6] According to Gartner research, firms desire integration services supporting “multiple protocols, multiple data formats, multiple on-boarding approaches, higher-order integration features (for example, in-line translation, data validation and business process management), BAM (for example, process visibility and compliance management) and hosted applications (for example, catalogues and global data synchronization” [15, p.4]. Summing up, **existing solutions** only provide **limited richness** (functional scope) **and reach** (amount of connected organizations).

4 A Service-Oriented Reference Architecture for Decentral Service Networks

In order to allow for loosening up the hierarchies of EIIS organization, we propose a novel, service-oriented reference architecture [18] for electronic business media. The reference architecture comprises three major components (according to the St. Gallen Media Reference Model (MRM) [19]): The physical component (infrastructural basis of the medium enabling interaction on a technical level), the logical component (ensuring a common understanding between interacting agents) and the organizational component (organizing social interaction). Figure 3 visualizes its major components.

Organizational Component: The structural and the process-oriented organization are defined in the organizational component of the MRM: The medium provides a registry of participating agents as well as a role model to define rights and obligations for each participant. To account for the process-oriented organization, each role may execute certain actions (send messages to other roles) depending on the current process status. The organizational component also comprises the specification of the information objects (messages) the agents may send and receive via the medium.

Logical Component: The organizational component is complemented by the logical component which ensures that agents can seamlessly interact on the basis of a common understanding. To this end, the semantics and the structure of exchanged messages are defined and made known to all participants.

Physical Component: As an infrastructural component, the service bus enables the interaction of the different stakeholders by providing the physical means for exchanging messages. The bus relies on the principles of event-driven architectures and provides the services as well as the infrastructure for the actual realization of the “ideas” described as part of the organizational component. Both coordination services (application-specific services which facilitate the interaction of agents) and more operational services (which enable interaction) such as routing services (for routing messages from the sender to the receivers), agent directory services, event catalogue services (providing information about all supported events), abo services (allowing publish-subscribe-based information dissemination) and security services as well as services for handling technical exceptions (Figure 3) are part of this component.

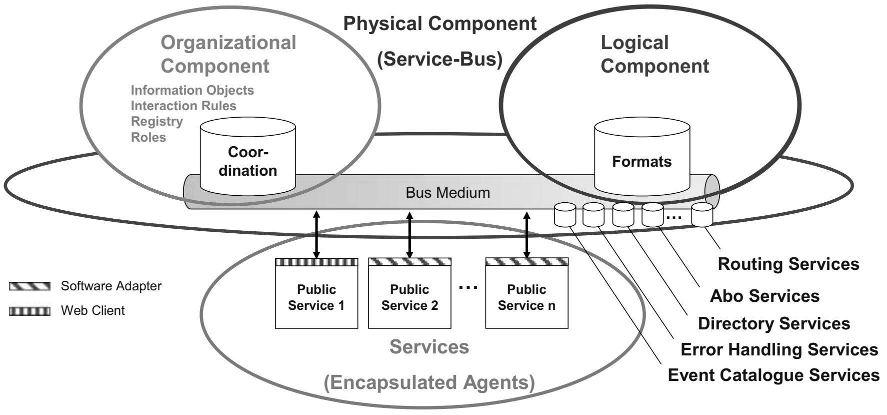


Fig. 3. Service-oriented Reference Architecture

Finally, the actual **agents** (with their associated roles) encapsulate the services they offer with the help of standardized adapters based on web services. In this way, proprietary applications can be made available via the medium, permitting agents to communicate with the business medium using their local software applications. Web Clients represent alternative options for accessing the medium. It is important to mention that with the help of software adapters, **architectural recursivity** can be achieved: Adapters enable seamless mediation between the business medium and local applications, but also between the medium and another medium which again connects a number of agents (in this case, we refer to the adapter as bus connector [18]). As different business ecosystems have different requirements with respect to all three abovementioned components but still need to communicate with each other, this modular, decentral approach is ideal: By relying on a minimal set of globally valid standards (such as a global routing table and a common message envelope), interaction is possible across different business media.

In the course of the Swiss government-funded project HERA [20], this reference architecture is currently being applied. As a first goal, an ecosystem of companies, auditors, accountants and governmental institutions had to be provided an adequate business medium for the organization and implementation of a collaborative tax declaration scenario (right side of Figure 4). In order to avoid the creation of yet another isolated electronic service hierarchy, we applied our reference architecture ("HERA bus") and thus guaranteed seamless interoperability with the media setup by other business ecosystems. In parallel, the governmental SEDEX initiative[2] had started to build an electronic medium for the interaction of different stakeholders in the field of resident data management (middle of Figure 4). By ensuring adherence to the proposed reference architecture, seamless interaction is enabled also across different media. The existence of bus-connectors and the compliance to a nation-wide routing table and common message envelopes (which are base on the Event-Bus

[2] www.bfs.admin.ch/

Switzerland[3]-standard) agents on all different levels of the federal Swiss political system can interact, independent of which medium they are connected to. In case of a resident's relocation, for example, the resident management office of the persons target municipality can seamlessly send a proper message to the proper cantonal tax authority (as the person is likely to be taxable).

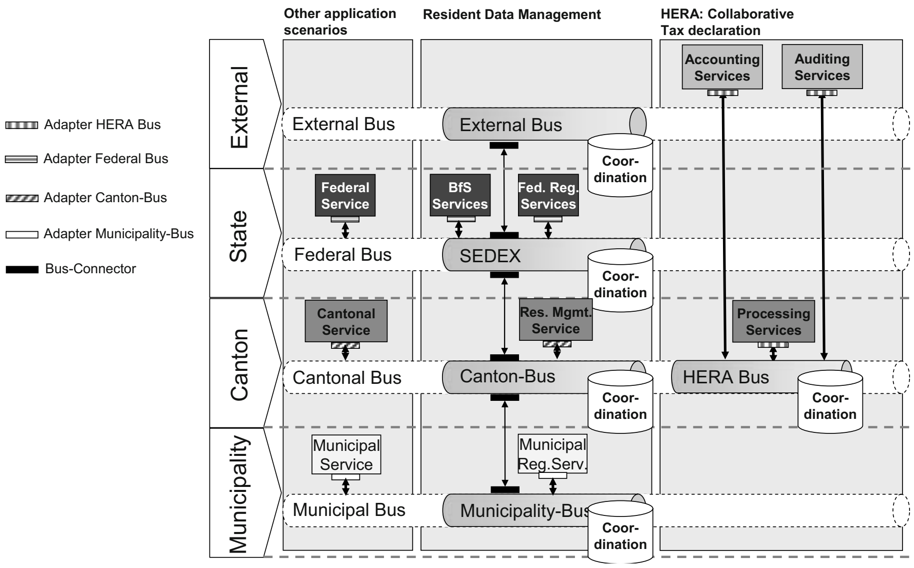


Fig. 4. Decentral network of electronic business media

Again, similar to the evolution of societies and organizations, the reduction of communication and coordination costs has enabled for significant decentralization. Several different factors also contribute to its attractiveness: The creativity of the individual EIIS operators can be better leveraged than in case of centrally prescribed, comprehensive standards and protocols. The flexibility of services and their interconnections is improved as well. Building on the principles modularization and hiding of complexity, services can easily be reused for different purposes if relevant decisions regarding design and execution specificities remain at the individual service operators rather than one central instance. Also, individual freedom allows for a simplified reuse of existing services (which are encapsulated with the help of adapters) as well as a better match of service interconnections and the individual needs (e.g., with respect to individual data formats or coordination services). Summing up, decentral EIIS markets as shown in Figure 4 are enabled trough lowered transaction costs (as a consequence of maturing e-Business technologies) and allow for increased freedom to chose standards, data formats, process-oriented organizational patterns which best suit individual requirements, while keeping the advantage of bigness such as the above discussed electronic integration effect, economies of scale, specialization and division of

[3] www.isb.admin.ch/

labour. However, as already discussed in the societal and in the business context, decentralization is not the preferable option in all cases. Decentral EIIS markets do not represent the better solution unless the benefits of loosening up service hierarchies outweigh their advantages. As argued above, the main advantages of decentralization comprise individual freedom in many respects, specialization and flexibility. When deciding whether to setup or join decentral or hierarchical service organizations, stakeholders should first of all analyze the relevance of these benefits in their specific situations. Secondly, they will have to investigate whether the realized benefits of decentralization compensate for the costs incurring from decentralization (higher transaction costs due to more complex decision making, risk and quality management as well as the realization of economies of scale). In certain cases of service interconnections where EIIS are highly complex to describe (and thus complicate their transaction from service provider to consumer via markets) or require the setup of proprietary technology because none of the existing technical means is adequate (highly asset specific), a centralized form of EIIS organization is often preferable.

5 Conclusion

In this article, we elaborated on current and upcoming organizational forms of electronic information-intensive services which span across corporate boundaries. Stakeholders who try to organize and implement their business relationships electronically today often face enormous costs for the seamless communication and coordination of services, leading to centralized and inflexible service structures where few influential stakeholders prescribe standards and protocols which other stakeholders have to comply with. As service coordination and communication costs decline even more, the emergence of more decentralized "service markets" becomes possible. Benefits of increased decentralization of service organization such as more individual freedom (with respect to the design of services and protocols), improved flexibility and productivity are expected to drive this development. To thoroughly investigate major enablers and driving forces of this upcoming service decentralization, we draw an analogy to both the societal and the business context. The hypothesis underlying this article is that the organizational evolution of EIIS shares significant similarities with the development of societies and businesses which have evolved from unconnected groups of people over large hierarchies to more decentral forms. On the basis of case studies, we showed that this hypothesis holds. EIIS organization has evolved from local, unconnected services over large, hub-based hierarchies towards the currently emerging service marketplaces: In the course of the HERA project, for example, a novel, Swiss-wide infrastructure is being setup which allows stakeholders to seamlessly connect their EIIS in a highly decentralized and flexible manner. Information-intensive services may adhere to different architectural styles, be based on different message formats and are allowed to control message choreographies on their own, rather than exactly following comprehensive central standards regulations. Only minimal compliance with global and common standards is required to ensure seamless interoperability. Such upcoming service markets enable efficient and effective service interconnections while requiring comparably low transaction costs as technical complexity and uncertainty as well as asset specificity of involved technologies are heavily reduced.

References

1. Malone, T.: The Future of E-Business. Sloan Management Review 43(1), 104 (2001)
2. McAfee, A.: Will Web Services Really Transform Collaboration. Sloan Management Review 46(2), 64–78 (2005)
3. McAfee, A.: Enterprise 2.0: The Dawn of Emergent Collaboration. Sloan Management Review 47(3), 21–28 (2006)
4. Hill, T.P.: On goods and services. Review of Income and Wealth 23, 315–338 (1977)
5. Apte, U., Goh, C.: Applying Lean Manufacturing Principles to Information-Intensive Services. International Journal of Service Technology and Management 5(6), 488–506 (2004)
6. Fitzsimmons, J.A., Fitzsimmons, M.J.: Service management. McGraw-Hill, New York (2003)
7. Smith, A.: An Inquiry into the Nature and Causes of the Wealth of Nations, UK (1776)
8. Coase, R.H.: The Nature of the Firm. Economica 4(16), 386–405 (1937)
9. North, D.C. Institutions, Institutional change and economic performance, Cambridge University Press, Cambridge, UK (1990)
10. Malone, T.W.: The Future of Work: How the New Order of Business Will Shape Your Organization, Your Management Style, and Your Life. Harvard Business School Press, Boston (2004)
11. Schroth, C.: Web 2.0 and SOA: Converging Concepts Enabling Seamless Cross-Organizational Collaboration. In: Proceedings of the IEEE Joint Conference CEC 2007 and EEE 2007, Tokyo, Japan (2007)
12. Simon, H.A.: Models of Bounded Rationality. MIT Press, Cambridge (1997)
13. Malone, T.W., Yates, J., Banjamin, R.I.: Electronic Markets and Electronic Hierarchies. Communications of the ACM 30(6) (1987)
14. e-Business W@tch: The European e-Business Report 2006/07 edition – A portrait of e-Business in 10 sectors of the EU economy, `http://www.ebusiness-watch.org/resources/synthesis.htm`
15. Lheureux, B.J., Malinverno, P.: Magic Quadrant for Integration Service Providers, 1Q06. Gartner Research Paper, USA (2006)
16. Lheureux, B.J., Biscotti, F., Malinverno, P., White, A., Kenney, L.F.: Taxonomy and Definitons for the Multienterprise/B2B Infrastructure Market. Gartner Research Paper, USA (2007)
17. White, A., Wilson, D., Lheureux, B. J.: The Emergence of the Multienterprise Business Process Platform. USA: Gartner Research Paper (2007)
18. Schmid, B. F., Schroth, C.: Organizing as Programming: A Reference Model for Cross-Organizational Collaboration. In: Proceedings of the 9th IBIMA Conference on Information Management in Modern Organizations, Marrakech, Morocco (2008)
19. Schmid, B.F., Lechner, U., Klose, M., Schubert, P.: Ein Referenzmodell für Gemeinschaften und Medien. In: Englien, M., Homann, J. (Hrsg.) Virtuelle Organisation und neue Medien, pp. 125–150. Eul Verlag, Lohmar (1999)
20. HERA project, `http://www.hera-project.ch`

(Meta-)Models, Tools and Infrastructures for Business Application Integration

Ralf Kutsche and Nikola Milanovic

TU Berlin, Computergestützte Informationssysteme CIS
{rkutsche,nmilanov}@cs.tu-berlin.de

Abstract. In the context of the *Berlin Brandenburg Business Process Integration and Evolution framework BIZYCLE*, we are using modeling and metamodeling strategies in order to achieve a platform for the (semi) automatic integration of software components. Our general methodology is based on the principles of the MDA paradigm, distinguishing between platform specific (PSM), platform independent (PIM) and computation independent (CIM) models, and on the general philosophy of solving integration problems on higher levels of abstraction. The basic models and metamodels on CIM, PIM and PSM level, forming the pre-requisite for our further project work, are examined in this article.

Keywords: Modeling, metamodels, software and data integration.

1 Introduction

The Berlin-Brandenburg Regional Business Initiative BIZYCLE, was established in 2007 in order to examine the potential of model-based software and data integration methodologies, tool support and practical applicability in large-scale, conforming a consortium of six industrial partners (from sectors health, production and logistic, facility management and publishing) and academia, under guidance of the German Federal Minister for Education and Research.[1]

The basic idea of BIZYCLE is to create a model-based tool generation and interoperability operation platform, in order to allow for improved software and data integration processes in the area of software component integration. Classically, in daily business, software – and, as an integral part, data – integration tasks are performed by well-experienced software engineers and application experts, manually programming the *connections*. In general, this requires rather professional skills in the software systems/ components/ applications on one-hand side, and integration requirements' analysis skills on the other. This is a very expensive procedure, claming significant part of the IT budgets.

The BIZYCLE initiative follows the strategy of model- and metamodel- based abstractions for system integration. Our strategy focusses on (semi-)automated conflict analysis and connector generation as competitive advantage.

[1] This work is partially supported by the *Bundesministerium für Bildung und Forschung BMBF* under grant number 03WKBB1B.

2 General Background and Related Work: Continuous Software Engineering and Information Systems Evolution

Since the early 90s, integration issues have become a central challenge in many industry sectors: heterogeneous, distributed software solutions under autonomous responsibilities brought up the definite need for data integration in heterogeneous, distributed environment, based on the strongly increasing capabilities of (cheap!) small and medium computer systems. Federated Information Systems [12] and, their special case of read-only accessible Mediator-Based Information Systems ([11], [13], and our Reference Model Federated Information Systems [12]), are the examples of results achieved in this context. Finally, opening this area towards new paradigms like, e.g. Peer Data Management Systems have consolidated this very challenging and rapidly emerging research area under the name information integration [14]. Moreover, the Service Oriented Architecture [4] and Semantic Web efforts have added the very important aspects of communication and interoperability semantics to the integration business.

These activities perfectly meet with a tremendous pressure in business, where integration is a crucial aspect of the success (or only: survival) in a global marketplace. However, simply taking information integration as our challenge is, under a holistic view of 'I&C infrastructures', taken too short, because full interoperability, i.e. functional and behavioral integration to full extent in conjunction with structural integration, is needed in real business. This requirement additionally is complemented by an *evolution view* on software development, in our terminology expressed in the term Continuous Software Engineering [9] [3], meaning that a consistent view and continuous development and 'maintenance' of *all* artifacts in the complex circles of forward/reverse/re-engineering processes must be achieved. These two holistic approaches of software and systems engineering, namely integral view of integration issues across wide-scale I&C infrastructures, combining all informational aspects with full interoperability requirements, and integral view on all artifacts in a continuous software engineering process, yielding benefits wrt. consistency goals, constitute the essence of our BIZYCLE philosophy, methodology and concrete work. Basic paradigm of our approach is the use of models and metamodels as 'containers' of the 'engineering knowledge' on all levels of abstraction, to be complemented by special model extensions wrt. to semantic issues or metadata management.

3 BIZYCLE Methodology in Detail

In this section internals of the BIZYCLE Interoperability Platform will be investigated. We will describe BIZYCLE integration process, as well as the novel roles involved in the integration lifecycle. The abstraction levels already mentioned (CIM, PIM, PSM) will be examined in more detail, together with the process of model transformation, conflict analysis and connector generation.

3.1 BIZYCLE Integration Process and Roles

Four levels of abstractions are supported in the BIZYCLE integration process with appropriate models: computational independent level, platform independent level, platform specific level and deployment/code level (Figure 1).

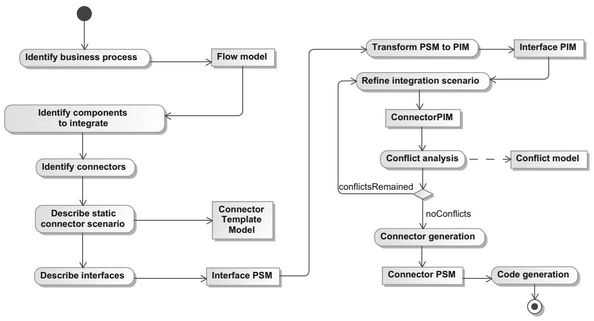


Fig. 1. BIZYCLE Integration Process

Computational Independent Model (CIM) captures essentials of the business process enacted by the integrated systems. This step consist of identifying business process, components and connectors, resulting in flow and connector template models. They abstract ICT related properties from the integration solution and describe only process-related issues. Flow model describes data and control flow. Abstract connectors are described in the connector template model. This model will be subsequently refined. The following step is identification and definition of interface PSMs, which are then abstracted and generalized to PIM level, for the purpose of conflict analysis, as is not possible to directly define and analyse integration conflicts between heterogeneous platforms. The process of conflict analysis generates a set of integration conflicts at the PIM level. It is performed iteratively until all conflicts are solved (or only non-solvable conflicts remain). Connector is then generated and deployed as a mediator component that matches requirements and capabilities of the observed interfaces.

Several roles emerge within BIZYCLE integration process. They enable modularization and separation of concerns in the integration process, but primarily aim to support automation and optimal (re)use of the domain-expert knowledge. Integration solution architect develops flow and connector template models, based on the requirement analysis. Platform specialists create and update metamodels for the supported platforms. Application specialists refine the work of solution architect, and create interface PSMs. System integrator generalizes PSMs to PIMs and guides conflict analysis process. It is his responsibility to manually intervene when the interoperability platform is unable to solve the conflicts automatically. Solution deployer oversees code generation and connector deployment and maintains the runtime environment.

3.2 Metamodels and Model Transformation

The BIZYCLE interoperability platform is based on the principles of model-driven engineering and Model Driven Architecture (MDA) [5]. Therefore, metamodels and model transformation form the core BIZYCLE elements. They are defined at the M2 level of the Meta Object Facility (MOF) hierarchy.

Computational Independent Metamodels. (CIMM) comprise two metamodels: Flow Metamodel (FMM) and Connector Template Metamodel (CTMM). FMM is based on the UML2 Activity Diagram and describes business data and control flow between the integrated systems. For that purpose a signaling mechanism is used: `SendSignalAction` represents a source system, while `AcceptEvent Action` represents target systems. Class `ConnectorNode` connects one sending action with one or more receiving actions. CTMM is used to describe potential conflicts and connector logic from the business-process perspective. The core element of CTMM is `Connector` class which connects two or more `IntegratableElement` classes. Connector template defines several `DataProcessor` classes. They capture integration semantics (e.g., aggregation, filtering). Technical aspects (e.g., tables to be exported or database drivers) are not addressed at this level. CTMM will be refined at the subsequent levels. The connection between FMM and CTMM is performed by `Connector` activating one or more `ConnectorNodes`. Activities between send and receive signal actions represent business connector template. The relation between `Connector` and `ConnectorNode` thus also has an activation semantics.

Platform Specific Metamodels. (PSMM) cover a class of metamodels describing technical aspects of systems that are to be integrated. Systems under study, for which we currently provide support, are relational databases (including extensions for MySQL, Microsoft SQL Server, Oracle, and Firebird), J2EE applications, .NET Framework applications (version 1.1 and 2.0), ERP applications (SAP R/3 BAPI/IDOC and AvERP), XML flat files and SOAP-based Web Services. In accordance with the view that all systems are treated as components, that is, as black boxes that expose their functionality via *interfaces*, PSMM offers a platform specific interface description. Aspects such as remote addresses, function and parameter signatures and structure, communication properties, diverse non-functional properties specific for each platform (e.g., security or transactions) are supported, as well as interface semantics. The goal of PSMM is to provide enough platform-specific information on interface properties, such that: 1) they can be abstracted to platform independent level and be used in the process of conflict analysis (design time requirement) and 2) an automatic connector can be generated, that invokes the given interface (runtime requirement).

All PSMM share the similar structure, where endpoint is described, parameters, communication properties and type system. Non-functional part of interface description has been left out of this example, as it is orthogonal to all PSMMs and will be commented on in the next section within PIMM context.

Platform Independent Metamodels. (PIMM) are used to abstract platform-specific properties from interface descriptions, which is precondition for the conflict analysis process. PIMM comprises the following packages: interface, structure, property, behavior, communication and semantic. Depending on the source PSM, interface can be functional, method or document. Each interface has a structure defined within platform independent type system. Platform specific types are then mapped into this type system in the process of model transformation. Object mapping is also supported through `Object` class, where object attributes are further PIMM-typed. Properties of platform specific communication channels are abstracted using `AccessPattern` and `MessageExchangePattern` classes, describing function- and document-based communication styles. Behavior is associated with each interface, describing pre-conditions, post-conditions and invariants. Properties define quality of service attributes, such as security, performance, reliability, etc. Using combination of behavior and properties, it is possible to express requirements and capabilities of each interface, e.g., login, encryption or transactional behavior. Interface can carry semantic description via association to `BusinessFunction`. Domain ontologies are used to describe business semantics, together with business objects that are mapped to physical interface parameters. Model transformation rules, that perform generalization from PSM to PIM level, are defined and implemented in ATL language [2].

Connector and Conflict Metamodels. represent the final step in the process of interface integration. Integration data and control flow are determined within a connector, which uses channels to transport messages between source and target interfaces. Message processors operate on the message content, by routing messages or transforming message content. A message-middleware independent transformation language is developed for expressing message content transformation rules. Business goals, as well as technical requirements are therefore matched within a connector. Connector also contains an application endpoint, which is an abstract placeholder for the interface client-side code. Our understanding of a connector is pro-active, where it describes not only behavior protocol between two components, but can activate that behavior in runtime. According to the principles of generative programming [6], BIZYCLE connectors do not require glue code, as they are generated up to the code/deployment level and can communicate autonomously with the target interfaces.

The BIZYCLE interoperability platform recognizes the following conflict types: structural, behavioral, communication and property. To represent each conflict type, an appropriate formal abstraction is used. Structural conflicts are type conflicts which are detected and solved using typing and subtyping rules. Behavior conflict are solved using either simulation and bisimulation [1], or transferring interface descriptions into abstract machine notation and performing invariant preservation proofs [10]. Process algebra [8] is used to perform communication and protocol compatibility check, while frame-logic [7] is suitable to perform semantical and property checks. The overall strength of this approach is the possibility to use model transformation (with PIM as a source), to choose adequate target formalisms and languages for verifying properties of interest.

4 Conclusion and Outlook

We have sketched the results of the first phase of our BIZYCLE initiative, by clarifying the general philosophy and strategy of our model- and metamodel-based approach towards (semi)automatization of software and data component integration. New process and roles which will be used in the future when the BIZYCLE platform has become operational have been defined, and, finally a sketch of our (meta)modeling activities which constitute the basis for the BIZYCLE framework's toolboxes and interoperability platform has been given.

At present, detailed specifications of concrete integration scenarios are being finalized by our industrial partners, which already (in the first round of our cyclic procedures) have been used for the platform specific models partially presented in this paper. From that we shall be able to revise and to finalize the PSMs and then integrate them into our tool generator.

References

1. Leicher, A.: Analysis of Compositional Conflicts in Component-based Systems. PhD Dissertation, TU Berlin (2005)
2. LINA & INRIA Atlas Group. ATL: Atlas Transformation Language User Manual (2006), `http://www.eclipse.org/m2m/atl/doc/ATL_User_Manual[v0.7].pdf`
3. Pons, C., Kutsche, R.D.: Model Evolution and System Evolution. In: Proceedings of the CACIC 99, Buenos Aires, Argentina (1999)
4. Krafzig, D., Banke, K., Slama, D.: Enteprise SOA. Prentice-Hall, Englewood Cliffs (2004)
5. Miller, J., Mukerji, J.: Model Driven Architecture (MDA) (2001), `http://www.omg.org/cgi-bin/apps/doc?ormsc/01-07-01.pdf`
6. Czarnecki, K., Eisenecker, U.: Generative Programming - Methods, Tools and Applications. Addison-Wesley, Reading (2000)
7. Kiefer, M., Lausen, G., Wu, J.: Logical Foundations of Object-Oriented and Frame-Based Languages. Journal of the Association for Computing Machinery (ACM) 42(4), 741–843 (1995)
8. Milner, R.: The Polyadic π-calculus: A tutorial. In: Logic and Algebra of Specification, Springer, Heidelberg (2003)
9. Mueller, H., Weber, H. (eds.): Continuous Engineering of Industrial Scale Software Systems. Dagstuhl Seminar 98092, Seminar Report No. 203, IBFI, Schloss Dagstuhl(1998)
10. Milanovic, N.: Contract-based Web Service Composition. PhD Dissertation, HU Berlin (2006)
11. Kutsche, R.-D., Conrad, S., Hasselbring, W. (eds.): Proceedings of the 4th Workshop Engineering Federated Information Systems (EFIS 2001). Akademische Verlagsgesellschaft Aka GmbH (2001)
12. Busse, S., Kutsche, R.-D., Leser, U., Weber, H.: Federated Information Systems - Concepts, Terminology and Architectures. Forschungsberichte des Fachbereichs Informatik Nr. 99-9, TU Berlin (1999)
13. Sigel, W., Busse, S., Kutsche, R.-D., Klettke, M.: Designing a Metadata-based Infrastructure for E-Learning. In: Proceedings 5th Int. Workshop EFIS 2003 - Engineering Federated Information Systems, pp. 89–99. Akademische Verlagsgesellschaft Aka GmbH, Berlin (2001)
14. Leser, U., Nauman, F.: Informationsintegration. Dpunkt Verlag (2007)

The Impact of Structuredness on Error Probability of Process Models

Ralf Laue[1] and Jan Mendling[2]

[1] Chair of Applied Telematics, University of Leipzig
Klostergasse 3, 04109 Leipzig, Germany
laue@ebus.informatik.uni-leipzig.de
[2] BPM Cluster, Faculty of IT, Queensland University of Technology
Level 5, 126 Margaret Street, Brisbane QLD 4000, Australia
j.mendling@qut.edu.au

Abstract. Recent research has shown that business process models from practice suffer from several quality problems. In particular, the correctness of control flow has been analyzed for industry-scale collections of process models revealing high error rates. In the past, structuredness has been discussed as a guideline to avoid errors, first in research on programming, and later also in business process modeling. In this paper we investigate the importance of structuredness for process model correctness from an empirical perspective. We introduce definitions of two metrics that capture the degree of (un)structuredness of a process model. Then, we use the Event-driven Process Chain models of the SAP Reference Model for validating the capability of these metrics to predict error probability. Our findings support the importance of structuredness as a design principle for achieving correctness in process models.

1 Introduction

The quality of business process models has been discussed for a while [1], but there are only few empirical contributions reported that partially help the modeler to come up with a good design. In particular, the works of [2,3,4] show that business process and workflow models from practice have considerable error ratios with 5% to 30% of the models having control flow problems such as deadlocks. [5,6] reveals that unstructuredness is an important property of a process model that is connected with a higher probability of errors.

Earlier research into software engineering, e.g. [7,8], emphasizes structuredness as a desirable property and proposes techniques for translating unstructured flowcharts into structured ones. However, these techniques are only partially applicable when concurrency is introduced. Therefore, the transformation of process models to executable BPEL (that impose certain structuredness constraints) becomes a sophisticated problem that cannot be automated in the general case [9,10]. While the negative impact of unstructuredness on quality of process models is supported by previous research, there is an ongoing debate on how to measure the degree of (un)structuredness. This paper contributes to this discussion by comparing the recently introduced Degree of Structuredness measure with a new metric called Unmatched Connector Count. Beyond the conceptual definition of the metrics, we utilize the SAP reference model as a sample to evaluate in how far the different metrics are associated with formal control flow errors. We use Event-driven Process Chains (EPCs) to illustrate our arguments, basically because EPCs cover the typical routing elements that are also used in other languages like YAWL and BPMN, and because we can use existing EPCs from the SAP reference model to test the suitability of our metrics. For details on EPCs refer e.g. to [11].

2 Structuredness of EPC Business Process Models

2.1 Informal Description of Structuredness

In a well-structured EPC model, splits and joins are properly nested such that each split has a corresponding join of the same type. Structured models can be built iteratively from the building blocks shown in Figure 1.

In the following sections we discuss how structuredness can be measured.

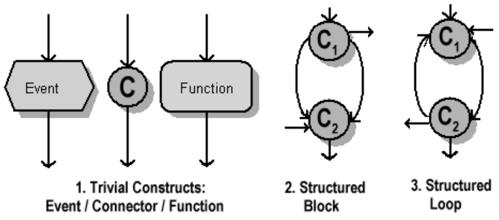


Fig. 1. Building Blocks of Structured EPCs [5]

2.2 Degree of Structuredness

The idea behind the *Degree of Structuredness* metric is to apply reduction rules and comparing the size of the reduced model to the original size. Due to space restrictions, we have to omit the formal definition of those reduction rules. It can be found in [5]. Informally spoken, reduction rules remove a structured modelling element from an EPC. These rules can be applied iteratively until the EPC cannot be reduced any further. We refer to this resulting reduced model as Λ'. If an EPC is constructed from well-structured elements only, a repeated use of reduction rules gives a Λ' which contains a single node. We define the Degree of Structuredness (DoS) as follows:

Definition 1 (Degree of Structuredness). *Let Λ be an EPC and Λ' the reduced model derived by iteratively applying the reduction rules described in [5]. We define the degree of structuredness (DoS) of the EPC Λ as $1 - \frac{|N_{\Lambda'}|}{|N_{\Lambda}|}$, where $|N_x|$, is the number of nodes in an EPC x.*

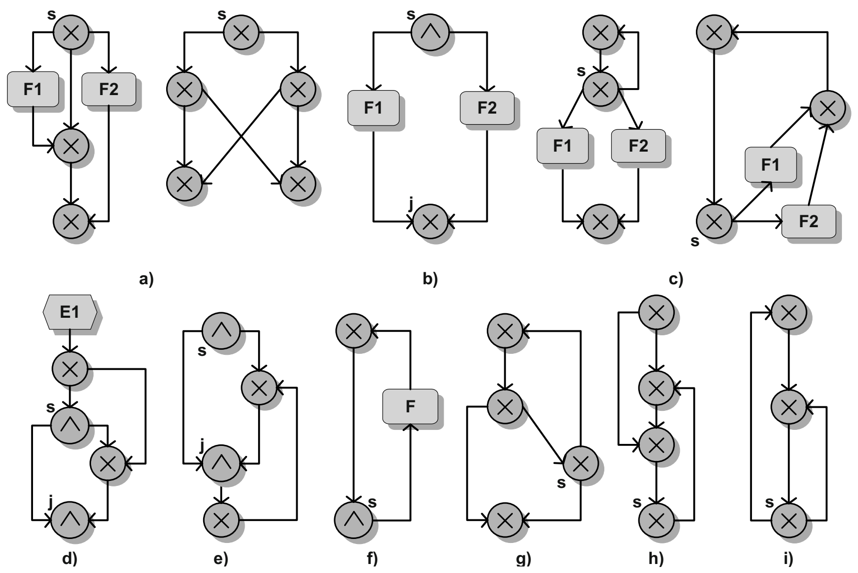


Fig. 2. Unstructuredness according to Definition 3

2.3 Unmatched Connector Count (UCC)

In this section we will define the concepts to measure how much an EPC deviates from structuredness. First, we give two auxiliary definitions:

Definition 2 (Cycle Entry and Cycle Exit). *Let Λ be an EPC and N its set of nodes. Then we define the following notations:*

- *The set of nodes on a directed, non-empty path starting and ending with the same node a (i.e. $a = n_1 \rightarrow \ldots \rightarrow n_k = a$) is called a cycle, symbol: $a \hookrightarrow a$. We refer to its elements as $N_{a \hookrightarrow a} = \{n_1, \ldots, n_k\}$.*
- *In a cycle $N_{j \hookrightarrow j}$, a node $n \in N$ is called*
  - *cycle entry, if there exist a directed path from a start event to j such that j is the only element of this path that is in the cycle.*
  - *cycle exit, if there exists a directed path from s to an end event such that s is the only element of this path that is in the cycle.*
- *The relation $match(s, j)$ (split s is matched by join j) holds iff there exist two directed paths from s to j whose only common elements are s and j.*

The Unmatched Connector Count metric counts symptoms for unstructuredness as shown in Fig. 2. The indices a) to i) in Def. 3 directly refer to Fig. 2.

Definition 3. *Let Λ be an EPC, $s \in \Lambda$ be a split node and $j \in \Lambda$ a join node such that $match(s, j)$ is true. We call s not properly matched, if at least one of the following properties holds:*

(a) There is more than one j for which $match(s, j)$ holds.
(b) For a j with $match(s, j)$, the types differ (for example, an AND-split is matched by a XOR-join).
(c) Beyond $match(s, j)$, s is also a cycle exit.
(d) For a j with $match(s, j)$, there is a path from a start node to j which does not pass s or a path from s to an end node which does not pass j.
(e) For a j with $match(s, j)$, there is a cycle $j \hookrightarrow j$ which does not pass s.
(f) The split s is a cycle exit and its type is not XOR (i.e. is AND or OR).
(g) The cycle $s \hookrightarrow s$ has more than one cycle exit.
(h) The cycle $s \hookrightarrow s$ has more than one cycle entry.
(i) There exist two cycles $N^1_{s\hookrightarrow s}$ and $N^2_{s\hookrightarrow s}$, and one cycle entry into $N^1_{s\hookrightarrow s}$ is not a cycle entry into $N^2_{s\hookrightarrow s}$.

Def. 3 considers only the splits for which no matching join can be found. In order to consider unmatched joins as well, we define for an EPC Λ its inverse Λ^{-1} such that Λ and Λ^{-1} have the same sets of nodes and (y, x) is an arc in Λ^{-1} iff (x, y) is an arc in Λ. It can be shown that the inverse of an EPC is an EPC as well. Using this definition, we formalize the Unmatched Connector Count as follows:

Definition 4. *Let Λ be an EPC, and Λ^{-1} its inverse. Then, the Unmatched Connector Count (UCC) is the sum of the number of not properly matched splits of Λ plus the number of not properly matched splits in the inverse Λ^{-1}.*

3 Validation

In this section we investigate in how far the two metrics defined in the previous section relate to errors (i.e. violations of the soundness property described in [11]) found in EPC process models. We assume that a deviation from structuredness is likely to result in errors. We use the EPCs of the *SAP Reference Model* for our study. It contains 604 non-trivial EPCs. From these models, we had to exclude some syntactically incorrect models (for example not strongly connected ones). Altogether, we considered 540 syntactically correct EPCs in the analysis.

As a first step, we use *correlation analysis*. We investigated in how the two metrics are capable to rank non-error and error models. This capability can be estimated using the Spearman rank correlation coefficient. It is +0.777 for *UCC* and -0.500 for *DoS*, i.e. for both metrics there is a strong and 99% significant correlation. This matches the expectation of the hypotheses that there is a relationship between *UCC* and *DoS* and the presence of control flow errors.

In a second step, we use *multivariate logistic regression* which estimates the coefficients of a linear combination of input parameters for predicting event versus non-event based on a logistic function. In our case, we predict error versus non-error for the EPC models based on the two metrics and a constant. The accuracy of the estimated model is assessed based on the significance level of the estimated coefficients, the percentage of cases that are classified correctly, and the share of the variation that is explained by the regression. This share is typically measured using the Nagelkerke R^2 ranging from 0 to 1. For technical

Table 1. Six Logistic Regression Models

	UCC	**C-UCC**	**DoS**	**C-DoS**	**UCC-DoS**	**C-UCC-DoS**
Constant		-3.673		6.534		-0.054
sign.		0.000		0.000		0.954
DoS			-1.858	-10.063	-4.368	-4.307
sign.			0.000	0.000	0.000	0.000
UCC	0.156	0.865			0.748	0.75
sign.	0.000	0.000			0.000	0.000
Classification	21.7	90.6	78.3	85.9	90.2	90.2
Nagelkerke R^2	0.097	0.707	0.482	0.416	0.835	0.732

details of logistic regression refer to [12]. For applications in predicting errors in process models see [2,5,6].

Table 1 summarizes the six logistic regression models that we estimated for all combinations of Unmatched Connector Count (UCC), Degree of Structuredness (DoS), and a constant (C). The following conclusions can be drawn:

1. Both metrics perform well in predicting error probability. The univariate regression models for both metrics (UCC, C-UCC, DoS, C-DoS) classify 85.9% and 90.6% of the EPCs correctly (using 0.5 as cut-value) with Nagelkerke R^2 ranging between 0.416 and 0.707 if a constant is included. Values greater than 0.4 already indicate an excellent explanatory power of the regression which are rare in real-world applications.
2. Both metrics complement each other. The multivariate regression models including both metrics (UCC-DoS, C-UCC-DoS) have a much better explanation (an excellent Nagelkerke R^2 of up to 0.835) than each univariate regression model alone. Since the constant is not significant in the multivariate model (significance of 0.954), it can be concluded that both metrics together leave only a little share of variation unexplained.

Apparently both metrics complement each other – they measure different aspects of structuredness. *DoS* quantifies the relative share of the EPC that is structured. Still, when the application of reduction rules stops there may be structured parts left in which the unstructured component is nested. In this regard, it captures a lower bound for relative structuredness. *UCC* provides absolute information of unstructuredness which might be difficult to compare across EPCs of different size. Both together perform better than the count metrics in [2] and almost as good as the regression model in [5,6] that requires seven metrics.

4 Conclusion and Future Work

In this paper we discussed the relationship between structuredness and correctness of process models. In particular, we defined two metrics that capture different aspects of (un)structuredness. Furthermore, we automatically calculated these metrics for the EPCs from the SAP reference model and used the results for estimating a logistic regression model. The multivariate regression based on both metrics performed almost as good as the regression model in [5,6] that requires seven metrics.

Our findings strongly support the importance of structuredness for the quality of process models. We agree with van der Aalst [13]: "If possible, non-well-structured constructs should be avoided. If such a construct is really necessary, then the correctness of the construct should be double-checked, because it is a potential source of errors." The metrics discussed in this paper support the modeler to check the degree of deviation from this ideal.

We are aware of the fact that in certain situations, unstructured models can be more expressive or easier to understand than structured ones. Examples for such cases can be found in [14]. To judge about problems that could arise from unstructured parts of a model, we aim to extend the research on such patterns.

References

1. Krogstie, J., Sindre, G., Jørgensen, H.: Process models representing knowledge for action: a revised quality framework. Europ. J. of Inf. Systems 15, 91–102 (2006)
2. Mendling, J., Verbeek, H., Dongen, B., van der Aalst, W., Neumann, G.: Detection and Prediction of Errors in EPCs of the SAP Reference Model. Data & Knowledge Engineering (accepted for publication, 2007)
3. Gruhn, V., Laue, R.: What business process modelers can learn from programmers. Science of Computer Programming 65, 4–13 (2007)
4. Vanhatalo, J., Völzer, H., Leymann, F.: Faster and more focused control-flow analysis for business process models through sese decomposition. In: Krämer, B.J., Lin, K.-J., Narasimhan, P. (eds.) ICSOC 2007. LNCS, vol. 4749, pp. 43–55. Springer, Heidelberg (2007)
5. Mendling, J.: Detection and Prediction of Errors in EPC Business Process Models. PhD thesis, Vienna University of Economics and Business Administration (2007)
6. Mendling, J., Neumann, G., van der Aalst, W.: Understanding the occurrence of errors in process models based on metrics. In: Meersman, R., Tari, Z. (eds.) OTM 2007, Part I. LNCS, vol. 4803, pp. 113–130. Springer, Heidelberg (2007)
7. Oulsnam, G.: Unravelling unstructured programs. Comp. J. 25, 379–387 (1982)
8. Ammarguellat, Z.: A control-flow normalization algorithm and its complexity. IEEE Trans. Software Eng. 18, 237–251 (1992)
9. Ouyang, C., Dumas, M., Breutel, S., ter Hofstede, A.: Translating standard process models to BPEL. In: Dubois, E., Pohl, K. (eds.) CAiSE 2006. LNCS, vol. 4001, pp. 417–432. Springer, Heidelberg (2006)
10. van der Aalst, W., Lassen, K.: Translating unstructured workflow processes to readable BPEL: Theory and implementation. Inf. & Softw. Techn. (In Press, 2006)
11. Mendling, J., van der Aalst, W.: Formalization and Verification of EPCs with OR-Joins Based on State and Context. In: Krogstie, J., Opdahl, A., Sindre, G. (eds.) CAiSE 2007 and WES 2007. LNCS, vol. 4495, pp. 439–453. Springer, Heidelberg (2007)
12. Hosmer, D., Lemeshow, S.: Applied Logistic Regression, 2nd edn. Wiley, Chichester (2000)
13. van der Aalst, W.M.: Formalization and verification of Event-Driven Process Chains. Information & Software Technology 41, 639–650 (1999)
14. Gruhn, V., Laue, R.: Good and bad excuses for unstructured business process models. In: EuroPLoP 2007, Proceedings (to appear)

Author Index

Abramowicz, Witold 14
Abusukhon, Ahmad 380
Acker, Hilmar 72
Al-Kamha, Reema 25
Anjomshoaa, Amin 37
Aoumeur, Nasreddine 393
Argiolas, Carlo 555
Avola, Danilo 405

Bassara, Andrzej 14
Best, Eike 45
Boronowsky, Michael 135
Böszörmenyi, Laszlo 52
Breu, Ruth 59
Brumen, Boštjan 365
Buddendick, Christian 445

Dadam, Peter 72
Desel, Jörg 84
Dessì, Nicoletta 555
Dzhumalieva, Stefka 417

Eder, Johann 96
Embley, David W. 25
Ermolayev, Vadim 108, 429
España, Sergio 258

Ferri, Fernando 405
Fill, Hans-Georg 175
Frank, Ulrich 111
Fugini, Mariagrazia 555

Gamzayev, Rustam 508
Ginige, Athula 123
Golob, Izidor 365
González, Arturo 258
Gonzalez-Perez, Cesar 1
Göser, Kevin 72
Grifoni, Patrizia 405
Grob, Heinz Lothar 445

Hafner, Michael 59
Helfert, Markus 417
Hellwagner, Hermann 213
Henderson-Sellers, Brian 1
Herzog, Otthein 135
Hesse, Wolfgang 136
Höfferer, Peter 175
Hölbl, Marko 365

Innerhofer-Oberperfler, Frank 59

Jarke, Matthias 148
Jurisch, Martin 72

Kabilan, Vandana 531
Kamišalić, Aida 365
Kappel, Gerti 158
Karagiannis, Dimitris 175
Kargl, Horst 158
Karle, Thomas 189
Keberle, Natalya 429
Kirchberg, Markus 457
Klar, Felix 201
Kreher, Ulrich 72
Kutsche, Ralf 579

Lachner, Janine 213
Laue, Ralf 585
Lauer, Markus 72
Lawo, Michael 135
Lehner, Franz 225
Liddle, Stephen W. 25
Lockemann, Peter C. 232
Lux, Mathias 496

Matzke, Wolf-Ekkehard 108, 429
Mendling, Jan 585
Menendez, Victor H. 469
Milanovic, Nikola 579
Mitlöhner, Johann 475
Mittermeir, Roland T. 246

Nemetz, Martin 175
Noack, René 484

Oakes, Michael P. 380
Oberweis, Andreas 189

Paolozzi, Stefano 405
Pastor, Óscar 258
Prieto, Manuel E. 469

Raibulet, Claudia 519
Rakow, Astrid 45
Reichert, Manfred 72
Reisig, Wolfgang 271
Reiss, Dirk 282
Reiter, Thomas 158
Retschitzegger, Werner 158
Reuter, Andreas 13
Riaz-ud-Din, Faizal 457
Rinderle, Stefanie 72
Rose, Sebastian 201
Rumpe, Bernhard 282
Russ, Christian 496

Saake, Gunter 393
Schewe, Klaus-Dieter 296, 318, 457
Schroth, Christoph 567
Schulze-Quester, Marvin 282
Schürr, Andy 201
Schwinger, Wieland 158
Sohnius, Richard 108, 429
Stănculescu, Raluca 544
Stănescu, Emil 544
Stănescu, Ileana 544
Stănescu, Laura 544
Stein, Mark 282
Strauch, Gereon 445
Strommer, Michael 158
Stucky, Wolffried 306

Tahamtan, Amirreza 96
Talib, Mohammad 380
Thalheim, Bernhard 296, 318, 484
Thoma, Helmut 332
Tjoa, A. Min 37
Tkachuk, Mykola 340, 508

Ubezio, Luigi 519

Valle, Enrico 519
van de Riet, Reind 352
Volčič, Zala 365

Weiß, Peter 306
Welzer, Tatjana 365
Wieringa, Roel 371
Wimmer, Manuel 158
Wisniewski, Marek 14
Witt, Hendrik 135
Wozak, Florian 59

Zanker, Markus 379
Zdravkovic, Jelena 531
Zebrowski, Pawel 14
Zemlyanoy, Andriy 508
Zota, Răzvan Daniel 544

Printing: Mercedes-Druck, Berlin
Binding: Stein+Lehmann, Berlin