Gesellschaft für Informatik (GI)

publishes this series in order to make available to a broad public recent findings in informatics (i.e. computer science and information systems), to document conferences that are organized in cooperation with GI and to publish the annual GI Award dissertation.

Broken down into the fields of
• Seminar
• Proceedings
• Dissertations
• Thematics
current topics are dealt with from the fields of research and development, teaching and further training in theory and practice. The Editorial Committee uses an intensive review process in order to ensure the high level of the contributions.

The volumes are published in German or English.

Information: http://www.gi-ev.de/service/publikationen/lni/

ISSN 1617-5468
ISBN 978-3-88579-241-3

This volume contains papers from the BPSC 2009 Conference on Business Process and Services Computing, the ISM 2009 International Workshop on Intelligent Service Management and YRW-MBP 2009 Young Researchers Workshop on Modeling and Management of Business Processes held in Leipzig Germany March 23-24, 2009. The papers discuss research and industry experiences with relation to process-centric service-oriented paradigm as it applies to the development and integration of enterprise and e-business information systems.

Business Process, Services Computing and Intelligent Service Management 2009

147

GI-Edition

Lecture Notes in Informatics

Witold Abramowicz, Leszek Maciaszek, Ryszard Kowalczyk, Andreas Speck (Eds.)

Business Process, Services Computing and Intelligent Service Management

BPSC 2009
ISM 2009
YRW-MBP 2009

March 23 – 25, 2009, Leipzig, Germany

Proceedings

Witold Abramowicz, Leszek Maciaszek
Ryszard Kowalczyk, Andreas Speck (Eds.)

Business Process, Services Computing and Intelligent Service Management

March 23 – 25, 2009,
Leipzig, Germany

Gesellschaft für Informatik e.V. (GI)

Lecture Notes in Informatics (LNI) - Proceedings
Series of the Gesellschaft für Informatik (GI)

Volume P-147

ISBN 978-3-88579-241-3
ISSN 1617-5468

Volume Editors
Witold Abramowicz
Poznan University of Economics, Department of Information Systems,
61-875 Poznan, Poland
Email: Witold@Abramowicz.pl
Leszek Maciaszek
Macquarie University, Department of Computing
Sydney, NSW 2109, Australia
Email: leszek@ics.mq.edu.au
Ryszard Kowalczyk
Swinburne University of Technology, Centre for Information Technology Research
Hawthorn, VIC 3122, Australia
Email: rkowalczyk@it.swin.edu.au
Andreas Speck
Christian-Albrechts-University Kiel, Department of Computer Science
24098 Kiel, Germany
Email: Andreas.Speck@email.uni-kiel.de

printed by Köllen Druck+Verlag GmbH, Bonn

2nd International Conference on Business Process and Services Computing BPSC 2009

Introduction

The papers published in this volume were presented at the 2nd International Conference on Business Process and Services Computing (BPSC 2009) held in Leipzig Germany on 23 – 25 March 2009. The book includes papers selected for presentation in the rigorous review process conducted by the BPSC Program Committee.

The paper selection process considered the mission of BPSC conferences to become a prime international forum to discuss and publish research findings and IT industry experiences with relation to process-centric service-oriented paradigm as it applies to the development and integration of enterprise and e-business information systems. By looking at the business process as a first-class citizen in the IT world and by using the potential of services computing for creation of adaptive process-centric business solutions, BPSC conferences identify most hopeful trends and propose new directions for consideration by researchers and practitioners involved in large-scale software development and integration.

The Business Process Management (BPM) is based on the premise that applications (of business processes) can be evolving independently from process management, very much like they have been evolving independently from data management. The technology of web services and Service Oriented Architecture (SOA) are at the forefront of enabling a desired degree of process independence. The related technology stack includes document management and workflow solutions as well as enterprise integration and e-business interoperability solutions. The related research trends include integration of SOA with Event-Driven Architecture (EDA) and AI-inspired ideas of autonomic computing, multi-agent systems or SWS (Semantic Web Services).

Services within SOA are units of processing logic that collaborate to deliver enterprise logic as a combined effect of business process logic and application control logic. In other words, services apply to both kinds of logic and create a connectivity layer that enables independence of processes and applications. Services can ensure that processes and applications evolve gracefully together (very much like the aspect code and the base code in aspect-oriented programming) and the “crosscutting concerns” are well-documented and tractable. The SOA paradigm redefines the concept of an application as a distributed set of implementation-independent services executing as an orchestrated sequence of messaging and event processing. The confluence of SOA and BPM is resulting in a new process-centric paradigm that holds great promise for enterprise and B2B computing.

Moreover, one of the major BPM tasks is the constant need to adapt implemented processes to the changing business needs. As the degree of automation in BPM is currently rather limited, a great potential lies in the attempts to automate BPM a little further by the use of Semantic Web services and technologies. Founded on ontologies, Semantic Web provides methods and tools for the machine-understandable representation of collective knowledge and business processes in which such knowledge resides. Semantic Web Services (SWS) make use of Semantic Web technology to support the automated discovery, substitution, composition, and execution of SOA-based applications. Semantic Web goes as far as expecting that intelligent software agents can use semantic descriptions of Web services and resources to automate their use to accomplish user goals. Current research shows that combining the worlds of BPM and SWS may be very fruitful.

Acknowledgements

Our gratitude goes first of all to the Organizing Committee of BPSC 2009 – Karol Wieloch of Poznan University of Economics, Poland and Martin Matzner of University of Leipzig, Germany. Between two of them, they have taken care of coordinating the work of PC members, communicating with the authors, enabling electronic paper submissions and consequent reviewing tasks, registering of participants, as well as formatting this volume for publication.

The most responsible work was of course placed on the PC members who had to decide which papers were worthy of presentation at the conferences. We would like to thank them for marvelous work done.

Leszek Maciaszek
BPSC 2009 General Chair

Witold Abramowicz
BPSC 2009 Program Chair

Program Committee

The reviewing process was carried out by the BPSC 2009 Program Committee members. The final decision of acceptance/rejection was strictly based on the reviews. The PC members who contributed reviews were:

Rainer Alt	University of Leipzig, Germany
Giuseppe Berio	Université de Bretagne Sud, France
Miriam Capretz	University of Western Ontario, Canada
Schahram Dustdar	Vienna University of Technology, Austria
Bogdan Franczyk	University of Leipzig, Germany
Cesar Gonzalez-Perez	IEGPS, Spanish National Research Council, Spain
Paweł Kalczyński	J. California State University, Fullerton, USA
Marek Kowalkiewicz	SAP Australia Pty Ltd, Australia
Lech Madeyski	Wroclaw University of Technology, Poland
Mariusz Momotko	General Electric Money Bank, Poland
Mitsunori Ogihara	University of Rochester, USA
Eric Paquet	National Research Council, Canada
Václav Repa	Prague University of Economics, Czech Republic
Jürgen Sauer	University of Oldenburg, Germany
Janice C. Sipior	Villanova University, USA
Yoichi Takayama	Macquarie University, Australia
Ioan Toma	University of Insbruck, Austria
Krzysztof Wecel	Poznań University of Economics, Poland
Mathias Weske	University of Potsdam, Germany

Organizing Committee

Martin Matzner	University of Leipzig, Germany
Karol Wieloch	Poznań University of Economics, Poland

Organizers

- University of Leipzig, Germany
- Macquarie University, Australia
- Poznan University of Economics, Poland

UNIVERSITAT LEIPZIG

Supporters

- Service Web 3.0
- SUPER Project

The International Workshop on Intelligent Service Management ISM 2009

Introduction

The papers published in this volume were presented at the International Workshop on Intelligent Service Management 2009 (ISM 2009) held in Leipzig, Germany on 24 March 2009. The book includes papers selected for presentation in the rigorous review process conducted by the ISM Program Committee.

Service-oriented computing has emerged as the most promising design paradigm for distributed information systems. The vision of service-oriented computing is to capture business relevant functionalities of existing software systems as services and use service composition to form composite applications. While this vision has yet to be achieved in practise, in particular the application of intelligent systems and techniques promises significant advancements for an adaptive and reliable construction and management of service-oriented applications and systems.

The ISM 2009 workshop provided an international forum for presenting and discussing recent significant developments and practical results at the intersection of service-oriented computing and intelligent systems and technologies and promoted cross-fertilization of ideas and techniques between these fields. The workshop encouraged a multidisciplinary perspective and aimed at bringing together researchers from diverse fields and interests, including multi-agent systems and artificial intelligence, automated construction and management of service-oriented applications/composite services, intelligent management of service quality concerns, and adaptive and reliable evolution and optimization of services.

The papers published in this book were accepted for publication at the ISM 2009 workshop as a result of a thorough peer review process. All submitted papers were reviewed by at least three members of the international ISM Program Committee and assessed by the conference chairs. The final decision of acceptance/rejection was strictly based on the reviews of the Program Committee members.

We would like to thank all members of the international Program Committee for their excellent work, effort, and support in ensuring the high-quality program and successful outcomes of the ISM 2009 workshop. Our thanks go also to the German Computer Society (Gesellschaft für Informatik) for their cooperation and help in putting this volume together.

Ryszard Kowalczyk and André Ludwig
Conference Chairs

Conference Chairs and Organizing Committee

Ryszard Kowalczyk	Swinburne University of Technology, Australia
André Ludwig	University of Leipzig, Germany
Rainer Unland	University of Duisburg/Essen, Germany
Dominik Zyskowski	Poznan University of Economics, Poland

International Program Committee

Stanislaw Ambroszkiewicz	Polish Academy of Sciences, Poland
Youcef Baghdadi	Sultan Qaboos University, Oman
Jamal Bentahar	Concordia University, Canada
M. Brian Blake	Georgetown University, USA
Peter Braun	The Agent Factory GmbH, Germany
Paul Buhler	College of Charleston, USA
Jiangbo Dang	Siemens Corporate Research, USA
Ian Dickinson	HP Laboratories, UK
Agata Filipowska	Poznan University of Economics, Poland
Mauro Gaspari	University of Bologna, Italy
Michael Gerndt	Technische Universität München, Germany
Dominic Greenwood	Whitestein Technologies, Switzerland
Thomas Hering	Universität Leipzig, Germany
Jingshan Huang	University of South Carolina, USA
Monika Kaczmarek	Poznan University of Economics, Poland
Sebastian Kiebusch	Weberbank Actiengesellschaft, Germany
Marek Kowalkiewicz	SAP Research, Australia
Joerg Leukel	University of Hohenheim, Germany
Margaret Lyell	Intelligent Automation Inc., USA
Zakaria Maamar	Zayed University, United Arab Emirates
Mercedes G. Merayo	Universidad Complutense de Madrid, Spain
Harald Meyer	University of Potsdam, Germany
Manuel Núñez-García	Universidad Complutense de Madrid, Spain
Giovanna Petrone	University of Torino, Italy
Ajith Ranabahu	Wright State University, USA
Marwan Sabbouh	The MITRE Corporation, USA
Francisco Garcia Sánchez	University of Murcia, Spain
Michael Sheng	University of Adelaide, Australia
Mohammed Sellami	INT Telecom, France
Andreas Speck	Christian-Albrechts-Universität zu Kiel, Germany
Xuan Thang-Nguyen	Swinburne University of Technology, Australia
Jun Yan	Wollongong University, Australia
Xiaohui Zhao	Swinburne University of Technology, Australia
Wolfgang Ziegler	Fraunhofer-Institute for Algorithms and Scientific Computing, Germany

Organizers

- University of Leipzig, Germany
- Swinburne University of Technology, Australia
- Poznan University of Economics, Poland

Supporters

- Logistik Service Bus Project
- SUPER Project

The Young Researchers Workshop on Modeling and Management of Business Processes YRW-MBP 2009

Management and Modeling of Business Processes is among the most important, but also most complex problems of modern computer science. The rise in electronic business also leads to a major, industry-driven request for appropriate support in defining and maintaining business process execution engines. Additionally, the new paradigms of service-oriented architectures (SOA) and extensible service definitions (e.g. Web Services) require a thorough re-engineering on the existing approaches in Business Process Modeling.

The Young Researchers Workshop Series on Modeling and Management of Business Processes aims at providing a platform for young researchers to present their work in this particular area. The workshop is intended to serve as a forum for the participants to get in contact with other researchers in the field and to become familiar with other approaches and future research topics.

We would like to express our thanks to all authors who submitted their papers and provided a presentation at the workshop. Further, we want to thank our reviewers and on-site moderators for their outstanding efforts and support. Though two pages per paper are not supposed to completely cover the author's full contributions, we think the high quality of the resulting proceedings illustrates the importance and necessity of discussion on this hot research topic.

March 2009,

Andreas Speck
Sven Feja
Meiko Jensen

Panelists and Reviewers

Leszek Maciaszek	Macquarie University, Australia
Krzysztof Czarnecki	University of Waterloo, Canada
Jens Weiland	Reutlingen University, Germany

Moderator and Reviewer

Andreas Speck	Christian-Albrechts-University Kiel, Germany

Organizing Committee

Sven Feja	Christian-Albrechts-University Kiel, Germany
Meiko Jensen	Ruhr-University Bochum, Germany

Organizers

Christian-Albrechts-University Kiel, Germany
Ruhr-University Bochum, Germany

Table of Contents

BUSINESS PROCESS AND SERVICES COMPUTING

INTELLIGENT SERVICE MANAGEMENT

MODELING AND MANAGEMENT OF BUSINESS PROCESSES

Business Process and Services Computing
BPSC 2009

Towards a Service-Oriented and Model-Driven framework with business processes as first-class citizens

Andrea Delgado[1], Francisco Ruiz[2], Ignacio García-Rodríguez de Guzmán[2]
Mario Piattini[2]

[1]Computer Science Institute, Engineering Faculty, University of the Republica
Julio Herrera y Reissig 565, 5to. Piso
CP 11300, Montevideo,Uruguay
[2]Alarcos Group, Technology and IS Department, University of Castilla-La Mancha
Paseo de la Universidad No.4
CP 13071, Ciudad Real, España
adelgado@fing.edu.uy
{francisco.ruizg, ignacio.grodriguez, mario.piattini}@uclm.es

Abstract: One challenge that organizations face nowadays is to agilely react to changes in their business, adapting their business processes and technologies to new possibilities. To do so, organizations must be capable of separating the definition of their business processes from their technical implementation, which most are currently streaky. Applying the Service Oriented Computing (SOC) and Business Process Management (BPM) paradigms in conjunction, is an important but not trivial, step to take, involving different visions of business and technological challenges. The Model Driven Development (MDD) paradigm is also applied to serve as a bridge between business process models and technical models of the software to implement them. In this paper, the further work done on a service oriented methodology defined years ago is presented, considering business processes as the centre of software development. From business process models, software services are derived in a straightforward way, which will be automated by model transformations using the OMG service profile.

1 Introduction

Service Oriented Computing (SOC) involves the integration of technologies and concepts from various disciplines of the area of computation [PTDL07], and the Service Oriented Architecture (SOA) is a key element of its realization. In [KBS05] SOA is an architecture style of reusable software-based services, with well-defined public interfaces, where suppliers and consumers of services interact in a decoupled way for conducting business processes. A service provides business logic and data, a service contract, restrictions for the consumer, an interfaces that exposes functionality. A repository for storing service contracts and a service bus for connecting those involved are also defined. This vision is related to Business Process Management (BPM) which deals with efforts to optimize or adapt the organizational needs of business processes [BPMI] and BPM Systems (BPMS) as support tools that allow, among others, modeling and implementation of these processes in sequences of invocations to services (orchestration, choreography) [KBS05] [SF03].

The joint application of the SOC and BPM paradigms is the ideal way to model business processes implementing them independently of technology. An important feature is that it allows the logic of business processes to be clearly separated from the core business logic located in lower-services of finer granularity. Using a BPMS for the definition and control of these processes avoids codifying information and business rules directly in the software, thus facilitating the modification, re-configuration and optimization of the processes through graphical tools to define process flows, i.e., using the Business Process Modeling Notation (BPMN) [BPMN]. Processes could be added or modified with few adjustments, the business logic will be implemented only once in a service, increasing the reuse of knowledge and reducing inconsistencies and redundancies.

While progress has been made in the conceptual and technological aspects of SOC and SOA, BPM and business processes, there is still a lack of methodologies and guides for their joint application. The methodology presented here was initially proposed in 2005 [DGP06][De06][De07] in 2005, as an extension to a base process [BP00][BP06] which is an adaptation of RUP[RUP]. In this paper a modified methodology is presented for its application along with any software development process, with special focus on the business process models to derive software services. At this stage, this derivation is only methodological and conceptual, but it would be automated applying the Model Driven Development (MDD) paradigm, by defining models, metamodels and transformations between them, in a framework which integrates the three paradigms.

The rest of the paper is structured as follows: section 2 will discuss the related work, section 3 will present the business process-driven methodology, including the first proposal and the further work done, and section 4 will illustrate the use of the methodology through a case study based on the generic "Grant Loan" business process of a bank. Finally, in section 5 some conclusions and future work will be presented.

2 Related work

There are many guidelines, techniques and recommendations, and some methodologies that prescribe key aspects of the development process for service oriented software or business process implementation, but very few combine them. [EA04][Er05][KBS05] provides insight of the main aspects for SOA development, but in a general way, enumerating the features and elements of service orientation, and discussing specific technologies to implement it, i.e., Web Services. The SOMA plug-in [SOMA] of RUP in Rational Method Composer (RMC) [RMC], is contemporaneous with the first proposal of the methodology presented here, and similarly, it is focused on disciplines of Business Modeling, Design and Implementation. It defines more elements (activities, products, roles) making it more complex. A comparison is presented in [DGRP08].

Some of the most relevant works in the area include the following: [PJ06] defines phases, activities and artifacts for the development of services associated with business processes. It differs from ours in that although it defines guides for a service oriented development, it is also focused on the implementation of services as Web Services, defining technical aspects that cannot be applied with other technologies. Other works also include the MDD paradigm and its realization the Model Driven Architecture (MDA) [MDA03], as in [DML06] in which the main focus is the development of service oriented web systems defining models, metamodels and transformations between them to obtain a service composition model which expresses the interaction of services to perform business processes. Although they plan to, the method does not start with the modeling of business processes, as the methodology presented here does.

In [ZHD07] patterns are defined to guide the definition, transformation and implementation of technical processes using software services from business processes in which they call process-driven service oriented architecture. The macroflow pattern represents long-running business processes and the microflow pattern short-running technical processes, starting top-down from business process and bottom-up from software systems, joining them in the middle. Differently, our proposal does not classify types of processes, and does not use patterns to link models, as they are successively refined starting from business process models. In [RBM06] models and metamodels for services are defined to relate them to business processes and the underlying architecture, focusing the derivation of services to three architectures: brokerless, centralized and decentralized broker, providing a technical focus, which is not treated in our proposal.

Other approaches aims to relate business processes and software services as in [QDV05] which uses models expressed in ISDL to relate conceptual models and to asses conformance between them, which in our work is not made formally. In [LKT04] UML [UML] software artifacts like use cases, activity and collaboration diagrams are obtained automatically from business processes expressed in BPMN [BPMN], which is complementary to ours since we derive from BPMN business processes the needed services. In [HZ05] business processes are related to technical processes which use existing services, defining types of realization to identify the quality of the transformation between them. Although our proposal also takes into account existing services, it doesn't relate or constraint the business process because of services.

3 Business process-driven framework

Most of the current software development processes used are based on the philosophy of interaction and change, given the fact that the requirements of systems are usually unstable and it should be possible to incorporate them as they arise. These models classified as "heavy" or "agile", typically indicate that frequent releases are used to obtain feedback from users, and the construction of the system in iterative way is based on these incremental releases. A methodology for development-oriented services does not need to be a completely new methodology, but it could be built over the process or approach used in the organization, adding specific activities and artifacts for the development of services, and therefore any process could serve as a base.

The first proposal of the methodology [DGP06][De06][De07] made in 2005, was defined over a base process [BP00][BP06] adapted from RUP [RUP], and later RUP aspects like use cases or models were generalized to suit the process model of COMPETISOFT [CO06]. The core set defined emphasizes the Disciplines of Business Modeling, Design and Implementation to identify and model business processes, identify and derive the services making up these processes, and to design and build them.[EA04][Er05][KBS05] were references for general aspects and the Business Modeling Discipline was based on [RUP]. The core methodology was validated with cases studies in an academic context [DGP06][De07] and improved adding activities and deliverables in other Disciplines -which are not presented here- such as: testing, quality assurance, configuration management, deployment and management of services.

The core Disciplines, Activities, Deliverables and Roles presented here differ from the first proposal in many ways. First of all, they are independent of the RUP and its elements, and secondly, they are focused on modeling business processes and sub-processes, indicating how to derive the services from them. In our current work, the transformations between models are being defined, including the use of metamodels to make this derivation as automatic as possible.

3.1 Elements of the methodology

As we see in the service orientation paradigm, a key aspect is focused on business process modeling, taking them as first-class citizens from which to derive the required software services. The methodology proposed here defines two main activities in the Business Modeling Discipline to identify and model business processes. Besides, it defines five activities in the Design Discipline, which are key factors to identify, categorize, reuse, specify, and define the needed services, and their orchestration or choreography. The Implementation Discipline indicates the development of services as designed. Each activity, its input and output deliverables, and associated roles are clearly defined, as well as associated responsibilities. The roles defined are Software Architect, Analyst of business and requirements, and Developer specialized in technology.

Business Modeling Discipline

The purpose of the Business Modeling Discipline is to ensure that developers and others stakeholders have a common understanding of the Organization and derive the requirements for the software system, linking them to the identified business processes. The goal is to obtain a map of the organization and its processes to gain a better understanding of the business and requirements for the software system. The roles involved are Analyst and Architect, who have meetings with the client to identify them. The main deliverables are the Assessment of the target Organization with the identified key aspects and the Business Processes document with the specified business processes.

Assess the target Organization (MN1)

This activity aims to involve the project team with the organization for which the development is being carried out, in issues such as: the area of business, operation, employees, etc. of the organization, its current business processes, tools, skills of

people, customers, competition, technological challenges, problems and areas of improvement, clearly identifying those stakeholders involved in the business modeling effort. The role responsible is Analyst but Architect also participates, as both roles are involved in the business process model effort from which to derive the requirements for the application. The input of this activity is information about the business obtained from the stakeholders while the output is the assessment of the target organization which details its key aspects.

Identify Business Processes (MN2)

This activity has the objective of understanding as well as describing the business processes in the organization, mainly those related to the application being developed. Business processes are modeled with the selected notation, i.e. BPMN [BPMN], or Activity Diagrams of UML [UML], among others. The description must include: actors involved, control flow including sequence of activities, flow decisions and business rules. It is recommended to use process patterns (workflow patterns [VTKB03]) to help in various modeling aspects of business processes. The boundaries of business are clearly stated, indicating who and what interacts with the organization, specifying processes both in natural language and graphically. The role responsible is Analyst although Architect takes part too. As input, this activity has the Assessment of the Target Organization document, and minutes of meetings with customers. The output will be the document of the specified Business Processes.

Design Discipline

The purpose of the Design Discipline is that of identifying and cataloging services to perform the defined business processes , specify their interfaces and define their operations, the components that will implement them, reusing existing services in the organization. In addition, it has the target of defining the interaction of services in orchestrations or choreographies, to carry out the modeled business processes, including a BPMS when possible, for modeling, implementation, and monitoring of processes, among others. The roles involved are Architect, Analyst and Developer, being the Architect responsible for the whole discipline. The deliverables are the Services document, with specified services, and the Services Catalogue, to include the new services and search for existing ones.

Identify and categorize services (D1)

This activity is aimed at identifying the services needed to perform the modeled business processes, classifying them by type of service. Services represent features related to business concepts, and their categorization into a conceptual hierarchy helps in guiding the development and avoiding their proliferation in an indiscriminate manner, which is known as "services syndrome". Services are derived from business process models and their sub-processes and required functionalities, which have to be provided by services mapped to software subsystems. The role responsible is the Architect while Analyst and Developer participates as well. This activity has as input the document of Software Architecture, the Business Processes document, Requirements document and Services document. As output, it has the Services document including the identification and categorization of the informed services.

Specify services (D2)

This activity aims to specify the services, defining for each one its contract with the interfaces it provides and its operations and parameters. Each operation in the interface must specify: a) name of the method, b) required parameters and for each one, its name, type and description, c) return value, indicating name, type and description d) list of exceptions, e) brief description of the provided functionality f) pre-conditions for the successful execution, g) post-conditions that have to be valid after execution. The role responsible is Architect and Analyst is also involved, due to the fact that the specification can be carried out by both roles. It has as input and output the Services document, as input the identification and categorization of services, and as output the specification of services.

Investigate existing services (D3)

The purpose of this activity is to find services that are already implemented in the organization and can be reused in the application under development. It could be necessary to implement intermediary services for reusing other services that are not completely suited to the application, which is preferable than implement them again from scratch. To keep track of the services, their contracts and associated software systems, the Catalogue of services for its registration is defined. The role responsible is Architect and Analyst is also involved, as it is related to business modeling. The inputs of this activity are the Catalogue of services where existing services are listed and the Services document with the defined services, while the output is the Services document with existing services to reuse and the updated Catalogue of services.

Assign services to components (D4)

This activity aims to define the components to be implemented for providing the specified services. To assign services to components the question of whom (which component) will be provided with the services defined by the specified interfaces must be answered. The existing services also count if there are components providing the required functionality, or if an intermediary service could be developed to provide it using existing services. The role responsible is Architect and Developer also participates, as it involves mapping from design to implementation. The inputs are the Design and the Services documents, with information of software and services. As output it has the Development and Services documents, with the components defined for the services.

Define services interaction (D5)

The target of this activity is to define the sequence of interaction between services needed to perform the identified business processes. It will be an orchestration of services if it is internal to the organization or choreography if it is a collaborative process with communication with other organizations or entities. The invocation sequence of services is shown for each process or sub-process in a sequence diagram describing the interaction between the services involved. In addition, it is recommended to use a BPMS to define, implement, enact and manage the invocation sequences in languages such as BPEL[WSBPEL] or XPDL[XPDL]. The role responsible is the Architect but Analyst and Developer also participate because interaction of services needs business and technical knowledge. The inputs are Business Process, Requirements and Services documents, with information of needed services interaction to perform business

processes. As output, it has the Services document with the defined interaction and the Implementation document with the associated details.

Implementation Discipline

The Implementation Discipline aims to develop components providers of services, according to the assignment of services to components previously performed, the binding to them for invocations at the application level or between services, using the defined strategy.

Implement services (I1)

This activity has the objective of implementing the described services , taking into account the type of service, the designed interfaces, the interaction with other services (with or without a repository of services, binding in development or execution time). The only participating role is Developer who is in charge of the implementation of services. The inputs are the Services document, with the correspondence between design and implementation, Design and Implementation documents with associated details. Its output is the service implemented in the specified component.

4 Case study – the "Grant Loan" business process

The "Grant Loan" business process of a bank specifies the procedure defined to carry out the steps for the granting of loans to its customers. The bank is the target Organization in which the software project is taking part. The business process starts when a customer goes to the bank to request a loan, filling out the documentation. The request is sent to the Loan Authorization section, where the personnel study the documentation assessing the client's loan history, requested amount , and the client's credit information provided by the Credit Information Centre. The loan is approved or rejected and the resolution is sent back to the Customer Service area, which informs the client. If approved the client has to sign a loan contract and withdraw the money. In either case, the resolution is registered in the client's loan history. There are three actors involved: Client, Bank and Credit Information Centre, and inside the Bank there are also two different sections. It is worth mentioning that although the Bank owns the "Grant Loan" business process, it is a collaborative process since it has communications with other organizations.

4.1 Using the methodology

The first activities to be performed are the two defined in the Business Modeling Discipline, so the Analysts and the Architect have meetings with stakeholders and identify, among other things, the main business processes to develop. In this case study, it is the "Grant Loan" business process, which is shown in figure 1 using BPMN.

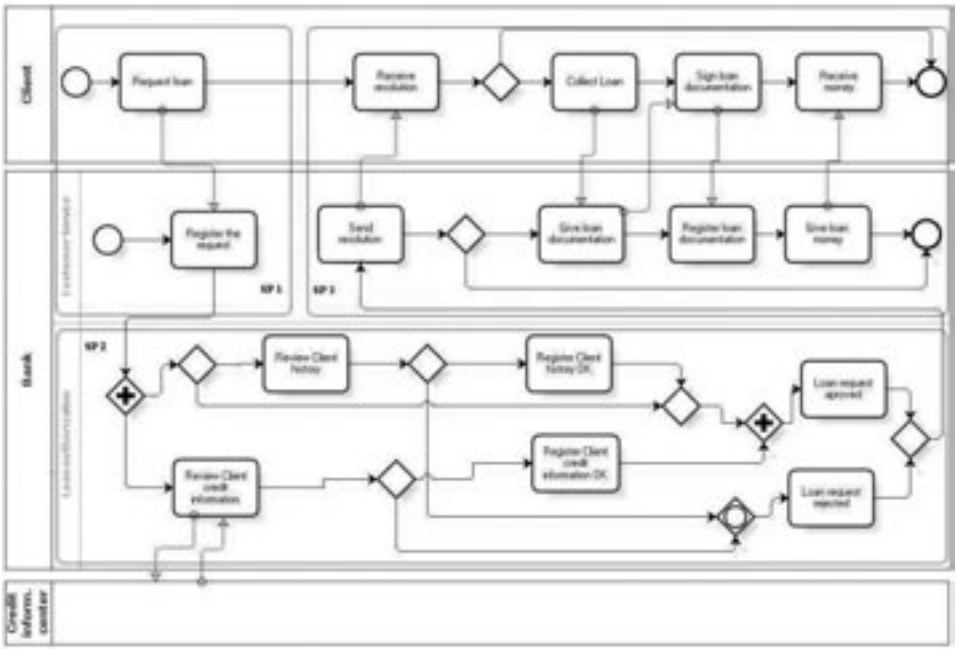


Figure 1: business process model of "Grant Loan" in BPMN

The notation and tool to be used in modeling the business processes have to be selected accordingly with the Organization's capabilities, including the economic ones; the important thing is to express the business processes graphically to help the understanding and further analysis for their implementation. Other important thing to keep in mind while carrying this selection out is the scope that the business process modeling effort would have. If it is mainly for better communication of ideas between the stakeholders, including business persons, it would be preferable to use a simple and business oriented notation like BPMN. Moreover, this notation could be transformed both into BPEL and XPDL, so business processes could also be deployed into a compatible business process engine, which supports the enactment of processes.

In the model presented in figure 1, it can be seen that there are three red rectangles which contain subparts of the process. These rectangles correspond to the following sub-processes: SP1-"Loan Request", SP2-"Loan Authorization" and SP3-"Loan Delivery", identified in the business process, which make the general process more manageable and simpler, facilitating the derivation of services from it. If uses cases are used, the identified sub-processes correspond to the system use cases performing the business process. To identify the needed services as defined in the Identify and categorize services (D1) activity, the functional requirements of each sub-process have to be identified and specified, and further refined, accordingly to the design activities. In the case study, the high level services identified from the "Grant Loan" business process are shown in table 1, together with the sub-process and a description.

Table 1: Definition of services from requirements of sub-process

Service	Sub-process	Description
RegisterLoanRequest	SP1–"Loan Request"	It models the interaction between client and Customer Service when the loan is requested.
ReviewClientHistory (defined for reuse)	SP2–"Loan Authorization"	It abstracts the management of the client's history.
ReviewClientCreditInfo (defined for reuse)	SP2–"Loan Authorization"	It abstracts the interaction with the Credit Information Center.
RegisterClientInfo (defined for reuse)	SP2–"Loan Authorization"	It registers Client's information.
RegisterLoanGranted	SP3-"Loan Delivery"	It models the interaction between client and Customer Service when the loan is approved.

As we can see in table 1, for the SP2-"Loan Authorization" sub-process there are three services identified and defined for reuse: ReviewClientHistory, ReviewClientCreditInfo and RegisterClientInfo. This is carried out since the probabilities that other applications need to obtain the client's history of loans in the bank as well as the client's credit information in the Credit Information Centre, or to register client's information, are high. Also, it is advisable to design services for reuse in the organization, since the Catalogue of services is searched every time to find services to be reused in new applications. Once a service is identified and categorized, its functional contract is specified, as indicated in the Specify services (D2) activity. The services are iterative and incrementally identified, defined, categorized and specified, performing the two first activities once and again as the projects evolve and the team gains knowledge of the organization and its business processes. Once the main business processes have been identified and specified, the Investigate existing services (D3) activity is performed, searching in the Catalogue of services for services or system functionalities that bring the desired behavior, or for a composition that could bring it. After that, the components to provide the functionalities of the services are designed, following the guides included in the Assign services to components (D4) activity.

Finally, the interaction of services into orchestrations or choreographies is defined, as the Define services interaction (D5) activity indicates, that is, taking a business process and its identified sub-process, the flow of interaction of services has to be defined, following the business process model flow. This interaction could be presented by sub-processes to make the whole process more understandable, joining them to make up the general process. This interaction between defined services is shown in a UML sequence diagram, as it clearly describes the interchange of messages between the participants. In figure 2 the sequence diagram in UML for the SP2-"Loan Authorization" sub-process is shown, including the interaction with internal defined services and external services from other organizations.

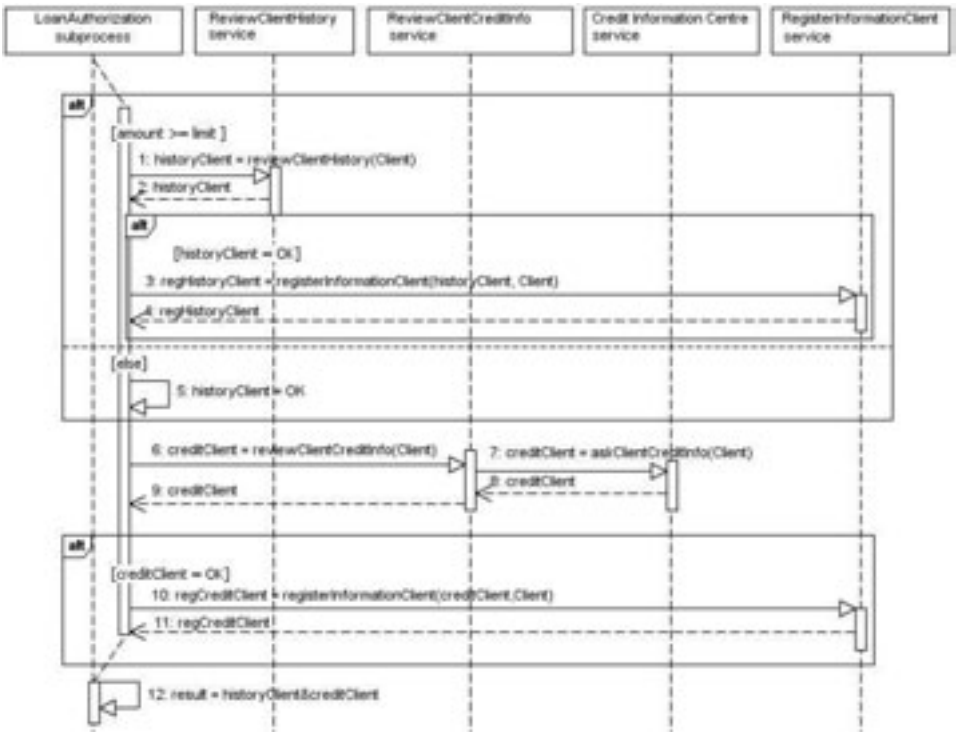


Figure 2: UML sequence diagram for SP2-"Loan Authorization" sub-process

As it can be seen in figure 2, the two invocations in the SP2-"Loan Authorization" sub-process to the services performing it, are concurrent (as it is stated in the business process model), and the result it returns is made up of the logical AND of the results from its invocations. The sequence diagram also helps us clarify operations in the contract of each service, or refine them if it shows that something is missing or misunderstood. So, it could be carried out when defining the service's contract, although the whole interaction of services is still unknown. The diagram shows how and with which other services each one interacts, and clearly identifies external services to be negotiated with partner organizations.

As service functionality has to be implemented, classes, subsystems and/or components associated with each service have to be defined too. In the example, the subsystems defined could be "ClientManagement", "LoanManagement", "AccountsManagement", assigning the services identified to them, i.e. the RegisterLoanRequest and RegisterLoanGranted services are assigned to the "LoanManagement" subsystem, and the "ReviewClientData" service -not shown here- is assigned to the "ClientManagement" one. Inside each subsystem the classes to implement the operations defined are designed, for now, in a conceptual way. We are working in defining transformations from BPMN to OMG service profile [UPMS07] to support the automatic derivation, when possible, of services classes from business processes.

5 Conclusions and future work

In this work, we have presented a service oriented methodology for the derivation of software services from business process models, which is included in a framework under development for the improvement of business processes. The methodology is composed of a core set of Disciplines, Activities, Deliverables and Roles, to guide the development of this type of software, taking into account the main characteristics defined by the paradigm.

Business processes are first-citizen classes since one of the most important activities prescribed is the business process modeling in the selected notation, applying process patterns to reuse known solutions to modeling problems. As part of the methodology, the derivation of services from business processes and sub-processes is stated following the guides of the defined activities. To illustrate its use, a case study is presented based on the generic "Grant Loan" of a bank, which explains step by step how to identify and obtain the needed services from business process models, and how to define the interaction of services to perform the identified business processes.

In this way, software services are derived from business process models in a straightforward way, so the next step to complete the methodological aspect of the framework we are working on, is to define transformations between models and metamodels from BPMN and the OMG service profile, to automate the defined derivations as much as possible. We will also investigate for other notations. We think that the methodology proposed here is simple but powerful enough to be easily applied in various environments complementing almost any software development process which needs to incorporate guides for service oriented development based on business processes.

References

[BPMI] Business Process Management Initiative,http://www.bpmi.org/

[BPMN] Business Process Modeling Notation (BPMN) Spec.v.1.1, Object Management Group (OMG),<http://www.bpmn.org/Documents/BPMN%201-1%20Specification.pdf>, 2008.

[CO06] COMPETISOFT – Process Improvement to increase competitively of Small and Medium Iberoamerican Software Industry, CYTED, http://alarcos.infcr.uclm.es/Competisoft/

[De06] Delgado, A.: Metodología para desarrollo de aplicaciones con enfoque SOA (Service Oriented Architecture), LatinAmerican Informatics Conference (CLEI'06), Santiago de Chile, Chile, 2006.

[De07] Delgado, A.: Metodología de desarrollo para aplicaciones con enfoque Service Oriented Architecture (SOA),Master Thesis, PEDECIBA, UdelaR, Montevideo, Uruguay, 2007.

[DGP06] Delgado, A.; González, L.; Piedrabuena, F.: Desarrollo de aplicaciones con enfoque SOA (Service Oriented Architecture), Iberoamerican conference on Software Engineering and Knowledge Engineering (JIISIC'06), Puebla, México, 2006.

[DGRP08] Delgado, A.; García-Rodríguez de Guzmán, I.; Ruiz, F.; Piattini, M.: Metodologías de desarrollo para Service Oriented Architectures con Rational Unified Process, Scientific-technical Conference on WS and SOA (JSWEB'08), Sevilla, España, 2008.

[DML06] De Castro, V.; Marcos, E., López Sanz, M.: A model driven method for service composition modelling: a case study, J.WebEngineering&Technology,Vol.2,No.4, 2006.

[DP00] Delgado, A.; Pérez, B.: Modelado de proceso de software, Proyecto Taller V, Instituto de computación, Facultad de Ingeniería, Universidad de la República, 2000.

[DP06] Delgado, A.; Pérez, B.: Modelo de Desarrollo de Software OO, Experimentación en un curso de Ingeniería de Software, Jornadas Iberoamericanas de Ingeniería de Software e Ingeniería del Conocimiento, JIISIC'06, Puebla, México, 2006.

[EA04] Endrei, M.; Ang, J.; Arsanjani, A. et. al: Patterns: Service Oriented Architecture and Web Services, IBM Redbook, SG24-6303-00, 2004.

[Er05] Erl,T.:Service OrientedArchitecture:Concepts,Technology,andDesign,PrenticeHall,2005.

[HZ05] Henkel, M.; Zdravkovic, J.: Supporting Development and Evolution of Service-based Processes, International Conference on e-Business Engineering (ICEBE'05)

[KBS05] Krafzig, D.; Banke, K.; Slama, D.: Enterprise SOA, Service Oriented Architecture Best Practices, Prentice Hall, 2005.

[LKT04] Liew, P.; Kontogiannis, K.; Tong, T.: A Framework for Business Model Driven Development,International Workshop on Software Technology and Engineering Practice (STEP'04), 2004.

[MDA03] Model Driven Architecture (MDA) Guide version 1.0.1, Object Management Group (OMG), <http://www.omg.org/mda/>, 2003.

[PJ06] Papazoglou, M.; Jan van den Heuvel, M.: Service Oriented design and development mehodology, J.WebEngineering&Technology,Vol.2,No.4, 2006.

[PJ07] Papazoglou, M.; Jan van den Heuvel, M.: Business Process development life cycle methodology, Communications of the ACM, Vol. 50, No. 10, October 2007.

[PTDL07] Papazoglou, M.; Traverso, P.; Dustdar, S.; Leymann, F.: Service Oriented Computing: State of the Art and Research Challenge, IEEE Computer Society, 2007.

[QDV05] Quartel, D.; Dijkman, R.; van Sinderen, M.: An approach to relate business and applications services using IDSL, International Enterprise Computing Conference (EDOC'05)

[RMC] IBM - RMC, <http://www- 306.ibm.com/software/awdtools/rmc/>

[RUP] IBM - RUP, <http://www- 306.ibm.com/software/awdtools/rup/>

[RBM06] Roser, S.; Bauer, B.; Müller, J.: Model- and Architecture-Driven Development in the Context of Cross-Enterprise Business Process Engineering, International Conference on Services Computing (SCC'06), 2006.

[SF03] Smith, H.; Fingar, P.: Business Process Management: The third wave, Meghan-Kieffer Press, 2003.

[SOMA] IBM – RMC - RUPforSOMA, Plug-inV2.4,<http://www.ibm.com/developerworks/rational/downloads/06/rmc_plugin7_1 /#12>

[SV06] Stahl, T.; Volter, M. et. al.: Model-Driven Software Development, Technology, Engineering, Management, John Wiley & Sons, Ltd., 2006.

[UML] Unified Modeling Language (UML) 2, Object Management Group (OMG), <http://www.omg.org/spec/UML/2.1.2/ >, 2007.

[CBK06] UML Profile and Metamodel for Services (UPMS),Object Management Group (OMG),<http://www.omg.org/docs/ad/07-06-02.odt>,<…/ad/08-05-03.pdf>, 2007/08

[VTKB03] van der Aalst, W.; ter Hofstede, A.; Kiepuszewski, B.; Barros, A. Workflow Patterns,
Distributed and Parallel Databases, 14(3), pages 5-51, July 2003

[WSBPEL] Web Services Business Process Execution Language Version 2.0, OASIS, <http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html>

[XPDL] XML Process Definition Language (XPDL), Workflow Management Coalition (WfMC)
<http://wfmc.org/index.php?option=com_docman&task=cat_view&gid=42&Itemid=72>

[ZHD07] Zdun, U.; Hentrich, C.; Dustdar, S.: Modeling Process-Driven and Service Oriented Architectures Using Patterns and Pattern Primitives, ACM Transaction on the Web, Vol.1, No.3, Article 14, September 2007.

A Formal Temporal Semantics for Microsoft Project based on Allen's Interval Algebra

Denis Gagné
Trisotech Inc.
3100 Cote Vertu, B380
Montreal, QC, H4R 2J8, Canada
dgagne@trisotech.com

André Trudel
Jodrey School of Computer Science
Acadia University
Wolfville, NS, B4P 2R1, Canada
andre.trudel@acadiau.ca

Abstract: Process modeling systems are complex and difficult to compare. A key attribute of any process modeling formalism or tool is time which involves how it handles and represents temporal dependencies and constraints. We are interested in doing a temporal based comparison between process modeling formalisms and tools by first converting them to a common representation. The temporal representation chosen is Allen's interval algebra. In this paper, we explain how to convert a project specified in Microsoft Project to a set of logical formulas. This conversion provides a formal temporal semantics for Microsoft Project.

.

1 Introduction

Business processes require the coordinated execution of individual activities to achieve a common business goal. A process model formally describes the structure of the workflow and how these atomic activities are coordinated and enacted by central or distributed workflow enactment services [Wf95].

There exists a variety of process modeling formalisms and tools allowing one to capture process models at different levels of abstraction and addressing various business and-or technological concerns.

In recent years, many research initiatives have been dedicated to defining various workflow patterns. Each of the resulting pattern frameworks provides process constructs from a different perspective such as control [Aa03], data [Ru04a], resources [Ru04b], and exceptions [RAH06]. These pattern frameworks are useful for:

- supporting business process modeling efforts,
- as a reference for learning basic and complex process concepts, and
- to assess and compare the expressiveness of various formalisms and tools.

We are primarily interested in the last use above but, from a temporal perspective. We do not use the traditional pattern approach of defining recurring temporal constructs within processes and then using these constructs to compare process models. We instead propose to compare process models in a more meaningful manner on the basis of formal semantics.

For each process modeling formalism or tool, we plan to convert its temporal constructs into first order temporal logic formulas derived from Allen's interval algebra. The resulting set of formulas will provide semantics for the process modeling formalism or tool. From there, the set of formulas derived from one formalism or tool will be compared against sets of formulas derived from other formalisms or tools.

Microsoft Project (MS Project) is not a process modeling tool per se, as it cannot capture some of the basic control constructs of process models such as decision (or-split) and loops. Nevertheless, MS Project can still be used to model certain simple classes of processes. For example, we can capture in MS Project all activities related to the opening of a new store. We can then use this MS Project file as a franchise template by instantiating the series of coordinated activities captured in the MS Project file for all future store openings.

In this paper, we convert a project specified in MS Project to a set of logical formulas. We show how to convert each of MS Project's temporal constructs to a formula in Allen's interval algebra. One advantage of our semantics is that the formulas clarify some unexpected patterns allowed by MS Project. As a direct result of our deeper understanding of MS Project's semantics, we were able to correctly semantically parse MS Project files as process model inputs for a workflow management system which is currently under development. In our ongoing research, we are using the formulas to compare MS Project from a temporal perspective to various process modelling formalisms and tools.

The management of time in the context of business processes has received some attention in the past few years (e.g. [CP02, CP03, CP04, CP06, EGP00, EP01, Lu06, Me03, Me04, and ZS99]). But, none of these previous papers deals with the problem of capturing the temporal constraints within a specific process modelling tool that is currently used and popular in the real world. Another difference between this paper and previous work in the area is the adoption of Allen's relations. Although Allen's interval algebra has been applied to many application areas, it has not received much attention in the business process community. To the best of our knowledge, only [Lu06] uses Allen's algebra as the basis for flexible business process execution via constraint satisfaction.

The remainder of this paper is organized as follows. After providing a brief overview of Allen's interval algebra, we introduce various temporal constructs in MS Project. For each construct, we provide a brief overview, followed by its semantics based on Allen's interval algebra. We conclude with an example.

2 Allen's interval algebra

The most popular temporal reasoning approach in Artificial Intelligence is due to Allen [Al83]. Allen's approach is based on intervals and the 13 possible binary relations between them. The first relation is "precedes" which is represented by the letter "p". Interval A precedes interval B if A ends before B starts, and is written as "A p B". If A

precedes B, then it is also the case that B is preceded by A. This inverse relation for precedes, “preceded by”, is represented by “pi”. The precedes relation is shown at the top of figure 1. The diagram for precedes shows interval A to the left of interval B:

The diagram’s representation is shown to its right and left. In this case, we have “A p B” and its equivalent inverse notation “B pi A”.

The other relations are meets (m), overlaps (o), during (d), starts (s), finishes (f), and equals (e). As with proceeds, each of these relations has an inverse which is represented by appending an “i” to the relation symbol: mi, oi, di, si, and fi. The inverse of equals is equals. We refer to the 13 relations as the basic labels and they are all shown in figure 1.

Allen’s interval relations are mutually exhaustive. For example, given two intervals, exactly one of the 13 relations will hold between them. It is impossible to have none or, two or more relations true between two temporal intervals.

Often, there is uncertainty as to exactly which relation holds between two intervals. For example, supper may be before or after going to the movies. We write this as:

(supper p movies) xor (supper pi movies)

Since exactly one of “p” or “pi” will be true, we use an exclusive-or (i.e., “xor”). A shorthand notation for the above formula is:

supper {p, pi} movies

In general, the relation between two intervals is allowed to be any subset of

I = {p,pi,m,mi,o,oi,d,di,s,si,f,fi,e}

including I itself.

We extend Allen`s approach with points. We represent a temporal interval A by its endpoints written as (A-, A+). For example, the interval A = (1,10) has a left endpoint A- = 1, and right endpoint A+ = 10. All intervals are convex (i.e., there are no gaps in the interval).

The underlying temporal structure is assumed to be discrete, linear, and totally ordered. An example of such a structure is the integers.

3 MS Project

MS Project [Ms07] is a product within the Microsoft Office Suite that allows the user to specify, plan and schedule various types of projects. A feature of MS Project is that it can calculate a realistic schedule for a project based simply on task durations and task dependencies. Other basic scheduling controls within MS Project include the Start and

Finish dates of tasks, and the various available Calendars. MS Project uses four types of calendars: the base calendar, project calendar, resource calendar, and task calendar.

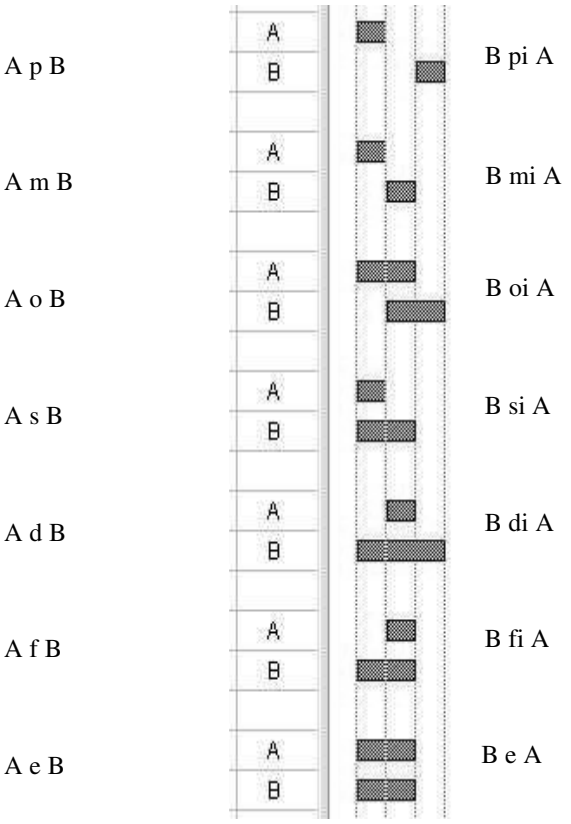


Figure 1. Allen's 13 relations

When a new project is created, the user must specify whether MS Project is to schedule the project from a specified start or finish date (start date: The date when a task is scheduled to begin. This date is based on the duration, calendars, and constraints of predecessor and successor tasks. A task's start date is also based on its own calendars and constraints.). As a default, MS Project will assume that the project is to be scheduled from the project start date, and that the start date is the current date at the time the new project is created.

Tasks within MS Project are entered in a task list. For example, tasks A, B, and C in figure 2. These tasks can be further organized and structured with the Summary Tasks feature. For example, the summary task at the top of figure 2 contains sub-tasks A, B, and C. Summary tasks can be used to summarize, show, or hide subtasks (e.g., compare

the top expanded view to the bottom collapsed view of the same summary task in figure 2).

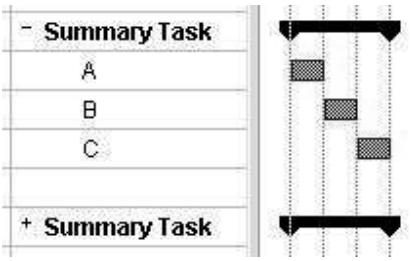


Figure 2. An expanded and collapsed summary task with subtasks

Placing tasks in such a hierarchical structure does not create or imply any task dependencies among themselves. It is possible for a pair of tasks, whether they contain subtasks or not, to be related by any of Allen's thirteen basic interval relations. For example, even though tasks A and B in figure 2 are drawn as if there is a meets relationship between them, this is not necessarily the case. Any of the 13 relations can hold between A and B.

The sub-tasks within a summary task must occur during the summary task. For example, the summary task in figure 2 can only terminate after A, B, and C have all terminated.

Note that upon creation of a Task, MS Project by default assigns an "As Soon As Possible" start constraint to the newly created task (this constraint is discussed later).

Non trivial projects contain many tasks which depend on one another. A Task Dependency is a relationship between two tasks in which one task depends on the start or finish of another task in order to begin or end. The task that depends on the other task is the Successor, and the task it depends on is the Predecessor. Together, these two types of tasks help bind and give structure to a project.

Task dependencies allow MS Project to shift (recalculate) the schedule of all successor tasks whenever the start, finish or duration of any predecessor task is changed. The shift usually has a cascade effect throughout the project.

Task dependencies can be further refined with delays called Lead and Lag Time. Lag Time can be used to specify a delay between the finish of the predecessor and the start of the successor task. For example, we need a delay between the finish of the task "Paint wall" and the start of the next task "Hang pictures" to allow the paint to dry. Lag time can be specified either as a duration, such as 1 day, or as a percentage of the predecessor's duration, such as 25%. For example, if "Paint wall" has a 4 day duration, entering 1day or 25% would result in a 1 day delay to allow the paint to dry before starting the task "Hang pictures".

Lead Time causes the overlap of two tasks. In this case, the successor starts before the predecessor finishes. Lead time is useful when a successor task requires a head start. Lead time is entered as negative lag, such as -1 day or -25%. For example, for the tasks "Construct walls" and "Plaster walls," we can use lead time to begin "Plaster walls" when "Construct walls" is half done.

In MS Project, both lag and lead time are specified using a single value. A positive lag specifies a Lag Time, a negative lag specifies a Lead Time, and a zero lag specifies that there is neither a Lag nor a Lead Time.

4 Temporal semantics

In this section, we provide a temporal semantics for MS Project based on Allen's relations. Specifically, we give an axiomatization for summary tasks, constraints on individual tasks, and constraints between tasks. First, we introduce a function and some constants.

Lag is always specified relative to two tasks. We represent the lag as a binary function. The function lag(A,B) returns the lag between tasks A and B (i.e., the delay desired between the termination of task A and the start of task B). For example, a lag of 5 between tasks A and B is written as lag(A,B)=5. If lag(A,B) is negative, it represents a lead time.

We represent the project's start and finish date with the constants project-start-date, and project-finish-date respectively. Note that these constants represent a point and not an interval.

For summary tasks, assume a summary task A contains sub-tasks A1, A2,…, An, then each sub-task occurs during A:

$$\forall Ai \quad A- \leq Ai- \leq Ai+ \leq A+ \qquad \text{(ST)}$$

Note that one or more sub-tasks can start or end at the same instant as the summary task A.

MS Project allows the user to specify temporal constraints between tasks, and also apply them to an individual task. In the next sub-section, we describe the possible temporal constraints between two tasks. For each, we provide a definition followed by a first order logical representation using Allen's Interval Algebra. The second sub-section does the same with the temporal constraints applied to a single task.

4.1 Temporal relationships between tasks

Temporal relationships between two tasks in MS Project are represented with links between the tasks. The links can be used to specify four types of task dependencies:

1. Finish-to-start (FS),
2. Finish-to-finish (FF),
3. Start-to-start (SS), and
4. Start-to-finish (SF).

Each of the four types of dependencies is described below. For each, we also provide a formal semantics based on Allen's interval algebra.

4.1.1 Finish-to-start (FS)

A Finish-to-start link or dependency between tasks A and B means that B cannot start until A finishes. For example, the task "Paint fence" cannot start until "Construct fence" finishes. This is the most common type of dependency and is pictured at the top of figure 3.

The Finish-to-start semantics is not obvious. When the lag variable is zero, we can have a meets relationship between the tasks as shown at the top of figure 3. But, MS Project also allows the user to move either task A to the left or move B to the right to create a precedes relationship between the tasks. Note that in this case the lag variable remains at zero. The Finish-to-start semantics in terms of Allen's interval algebra is:

$$lag(A,B)=0 \rightarrow A\{p,m\}B \qquad \text{(FS)}$$

We can specify a Finish-to-start relationship with a Lag Time by setting the lag variable to a positive value. This has the effect of separating the tasks and the predecessor is always to the left of the successor. For example, if we change the lag variable from zero to 5, we have the situation shown in the middle of figure 3. The relationship between tasks A and B is precedes. Either task can be shifted to increase the distance between them. Note that increasing the distance does not change the value of the lag variable nor the precedes relation between the tasks. When lag is set to 5, MS Project will not allow tasks A and B to be closer than 5 units apart. The semantics of a Finish-to-start with a Lag Time is:

$$lag(A,B)>0 \rightarrow [[B- - A+ \geq lag(A,B)] \,\&\, A\{p\}B] \qquad \text{(FS lag)}$$

A negative lag variable is used to specify a Finish-to-start relationship with a Lead Time. The bottom diagram in figure 3 shows a Finish-to-start relationship with lag=-3. In this case, A is preceded by B. If we hold A fixed, B cannot be shifted to the left. But, B can be shifted anywhere to the right without altering the value of lag. The result is that any relationship is possible between A and B:

$$lag(A,B)<0 \rightarrow [[B- \geq (A+ - | lag(A,B) |)] \,\&\, AIB] \qquad \text{(FS lead)}$$

4.1.2 Start-to-start (SS)

A Start-to-start link or dependency between tasks A and B means that B cannot start until A starts. For example, if A is "Pour foundation" and B is "Level concrete," "Level

concrete" cannot begin until "Pour foundation" begins. The semantics for Start-to-start is:

lag(A,B)=0 → A{p,m,o,fi,di,s,e,si}B (SS)
lag(A,B)>0 → [[B- - A- ≥ lag(A,B)] & A{p,m,o,fi,di}B] (SS lag)
lag(A,B)<0 → [[B- ≥ (A- - | lag(A,B) |)] & AIB] (SS lead)

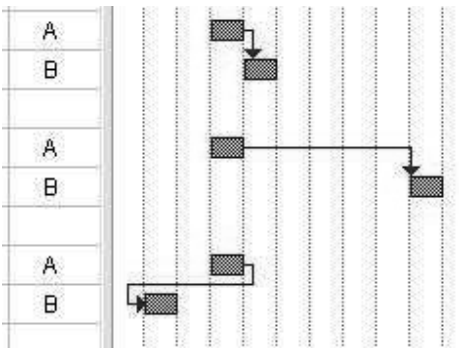


Figure 3. Finish-to-start dependencies, with Lag Time and Lead Time

4.1.3 Finish-to-finish (FF)

A Finish-to-finish link or dependency between tasks A and B means that B cannot finish until A finishes. For example, if A is "Add wiring" and B is "Inspect electrical," "Inspect electrical" cannot finish until "Add wiring" finishes. Note that with a Finish-to-finish dependency, B can finish later than A's finish time. The semantics is:

lag(A,B)=0 → A{p,m,o,fi,s,e,d,f}B (FF)
lag(A,B)>0 → [[B+ - A+ ≥ lag(A,B)] & A{p,m,o,s,d}B] (FF lag)
lag(A,B)<0→[[B+ ≥ (A+ - | lag(A,B) |)] & AIB] (FF lead)

4.1.4. Start-to-finish (SF). A Start-to-finish link or dependency between tasks A and B means that B cannot finish until A starts. This dependency type can be used for just-in-time scheduling up to a milestone or the project finish date to minimize the risk of a task finishing late if its dependent tasks slip. The semantics is:

lag(A,B)=0 → [A I-{pi} B] (SF)
lag(A,B)>0 → [[B+ - A- ≥ lag(A,B)] & [A I–{pi,mi} B]] (SF lag)
lag(A,B)<0→[[B+ ≥ (A- - | lag(A,B) |)] & AIB] (SF lead)

4.2 Constraints on an Individual Task

Constraints can also be specified to control the start or finish date of a task. These constraints can be flexible (not tied to a specific date) or inflexible (tied to a specific date).

4.2.1 Flexible Constraints

A Flexible Constraint does not specify a specific date for a task. The flexible constraints are:

- As Soon As Possible (ASAP),
- As Late As Possible (ALAP),
- Finish No Earlier Than (FNET),
- Finish No Later Than (FNLT),
- Start No Earlier Than (SNET), and
- Start No Later Than (SNLT).

Flexible constraints work in conjunction with task dependencies to make a task occur as soon or as late as the task dependency will allow. For example, a successor task with an As Soon As Possible (ASAP) constraint and a finish-to-start dependency will be scheduled as soon as the predecessor task finishes.

Constraints with moderate scheduling flexibility restrict a task from starting or finishing before or after a specified date. For example, a successor task with a Start No Later Than (SNLT) constraint of June 15 and a finish-to-start dependency can begin any time its predecessor is finished up until June 15, but can't be scheduled after June 15.

The flexible constraints are described in detail below.

As Soon As Possible (ASAP)
If a task is assigned an As Soon As Possible constraint, MS Project schedules the task as early as it consistently can. No additional date restrictions are put on the task. This is the default constraint for newly created tasks in projects scheduled from the project start date. Note that the task will not be scheduled by MS Project before project-start-date. The semantics is:

minimize(A-) & A-≥ project-start-date (ASAP)

As Late As Possible (ALAP)
This constraint is analogous to the ASAP constraint, but relative to the end of the project. If a task is assigned an As Late As Possible constraint, MS Project schedules the task as late as it consistently can, given other scheduling parameters. No additional date restrictions are put on the task. This is the default constraint for newly created tasks in projects scheduled from the project finish date. If you choose to schedule your project from the project finish date, MS Project will determine how late you can start your project and still finish by the specified project finish date. The semantics is:

maximize(A+) & (A+ ≤ project-finish-date) (ALAP)

Finish No Later Than (FNLT)
This constraint indicates the latest possible date that this task is to be completed. It can be scheduled to finish on or before the specified date. A predecessor task cannot push a successor task with an FNLT constraint past the constraint date. The semantics for "task A FNLT a specific date d" is:

$$A+ \leq d \qquad \text{(FNLT)}$$

Note that every occurrence of "d" in the formula above and the remainder of this section is a point and not a temporal interval.

Start No Later Than (SNLT)
This constraint indicates the latest possible date d that this task can begin. The task can be scheduled to start on or before the specified date. A predecessor task cannot push a successor task with an SNLT constraint past the constraint date. For projects scheduled from the finish date, this constraint is applied when a start date is entered for a task. The semantics is:

$$A- \leq d \qquad \text{(SNLT)}$$

Finish No Earlier Than (FNET)
This constraint indicates the earliest possible date d that this task can be completed. The task cannot be scheduled to finish any time before the specified date. For projects scheduled from the start date, this constraint is applied when a finish date is entered for a task. The semantics is:

$$A+ \geq d \qquad \text{(FNET)}$$

Start No Earlier Than (SNET)
This constraint indicates the earliest possible date d that a task can begin. The task cannot be scheduled to start any time before the specified date. For projects scheduled from the start date, this constraint is applied when a start date is entered for a task. The semantics is:

$$A- \geq d \qquad \text{(SNET)}$$

4.2.2 Inflexible Constraints

An Inflexible Constraint ties a task to a date. In MS Project, the inflexible constraints are:

- Must Finish On (MFO) and
- Must Start On (MSO).

Inflexible Constraints override any task dependencies and restrict a task to a date. For example, a task with a Must Start On (MSO) constraint for September 30 and a finish-to-start dependency to another task will always be scheduled for September 30 whether its predecessor finishes early or late. The inflexible constraints are described below.

Must Start On (MSO)
This constraint indicates the exact date on which a task must be scheduled to begin. Other scheduling parameters such as task dependencies, lead or lag time and delay cannot affect scheduling the task unless this requirement is met. The semantics for "task A has an MSO of date d" is:

$$A- = d \quad \text{(MSO)}$$

Must Finish On (MFO)
This constraint indicates the exact date d on which a task must be scheduled to be completed. Other scheduling parameters such as task dependencies, lead or lag time, resource leveling, and delay cannot affect scheduling the task unless this requirement is met. The semantics is:

$$A+ = d \quad \text{(MFO)}$$

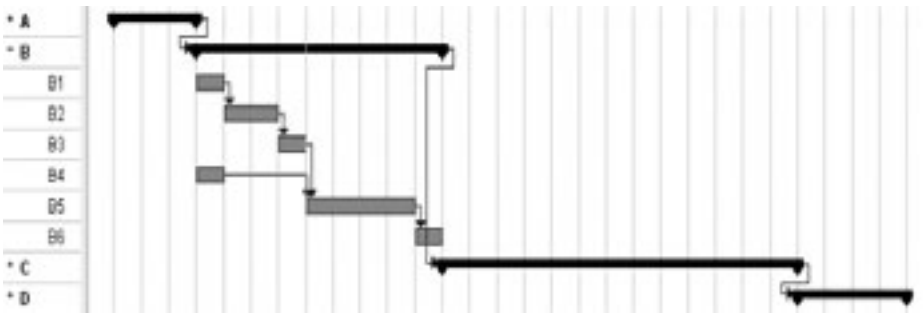


Figure 4. Finish-to-start dependencies, with Lag Time and Lead Time

5 Example

A simple example of an MS Project is shown in figure 4. We use this example to show how the formulas in the previous sections can be iteratively applied to provide a temporal semantics.

The example in Figure 4 captures a simple software development process where A stands for the "Inception" phase, B is the "Elaboration" phase, C is the "Construction" phase and D is the "Transition" phase. Further details of the "Elaboration" phase B are provided: B1 stands for "Complete Analysis", B2 is "Prepare Use Cases", B3 is "Use Cases Review Meeting", B4 is "Risks Analysis", and B5 and B6 are "Design Model" and "Design Review Meeting" respectively.

We derive the set of formulas (EX-1-20) for this example by iteratively applying the temporal formulas introduced in this paper. The example was drawn with a lag of zero between tasks and sub-tasks:

lag(A,B)=0 (EX-1)
lag(B,C)=0 (EX-2)
lag(C,D)=0 (EX-3)
lag(B1,B2)=0 (EX-4)
lag(B2,B3)=0 (EX-5)
lag(B3,B5)=0 (EX-6)
lag(B4,B5)=0 (EX-7)
lag(B5,B6)=0 (EX-8)

Note that the link between B4 and B5 in figure 4 may be misleading. It appears as if the lag is 3. It was indeed specified as zero.

There is a finish-to-start dependency between summary tasks A, B, C and D. From (EX-1-3) and (FS) we derive:

A{p,m}B (EX-9)
B{p,m}C (EX-10)
C{p,m}D (EX-11)

We derive similar formulas for the sub-tasks B1-B6:

B1{p,m}B2 (EX-12)
B2{p,m}B3 (EX-13)
B3{p,m}B5 (EX-14)
B4{p,m}B5 (EX-15)
B5{p,m}B6 (EX-16)

Using axiom (ST), we capture the temporal relationships between the summary task B and its sub-tasks:

$\forall Bi \quad B- \leq Bi- \leq Bi+ \leq B+$ (EX-17)

Optionally, we can explicitly represent the fact that there is no known temporal relationship between B4 and B1, B2, B3:

B1 I B4 (EX-18)
B2 I B4 (EX-19)
B2 I B4 (EX-20)

6 Conclusion & future work

There are many diverse process modeling formalisms and tools. Since they are often based on different paradigms, they are difficult to compare. Our long term goal is to compare these process modeling formalisms and tools based on their temporal capabilities.

Rather than simply identifying temporal modeling constructs or patterns, we have opted to compare process models in a more meaningful manner on the basis of formal

semantics. The advantage of this approach is that it leads to a deeper semantical notion of process model equivalence rather than simple comparison of expressiveness based on constructs.

Another advantage of our approach is that once a formal semantics has been specified for a particular project, the resulting formulas can be checked for consistency (e.g., an off the shelf constraint satisfaction package can be used). We may also be able to derive interesting conclusions from the formulas.

For our temporal formalism, we chose Allen's which is the most popular in Artificial Intelligence. As for a first process modelling framework, we chose MS Project because it is the most widely used and known project planning tool in industry. Although MS project is not a process modeling tool per se and its temporal constraints seem relatively simple, the study of their semantics leads to interesting results which are not immediately evident from naïve usage or simply reading the documentation provided.

Although MS Project is not as expressive with respect to the control perspective as other process modeling formalisms (e.g., lacking expressiveness for decision (or-split) and loops), it is in fact more expressive with respect to the temporal perspective [GT08].

As a direct result of our deeper understanding of MS Project's temporal semantics, we are developing an MS Project parser for a workflow management system. This parser allows us to convert MS Project files to equivalent process models that are semantically correct with respect to the temporal perspective. These initial models can then be augmented using another notation which is more expressive with respect to the control perspective (e.g., BPMN [Om06]).

Acknowledgements

The second author is supported by an NSERC Discovery Grant.

7 References

[Aa03] van der Aalst, W.M.P. et. al.: Workflow Patterns. Distributed and Parallel Databases, 14(3), 2003, pp. 5-51.

[Al83] Allen, J.F.: Maintaining Knowledge about Temporal Intervals, Communications of the ACM, 26, 1983, pp. 832-843.

[CP02] Combi, C., and Pozzi, G.: Towards Temporal Information Workflow Systems. In *ER 2002*, Berlin: Volume 2784, Lecture Notes in Computer Science, Springer-Verlag, 2002, pp. 13-25.

[CP03] Combi, C., and Pozzi, G.: Temporal Conceptual Modelling of Workflows. In *ER 2003*, Berlin: Volume 2813, Lecture Notes in Computer Science, Springer-Verlag, 2003, pp. 59-76.

[CP04] Combi, C., and Pozzi, G.: Architectures for a Temporal Workflow Management System. In Proceedings of the 2004 ACM Symposium on Applied Computing (SAC'04), Nicosias, Cyprus, 2004.

[CP06] Combi, C., and Pozzi, G.: Task Scheduling for Temporal Workflow Management System. In Proceedings of the Thirteenth International Symposium on Temporal Representation and Reasoning *(TIME'06)*: IEEE, 2006.

[EGP00] Eder, J., Gruber, W., and Panagos, E.: Temporal Modeling of Workflows with Conditional Execution Paths, In Database and Expert Systems Applications, Springer, 2000, pp. 243-253.

[EP01] Eder, J., and Paganos, E.: Managing Time in Workflow Systems. In Workflow Handbook 2001, Layna Fischer (Ed.), Future Strategies Inc., USA, 2001.

[GT08] Gagné, D., and Trudel, A.: The Temporal Perspective: Expressing Temporal Constraints and Dependencies in Process Models. To appear in The 2008 BPM and Workflow Handbook. Future Strategies, 2008.

[Lu06] Lu, R. et. al.: Using a Temporal Constraint Network for Business Process Execution. In Proceedings of the 17th Australian Database Conference(ADC2006*)*. Hobart, Australia: Volume 49, Conferences in Research and Practice in Information Technology (CRPIT), 2006.

[Me03] Meyer, A. et. al.: ICENI Dataflow and Workflow: Composition and Scheduling in Space and Time. In Proc of UK e-Science All Hands Meeting, EPSRC, 2003.

[Me04] Meyer, A. et. al.: Workflow Expression: Comparison of Spatial and Temporal Approaches. In Workflow in Grid Systems Workshop (GGF-10*)*, Berlin, 2004.

[Ms07] Microsoft Project: Online Documentation. Retrieved April 2, 2007. http://office.microsoft.com/enca/assistance/CH790018101033.aspx

[Om06] OMG.: Business Process Modeling Notation (BPMN) Specification. Version 1.0, OMG report: dtc/06-02-01, OMG, 2006

[Ru04a] Russell, N. et. al.: Workflow Data Patterns, QUT Technical report, FIT-TR-2004-01, Queensland University of Technology, Brisbane, 2004.

[Ru04b] Russell, N. et. al.: Workflow Resource Patterns, BETA Working Paper Series, WP 127, Eindhoven University of Technology, Eindhoven, 2004.

[RAH06] Russell, N., van der Aalst, W.M.P., and ter Hofstede, A.H.M.: Workflow Exception Patterns. In Proceedings of the 18th International Conference on Advanced Information Systems Engineering (CAiSE'06*)*, Berlin: Volume 4001, *Lecture Notes in Computer Science*, Springer-Verlag, 2006, pp. 288-302.

[Wf95] WFMC: The Workflow Reference Model, WFMC-TC-1003, 1.1, 1995.

[ZS99] Zhao, J.L., and Stohr, E.A.: Temporal Workflow Management in a Claim Handling System. In Proceedings of the international joint conference on Work activities coordination and collaboration (WACC '99*)*. New York: ACM Press, 1999, pp. 187-195.

Evolution of Business Process Models and Languages

Stefan Jablonski, Bernhard Volz, Sebastian Dornstauder

Chair for Applied Computer Science IV
University of Bayreuth
Universitätsstraße 30
95447 Bayreuth, Germany
{stefan.jablonski,bernhard.volz,sebastian.dornstauder}@uni-bayreuth.de

Abstract: "The only constant is change" is an often cited phrase. We regard it as predominant for the area of process based information systems. In this paper we investigate how evolution of process based information systems can be supported by a process modeling framework that easily can be adjusted to changing requirements of an application domain. Our key contribution is the provision of a system infrastructure that supports the adaptation of both process modeling languages and process models to evolving application requirements. Our approach is based on a multi level meta modeling framework.

1 Introduction

"The only constant is change" is a common quotation in literature when business process management is characterized. Without anticipating the introduction of a modeling hierarchy (Section 3), the phenomena of change can be classified according to the process modeling level it occurs. Starting at the "lowest" level, running process instances might have to be changed to react to a sudden shift in the application. Among others, [Aa00], [El95] and [Ri04] are investigating this issue and suggest adequate solutions. Stepping one level up, the process type (process model, process definition) might have to be changed since it has become obvious that from now on a certain application will be performed in a different way [Aa00] [He99]. Nevertheless, it is possible to even step up another level in a process modeling hierarchy. Change on this level means altering the modeling language used to define process models. We focus this third interpretation of change in this paper and investigate changes of process modeling languages (PMLs) since this is the kind of change which is not sufficiently dealt with in research and promises powerful means to implement process oriented information systems adequately. Nevertheless, we also discuss changes of process models since this is close to the change of a PML. Changing process instances is neglected because it is already discussed broadly in [Ja06].

Why is the change of a PML an issue that is worth to be investigated? One can argue that a PML should always remain untouched. However, we fully comply with the interpretation of change as given in [Cl08]; there, change is related to diversity. The authors of [Cl08] state that "life would be much easier if there was only one programming language and one deployment platform". They notice that diverse domains will be characterized by diverse customer requirements. This observation can seamlessly be adopted in the business process management domain. Here, the programming language is represented by the modeling language and the deployment platform corresponds to the process execution infrastructure.

We fully subscribe to the argument of [Cl08] that the right (process modeling) languages enable developers to be significantly more productive. Besides we agree with the requirement that "we need the ability to rapidly design and integrate semantically rich languages in a unified way". This means on the one hand that each domain may and finally has to create its individual, specific language (domain specific language). On the other hand it means that a common starting point for these language developments is assumed. It is important to sustain – despite the diversity of domain specific languages – a kind of comparability and compatibility between them. We finally agree that meta modeling provides capabilities to achieve this.

We started with the discussion of change and ended up in domain specific process modeling languages (DSPMLs). This is due to the fact that a DSPML is evolving from a standard PML. Thus, from a technical perspective evolution means changing models over time. Due to our specialized architecture being able to cope with evolution of a PML means to be also capable to cope with the evolution of process models.

We present a meta modeling approach which supports the definition of DSPMLs. The special feature of our approach is that domain specific languages are derived from a common basic language which most probable will be a sort of standard language. All language definitions will be based on a meta model. This strategy shows major advantages.

- All derived domain specific languages share a common set of modeling constructs. Thus, they remain compatible and comparable to a certain extent.
- The definition of a domain specific language can be done in a systematic way by extending the meta model of such a language.
- Extensions made for one domain specific language could be inherited by other domains, i.e. domain specific languages, if it is considered to be valuable for the new domain as well. This feature supports reuse of modeling constructs greatly.
- Tools can be built that support different domain specific languages at the same time. It is not necessary to build a special tool for each domain specific language.

The focus of this paper is on developing tools that support evolution of PMLs and process models. As we have discussed before, domain specific languages play an important role in that scenario. The foundation of a domain specific processes modeling tool is discussed in Section 2. Section 3 then illustrates its basic part, a meta model stack. Several use cases of evolution are analyzed in Section 4. Section 5 finally discusses related work.

2 The Foundation of Perspective Oriented Process Modeling

Perspective Oriented Process Modeling (POPM) is presented in general in [Ja94] and [JB96]. The runtime and visualization aspects of POPM are discussed in more detail in [Ja06] and [JG07], respectively. Since POPM itself is subject of an ongoing evolution, POPM now combines a couple of matured modeling concepts in a new and synergetic manner. These modeling concepts are introduced in the following.

2.1 Layered Meta Modeling

Meta modeling techniques are commonly used to describe the structure of models; thus "Meta Model" is often defined as "model of a modeling language" (e.g. [Se03]). The system under study in our case is a model, for instance a process model "Travel Claim Reimbursement Process". A PML has to be used to describe this process, i.e. this process model. We also use a model to describe such a PML; this model is indeed a meta model defining the structure (syntax) of our PMLs within the POPM framework. According to the Meta Object Facility (MOF, [OMG06a]) this model then becomes part of a meta model stack which consists of several, linearly ordered layers. Since MOF restricts modelers to a specific set of features which is not sufficient for our purpose, our solution is not strictly following the modeling rules of MOF.

In Fig. 1 (meta model stack of our POPM framework), actual process models are defined on M1 (right boxes). A process model uses process definitions (and data definitions, organization definitions etc.) which are all gathered in the "Type library" on M1 (left box). It is noteworthy to mention that on M1 the concept "type/usage" is applied: process types are defined (and put into the type library) and then are "used" in other process models (e.g. as sub-processes) to define the latter. M0 contains running instances of process types formerly defined on M1 (right boxes).

All process definitions on M1 are defined in an application specific language which must have previously been defined at M2. M2 contains the definition of an abstract process meta model (APMM) that defines a set of general language features, e.g. a standard language like BPMN. Basic concepts such as Processes, Data Flow or Control Flow are defined in such an APMM. All domain specific language dialects can also be found on M2 as a specialization of the APMM.

An abstract process meta meta model (APM^2M) at M3 defines basic modeling principles; for instance, it is defined that processes consist out of "bubbles and arcs" (directed graph) or that nesting of modeling elements is allowed. This APM^2M thus defines the fundamental structure, i.e. the structural templates, for PMLs specified on the lower level.

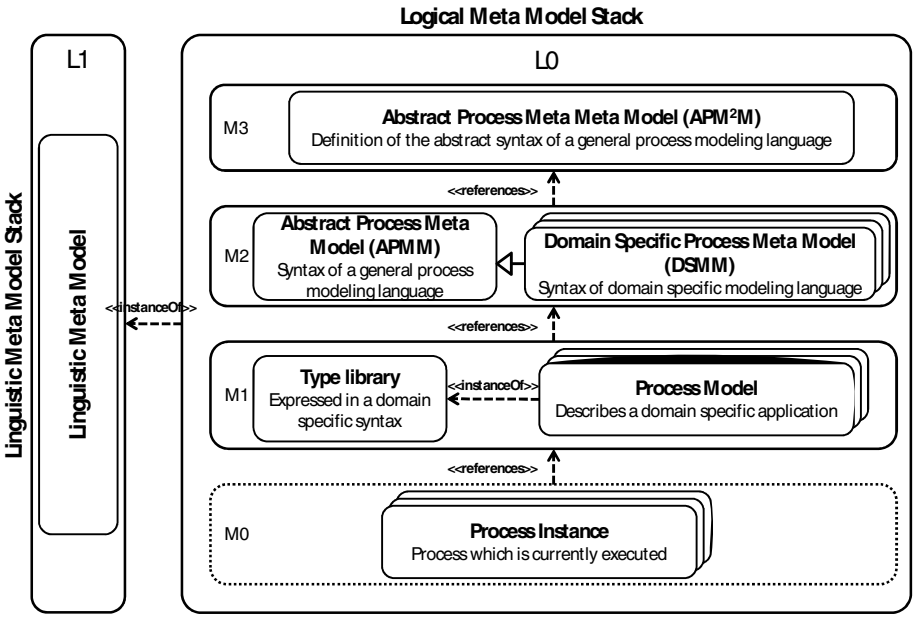


Fig. 1. The meta model stack of POPM

Following the architecture of Fig. 1 (Logical Meta Model Stack) allows for

- exchanging the modeling paradigm (graph based process models) at M3,
- defining DSPMLs at M2 as specializations of a general modeling language (APMM) and
- adapting process models at M1.

We believe that these capabilities provide a powerful basis for the evolution of PMLs and process models.

2.2 Extended Powertypes

We have mentioned that the APM^{2}M on M3 defines process models to be interpreted as graphs; for a tool it is then often necessary not only to recognize each element of such a graph together with its attributes (and operations) but also to know the capabilities (features) of each element. For example, both "Process" and "Start-Interface" are nodes of a process model graph. As in a graph each node can be connected with other nodes in principle, also the Start-Interface could have an incoming Control Flow arc. Obviously this needs to be prohibited since the Start-Interface denotes the beginning of a process and thus should not receive any incoming flows at all. Thus a capability "canHaveIncomingControlFlows" can be defined at M3. Such a capability (e.g. to have incoming flows) is defined as an attribute of the powertype. These values define which capabilities of the second type of the pattern, the so-called partitioned type, should be activated (they must be set to "true"). Furthermore only those attributes of the partitioned type are inherited by new constructs whose capability attribute has been set to "true". Thus our extension does not only activate or deactivate a set of certain features but also removes them physically from new constructs such that complex runtime checks dealing with disabled features can be avoided. But the main benefit is that this pattern eases the definition of completely new modeling constructs at M2 since the user can define which features are supported by a new construct easily.

2.3 Logical and Linguistic Modeling

In [AK05] an orthogonal classification approach is introduced; this approach contains two meta stacks instead of only one which are orthogonal to each other (cf. Fig. 1, Linguistic Meta Model Stack). One stack contains a meta model that describes how models (of the application domain that have to be defined) can be stored (linguistic model, e.g. "how is an attribute stored"). The other stack hosts the logical model which is purely content related.

It is crucial for this architecture that each layer of the logical stack can be expressed in the same linguistic model. As a result a modeling tool can be built that allows users to modify all layers of the logical stack in the same way since all logical layers are described by the same linguistic model. This is not the usual way modeling tools are built. Conventional modeling tools do not support an explicit linguistic model and thus can usually modify only one layer of a logical model hierarchy [AK05]. Therefore more than one tool is required if both, PMLs and process models should be modified. As the number of required modeling tools is directly proportional to the number of domain specific languages, it is better to provide only modeling tool that supports many DSLs along with the ability to modify process models defined in the corresponding DSLs.

Our goal is to implement a tool for the POPM framework that is capable of handling evolution on the various levels of our meta modeling hierarchy. In this section we will introduce the basic concepts for this tool implementation. Section 4 demonstrates how evolution can be accommodated by the tool.

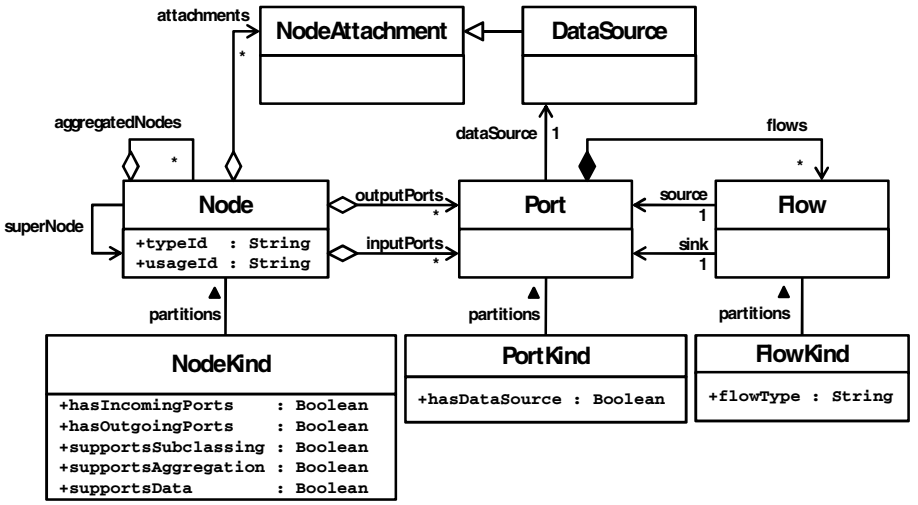


Fig. 2. The APM^{2}M – defining the fundamental modeling method

2.4 Abstract Process Meta Meta Model (APM^{2}M)

As explained in Section 2, the APM^{2}M is located at M3 and provides basic structures for PMLs defined on M2, i.e. it prescribes the structure of all modeling elements of a PML. The most common graphical notation for process models in POPM is the "bubbles & arcs" notation whose meta meta model is depicted in Fig. 2 (standard UML notation). It is important to differentiate between modeling and visualization in this context. In Fig. 2 only the (content related) structure of a PML – and respectively the process models derived from it – is defined. Visualization is defined in an independent – but certainly related and integrated – model (cf. [JG07]).

A process model in POPM can be regarded as graph whereby processes are the nodes of the graph. These nodes are represented by *Node* in the APM^{2}M (Fig. 2). *NodeKind* describes the characteristics (features) of a node where each characteristic corresponds to one attribute of *NodeKind*. The Powertype pattern between *Node* and *NodeKind* is then established with the “partitions” relationship; *Node* represents the partitioned type and *NodeKind* is the powertype of the Powertype pattern. Features can be defined (*NodeKind*) that individually determine the behavior of *Node*. The following featurs are available: *hasIncomingPorts, hasOutgoingPorts, supportsData, supportsSubclassing, supportsAggregation* – their meaning and purpose can easily be derived from their names. In summary, the features presented above determine whether elements of *Node* can establish relationships of a certain kind (e.g. *aggregatedNodes*, *superNode*, *inputPorts*) to other types of the APM^{2}M.

2.5 Abstract Process Meta Model (APMM)

Fig. 3 shows the APMM of POPM; in this model the fundamental components of a POPM-related process model are defined: process, connector, data container, control and data flow, organization etc.

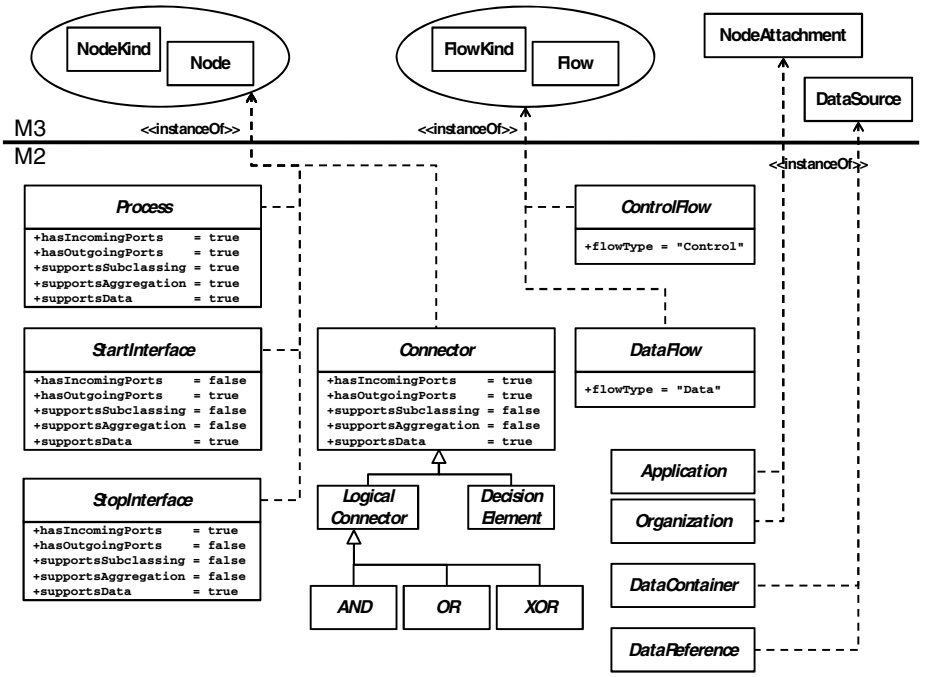


Fig. 3. The core of the Abstract Process Meta Model of POPM

In the APMM a process is an element in a graph that can be interconnected with other nodes (*hasIncomingPorts* = *true*, *hasOutgoingPorts* = *true*), can receive and produce data (*supportsData* = *true*), can be defined in terms of an already existing process (*supportsSubclassing* = *true*) and can be used as a container for other elements (*supportsAggregation* = *true*). A process – and in general every element of M2 – is an instance of a corresponding type – sometimes this is a Powertype – on M3. For instance, *Process* is an instance of the powertype *NodeKind* and inherits all activated features from the partitioned type *Node*. The types *StartInterface* and *StopInterface*) are also instances of the powertype *NodeKind* but do not support the creation of hierarchies since the corresponding feature attribute is not set (*supportsAggregation* = *false*); additionally, a *StartInterface* does not support incoming connections (*hasIncomingPorts* = *false*); analogously a *StopInterface* does not allow outgoing connections (*hasOutgoingPorts* = *false*).

2.6 Domain Specific Meta Models (DSMMs)

According to Fig. 1, DSMMs are specializations of the APMM. As with object oriented programming languages, abstract types cannot be instantiated. Thus, a DSMM must first provide specializations for each element of the APMM (abstract model) which can be instantiated. Then, a DSMM can be enriched by additional modeling constructs which determine its specific characteristics. Fig. 4 shows an example DSMM from the medical realm. This figure also depicts how domain specific modeling elements can furthermore be modified in order to capture specific characteristics. The attribute *stepType* for the modeling element *Medical Process* is introduced to determine whether a given step is an administrational task or a medical task; according to [Fa07] this is a fundamental distinction. Also the tags requested in [LS07] can be implemented in this way.

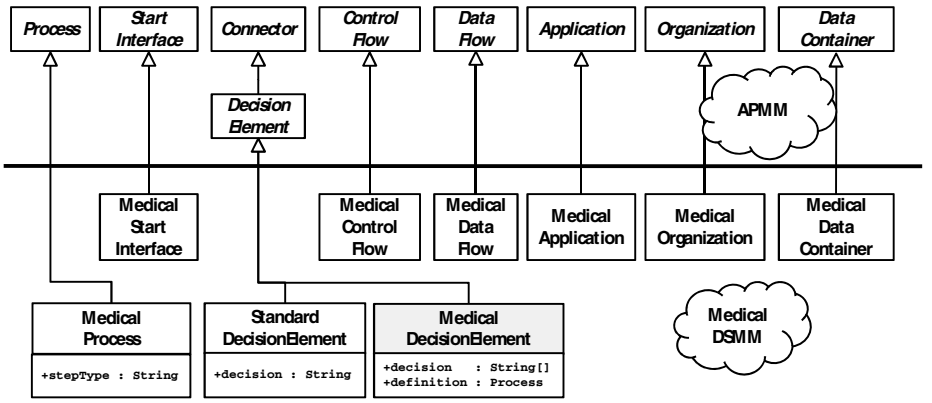


Fig. 4. A DSMM from the medical realm

It is possible to introduce completely new modeling constructs on this level as well. In Fig. 4, the so-called *MedicalDecisionElement* is presented. This modeling construct represents a complex series of single decisions.

2.7 Modeling Processes on M1

At level M1 "normal" process modeling takes place. We assume that a DSMM is defined on M2. Then real processes can be modeled on M1 and which are all derived from *MedicalProcess*. Accordingly, input and output data for each process can be defined; the same applies to organizations and operations. In Fig. 5c an example is shown. Regard, all modeling elements must be defined first, before they can be used within a process model. The process model consists of a start interface and two process steps namely *Anamnesis* and *Surgery*. The symbols (document, red cross) inside the two steps are tags that indicate whether a step is more of medical or administrational interest (this is valuable information when the process model has to be analyzed). The tags correspond to the attribute *stepType* defined in the Medical DSMM for *MedicalProcess*.

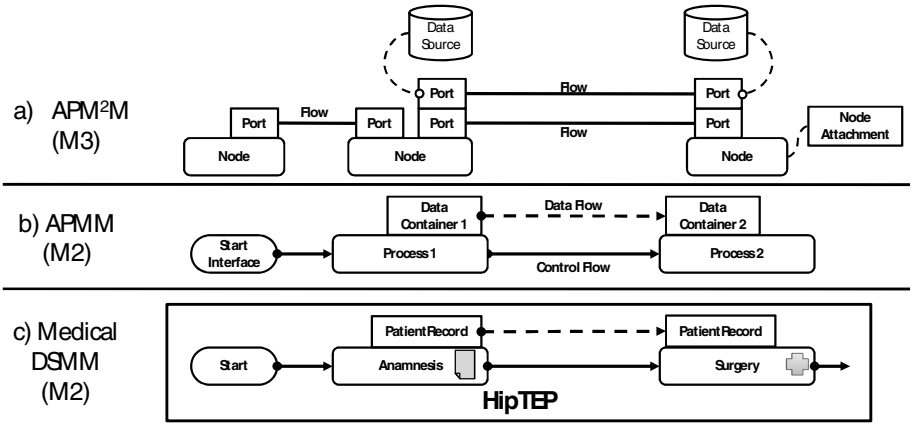


Fig. 5. Stepwise design of a process model

2.8 Stepwise design of a process model

In Fig. 5 the three decisive layers of a flexible modeling tool are clearly arranged. The figure illustrates nicely how concepts are evolving from very abstract (APM^2M), to more concrete (APMM), to domain specific (DSMM). Some of the metamorphoses of modeling elements are explained in detail. Through our integrated approach – models on M3, M2 and M1 can be manipulated by the same tool – consistency is best guaranteed on all layers of the meta modeling stack.

3 Evolution of process modeling languages and process models

We will now explain seven concrete use cases for evolution. These scenarios are sorted according to their relevance and frequency of occurrence in practice based on our experience gained in many industrial projects. The changes range over all levels of our logical meta stack; Change I affects M1, Changes II, III and IV concern M2 and Change V works on M3. A special role play Changes VI and VII; both affect layers not depicted in our meta hierarchy. We included them for completeness, however.

3.1 Change I - Adapting an existing process model

Adapting a process model to changing application requirements is a very frequent task in practice. Typical examples are the exchange of an application inside a process, the change of the execution order or the introduction / deletion of work steps. This kind of change is the normal use case for a modeling tool. It just has to be decided whether the generated new process models are replacements, versions or variants of the original process model. For process modelers this kind of change is uncritical.

3.2 Change II – Introducing new features for process modeling constructs (tagging)

Often it is necessary to distinguish processes from each other. Therefore, special tags are attached to processes and visualized in a suitable form [Fa07] [LS07]. Speaking in terms of our logical meta model stack this means that an attribute is added to the corresponding modeling element in the DSMM for storing the tag. In Section 3 we have shown such an extension: there the attribute *stepType* was added to the *MedicalProcess* type which was not part of the standard modeling. Depending on the actual value of this attribute a visualization algorithm can then for example display icons. The enactment of such a change can easily be performed by a domain expert.

3.3 Change III – Introducing a new process modeling construct

Modeling constructs have to be changed due an evolution of the application domain. For example, more powerful and semantically richer modeling constructs have to be created. Process modeling constructs are located at M2, usually in a specific DSMM. A new construct can either be defined "from scratch" or by redefining an already existing construct of the DSMM or APMM.

In case the new construct is defined from scratch, the creation task is facilitated enormously by the extended powertype concept. Merely a new construct must be defined on M2. If it is a kind of a process node it can inherit functionality provided by *Node; to selectively inherit this functionality* proper values for the feature attributes of the powertype *NodeKind* must be set. Nevertheless, this modification of a modeling language can be performed by any domain expert.

A new modeling construct can also be based on existing constructs like the medical decider from the medical application domain (Section 3). It is based on the standard decision construct of the APMM and summarizes a variety of single decisions into one compound construct (Fig. 6).

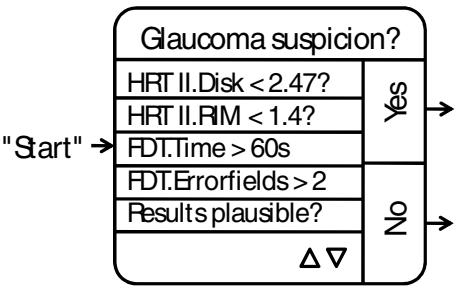


Fig. 6. The Medical Decision Element

In contrast to the standard decision construct the Medical Decision Element comprises many single decisions. Further, the output of the medical decision element is either "Yes" or "No" instead of an arbitrary result of the standard decision element. The introduction of this compact construct was one of the major factors why process modeling was accepted as adequate means to illustrate the medical applications in the Ophthalmological Clinics of the University of Erlangen [Ja05] where complex medical decisions had to be modeled often. This project convincingly demonstrated that a domain specific modeling language is not just "nice-to-have" but is crucial for the acceptance of process management in general. This kind of change corresponds to the introduction of macros in modeling languages.

3.4 Change IV – Adding a new perspective

Perspectives are in general defined at M2 – in the APMM for all domains as well as in DSPMMs for a single one. Similar to modeling constructs, new perspectives can be introduced in both. For example, it might become necessary to argue about process execution times in detail. So, various kinds of time should be introduced: preparation time, actual execution time, queue time and post processing time. A new perspective will be defined as an instance of the type *NodeAttachment* which is part of the APM^2M at M3. A perspective comprises multiple new modeling constructs. The constructs of this perspective can be associated with already existing modeling constructs like a process step or a flow construct.

3.5 Change V – Enhancing / changing the modeling method

So far all changes of PMLs were applied to DSLs individually. In our approach it is also possible to change the modeling method as such. This change happens on M3 and affects all PMLs defined below. For instance, from now on we will prohibit control flows between nodes. Referring to the APM^2M in Fig. 5 this means to remove ports which are not connected with data sources. Consequently all flow derived from this constellation must be removed from all PMLs on M2 and also from all defined process models on M1.

Another use case for changing the modeling method is if the used paradigm of directed cyclic graphs should be exchanged to something else, for instance petri nets. Also petri nets are graphs but they do not have the distinction of nodes, node attachments and flows. Instead they use states and transitions. Thus the presented model on M3 will not fit anymore and must be exchanged against a meta model for petri nets.

3.6 Change VI – Applying new semantics for a modeling construct

Without going into detail this change should be introduced. The idea is to change execution semantics for a process construct. For example, instead of interpreting a control flow arrow between two process steps as a mandatory order it should from now on be interpreted as recommended order. Details of the implementation of such kind of evolution can be found in [JI08] and [Ja06].

3.7 Change VII – Adapting the graphical representation of a construct

We were already mentioning that all models so far are just presenting conceptual issues. Visualization of process modeling constructs was excluded until now. In [JG07] we show how different kinds of visualizations can be associated with process modeling constructs according to evolving application requirements.

4 Related Work

We now give an overview on existing technologies and systems (beside those already introduced in Section 2) that aim at increasing the adaptability of information systems. We will show that these are – per se – not appropriate for domain experts because they require extensive programming skills or are not flexible enough.

Generative Programming [Cz00] and Software Factories [Gr99] are techniques for the reuse of code as known from object-oriented or component-oriented programming. Generative Programming aims at the generation of code out of a generic set of templates. These templates along with a specification of the outcome are handed to a code generator that automatically produces code. Because of the programming skills required to produce valid and correct results, Generative Programming is unusable for end-users or domain experts. Software Factories in contrast aim at reducing the costs (time, resources etc.) during the development of an application. This it is again an approach which is suitable for software developers but not for end-users or domain experts. Even more harmful is that both approaches are meant to be applied during the development phase of an application but not during runtime such that it is possible to introduce changes but a re-compilation and a re-deployment is needed.

Beside these programming techniques also complete systems exist that empower the user to build models such as the Microsoft Domain Specific Language Tools for Visual Studio [Mi07], the Eclipse Modeling Framework (EMF) [Ec07] (with related technologies for the generation of graphical editors) or MetaEdit+ [Ke96]. But nearly all of them use only two layers (definition of user model and instances of these) in which the type level defines the storage format for the user models. Beside this the modeling freedom is restricted as the underlying meta models often is fixed. For example EMF uses a subset of MOF and thus also inherits many restrictions from it – one is that it is not possible to use powertypes. Also many solutions are not able to use a new modeling language without generating a new modeling environment explicitly. Furthermore concepts for supporting more than one view are less sophisticated – if existing at all. But there are also some commonalities with these approaches from a technical point of view. We can reuse frameworks for code generation and graphical modeling that have already proven their strengths for meta modeling applications.

5 Conclusion

In this paper we were introducing our approach of a powerful process modeling infrastructure that supports evolution of PMLs and process models. We showed that we can leverage on some interesting concepts which unfold their real power after they were combined to form this unified and comprehensive approach. We have then shown how different evolution scenarios can be performed with the help of these concepts. Here the important key-point is that all those change requests that are most common can be performed without writing code. Thus domain experts are empowered to develop their modeling language(s) and applications by themselves and thus keep adapting the whole system perpetually to changing requirements. So as the overall system is able to handle changes by itself and it can be permanently improved by domain experts, we believe that this method supports sustainability greatly.

6 References

[Aa00] v.d. Aalst, W.; Jablonski, S.: Dealing with workflow change: identification of issues and solutions. International Journal of Computer Systems Science & Engineering (CSSE), Vol. 15 (2000), No. 5, 267 - 276

[AK01] Atkinson, C.; Kühne, T.: The Essence of Multilevel Metamodeling, 4th Int'l Conference on the Unified Modeling Language, Toronto, Canada, 1-5, 10.2001

[AK05] Atkinson, C.; Kühne, T.: Concepts for Comparing Modeling Tool Architectures, ACM/IEEE 8th Int'l Conference on Model Driven Engineering Languages and Systems, MoDELS / UML 2005, October 2-7, Montego Bay, Jamaica, 2005

[BS08] Berghoff, W.M.; Schlichter, J.H.: Computer-Supported Cooperative Work: Introduction to Distributed Applications. Springer-Verlag, 2000

[Cl08] Clark, T.; Sammut, P.; Willians, J.: Applied Metamodelling – A Foundation For Language Driven Development, 2nd Edition, CETEVA 2008, (visited: 2008-03-12) http://www.ceteva.com/book.html

[Cz00] Czarnecki, K.; Eisenecker, U.: Generative Programming: method, tools and applications. Addison-Wesley, 2000

[Ec07] The Eclipse Foundation: Eclipse Modeling Framework, (visited: 2007-03-29) http://www.eclipse.org/modeling/emf/?project=emf

[El95] Ellis, C., K. Keddara and G. Rozenberg (1995): Dynamic Change within Workflow Systems. In N. Comstock et al. (eds.): Proceedings of Conference on Organizational Computing Systems (COOCS'95). New York: ACM, pp. 10–21.

[Fa07] Faerber, M.; Jablonski, S.; Schneider, T.: A Comprehensive Modeling Language for Clinical Processes. 2nd European Conference on eHealth (ECEH'07), Oldenburg, Germany, 10.2007

[Gr99] Greenfield, J.; Short, K.: Software Factories: assembling applications with patterns, models, frameworks and tools. Wiley Publishing, 2004

[He99] Heinl, P.; Horn, S.; Jablonski, S.; Neeb, J.; Stein, K.; Teschke, M.: A comprehensive approach to flexibility in workflow management systems. SIGSOFT Softw. Eng. Notes, 24(2):79-88, 1999

[Ja94] Jablonski, S.: MOBILE: A Modular Workflow Model and Architecture. Proc. International Working Conference on Dynamic Modeling and Information Systems, Noordwijkerhout, NL, 1994

[JB96] Jablonski, S.; Bussler, C.: Workflow Management – Modeling Concepts, Architecture and Implementation. London: Int. Thomson Computer Press, 1996

[Ja05] Jablonski, S.; Lay, R.; Meiler, C.; Müller, S.; Hümmer, W.: Data Logistics as a Means of Integration in Healthcare Applications. Proc. 2005 ACM Symposium on Applied Computing (SAC) - Special Track on Computer Applications in Health Care, Santa Fe, New Mexico, 03.2005

[JG07] Jablonski, S.; Götz, M.: Perspective Oriented Business Process Visualization. 3rd International Workshop on Business Process Design (BPD) 5th International Conference on Business Process Management (BPM 2007). Brisbane, 9.2007

[JI08] Jablonski, S.; Igler, M.: Flexible Process Modeling and Execution based on Declarative Programming. Technical Report, University of Bayreuth, 2008

[Ja06] Jablonski, S.; Müller, S.; Faerber, M; Götz, M.; Volz, B.; Dornstauder, S.: Integrated Process Execution: A Generic Execution Infrastructure for Process Models. BPM Demo Session, 4th Int'l Confer¬¬¬ence on Business Process Management (BPM 2006). Austria, Vienna, 9.2006.

[Ke96] Kelly, S.; Lyytinen, K.; Rossi, M.: MetaEdit+: A fully configurable multi-user and multi-tool CASE and CAME environment. Proceedings of the 8th International Conference CAISE'96, Springer-Verlag, 1996, pp. 1-21

[LS07] Lu, R.; Sadiq, S.: On the Discovery of Preferred Work Practice Through Business Process Variants, 26th Int'l Conference on Conceptual Modeling (ER 2007), Auckland, New Zealand, 11.2007

[Me04] Melnik, S.: Generic Model Management: concepts and algorithms. Springer, 2004

[Mi07] Microsoft: Domain-Specific Language Tools, (visited: 2007-03-29) http://msdn2.microsoft.com/en-us/vstudio/aa718368.aspx

[OMG06a] Object Management Group: Meta Object Facility Core Specification version 2.0, 2006-01-01

[OMG06b] Object Management Group: Object Constraint Language Specification Version 2.0, 06-05-01

[OMG06c] Object Management Group: Business Process Modeling Notation Specification, 2006-02-01

[Od98] Odell, J.: Advanced Object-Oriented Analysis and Design using UML. Cambridge University Press, 1998

[Pe05] Petrov, I.: Meta-data, Meta-Modelling and Query Processing in Meta-data Repository Systems. PhD thesis, Univ. of Erlangen-Nürnberg, Germany, 12.2005

[Ri04] Rinderle, S.; Reichert, M.; Dadam, P.: Correctness Criteria for Dynamic Changes in Workflow Systems - A Survey. Data and Knowledge Engineering, Special Issue on Advances in Business Process Management 50(1):9-34 (2004)

[Se03] Seidewitz, E.: What models mean. IEEE Software 2003, 20(5): pp. 26-31

A Semantic Framework for Compliance Management in Business Process Management[1]

Marwane El Kharbili[1] and Elke Pulvermüller[2]

[1] IDS Scheer AG, ARIS Research
Altenkesseler Str. 17, 66115 Saarbrücken, Germany
marwane.elkharbili@ids-scheer.com
sebastian.stein@ids-scheer.com

[2] Institute of Computer Science, University of Osnabrück
Albrechtstr. 28, 49076 Osnabrück, Germany
elke.pulvermueller@informatik.uni-osnabrueck.de

Abstract: In process-centric enterprises, business processes (BPs) are at the center of value-creating activities. Governing enterprise BPs requires the ability to control and guide BP behavior. Ensuring compliance of processes to legal regulations and strategy directives becomes a critical requirement. Implementing business process compliance makes means for modeling and enforcing compliance measures necessary. In this work, we motivate the need for automation and semantic consistency in compliance management and defend the use of policies for this purpose. We then propose a policy-based framework for business process compliance management and further detail its architecture as part of the SUPER research project on semantic business process management (SBPM). Finally, we introduce the ontology stack we propose for compliance modeling and conclude by an investigation of the main challenges ahead in order to provide an implementation of the proposed framework. This work seeks to lay down the fundaments of a comprehensive architecture for semantic compliance modeling and enforcement in the context of BPM.

1 Introduction

Business Process Management (BPM) is the discipline of capturing, modeling implementing, and controlling all activities taking place in an environment defining the enterprise, and this, in an integrated manner [Sch00]. Organizations do not only own business processes, they are also subject to regulations. Not being compliant to regulations diminishes the added-value business processes represent for the organization, e.g. through non-optimal alignment with (i) quality standards, (ii) business partner service agreements or (iii) non-identified security flaws [Kh08a]. Non-compliance to regulations could also be the cause of judiciary pursuits, as in the case with laws such as the Sarbanes-Oxley-Act (SOX), which, among other aspects, seek to impeach financial manipulations in order to protect stakeholders in a company.

[1] **Acknowledgements:** We would like to thank the EU commission for supporting our research on semantic compliance management within the European research project SUPER IST-026850 (http://www.ip-super.org).

Consequently, non-compliance has both short-term (e.g. cost savings, reduced governance complexity) and long-term (e.g. judiciary pursuits, market confidence) consequences. Compliance management is the term covering *all activities and methods to ensure that a company follows all guidance and implements all measures required by an external or internal regulation* [Kh08a]. By extension, compliance management also *refers to standards, frameworks, and software used to ensure the company's observance of legal texts*. In the context of BPM, compliance management *applies on business processes and the related resources like data and systems*. Business processes are typically inter-departmental by nature. Similarly, inside organizations, compliance management spans the spectrum of horizontal activities (e.g. IT security or quality standard compliance) which are inter-departmental and inter-organizational by nature. Non-compliance at the level of business processes is critical because business processes control all value adding activities of a company. A comprehensive compliance management framework for BP-centered enterprises should take this aspect into account and permit *hiding the complexity of BPs* from compliance experts in order to *concentrate efforts on what should be checked instead of how it should be checked*.

A framework allowing organizations to integrate regulatory compliance tasks with business process management presents many advantages as we will show[2]. Requirements on such a framework have already been elicited in [Kh08a] and in more systematic and analytical fashion in [Ly08] as well as a high-level architecture proposed in [Kh08a]. Our approach to designing such a framework is based on policies. We argue that policies supported with semantic descriptions of business processes present many advantages for our purpose with regard to modeling, knowledge management and enforcement and monitoring.

More than the need for automation and complete coverage of enterprise models in compliance management, formal modeling of compliance is a requirement when considering the need for verification and validation of modeled compliance measures. Also automated compliance management implies compliance checking functionalities. In the following sections of this paper, we will show how policies and rules as enterprise model artifacts can be used for fulfilling these requirements and provide an answer to the aspects discussed in this section. In our work, we assume that an enterprise model is process-centered (such as is made in ARIS [Sch00]), and as such, we seek to model compliance on semantically modeled BPs which are used as the elements connecting enterprise model artifacts. This is for instance the approach taken by the SUPER project. Our work will also lead us to introduce an extension of the SUPER BPM ontology stack with an ontology for modeling policies and rules, thus providing an integrated way of managing compliance in BPM.

In the following, Section 2 introduces background knowledge about the context of our work, namely the SUPER semantic BPM platform, and then a presentation of our approach and an architecture for compliance management is given. In Section 2, we also proceed to a presentation of the policy ontology for modeling compliance measures.

[2] Interest in the issues tackled by this work is very high in the scientific community. The Compas (EU ICT 7FP: www.compas-ict.eu) and the Master (EU ICT 7FP: http://www.master-fp7.eu) projects illustrate this.

Section 3 contains a review of related work and finally Section 4 contains concluding remarks and a statement of future work.

2 An Architecture for Semantic Compliance Management

We have shown the need for a formal representation of compliance measures, as a mean for modeling regulations. In this work, we push forward the notion of policy and business rule as the tools necessary for linking enterprise models to regulations. This section will introduce the context of our work on compliance management, our approach to the problem of enterprise regulatory compliance, and will propose an architecture for this which is integrated in an SBPM platform.

2.1 The SUPER Platform for SBPM

The idea of the SUPER semantic BPM (SBPM) platform is to support modeling semantic business processes by delivering a stack of ontologies [Pe08a] for the domain of BPM as well as an architecture for supporting BPM with semantic web services (SWS), as described in [He05] and [He07]. BPM seeks to bring more automation in enabling business modelers to define inter-organizational applications, while offering a fully-fledged lifecycle for managing the resulting BPs. SUPER builds on previous results from semantic web services research in order to offer automation in composition, execution and analysis of BPs. It also seeks to integrate between the business layers of BPM where mostly conceptual BP models are managed and the execution layers where semantic web services as an EAI paradigm are used to implement BPs.

As is stated in [Pe08a], SUPER lifecycle for BPM is composed of four phases: SBP modeling (design of BP models), SBP configuration (tackles the deployment of SBP models on IT infrastructure, e.g. BP Management System, Web Services, ERP, etc.), SBP Execution for running SBP models and finally SBP analysis for assessing the quality and conformance of executed SBPs to initially drawn expectation in the modeling phase. SUPER defines a stack of ontology specifications which seek to cover various aspects of enterprise modeling while enriching it with semantics. Languages such as sEPC, sBPMN, and SBPEL have been defined as ontologized versions of respectively EPC, BPMN and BPEL languages. Also, ontologies for process monitoring, mining, resource, role, strategy and event modeling have been defined. These ontologies are used in order to enrich the description of SBP models with information relevant for both business modelers, and for semantic applications delivered by SUPER such as BP composition using SWS.

Because of this, SUPER allows defining a semantic enterprise model. Formal models for representing enterprise knowledge as enterprise models already exist (e.g. the non-semantic ones such as TOGAF [TOGAF08], ARIS [Sch00], Zachman [Za92], as well as semantic works such as the Enterprise Ontology by Dietz [Di06], by Uschold et al. [Us98] and the TOVE project [Fo92]) but none of these frameworks explicitly tackles the challenge of enterprise-wide corporate compliance management. Although works

such as TOVE do consider aspects such as quality management in enterprise modeling, no generic ways of modeling and enforcing regulatory compliance is given yet. Compliance management is still a discipline relying heavily on manual, error-prone, sample-based procedures undertaken by auditors, i.e. the level of automation is still very low.

2.2 Approach

We now proceed to the definition of an initial integrated approach to model, check and enforce compliance on enterprise models, as summarized on Fig. 1. In our approach, policy documents in natural language are the initial input. These should be first processed to fit into a pre-defined structured natural language dialect for expressing policies. In [ISO08], business rules are expressed in standardized languages (e.g. standardized English or standardized French – The ISO 639 standard) by relying on a pre-defined business vocabulary. An underlying formal model allows a logic-based representation of these natural language business rules. Another prominent approach to structured natural language business rules is the Attempto Controlled English (ACE) [Fu05]. This has the advantage of making rules understandable to people who are responsible for managing them: business analysts. It also makes it possible to compute machine-processable representations of regulations (provided that the latter be written/transformed in some structured language) thus avoiding multiple and possibly diverging interpretations of regulatory texts. Consequently, changes to regulations can be automatically processed by regenerating adapted representations than can be enforced on BPs.

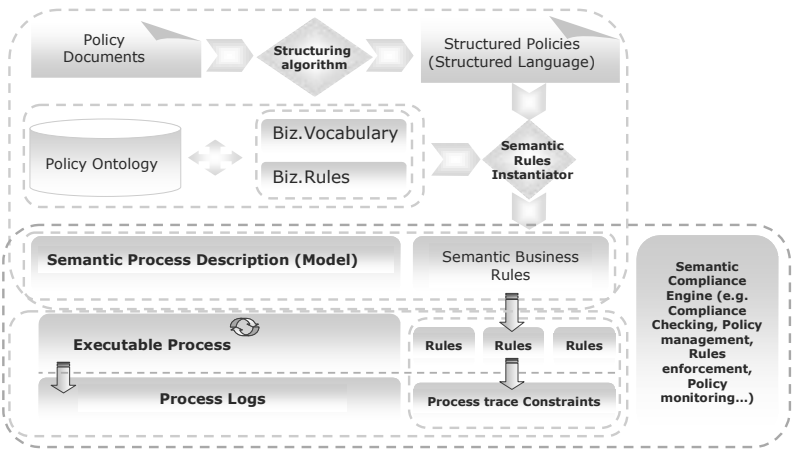


Fig. 1. An integrated approach for modeling and enforcing BP compliance

Once regulations can be represented as natural language policies in a structured language, they can be combined by a component (called the semantic rules Instantiator, see Fig. 1) with the following assets in order to become fully-exploitable:

(i) ***Policy Ontology:*** refers to both the generic policy ontology and the domain policy ontologies that might be necessary for regulation modeling. A generic policy ontology contains the high-level concepts that could be used for modeling a broad scope of policies regardless of the context to which these policies belong. On the other hand, a domain policy ontology contains specific constructs for expressing policies in a certain domain. For example, in the IT security domain, one particular class of policies which share a set of attributes and properties are role-based access control (RBAC) policies.

(ii) ***Business Rules Ontology:*** refers to the ontology providing constructs needed to express policies as business rules. The business rules ontology will typically allow serializing policies in some kind of mathematical logic (e.g. predicate logic, first order logic, description logic, temporal logic, etc.).

(iii) ***Business Vocabulary (Domain Ontology):*** is the agreed on conceptual model of a domain governed by policies. A business vocabulary could be represented by an ontology. Modeling axioms for the ontology would allow describing constraints on the business vocabulary's concepts. Policies and business rules are modeled on the elements of the business vocabulary. Attempts to provide generic quality management ontologies such as the ontology developed by the TOVE Project [Ki99] are a good example of this.

The result of this phase is a new ontology containing a semantic model of the policies and their further logical specification as business rules defined by the regulation (relevant for checking/enforcement). These semantic business policies can be used to govern regulatory compliance and the semantic business rules can be enforced on semantic enterprise models (i.e. BPs). Business rules are seen here as context-dependent representations of the decision logic required for policy enforcement. In order to be enforced on other representations of elements of this enterprise model, these semantic business rules need to be transformed into adequate representations. For semantic business process models (e.g. instances of a language such as BPMO[3] for semantic BP modeling [Pe08a], at least one transformation is needed.

Semantic BP (SBP) models can be transformed into executable semantic BP descriptions which can run on a specifically developed semantic execution engine. Consequently, semantic business rules need to be transformed into a language that can express the same logic for the executable semantic BP models. We call such rules "Semantic Operational Rules". The principle behind such a transformation is explained in [Kh08b]. An ontology needs to be defined for expressing such operational rules and our first two target languages for which to design such an ontology are the SBPEL [Ni07a] and the BPEL4SWS [Ni07b] languages. The latter are respectively a semantic representation of the BPEL standard and an extension of the BPEL [OASIS08] standard for invoking semantic web services.

In order to check regulatory compliance on process logs, a special transformation of the business rules into constraints verifiable directly on these logs is necessary. This is needed in case of the so-called Backward Compliance Checking, where compliant

[3] Business process Modeling Ontology

behavior of the business processes is checked after the concerned BP instances have finished running. Such checking techniques could be used when wanting to check behavior only for a sub-class of all process instances defined by the BP model.

Our future approach in realizing this relies on defining semantic LTL (Linear temporal Logic) formulae of semantic business rules and using the ProM [Do05] mining tool for checking if these formulae hold on process logs. As to now, no work provides performance comparison to normal model checking techniques. But it would still be reasonable to think that such a checking technique can be at least as cheap as the so-called Forward Compliance Checking techniques [Kh08a]. All checking operations such as activating and enforcing policies or executing rules are managed by a separate component called the Semantic Compliance Checking Engine (SCCE).

2.3 Architecture

The following figure shows a high-level description of the architecture of our compliance management framework, as it is explained in [Kh08a].

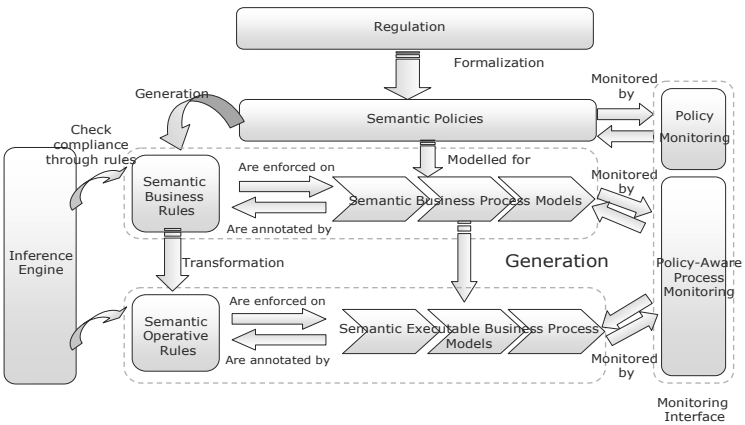


Fig. 2. A high-level architecture for a compliance checking framework

Figure 2 is aligned with Figure 1, and also separates the problem of compliance management into three distinct layers. Figure 2 also shows that our research efforts will concentrate on 5 distinct axes: (i) architecture, (ii) compliance management process, (iii) ontologies, (iv) algorithms, and (v) lifecycle. It divides compliance-related tasks into three layers: (i) regulation formalization, (ii) policy modeling and (iii) business/execution rule modeling. Compliance checking is realized by an inference engine for the policy and rules ontology. Additionally, monitoring components take policy models and policy enforcement monitoring in charge.

Semantic policies have to be modeled into the business processes (middle layer of Fig. 2). In the case of semantic business process management, this means extending the ontology for modeling business processes with an ontology for modeling policies. It also

requires defining anchor points between the various enterprise modeling ontologies in order to be able to annotate enterprise model elements with policy assertions. In the example of the SUPER project, the concerned ontologies would be the BP modeling ontologies and organizational ontologies as introduced by Pedrinaci et al. in [Pe08a].

Rules are an intuitive way of implementing policies. Policies' logic can be represented as sets of business rules. A comprehensive review of the state of the art is given by Bonatti et al. in [Bo04]. Decisions are made on the level of policies, as to which policies should be activated and consequently decide on the actions to be taken. Business rules partly encompass the actions to be taken, when the latter are of complex nature. Otherwise, a certain entity can be requested to execute an action required by the policy, without requiring a business rules to fire. Policies have to be monitored and controlled (as in [Ka06]), in order to discover discrepancies in policy and/or rule modeling. The logic implemented by the BRs can be represented using different formalisms depending in the needed use. For instance, a BR could decide to propose a 10% discount to a customer having purchased for more than 1000 euros merchandise in the last 6 months, and another BR could send an alert to an intrusion detection system if it discovers that two different credentials used by the same individual interacting with a BP instance are in contradiction with one another. The logic formalisms needed for these two examples are clearly distinct.

A semantic compliance checking engine (Left side of Fig. 2) has to be implemented by building on an inference engine. The inference engine will depend on the ontology language used to model the policies, business rules and business processes. In our case the language used is WSML[4] (respectively WSML-Flight[5] language variant for semantic business rules) since it is the ontology language used in the WSMO[6] framework. The WSMF[7] framework is the semantic web services framework used in the SUPER research project and WSMO is the ontology defined for it. The compliance checking engine stores procedures and algorithms that check and/or enforce policies on business process ontology instances (models). These algorithms implement non-routine elementary checks such as checking ontology axioms, implement complex compliance checking processes involving several policies, taking decisions based on policy violation, generating compliance reports, etc, which are tasks that could be realized by calling the inference engine for the WSML ontology language. Monitoring components (Right-side of Fig. 2) are needed to control the consistency of policies, but also to monitor the checking and enforcement operations on business processes. Besides policy monitoring, the monitoring component also takes in charge monitoring the activation of policies modeled into the enterprise model and informing about the causes of that activation.

---

[4] Web Service Modeling Language: http://www.wsmo.org/TR/d16/d16.1/v1.0/.

[5] WSML-Flight is the WSML-Core language enriched with logical programming with non-monotonic negation. Another WSML subset is the WSML-Rule language. WSML-Flight is based on a logic-programming variant of F-Logic. WSML-Rule is the WSML-Flight language enriched with features such as function symbols.

[6] Web Service Modeling Ontology: http://www.wsmo.org/TR/d2/v1.3/.

[7] http://www.swsi.org/resources/wsmf-paper.pdf and http://www1 c703.uibk.ac.at/~c70385/wese/index.html.

2.4 Integration in the SUPER Architecture for SBPM

We will now refine the architectural frame given in Fig. 2 with regard to the SUPER project architecture in order to illustrate a concrete implementation. On Fig. 3, a view on this architecture is given. This view includes the components related to semantic compliance checking in SUPER.

The SUPER architecture is designed to support a lifecycle for semantic BPM (SBPM) composed of SBP modeling, configuration, execution, and analysis [We07]. The architecture is made up of tooling components (SBP modeling, SBP monitoring and SBP mining tools etc.), execution components (e.g. semantic BPEL execution engine), which both sit on top of a semantic service bus (SSB). Platform services (semantic service and semantic goal mediation, semantic service discovery, semantic service composition etc.) and repositories (SBP models, SWS, SBP Execution History) are registered by the SSB and delivered upon request as services to the tooling and execution components.

SBP models can be represented using one of several ontologies. SBPs modeled in sEPC [Fi08] or sBPMN [Be08]. The sBPEL ontology also belongs to the SBP ontologies but lies on the execution level. The organizational ontologies are a stack of ontologies that allow modeling different enterprise modeling aspects such as roles, resources, functions, strategies, goals and governance guidelines. COBRA is the COre Business pRocess Analysis ontology and regroups the necessary knowledge for SBP mining and analysis [Pe08b].

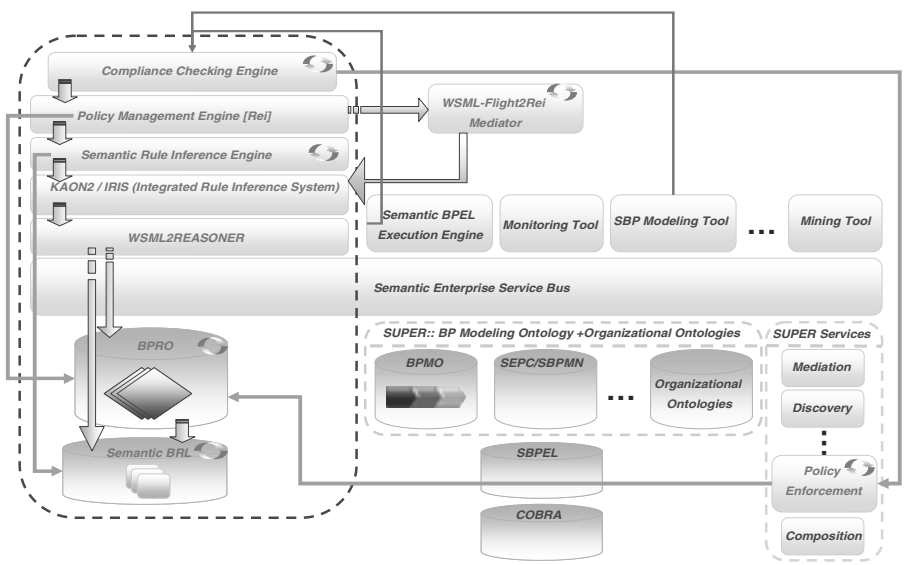


Fig. 3. Semantic Compliance Checking components – SUPER Architecture

Next to the previous ontologies is the Business Policy and Rule Ontology (BPRO). The BPRO is also part of the organizational ontologies but is represented separately Fig. 3 for ease of explanation. The BPRO is used to model policies and rules that can be

integrated into instances of the ontologies of design-time BP models layer (i.e. BPMO, sEPC, sBPMN and organizational ontologies). The distinction between design-time and run-time aspects of compliance checking have also been signaled by existing works [Ly08]. The authors of [Ly08] insist on the fact that a detailed analysis of the relationship between different types of constraints and corresponding use scenarios is necessary for optimizing interplay between design-time (e.g. allowed resource types to be allocated to an activity) and run-time (e.g. execution ordering) policies.

To enforce policy decisions on the executable SBP models layer, another adapted representation is required. The SBRL (Semantic Business Rule Language) ontology is responsible for this task. Rules expressed in the SBRL ontology do basically implement the same logic as in rules from the BPRO, with the difference being that SBRL instances can be run together with sBPEL instances for example. The use made of the SBRL ontology can be understood as the semantic equivalent to works seeking to implement constraint modeling and compliance checking on BPEL processes. The work by Liu et al. [Li07] on defining a graphical language for expressing compliance rules for BPEL in temporal logic and the work by Rossak et al. [Ro06] of extending the BPEL specification with elements for expressing quality constraints illustrate our idea.

The compliance checking engine (CCE) is a tooling component of the SUPER architecture and takes in charge the operations of policy interpretation and rule execution, necessary functionalities for realizing compliance checking. The CCE can be called either by the SUPER modeling tool or by the sBPEL execution engine. The CCE makes use of a policy management framework capable of inferring on the BPRO instances. As the BPRO contains both policy and rule descriptions, the CCE needs both an inference component to infer on policy descriptions and a semantic rule inference engine to infer on rule descriptions. These two inference components make use of inference engines necessary for the interpretation of WSML ontologies, as the BPRO itself is written in WSML-Flight and the SBP ontologies in WSML. Several engines are available for this and one possible combination would be to use the IRIS[8] (Integrated Rule Inference System) together with the WSML2Reasoner[9] interface.

The separation of policy management aspects from policy enforcement aspects in the BPRO is achieved by separating modeling of policies and rules. Requirements on policy management include (i) deciding when a policy is active, (ii) which decision has to be made by the policy, (iii) how the decision taken in to be enforced and finally (iv) which relationships link a policy to other policies that might as well be active.

In the field of policy management, there are several works which have proven able to fulfill these requirements. Due to space restrictions, no detailed comparison of these works will be given here. The foundational work on the Rei engine is presented in Lalana Kagal's PhD Thesis [Ka04]. Rei[10] is a policy specification language and

[8] http://iris-reasoner.org/.
[9] http://tools.sti-innsbruck.at/wsml2reasoner/.
[10] http://www.cs.umbc.edu/~lkagal1/rei/. See also: http://ebiquity.umbc.edu/project/html/id/34/Rei-A-Policy-Specification-Language.

methodology for building policy-directed architectures. Rei combines the following features: (i) it can describe both positive and negative authorization and obligation policies, (ii) it includes a policy engine, analysis tools, and a methodology for deploying policy frameworks, (iii) it allows policies to be described in terms of attributes of users, actions, and other context elements as well as supports meta-policies for conflict resolution, (iv) it provides greater extensibility as policies can be described over domain knowledge at different levels of abstractions, (v) it includes meta-policies for automated conflict resolution, and (vi) it supports dynamic policy modification via speech acts. Rei is written in OWL-Lite[11] and allows specifying declarative policies over domain ontologies written in RDF[12] and all OWL[13] variants. The main principle in Rei is that policies restrict actions than can be taken by resources in an ubiquitous environment. Concretely, this means that Rei policies can express authorizations (what an entity can do), obligations (what an entity must do) and prohibitions (what an entity can't do). Rei policies can be defined over an individual or class of actors, an action and a target. This approach fits well with our definition of business policies for SBPM as well, as is shown later in this document.

In order to allow the Rei engine to infer on BPRO policies and rules written in WSMLFlight, a mediator is needed to translate these BPRO instances into the Rei language. Once this mediator in provided, Rei inference results can be pushed down through the IRIS engine and the WSML2Reasoner component to the BPRO instances and modifications be made directly into the ontology. An example of this would be setting the status of a certain policy to "violated".

Another example would be triggering an action that is fired by a rule implementing a certain policy, when triggering this action requires setting attribute values in the BPRO ontology instance. One of the additional components of the SUPER architecture is the policy enforcement component (PEC). This component is called by the CCE, in order to check a certain policy in a BPRO instance on an enterprise model/SBP model and is available as a semantic web service. It also calls the policy management engine and the semantic rule inference engine.

2.5 Business Policy and Rules Ontology

The idea for the BPRO is to be independent from the concrete execution of the rules and to bring as much information from the rule language to the policy layers, while keeping policy definition independent from any rule language. We will start by listing some examples of competency questions that the ontology needs to be able to answer and then proceed to a short description of the main concepts.

The next two figures detail the business policy and the business rule ontologies which we introduced earlier. As already said, these ontologies seek to provide a generic and

[11] http://www.w3.org/TR/2004/REC-owl-features-20040210/#s3.
[12] Resource Description Framework: http://www.w3.org/RDF/.
[13] http://www.w3.org/TR/owl-features/.

high level model to be reused by other ontologies for concrete regulatory compliance modeling. In order to later validate the design of the ontology we first need to specify the competency questions it needs to be able to answer. This is not a sufficient validation criterion though, and a well-founded evaluation of the soundness of the ontology for the use that is to be made of still must be undertaken.

1. In a given BP model state, which policies are active?

2. Which policies are called by policy P once policy P is activated?

3. What policies apply to BP activity A?

4. To which activities, roles and resources does policy P apply?

5. Policy P1, P2 and P3 are active and need to take a decision on Resource R: Which policy has the highest priority?

6. Policy P took its decision: which rules need to fire in order to implement this decision?

7. Which business goals does policy P fulfill?

8. What is the jurisdiction of policy P?

9. What is the scope of a policy P inside a Jurisdiction J?

10. Which regulations is policy P part of?

11. What is the type of policy P?

12. What is the modality of policy P?

The main idea in the business policy ontology is that policies take decisions on whether a state of the business (enterprise model/BP) is allowed or not. In Rei, this is achieved by defining modalities upon actors, actions and targets of these actions. In the business policy ontology, this is achieved by setting modalities for attributes and properties of ontological concepts. These concepts can be concrete actions or simply the result of these actions. The modalities are combined with condition evaluation that is left for a rule to do. The decision taken by a policy is implemented by executing a rule too, and/or by triggering an event. Events are used to propagate decisions inside the policy framework. For example: to express the policy that a resource of type X cannot be accessed by a business process activity which has a responsible person of Role Y, it is necessary for the policy to check the attribute *Responsible* of a BP activity which is connected to the resource X on which it applies after having checked that resource X has an attribute *type=="X"*. In this scenario, both a passive and active approach to compliance checking work.

In the case of passive checking, the policy is modeled in a decision point. In this example, it means that the activation and evaluation of the policy is directly triggered by

the element governed by this policy: resource X. Before the concerned BP activity even executes, it checks which policies are active, by evaluating all the conditions to activate policies attached to it. These activation conditions are part of the definition of the business policy. In the case of active checking, it means that the policy enforcement component actively supervises all states traversed by the BP model and matches these states to policies that need to be activated for the current state. Changes in states are triggered by actions, or, in the case of BP models, of transitions from one activity to the other. Reasoning on the current transition to be made and matching it with policies to activate and then evaluating these policies, leads to the policy taking a YES/NO decision about the current state violating the policy (non-compliant) or not.

The policy ontology is represented on Fig. 12 and we will now proceed to a very concise description of its concepts. Due to space restrictions, this description does neither include relationships constraints nor constraints (axioms). The central concept of the ontology is the Policy concept (Fig. 12). A policy belongs to a regulation together with other policies. A policy also fulfills a goal and belongs to a strategy. Strategies are assigned to a goal. A policy can be a meta-policy, which is a policy acting on other policies. With regard to another policy, a policy has a priority set.

A constraint is one kind of policy, next to decision, functional and core policies. A constraint policy decides on how to constrain a resource in showing some behavior. It delivers one or many of several discrete allowed business artifact behaviors/states and does not provide a binary yes/no answer as a decision policy does.

A core policy is a policy which takes no decision that has to be enforced on business artifacts; it can only be invoked by other policies and delivers intermediary decisions. A functional policy applies for business functions that are able to execute differently depending on the parameters given to them.

A functional policy decides on which concrete action these business functions can take by setting these parameters. A policy takes decisions which can result in rule actions. Rule actions can be implemented outside the BPRO either by another ontology or by being implemented in another system and invoked from the BPRO.

A policy had two kinds of modalities: deontic and alethic. Deontic modalities allow expressing behavioral constraints and are of three types: Prohibition (interdiction), Permission and Obligation. Dispensation is an additional special type which frees a subject from a constraint rather than imposing it. Deontic constraints nap to those modeled in Rei. Alethic modalities allow expressing structural constraints and are of four types: Necessity, Non-necessity, Possibility and Impossibility. The SBVR (OMG, 2008) standard does include both alethic and deontic modalities for expressing business rules.

A policy has a subject, which is the entity (ies) on which it can apply. This subject can be a process model for example or any business artifact part of the enterprise model (e.g. role, resource, business function etc.). A process is composed of process fragments and the latter are composed of process constructs such as activities. The concepts related to

BP modeling have to be mapped to the used BPM ontologies. A policy has a jurisdiction and a scope.

A jurisdiction is the domain in which a policy has the right to take decisions. Outside its jurisdiction, a policy cannot take any decisions, cannot be solicited, and cannot communicate with other policies about subjects not belonging to its jurisdiction. A Jurisdiction is a set of subjects. These sets of subjects can be defined in a declarative way, such as using assertions on properties of subjects: all roles of type==[engineer | manager] where role.budget >= 1000 units. We do not take into account jurisdiction management (which would require a dedicated algebra) in order to define these inter-policy relations unambiguously. The latter are those managed by the Rei framework and make use of speech acts (policy delegation, policy revocation, policy invocation, policy cancellation) and conflict resolution.

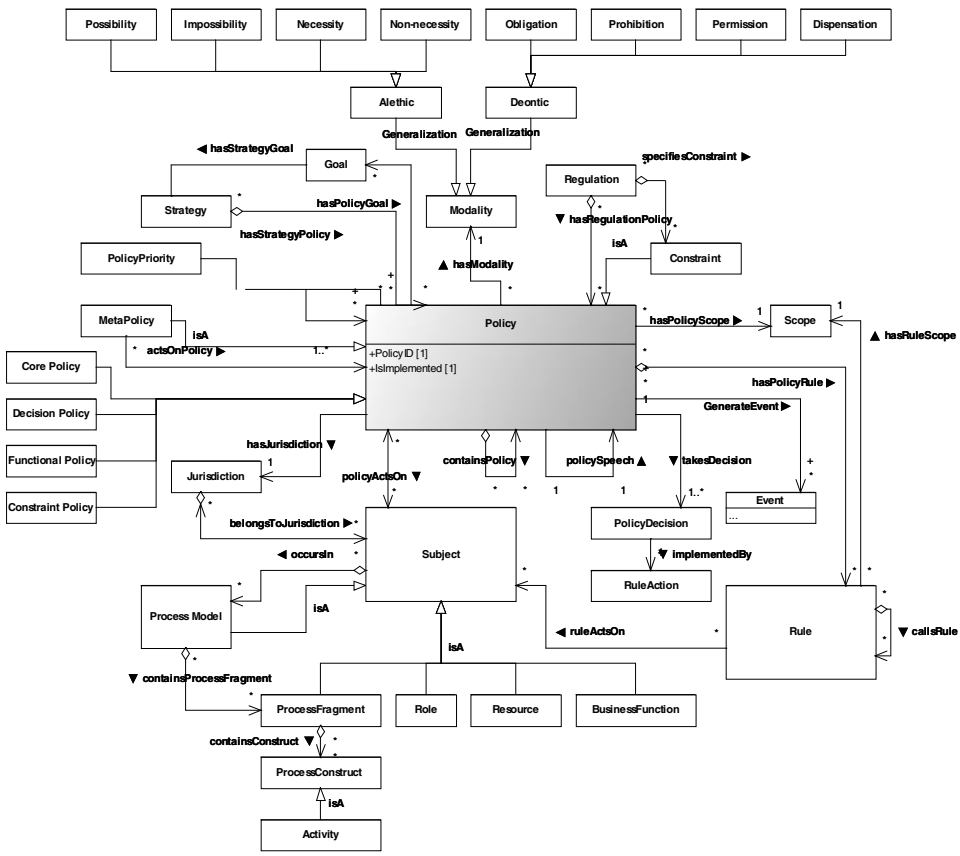


Fig. 4. The Business Policy Ontology

Scopes are different from a jurisdiction in that scopes are always strictly included in jurisdictions and define the set of subjects inside a given jurisdiction upon which a

policy can take a decision. Scopes introduce additional flexibility in managing policies, by allowing moving a policy's scope inside a given jurisdiction.

Fig. 4 displays all concepts of the business rule ontology, which we will shortly describe here. As the figure shows, rules allow modeling action taking, which is triggered by a policy decision. A rule belongs to one or several policies. That means rules can be composed in order to implement a certain policy, although the BPRO doesn't include concepts for expressing rule composition yet.

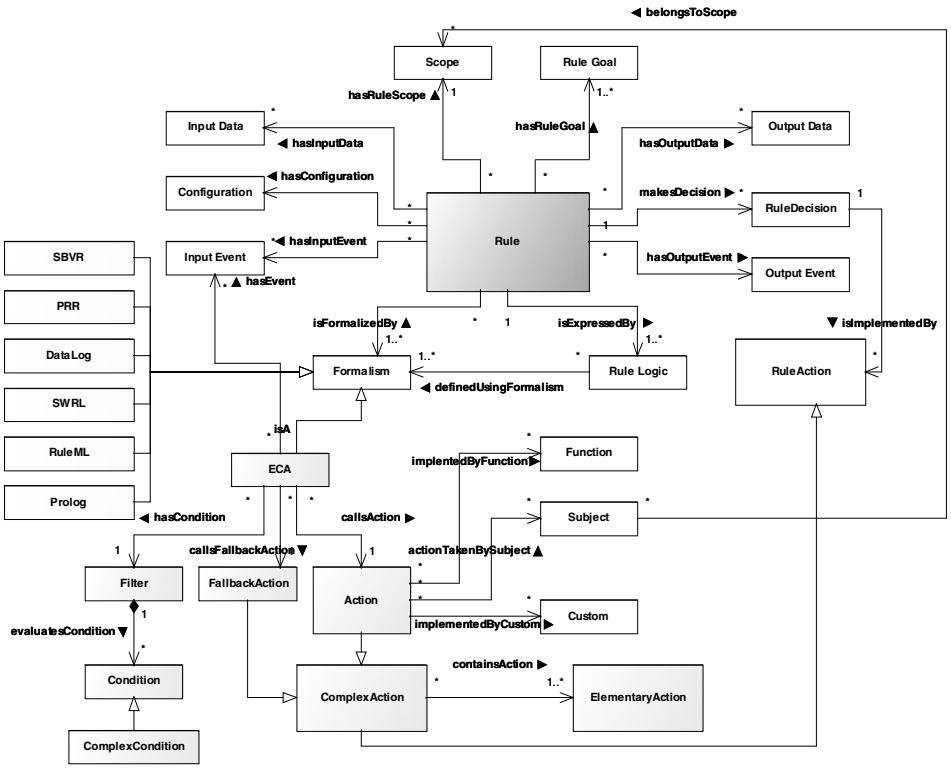


Fig. 5. The Business Rule Ontology

A rule is also attached to a business goal and has a scope. A rule has input and output data it processes and an input and output event. Input events can trigger the execution of a rule and output events are generated by a rule to trigger other rules or actions. A rule has configuration data which makes a rule able to execute different logic depending on its configuration.

A rule implements a policy decision and makes a rule decision. A rule decision allows modeling chains of rule executions which brings more expressiveness to rule modeling. A parallel to business processes would be that a chain of rule decisions is like a sequence pattern of business process activities, where the activities are all rule executions. A rule

contains rule logic which is expressed in a certain formalism. Rule actions support expressing complex actions which use logical operators such as (AND, OR, NOT, XOR) on rule calls.

Formalisms are many and in the figure above, the Event-Condition-Action (ECA) formalism has been used as an example. An ECA rule has a filter (condition), an action which can be a complex expression of actions, and a fallback action which is the action to be taken in case the condition evaluates to false. It is also triggered by one of the input events of the rule. The action taken by the ECA rule is done on a subject and can be implemented either by an available function (from the business functions ontology in SUPER) or be a custom action which means it is implemented as a concept in an external ontology or an external system.

2.6 A Simple Example

In this example, we will illustrate how an example instance of a policy together with a business rule instance would look like. The following policy is enacted on three instances of concepts declared in external ontologies: the CustomerService business role, the CustomerInformationAdministration business role and the ReviewCustomerFinancialDetails process task.

This policy basically expresses the following: "It MUST be guaranteed that only person with access to the CRM systems may perform the activity ReviewCustomer FinancialDetails". The policy is thus an obligation (see the ***hasModality*** attribute of the BizPolicy_12 business policy instance) The logic contained in this business policy is stored in the business rules instance "BizRule_12" and is coded in the WSML-Flight rule language. Another rule language could have been used for this.

```
instance Cust_Verif_obligation memberOf BPRO#Obligation

instance CustomerAssistant memberOf oso#OrganisationalPosition

instance CustomerServiceRole_59871239 memberOf
(bronto#CustomerServiceRole, bronto#OperationalROle)

instance CustomerInformationAdministeringRole_287334 memberOf
{bronto#OperationalRole, CustomerInformationAdministeringRole}

instance ReviewCustomerFinancialDetails_76765600 memberOf
ReviewCustomerFinancialDetails

instance BizRule_12_A memberOf BPRO#BusinessRule
       isFormalizedBy "WSML-Flight"
       isExpressedBy "?z memberOf
RolePerformsReviewCustomerFinancialDetails (?y memberOf
ReviewCustomerFinancialDetails, ?x) : ?x Customer memberOf
OrganisationalPosition and ?x[CRM_State hasValue TRUE]"

instance BizPolicy_12 memberOf UPO#BusinessPolicy
       hasID hasValue "zu873928_kuasi09"
       hasModality hasValue Cust_ Verif_obligation
       isImplemented hasValue TRUE
```

```
        hasTextualDescription hasValue "the formal verification of the
clients (i.e. the activity Review Customer Financial Details) may be only
performed by a person with access to the CRM systems - as no other person
should be allowed to view the financial information on clients"
        hasSubject CustomerServiceRole_59871239
        hasSubject CustomerInformationAdministeringRole_287334
        hasSubject ReviewCustomerFinancialDetails_76765600
        hasRule BizRule_12_A
```

Fig. 6. WSML Code Snippet – Business Policy Example

3 Related Work

There has been ongoing work on semantic compliance management, as shown in [Na07] where an approach for semantic compliance management for BPM is presented. However, the approach used concentrates on implementing internal controls and is restrictive because it relies on the necessary definition of risks. Another approach is presented in [Sa07] where the authors introduce the modeling of internal control objectives in business processes as a mean to integrate compliance requirements in business process design. The authors also relate their work to risk analysis and internal control modeling.

In [Ka08, Ka07], another approach for business process-based compliance management is presented. It defines an extension to a business process meta-model for regulatory compliance. However, the approach does not incorporate ontologies and thus, does not profit from the power of semantic technologies. In [Go06a, Go06b], deontic (obligations and permissions) constraints expressible for business processes are modeled using temporal deontic assignments. The latter can also be used in business process design and in expressing business process contracts.

The authors of [Sa06] present an approach to formalize contract documents and those aspects of BPs that relate to these business contracts. For this purpose, the semantics of business contracts and their violations are described using a specialized logic. Furthermore, the authors have shown how this formal specification of contracts can used to generate compliant processes. The work in [Sc07] is one of the rare semantic approaches to BP compliance where a compliance ontology is designed and proposed to be integrated in BP models. [Gh07] Proposes an approach based on so-called compliance patterns (i.e., pre-defined BP models proven to be compliant to regulations) where the deviation of a given BP model to a certain compliance pattern is computed.

While relying on the definition of compliance as SWRL rule, [Pa07] recognizes the limited expressiveness of the language and propose the use of extensions. The authors of [Li07] identify the need for separate modeling of compliance and processes. Process models are transformed from BPEL into Pi-Calculus (algebra for modeling concurrent communicating processes) and compliance rules are modeled in temporal logic using a special graphical notation. Model checking techniques are then used to formally check a process pool. In [Mi05], policy definitions are integrated into BPs and rely on BP events

and transactions for run-time compliance monitoring. In fact, this work poses fundamental questions about architectures for process compliance monitoring integrating events and policies such as the need for a formal definition of events, event triggers, event patterns, message handling as well as state management.

None of the previously presented works takes a (i) generic approach to regulatory compliance management, (ii) *seeks to cover all layers of BPM*, (iii) *makes use of the advantages of semantic technologies for compliance management*, (iv) allows *declarative modeling of policies and rules while separating these two concepts* and (v) takes *compliance modeling, enforcement and monitoring* as a target all at the same time. Our claim through this work is that our approach allows achieving these goals. However, the works presented all show up interesting challenges and ways of dealing with these. Many of the approaches used for this can be inspiring for realizing our own approach.

4 Concluding Remarks and Future Work

This work introduced the challenge of regulatory business process compliance management and motivates the use of policies for modeling and enforcing compliance on BP-centered enterprise models. This paper contributes an integrated architecture for compliance management and SBPM, thus bridging together the strategy level (policies and rules extracted from regulations) and the operational level (BP models) by making BP knowledge available to policies. We also contributed a core ontology for modeling policies and rules, which seeks to be used as a generic top-level ontology in compliance modeling. This framework is in its early stages and still requires further development. After having validated the policies ontology using real-world examples, we need to provide a reference implementation of the compliance framework. This requires a refinement of the technical architecture of the components we distinguished. We will also get to elaborate an approach for structuring regulations from their raw natural language representation into a form that can be easily represented using the policy ontology. Still, it is needed to show how inference engines integrate with the ontology and design the required ontology transformations and compliance checking algorithms. Once we dispose of the ontologies and the tools required, the next step will be to implement a prototype of an implementation of a concrete regulation on real world examples of business processes and regulations.

References

[Sch00] Scheer, A.W. ARIS - Business Process Frameworks. Third edition, Springer, Berlin. 2000.

[Kh08a] El Kharbili, M.; Stein, S.; Markovic, I. & Pulvermüller, E. Sadiq, S.; Indulska, M. & zur Muehlen, M. (ed.) Towards a Framework for Semantic Business Process Compliance Management. Proceedings of the workshop on Governance, Risk and Compliance for Information Systems (GRCIS 2008), CEUR Workshop Proceedings, Montepellier, France, June 2008, 339, pp. 1-15.

[Co02] Congress of the United States. Public Company Accounting Reform and Investor Protection Act (Sarbanes-Oxley Act). Pub. L. No. 107-204, 116 Stat. 745.2002.

[Ha07] Hartman, T.E. The Cost of Being Public in the Era of Sarbanes-Oxley. 2007.

[COSO08] Committee of Sponsoring Organizations of the Treadway Commission (COSO) framework. Retrieved on 01.11.2008 at: http://www.coso.org/default.htm.

[COIRT08] Control Objectives for Information and Related Technologies. Retrieved on 01.11.2008 from: http://www.isaca.org/Template.cfm?Section=COBIT6&Template=/TaggedPage/TaggedPageDisplay.cfm&TPLID=55&ContentID=31519.

[ITIL08] IT Infrastructure Library. Retrievd on 01.11.2008 from: http://www.ogc.gov.uk/guidance_itil.asp.

[TOGAF08] TOGAF Framework. Retrieved on 01.11.2008 from: http://www.togaf.org/, http://www-128.ibm.com/developerworks/ibm/library/ar-togaf1/.

[Fo92] M. S. Fox. The TOVE Project Towards a Common-Sense Model of the Enterprise. In IEA/AIE '92: Proceedings of the 5th international conference on Industrial and engineering applications of artificial intelligence and expert systems, pages 25–34, London, UK, 1992. Springer-Verlag.

[Us98] Uschold, M., King, M., Moralee, S., Zorgios, Y. (1998). The Enterprise Ontology. The Knowledge Engineering Review, 13(1):31–89, 1998. Retrieved August 1, 2008, from: http://www.aiai.ed.ac.uk/project/enterprise/enterprise/ontology.html.

[Di06] Jan Dietz. Enterprise Ontology - Theory and Methodology. Springer-Verlag Berlin Heidelberg. 2006.

[Za92] Zachman, J. A. (1992). Extending and Formalizing the Framework for Information Systems Architecture. IBM Systems Journal (Volume 31, No. 3).

[He05] Hepp, Martin et al.: Semantic Business Process Management: A Vision Towards Using Semantic Web Services for Business Process Management. IEEE International Conference on e-Business Engineering (ICEBE 2005). Beijing, China, October 18-20, 2005, pp. 535-540.

[He07] Martin Hepp, Dumitru Roman: An Ontology Framework for Semantic Business Process Management, Proceedings of Wirtschaftsinformatik 2007, February 28 - March 2, 2007, Karlsruhe.

[Pe08a] Pedrinaci, C., Brelage, C. van Lessen, T., Domingue, J., Karastoyanova, D. and Leymann, F. 2008. Semantic Business Process Management: Scaling up the Management of Business Processes. In 2nd IEEE International Conference on Semantic Computing (ICSC), IEEE Computer Society.

[SBVR08] SBVR, Object Management Group (OMG) 2008. Semantics of Business Vocabulary and Rules (SBVR), formal specification V1.0. Retrieved August 1, 2008, from: http://www.omg.org/spec/SBVR/1.0/.

[ISO08] ISO 639 Standard series. Retrieved on 01.11.2008 from: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=39534.

[Ki99] Kim, H.M., Fox, M.S., and Gruninger, M. An ontology for quality management - enabling quality problem identification and tracing. BT Technology Journal (Vol. 17, No. 4, pp. 131-140). 1999.

[Fu05] Norbert E. Fuchs, Kaarel Kaljurand, Gerold Schneider (2005). Verbalising Formal Languages in Attempto Controlled English I. In deliverable I2-D5 of the REWERSE research project. Retrieved August 15, 2008, from: http://rewerse.net/deliverables/m18/i2-d5.pdf.

[Kh08b] El Kharbili, M.; Stein, S. Pulvermueller, E. Policy-Based Semantic Compliance Checking for Business Process Management. In Proceedings of MOBIS Workshops - MOBIS Conference, Saarbrücken, November 2008.

[Ni07a] Nitzsche, J., Wutke, D. and van Lessen, T. An Ontology for Executable Business Processes. In Hepp, M., Hinkelmann, K., Karagiannis, D., Klein, R. and Stojanovic, N. (eds.): Semantic Business Process and Product Lifecycle Management, Proceedings of the Workshop SBPM 2007. Innsbruck, April 7, 2007, CEUR Workshop Proceedings, ISSN 1613-0073, online CEUR- S.org/Vol-251/.

[Ni07b] Nitzsche, J., van Lessen, T., Karastoyanova, D., and Leymann, F. BPEL for Semantic Web Services (BPEL4SWS). On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops (179-188). Springer. 2007.

[OASIS08] OASIS Web Services Business Process Execution Language (WSBPEL) TC. Web services business process execution language version 2.0 committee specification. http://docs.oasis-open.org/wsbpel/2.0/CS01/wsbpelv2.0- CS01.pdf, January 2007.

[Do05] B. van Dongen, A. K. Alves de Medeiros, H. Verbeek, A. Weijters, and W. van der Aalst. The ProM Framework: A New Era in Process Mining Tool Support. In Applications and Theory of Petri Nets 2005. 26th International Conference, ICATPN 2005, pages 444–454, Miami, USA, June 2005. Springer-Verlag.

[Bo04] Bonatti, P. A. , Shahmehri, N., Duma, C., Olmedilla, D., Nejdl, W., Baldoni, M., Baroglio, C., Martelli, A., Patti, V., Coraggio, P., Antoniou, G., Peer, J., Fuchs, N. E (2004). Rule-based Policy Specification: State of the Art and Future Work. In deliverable I2-D1 of the REWERSE research project. Retrieved August 15, 2008, from: http://rewerse.net/deliverables/i2-d1.pdf.

[Ka06] Karagiannis D., Nemetz, M., Schwab M. (2006). Dashboards for Monitoring Compliance to Regulations - A SOX-based Scenario. Proceedings of IGO'06 - International Conference on Integrating Global Organizations, Siena.

[We07] Wetzstein, B., Ma, Z., Filipowska, A. Kaczmarek, M., Bhiri, S., Losada, S., Lopez-Cob, J.-M., Cicurel, L. 2007. Semantic Business Process Management: A Lifecycle Based Requirements Analysis. In Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management (SBPM-2007). Vol-251, CEUR-WS, ISSN 1613-0073.

[Fi08] Filipowska, A., Kaczmarek, M., Stein, S. Semantically Annotated EPC within Semantic Business Process Management. Workshop on Advances in Semantics for Web Services (semantic4ws). Milan, Italy. 2008.

[Be08] Belecheanu, R. et al. Cabral, L., Domingue, J., Gaaloul, W., Hepp, M., Filipowska, A., Kaczmarek, M., Kaczmarek. T., Nitzsche, J., Norton, B., Pedrinaci, C., Roman, D., Stollberg, M., Stein, S. Business Process Ontology Framework. SUPER research project deliverable D1.1. 2007. Retrieved August 1, 2008 from: http://www.ip-super.org/res/Deliverables/M12/D1.1.pdf.

[Pe08b] Pedrinaci, C., Domingue, J., De Medeiros, A. A Core Ontology for Business Process Analysis. In Proceedings of the 5th European Semantic Web Conference 2008, Tenerife. 2008.

[Li07] Liu, A. Y., Müller, S., Xu, K. A Static Compliance-Checking Framework for Business Process Models. IBM Systems Journal (46(2), pp. 335–361). 2007.

[Ro06] Rossak, W., Foetsch, D., Pulvermueller, E. Modeling and Verifying Workflow-based Regulations. In Proceedings of the international workshop on regulations modeling and their validation and verification - REMO2V06 (pp. 825–830). CEUR-WS.org/vol-241, Luxemburg. 2006.

[Ka04] Kagal, L. (2004). A Policy-Based Approach to Governing Autonomous Behavior in Distributed Environments. PhD Thesis, Faculty of the Graduate School of the University of Maryland. 2004.

[Na07] Namiri, K., Stojanovic., N. (2007). A Formal Approach for Internal Controls Compliance in Business Processes. 8th Workshop on Business Process Modeling, Development, and Support (BPMDS07). Trondheim, Norway.

[Sa07] Sadiq S., Governatori G., Namiri K. (2007). Modeling Control Objectives for Business Process Compliance. Proceedings of the 5th International Conference, BPM 2007 (pp.149-164). Brisbane, Springer.

[Ka08] Karagiannis, D. (2008). A Business process Based Modelling Extension for Regulatory Compliance. Proceedings of the Multikonferenz Wirtschaftsinformatik 2008, Munich.

[Ka07] Karagiannis D., Mylopoulos J., Schwab M. (2007). Business Process-Based Regulation Compliance: The Case of the Sarbanes-Oxley Act. Proceedings of 15th IEEE International Requirements Engineering Conference, New Delhi.

[Go06a] Goedertier, S., Vanthienen, J. (2006). Designing Compliant Business Processes from Obligations and Permissions, 2nd Workshop on Business Processes Design (BPD'06).

[Go06b] Governatori, G., Milosevic, Z., Sadiq, S. (2006). Compliance checking between business processes and business contracts. 10th International Enterprise Distributed Object Computing Conference (EDOC 2006) (pp. 221-232). IEEE Press.

[Sa06] Sadiq, S., Milosevic, Z., Fiadeiro, J., Orlowska, M. 2006. Towards a methodology for deriving contract-compliant business processes. In Proceedings of the 4th International Conference on Business Process Management (BPM06), Vienna, Austria.

[Sc07] Schmidt, R., Bartsch, C., Oberhauser, R. (2007). Ontology-based representation of compliance requirements for service processes. Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management (SBPM 2007).

[Gh07] Ghose, A.K. and Koliadis, G. 2007. Auditing business process compliance. In Proceedings of the International Conference on Service-Oriented Computing (ICSOC-2007) (vol. 4749 of Lecture Notes in Computing Sicence, pp. 169–180).

[Pa07] Parameswaran, N., Ray, P., Yip, F.2007. Rules and Ontology in Compliance Management. In Proceedings of the 11th IEEE International Enterprise Distributed Object Computing Conference (page 435).

[Mi05] Milosevic, Z. 2005. Towards Integrating Business Policies with Business Processes. In W.M.P. van der Aalst, B. Benatallah, F. Casati, and F. Curbera (Eds.), Business Process Management (volume 3649, pp. 404–409).

[Ly08] Ly, L. T.; Göser, K.; Rinderle-Ma, S. & Dadam, P. Sadiq, S.; Indulska, M. & zur Muehlen, M. (ed.). Compliance of Semantic Constraints – A Requirements Analysis for Process Management Systems. Proceedings of the workshop on Governance, Risk and Compliance for Information Systems (GRCIS 2008), CEUR Workshop Proceedings, Montepellier, France, June 2008, 339, 31-45.

Verifying Business Rules Using an SMT Solver for BPEL Processes

Ganna Monakova[1], Oliver Kopp[1], Frank Leymann[1], Simon Moser[2], Klaus Schäfers[2]
[1]Institute of Architecture of Application Systems, Stuttgart, Germany
[2]IBM Deutschland Research & Development GmbH, Böblingen, Germany
[1]lastname@iaas.uni-stuttgart.de [2]{smoser|kschaef}@de.ibm.com

Abstract: WS-BPEL is the standard for modelling executable business processes. Recently, verification of BPEL processes has been an important topic in the research community. While most of the existing approaches for BPEL process verification merely consider control-flow based analysis, some actually consider data-flows, but only in a very restrictive manner. In this paper, we present a novel approach that combines control-flow analysis and data-flow analysis, producing a logical representation of a process model. This logical representation captures the relations between process variables and execution paths that allow properties to be verified using Satisfiability Modulo Theory (SMT) solvers under constraints represented by the modelled assertions.

1 Introduction

Many of today's enterprises model their business processes in BPEL [OAS07]. In order to ensure quality of the modelled process, its correctness has to be proved. A correct business process always terminates and produces valid results. Successful termination implies the absence of deadlocks and can be verified using several techniques, e.g. [Hol04, MM06, Loh07]. These techniques merely consider the control-flow of the process and abstract from the data-flow. A valid result for a business process is usually defined by business constraints on the produced output, e.g. "A customer who ordered less than 100 items should not receive a discount". The verification of such a business constraint depends on the relations between different process variables: to be able to verify the above constraint we need to know the relation between number of ordered items and calculated discount. Such relations depend on both control-flow and data-flow.

We use a simplified price calculation process that is a part of an onlineshop process to illustrate the concepts of our approach. The basic idea of the process presented in Figure 1 is to determine the price of an item depending on the amount of ordered items. A discount is given based on the subtotal. The process receives the number of ordered items n and determines the item price dependent on n. If the customer has ordered more than 100 items, they can buy each for 10 €, otherwise a price of 20 €per unit is offered. After the subtotal is calculated, the customer gets 10% discount if it is between 1.000 € and 2.000 € and 20% discount if the subtotal exceeds 2.000 €. The total sum s is sent back to the customer. The process contains both graph-based (`flow`) and block-structured (`if`) constructs to show

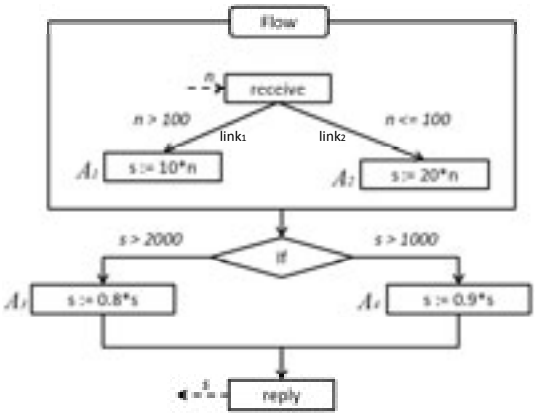


Figure 1: Price calculation process

that the presented approach is capable of handling both.

Assume that a business analyst wants to ensure that the modelled process satisfies the following business constraints: "A customer who ordered less than 200 items should not receive a discount of more than 10% " and "If the customer ordered 50 items or more, they should get a discount". To enable the verification of these constraints, the relations between the variables s and n have to be analysed: At first glance, the control flow decision made in the `if` activity considers the value of s only and does not depend on the decision made in the preceding `flow` activity. However, if the number of ordered items exceeds 100 ($n > 100$), then we know that A_1 will be executed, and thus $s = 10 * n > 10 * 100 = 1000$. With this knowledge, we can conclude that the `if` construct will never be skipped, meaning the customer will be assured of a discount. This analysis is only possible if we know the relation between s and n, namely $s = 10 * n$, and know the condition which enables this relation, namely $n > 100$. We also call such a condition the *Execution Condition (EC)* of activity A_1. Note that we do not make any assumptions on the values the variable n can take, but only analyse the connections between data conditions and data manipulations.

The above example shows that a relationship between variables enables restrictions on possible control flows to be analyzed. As an alternative example, the execution path $receive$, A_2, A_3, $reply$ is not possible: if n is 100, then the condition $s > 2.000$ evaluates to false, because s was assigned a value of 2.000 after A_2 occured.

In summary, the relationship between variables will have an impact on the evaluation of the conditions that drive the control-flow. For this reason they play an important role in the process analysis. In this paper, we present a technique that analyses these relations and models them using *logical assertions*. Logical assertions capture the execution semantics of a BPEL process and form the verification basis for the business constraints. The negation of a business constraint is added to the verification base and checked to ensure it is unsatisfiable. If it is unsatisfiable, then the business constraint is valid. Otherwise a counter example violating this constraint is found.

The satisfiability of the modelled assertions is checked using the Satisfiability Modulo Theories (SMT) solver Yices [DdM08]. An SMT solver solves satisfiability problems for

Boolean formulas containing predicates of underlying theories. Such theories can be, for example, theories of arrays, lists and strings [B+06]. In addition, an SMT solver can be extended with new theories as shown in [ND79]. To present the proposed approach, we consider the theory of the linear arithmetic.

The remainder of this work is organized as follows: Section 2 presents how the execution condition can be derived for each activity in a BPEL process. Section 3 shows how a BPEL process can be analysed and modelled with logical assertions. The verification of business constraints within the modelled context is presented in Section 4. Section 5 compares the presented approach with related work and 6 concludes and provides an outlook on future work.

2 Determining Execution Conditions

An execution condition of an activity is a Boolean expression that is constructed recursively by analysing all conditions that have to be satisfied to enable the execution of this activity. The following conditions have to be analysed:

1. A `flow` activity models concurrency. Additional synchronisation between activities can be modelled with links. Each activity in a flow can have an arbitrary number of incoming and outgoing links. Each link has exactly one source and one target activity and expresses the synchronisation dependency between them. Each link has a transition condition, which is evaluated if the source activity was successfully executed. The transition condition is an arbitrary[1] XPath expression of return type $Boolean$.

   If a transition condition is not explicitly defined, the link gets a default transition condition $true$. Each link has a status that can be either *unknown*, *negative* or *positive*. A link status can become positive if and only if the source activity of the link was successfully executed and the transition condition of the link evaluates to $true$. If the source activity was skipped or the transition condition evaluates to false, the link status becomes negative. This is called *dead path elimination (DPE)*. More information regarding `flow` semantics and DPE is given in [OAS07, C+03]. Each activity in a `flow` has a join condition. A join condition is a Boolean function over the status values of the incoming links. The default join condition is a disjunction of all incoming link status values.

   According to the flow semantics described above, the execution condition (EC) of an activity A can be recursively constructed as follows:

$$\begin{aligned} EC(A) &= JC(A) \\ JC(A) &= f(S(L_1)...S(L_n)) \\ \forall i \in [1..n] \ : \ S(L_i) &= L_i.tc \wedge EC(L_i.source), \text{ where} \end{aligned}$$

[1] Recall that in this work we use linear arithmetic theory and therefore consider linear arithmetic expressions only

A denotes an activity in a `flow` with L_1,..., L_n incoming links (or rather A is the target of L_1,..., L_n), $JC(A)$ denotes the join condition of the A, $L_i.tc$ denotes the transition condition of the link L_i, $L_i.source$ denotes the source activity of the link L_i, $S(L_i)$ denotes the status of the link L_i and *f(. . .)* is an arbitrary Boolean function, which specifies the join condition [OAS07].

In example of Figure 1 the `flow` activity contains three activities. The `receive` activity does not have any incoming links, therefore both its join and execution condition are true. Activities A_1 and A_2 have one incoming link and the default join condition. Therefore $EC(A_1) = JC(A_1) = S(link_1) = link_1.tc \wedge EC(receive) = link_1.tc \wedge true = link_1.tc = n > 100$

2. In an `if` activity, the branch conditions are evaluated from left to right and the first branch where a condition that evaluates to $true$ is taken. If no condition evaluates to $true$ and no else branch exists, the `if` activity immediately completes. Therefore, for an `if` with n conditional branches with the corresponding conditions $C_1, ..., C_n$ and an else-branch, the execution condition for each branch is constructed as follows:

$$\begin{aligned} \forall i \in [1..n] \ : \ EC(branch_i) \ &= \neg C_1 \wedge ... \wedge \neg C_{i-1} \wedge C_i \\ EC(else) \ &= \ \neg C_1 \wedge ... \wedge \neg C_n \end{aligned}$$

Note that this modelling ensures that the execution condition of the branch j cannot evaluate to $true$, although a branch i exists with $i < j$ and $EC(branch_i) = true$.

As an example, consider the `if` in Figure 1. It contains two branches with the conditions $cond_1 = s > 2.000$ and $cond_2 = s > 1.000$. Thus, the execution conditions of activities A_3 and A_4 become: $EC(A_3) = cond_1$ and $EC(A_4) = \neg cond_1 \wedge cond_2$.

3. In a `pick` activity, the first received message or the timeout event decides, which branch is taken. Without considering any interacting partner, the branch choice is non-deterministic. To model this, each of the branches gets a Boolean attribute *fired*. Assuming that this attribute can only be set to $true$ if and only if the corresponding branch is chosen, the execution condition of each branch in a `pick` P with $B_1, ..., B_n$ branches is modelled as follows:

$$\forall i \in [1..n] \ : \ EC(B_i) \ = \ B_i.fired$$

To model the property that only one of the branches can actually be chosen, the following constraints are used:

$$\begin{gathered} \forall i \in [1..n] \ : \ B_i.fired \rightarrow \forall j \in [1..n](j \neq i \rightarrow \neg B_j.fired) \\ \bigvee_{i \in [1..n]} B_i.fired \end{gathered}$$

A more precise branch-choice semantic has to consider the partner process. Because of the space limitations, we don't consider it here. The complete `pick` semantic modelling is described in [Mon08].

In addition, each structured activity influences its children: if a structured activity is skipped, then all of its children are also skipped. An activity will only be skipped if its execution condition evaluates to false. That means the execution condition of an activity depends on the execution conditions of its parent: if a structured activity P with the execution condition C contains children activities A_1,..., A_n, then the execution conditions of each child A_i takes the form $C_i \wedge C$. Here C_i denotes the combination of other conditions that have impact upon the execution of A_i and is derived as presented above.

In the example shown in Figure 1 the execution conditions of all structured activities, namely `if` and `flow`, are equal $true$. Therefore all derived execution conditions remain unchanged.

Currently the loops are unfolded and thus can be mapped to a set of the `if`-constructs. We investigate the loop invariants to improve this technique. The extension of the proposed approach with `scopes`, `fault-` and `compensation` handlers is also addressed in ongoing work.

3 Analysing BPEL Processes

There are several possible executions of the same BPEL process. Executions vary in the executed activities and in their execution order. Input to the process and variable relations determine which activities are executed and the synchronisation dependencies between activities determine their execution order. The relations between variables are defined in assign activities. If an assign activity is executed, then the relations defined in this activity become valid. Therefore, the execution condition of an assign activity is the enabling condition for the relations defined in this activity. To be able to evaluate an execution condition, we need to know which relations are valid at the point of evaluation time. For this purpose, we need to know which values each variable can take at the point of evaluation. This depends on the activities that may have written to this variable. Such activities are called *possible writers* and can be determined for each variable access using the *Concurrent Static Single Assignment (CSSA)* representation of a BPEL process. In the following, we begin by describing the CSSA form of BPEL in Section 3.1. In Section 3.2 we show how the synchronisation dependencies captured in the CSSA form are modelled using logical assertions. Finally, Section 3.3 presents how the relations between variables are modelled.

3.1 CSSA Representation of BPEL Processes

The *Static Single Assignment (SSA)* form is an intermediate representation that is used to facilitate program analysis and optimisation [C+91]. The SSA form can be characterised through two properties. First, each reference to a name corresponds to the value produced at precisely one definition point giving the single assignment property. The single assignment property is achieved by giving a unique index to each occurrence of the original variable on the left side of an assignment (when it is reassigned). Second, it identifies the points in

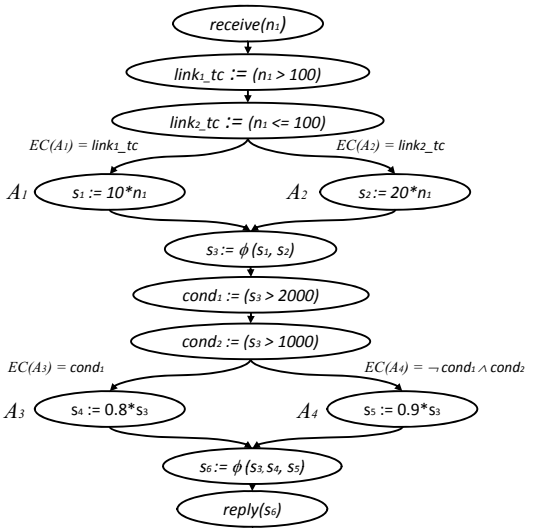


Figure 2: Simplified CSSA representation of the price calculation process

the computation where values from different control-flow paths merge. At a merge point, several different SSA names, corresponding to different definitions of the same original name, may flow together. To ensure the single-assignment property, the construction inserts a new definition at the merge point; its right hand side is a pseudo-function called a ϕ-function that represents the merge of multiple SSA names. As parameters the ϕ-function contains all variables written by possible writers. Due to the uniqueness of variable names, there is no need to distinguish between variables and activities. Thus, we use the term "possible writers" also for the variables, which can be uniquely mapped to the corresponding activity.

While SSA form is suitable for the representation of sequential program execution, it cannot deal with the parallel constructs. To analyse the parallel execution, an extension to the SSA form, called *Concurrent Static Single Assignment (CSSA)* is used [LMP97]. The idea of the CSSA form is that it summarises the interleaving information for conflicting variables in an explicitly parallel program through the use of π-functions. The values of all conflicting variables are well defined by the π-function at the point where the π-function is placed and are represented via parameters of this function. Like the SSA form, the CSSA form has the property that all uses of a variable are reached by exactly one assignment to the variable.

The approach described in [M$^+$07] shows how to obtain a CSSA representation for a BPEL process. Figure 2 shows the simplified CSSA representation of the price calculation process from Figure 1. For readability, the nodes representing implicit join conditions, which in our case are equal to the status of the incoming link, are skipped.

The CSSA representation captures all accesses to the process variables. The link transition conditions, the activity join conditions and the `if` branch conditions are each represented as single nodes. Each node defines a unique variable that represents the

corresponding condition. These variables are used for modelling an activity's execution conditions.

3.2 Modelling Synchronisation Dependencies

The synchronisation dependencies are captured within the CSSA representation. An activity B has a synchronisation dependency on activity A if there exists a path from A to B in the CSSA graph. We assume that each assign activity writes only one variable, and thus that dependencies between activities can be considered as dependencies between corresponding written variables.

For each assign activity A let W_A denote the variable written in A. We define a *Direct Dependency Set* $D(W_A)$ as follows: $D(W_A) = \{W_{A'} \mid$ there is a path from A' to A which does not contain any other assign activity$\}$

The synchronisation dependencies define the irreflexive partial order on activities execution. To model this partial order, and therefore the constraints on possible executions, each variable W_A gets an $order$ attribute. This attribute is of type Integer. Our goal is to specify the constraints on how these attributes can be assigned and an SMT solver assigns the concrete values.

Let $\mathcal{A}$ denote the set of all assign activities. We constrain all possible executions of the assign activities to those that are allowed by the specified synchronisation dependencies as follows:

$$\forall A \in \mathcal{A}, \forall W_{A'} \in D(W_A) : \; W_A.order > W_{A'}.order$$

Note that parallel activities do not have any synchronisation dependencies. Therefore, the variables written in such activities do not get mutual constraints. This corresponds to the non-determinism within parallel constructs.

3.3 Modelling Relations Between Variables

The relations between variables are defined by assign activities. The evaluation of the execution condition of an assign activity decides whether this relation is valid. To model these relations each variable gets an ec and a val attributes. The Boolean ec attribute defines the execution condition of the activity, in which this variable is written. The val attribute denotes the value of this variable. Note that the actual values are not necessarily known. For example, for the assign activity A_1: $s_1 := 10 * n_1$ we specify a constraint $s_1.val = 10 * n_1.val$. Even if the actual value of n_1 is not known, the assertion captures the dependency between the values of s_1 and n_1.

An assign activity A will only be executed if its execution conditions $EC(A)$ evaluates to true. Let W_A denote the variable written in A and $f(x_1, ..., x_n)$ denote the right side of A. In this work $f(x_1, ..., x_n)$ can only be a ϕ-, π or a linear arithmetic function. The

relation between variables defined by A is modelled as follows:

$$W_A.ec = EC(A)$$
$$W_A.ec \rightarrow W_A = f(x_1, ..., x_n)$$

According to these rules, we specify the following assertions for A_3 and A_4:

$$cond_1 = s_3 > 2000 \qquad cond_2 = s_3 > 2000$$
$$s_4.ec = cond_1 \qquad s_4.ec \rightarrow s_4 = 0.8 * s_3$$
$$s_5.ec = \neg cond_1 \wedge cond_2 \qquad s_5.ec \rightarrow s_5 = 0.9 * s_3$$

In case $f(x_1, ..., x_n)$ is a linear arithmetic expression, it is mapped one to one, as the example for the activities A_3 and A_4 above shows. To model the assigns with the ϕ- and π- functions on the right side, we need to specify their selection semantics. The *selection semantics* defines the rules for the selection of the effective writer. In each single execution of a process each variable has only one effective writer, namely the one that wrote the variable which is used in the current execution. For example, consider the `if` construct in Figure 1. There are three possible executions for this `if`: the left branch is taken, the right branch is taken or the `if` is skipped. For each of these executions the value of s_6 is clearly defined: in the case of the left branch it becomes s_4, in the case of the right branch it becomes s_5 and in the skipped case it becomes s_3. This describes the selection semantic of the $\phi(s_3, s_4, s_5)$-function for our example.

The ϕ-function is synchronised on the possible writers listed as parameters of the ϕ-function. Therefore, the last writer is the effective writer. Thus, the selection semantic of a ϕ-function is modelled as follows: If $x_i = \phi(x_{i_1}, ..., x_{i_n})$, then we define the following constraints on the value of x_i, where x_{i_k} denotes the effective writer:

$$\bigvee_{k \in [1,n]} \left((x_i.val = x_{i_k}.val) \wedge x_{i_k}.ec \wedge \bigwedge_{l \in [1,n], l \neq k} (x_{i_l}.ec \rightarrow (x_{i_k}.order > x_{i_l}.order)) \right)$$

For our example, the ϕ-node $s_6 = \phi(s_3, s_4, s_5)$ is modelled by the following assertion:

$$\begin{aligned}
&(s_6.val = s_3.val) \wedge s_3.ec \wedge (s_4.ec \rightarrow (s_3.order > s_4.order)) \\
&\qquad \wedge (s_5.ec \rightarrow (s_3.order > s_5.order)) \\
\vee\ &(s_6.val = s_4.val) \wedge s_4.ec \wedge (s_3.ec \rightarrow (s_4.order > s_3.order)) \\
&\qquad \wedge (s_5.ec \rightarrow (s_4.order > s_5.order)) \\
\vee\ &(s_6.val = s_5.val) \wedge s_5.ec \wedge (s_3.ec \rightarrow (s_5.order > s_3.order)) \\
&\qquad \wedge (s_4.ec \rightarrow (s_5.order > s_4.order))
\end{aligned}$$

The assertions specified for A_3 and A_4 together with the above assertion for the ϕ-function model the relations between the variables s_3, s_4, s_5, s_6, $cond_1$, $cond_2$. For example, the value assignments $s_3 = 1000$; $cond_1 = false$; $cond_2 = false$; $s_4 = any$; $s_5 =$

any; $s_6 = s_3 = 1000$ satisfy these assertions, while the assignment $s_3 = 1000$; $cond_1 = false$; $cond_2 = false$; $s_4 = any$; $s_5 = any$; $s_6 = s_3 = 900$ does not.

While a ϕ-function chooses an effective writer after all possible writers are executed, the assertions for the π-function should consider the possibility that some of the possible writers defined in the π-function can actually be executed after the assign activity that uses the value of the π-function. In addition, because the possible writers of the π-function are not necessarily synchronised, we have to explicitly model the property that a variable has to be written before it can be used as the value of the π-function. Collectively, we model the selection semantic of a π-function as follows:

If $x_i = \pi(x_{i_1}, ..., x_{i_n})$, then we define on the value of x_i the following constraints:

$$\bigvee_{k \in [1,n]} \Bigg((x_i.val = x_{i_k}.val) \wedge x_{i_k}.ec \wedge (x_i.order > x_{i_k}.order) \wedge \bigwedge_{l \in [1,n], l \neq k} (x_{i_l}.ec \rightarrow ((x_{i_k}.order > x_{i_l}.order) \vee (x_{i_l}.order > x_i.order))) \Bigg)$$

Compared to the ϕ-function, we have two new clauses. $x_i.order > x_{i_k}.order$ models the fact that x_{i_k} has to be written before x_i can read it and $x_{i_l}.ec \rightarrow ((x_{i_k}.order > x_{i_l}.order) \vee (x_{i_l}.order > x_i.order))$ states that if any other possible writer is written ($x_{i_l}.ec = true$), then it is either written before x_{i_k} or after x_i. Otherwise x_i would read the value of x_{i_l}.

4 Business Constraints Verification

In the previous section we showed how the logical assertions could be used to capture the relationships between BPEL process variables. These assertions form the basis for the business constraints verification. To verify a business constraint, its negation is modelled as an assertion and added to the evaluation basis. If the obtained combination of assertions cannot be satisfied, then the business constraint itself is fulfilled. Otherwise an assignment to the process variables violating this constraint will be found.

Let C denote the conjunction of the assertions modelling the verification basis. Note that C is fulfilled for the modelled process. For a given business constraint let B denote the corresponding logical assertion. For the business constraint to be always fulfilled for the modelled process the formula $C \rightarrow B$ has to be valid. To prove this, we verify the satisfiability of its negation:

$$\neg(C \rightarrow B) = C \wedge \neg B$$

If it cannot be satisfied, then $C \rightarrow B$ is valid and therefore the business constraint B is fulfilled for our process model.

As an example we show how the business constraints defined in Section 1 can be verified for the price calculation process. We assume that the basis is already modelled as shown in

Section 3. So far this basis is satisfiable with all assignments to the variables, which are possible in the real process execution.

Assume we want to verify the following business constraint: "A customer who ordered less than 200 items should not receive a discount of more than 10%". The negation of this rule is represented with the following assertion:

$$(n_1.val < 200) \wedge s_4.ec$$

Here s_4 is the variable written in the activity A_3, which calculates a 20% discount. This assertion assumes that there is a possible configuration of the variables that satisfies the fact that the customer orders less than 200 items and gets 20% discount. Alternatively it can be expressed with the following assertion:

$$(n_1.val < 200) \wedge (s_6.val < 0.9 * s_3.val)$$

If one of the above assertions is added to the modelled verification basis, it becomes unsatisfiable. In other words, the business constraint is fulfilled. This is due to the fact that the dependency between n and s is captured within the basis assertions. Therefore if $100 < n_1 < 200 \Rightarrow link_1\_tc = true \Rightarrow s_1 = 10 * n_1 < 2000 \Rightarrow s_3 = s_1 \Rightarrow cond_1 = false \Rightarrow$ the customer will not get 20% discount. If $n_1 \leq 100 \Rightarrow link_2\_tc = true \Rightarrow s_2 = 20 * n_1 < 2000 \Rightarrow s_3 = s_2 \Rightarrow cond_1 = false \Rightarrow$ customer will not get 20% discount.

Analogous to the above, for the property "If the customer ordered 50 items or more, then they should receive a discount" the corresponding assertion, which corresponds to the negation of this property, is:

$$\neg((n_1.val \geq 50) \rightarrow (s_4.ec \vee s_5.ec))$$

This assumes that a customer who ordered 50 or more items did not receive a discount. Alternatively, this rule can be modelled as follows:

$$\neg((n_1.val \geq 50) \rightarrow (s_6.val < s_3.val))$$

In this case, Yices finds variable assignments that satisfy the modelled context. This model contains $n_1 = 50$, which means that when the customer orders 50 items, they do not receive any discount[2].

5 Related Work

An overview of existing BPEL formalizations and verification approaches is provided in [BK06]. We present a summary of the presented approaches here.

[2]The input for Yices is available at `http://www.iaas.uni-stuttgart.de/forschung/pricecalculation.ys`.

Petri net approaches abstract from data-flow [MM06, Loh07]. For example, the decision of which `if` branch is taken is made non-deterministically. Thus, in the example from Figure 1 the execution path $receive$, A_2, A_3, $reply$ becomes possible, which can never happen and thus leads to wrong verification results. The approach of [YTYL05] transforms BPEL to Coloured Petri nets. Here, each type of message is transformed to a token with a different colour. However, the mapping does not consider relations between the conditions and variables.

The Promela approach transforms a BPEL process into Promela and verifies it with the SPIN model checker [FBS04, Nak05, FFK05]. SPIN itself cannot handle large data domains. SPIN works with explicit states and therefore has to check all possible values an integer variable n can take, which is infinite [Hol04]. Even if the value of n is bounded by the maximum integer value, the number of states explodes. For example, if x is bound to $[r_1 \dots r_2$ and y to $[s_1 \dots s_2$, then SPIN has to consider $(r_2 - r_1 + 1) * (s_2 - s_1 + 1)$ states. The approach presented in [BGS07] is similar to the Promela approaches, but uses Bogor [RDH03] to do the actual model checking. As the Promela approach, it cannot handle the large data domains.

An approach based on abstract state machines (ASM) is presented in [FR05]. While the mapping covers scopes, it does not consider the relations between conditions and variables. Approaches based on the π-calculus are presented in [WDW07, Fad04, LM07]. As with the ASM approaches, these do not consider the relations between conditions and variables.

The approach presented in [PA08] transforms a BPEL process into a Java program using B2J [Ecl08]. The Java model checker Java PathFinder (JPF, [VHB+03]) is then used with the transformed program. The model checker uses explicit states for each variables combination and therefore cannot handle large or unbound data domains.

When it comes to the determination of data-flow in BPEL processes, the control-flow has to be analyzed. Current work on data-flow analysis are presented in [M+07], [KKL08] and [ZZK07]. The approach presented in [M+07] is based on CSSA, the approach of [KKL08] is based on abstract interpretation, and the approach of [ZZK07] is based on automatons. All of these approaches determine a set of possible writers for each use of a variable. However, all of them do not consider variable relations and the selection semantics. Thus, all of them return $\{A_1, A_2\}$ as possible writers for A_3, which is an over-approximation. This over-approximation can be improved if the execution conditions for each activity were considered.

We showed in [Mon08] how the approach can be used to model and to verify service communication. A proof of concept has been provided using IBM WebSphere as implementation platform.

6 Conclusion and Outlook

We analysed the relations between variables and showed their influence on the data-flow. The presented verification algorithm uses the results of this analysis and enables verification of business constraints. The BPEL process execution semantics, the variable relations and

the business constraints were modelled using logical assertions. These assertions were verified using the SMT solver Yices, that is extensible with different theories. In this work we presented how the linear arithmetic theory can be used to enable business rules verification. Our approach is the first one which goes beyond simple control flow analysis and considers the dependency between control flow and data flow. That is not possible with the other approaches.

The mapping to logical assertions presented in this work excluded BPEL scopes, which is a part of our ongoing work. Furthermore, we investigate the other possibilities to handle the loop constructs, e.g. using loop invariants. We also plan to use the results of this work to analyse interacting processes and choreographies expressed in BPEL4Chor [D+07]. This should enable the analysis and combination of the variables relationships between different partners as shown in [Mon08].

Acknowledgments The work published in this article is partially funded by the MASTER project under the EU 7th Framework Programme (contract no. FP7-216917). Oliver Kopp is funded by the German Ministry of Education and Research (project Tools4BPEL, project number 01ISE08B).

References

[B+06] Bernhard Beckert et al. Intelligent Systems and Formal Methods in Software Engineering. *IEEE Intelligent Systems*, 21(6):71–81, 2006.

[BGS07] Domenico Bianculli, Carlo Ghezzi, and Paola Spoletini. A Model Checking Approach to Verify BPEL4WS Workflows. In *IEEE International Conference on Service-Oriented Computing and Applications (SOCA '07)*, pages 13–20. IEEE computer society, 2007.

[BK06] Franck van Breugel and Maria Koshkina. Models and Verification of BPEL. `http://www.cse.yorku.ca/~franck/research/drafts/tutorial.pdf`, 2006.

[C+91] Ron Cytron et al. Efficiently Computing Static Single Assignment Form and the Control Dependence Graph. *ACM Transactions on Programming Languages and Systems*, 13(4), October 1991.

[C+03] Francisco Curbera et al. Exception Handling in the BPEL4WS Language. In *Conference on Business Process Management*, pages 276–290. Springer, 2003.

[D+07] Gero Decker et al. BPEL4Chor: Extending BPEL for Modeling Choreographies. In *IEEE International Conference on Web Services*. IEEE Computer Society, 2007.

[DdM08] Bruno Dutertre and Leonardo de Moura. The YICES SMT Solver, 2008. Available at `http://yices.csl.sri.com/`.

[Ecl08] Eclipse Foundation. BPEL to Java (B2J) Subproject, 2008. `http://www.eclipse.org/stp/b2j/`.

[Fad04] M. Fadlisyah. Using the π-Calculus for Modeling and Verifying Processes on Web Services. Master's thesis, Insitute for Theoretical Computer Science, Dresden University of Technology, 2004.

[FBS04] Xiang Fu, Tevfik Bultan, and Jianwen Su. Model checking XML manipulating software. In *IEEE Int. Symp. on Software Testing and Analysis*. ACM, 2004.

[FFK05] Jesús Arias Fisteus, Luis Sánchez Fernández, and Carlos Delgado Kloos. Applying model checking to BPEL4WS business collaborations. In *Proceedings of the 2005 ACM symposium on Applied computing*, pages 826–830. ACM, 2005.

[FR05] D. Fahland and W. Reisig. ASM-based semantics for BPEL: The negative Control Flow. In *12th International Workshop on Abstract State Machines*, pages 131–151, March 2005.

[Hol04] Gerard J. Holzmann. *SPIN Model Checker, The: Primer and Reference Manual*. Addison-Wesley Professional, 2004.

[KKL08] Oliver Kopp, Rania Khalaf, and Frank Leymann. Deriving Explicit Data Links in WS-BPEL Processes. In *IEEE International Conference on Services Computing*. IEEE Computer Society Press, 2008.

[LM07] Roberto Lucchia and Manuel Mazzara. A pi-calculus based semantics for WS-BPEL. *Journal of Logic and Algebraic Programming*, 70(1):96–118, January 2007.

[LMP97] Jaejin Lee, Samuel P. Midkiff, and David A. Padua. Concurrent Static Single Assignment Form and Constant Propagation for Explicitly Parallel Programs. In *International Workshop on Languages and Compilers for Parallel Computing*. Springer, 1997.

[Loh07] Niels Lohmann. A Feature-Complete Petri Net Semantics for WS-BPEL 2.0. In *International Workshop on Web Services and Formal Methods*. Springer, 2007.

[M+07] Simon Moser et al. Advanced Verification of Distributed WS-BPEL Business Processes Incorporating CSSA-based Data Flow Analysis. In *IEEE International Conference on Services Computing*, July 2007.

[MM06] Axel Martens and Simon Moser. Diagnosing SCA Components UsingWombat. In *Conference on Business Process Management*. Springer, 2006.

[Mon08] Ganna Monakova. Ontology Based Partner Service Discovery Using a First-Order Logic Representation for BPEL Process Models. Diploma thesis, University of Stuttgart, Institute of Architecture of Application Systems, 2008.

[Nak05] Shin Nakajima. Lightweight formal analysis of Web service flows. *Progress in Informatics*, 1:57–76, November 2005.

[ND79] G. Nelson and Oppen D. Simplification by Cooperating Decision Procedures. *ACM Transactions on Programming Languages and Systems*, 1(2):245–257, 1979.

[OAS07] OASIS. *Web Services Business Process Execution Language Version 2.0*, 2007.

[PA08] P. Parizek and J. Adamek. Checking Session-Oriented Interactions between Web Services. In *Proceedings of 34th EUROMICRO SEAA conference*, pages 3–10. IEEE Computer Society, 2008.

[RDH03] Robby, Matthew B. Dwyer, and John Hatcliff. Bogor: an extensible and highly-modular software model checking framework. In *9th European software engineering conference (ESEC/SIGSOFT FSE)*, pages 267–276. ACM, 2003.

[VHB+03] Willem Visser, Klaus Havelund, Guillaume Brat, SeungJoon Park, and Flavio Lerda. Model Checking Programs. *Automated Software Engineering*, 10(2):203–232, April 2003.

[WDW07] Matthias Weidlich, Gero Decker, and Mathias Weske. Efficient Analysis of BPEL 2.0 Processes using pi-Calculus. In *Proceedings of the IEEE Asia-Pacific Services Computing Conference (APSCC)*. IEEE Computer Society, 2007.

[YTYL05] YanPing Yang, QingPing Tan, JinShan Yu, and Feng Liu. Transformation BPEL to CP-Nets for Verifying Web Services Composition. In *International Conference on Next Generation Web Services Practices (NWeSP)*, pages 137–142. IEEE Computer Society, 2005.

[ZZK07] Yongyan Zheng, Jiong Zhou, and Paul Krause. Analysis of BPEL Data Dependencies. In *Proceedings of the 33rd EUROMICRO SEAA conference*, pages 351–358. IEEE Computer Society, 2007.

All links were last followed on October 20, 2008.

Validating Auction Business Processes using Agent-based Simulations

Emilian Pascalau[1], Adrian Giurca[2] and Gerd Wagner[2]
[1]Hasso Plattner Institute, University of Potsdam
{emilian.pascalau}@hpi.uni-potsdam.de
[2]Brandenburg University of Technology
{giurca, wagnerg}@tu-cottbus.de

Abstract: Business Process Modeling and particularly modeling business processes as collaborations is one of the challenges of today enterprise software development. The industry of software development becomes more and more expensive making crucial a correct translation from business idea to implementation to allow for a complete understanding and complete exchange of information between development participants. Particularly, large established software infrastructures are critical with respect to the integration of new components. In this paper, we describe an automated mapping from Single Item English Auction BPMN model to an Agent-Object Relationship simulation towards validation of this business process. The mapping underlines the capabilities of agent-based simulations to execute business processes as well as a number of open questions with respect of BPMN collaboration models.

1 Motivation and Related Work

The Business Process Management Initiative (BPMI) has developed a standard Business Process Modeling Notation (BPMN) [OMG08]. The primary goal of BPMN is to provide a notation that is readily understandable by all business users such that business processes can be illustrated using a standard notation understandable also to business analysts that create the initial drafts of the processes, continuing towards technical developers responsible for implementing the technology that will perform those processes, and finally going on, towards the business people who will manage and monitor those processes. A complete introduction to business processes is given in [Wes07]. This work tries to narrow the gap between two different points of view underlined by two different communities: researchers with a background in formal methods interested in investigating structural properties of processes and the second group consisting of software community interested in providing robust and scalable software systems.

However, building complex business processes on already existent enterprise infrastructure it might be an expensive task. Usually, corporations spent millions integrating heterogeneous applications (otherwise known as Enterprise Application Integration or EAI). Why? Because one of the best ways to generate more profits out of a company is to reduce the costs of doing business.

BPMN helps us to design a business process, but is it able to guarantee the efficiency of the implementation?

As stated in [WG08] the BPMN notation does not have a formal behavioral semantics, which is important in behavioral specification and behavior verification. There are several approaches that try to overcome this inconvenience. The traditional approaches are based upon Petri Nets (e.g. [vdAtHW03, vdAdMW06]), Event-driven Process Chains (e.g. [DvdAtH05]) and Workflow Nets (e.g. [HTvdAW01]).

We propose a new approach for defining behavior semantics for business processes as Agent-based simulations.

How can we validate a BPMN model with respect of its execution? In [Wes07] (page 7) was argued that "business process models are the main artefacts for implementing business processes". In addition in [Wes07] was defined a business process management system as *"a generic software system that is driven by explicit process representations to coordinate the enactment of business processes."* On the other side in [Woo02] was argued that *"an agent is a computer system that is capable of independent action on behalf of its user or owner."*. In this context at least semantically it is obvious that the generic software system in the form of a business process management system can be in fact a multi agent system.

As far as we know the literature concerning this approach is in a starting point, [EKHA07, EHKA07]. The first paper addresses a similar task as this work, while the second one abandon this approach and uses a different one by trying to *normalize* BPMN towards a better mapping to Petri Nets. However, the work in [EKHA07] is in a beginning stage and is not really related to the BPMN validation but towards "to use an intuitive graphical process notation for designing multi-agent systems" (page 104). By contrast our work proposes an use case investigation i.e. a BPMN mapping to an established agent-based platform, Agent-Object Relationship (AOR) introduced in [Wag03, Wag04], with the goal of validation of BPMN based business process models. The main research behind this approach is to investigate if and to what extent BPMN can be executed in an agent-based simulation environment.

There is at least one more reason, why this approach is important and adequate. According to [Wes07] *"a business process consists of a set of activities that are performed in coordination in an organizational and technical environment"* (page 5). Moreover, Service Oriented Architectures (SOA) and business process modeling go hand in hand. Very often the activities modeled within BPMN models are typically implemented as services. The relationship between agent-based simulation environments and services is a critical one since may help understanding organizational models as services.

2 Single Item English Auction

Automated negotiations including here electronic auctions are very well suited to be modeled with BPMN and also with rules, and research focused on defining and on development of protocols and strategies to be used in multi agent systems that are to perform negotiations [BPJ02], [BPJ03], [JFL$^+$01],[Kra97]. The distinction between the negotiation mech-

anism and negotiation strategies is their access modifier. The access modifier refers to the fact that rules describing the negotiation mechanism are *public* and rules defining behavior or strategies are *private*, they belong to agents. Auctions are a form of negotiation mechanism for electronic commerce discussed also in many papers such as [RE05], [WWW01], [WWW02].

English auction also called *Open-outcry auction* is an ascending type of auction. It is an important type of auction discussed in a wide range of papers such as [DASK02], [DRS+05], [BGW06], and we consider that the subject is far from being finished. In Single Item English auction only one item is sold at a time. Bidding is open; all participants bid against each other openly. Each successive bid must be higher than the old one. The seller begins the auction by asking for bids at a low price. Buyers bid against each other by raising the price, until only one willing buyer remains.

2.1 The BPMN Model

The model presented in Figure 1 is based on the protocol defined in [BPJ02], [BPJ03] for multi agent systems and underlines that agent models can be modeled with BPMN. The model consists of three pools: `Seller`, `AuctionHost` and `Bidder`. Usually in an auction participate more bidders, but for simplicity the BPMN diagram (1) shows us only one.

An auction is started by the `Seller` with the `AuctionRequest` activity. This activity creates a process starting message to be consumed by the `AuctionHost`. The message transports both the `sellerID` and `startPrice` of the auction. After that the seller's process waits for the `AuctionHost`'s `AuctionCreationRequestResponse` message.

Bidders have to be *admitted* to the auction i.e. they must do admission requests and wait for confirmations.

So following the same pattern (as in the `AuctionRequest` activity), bidders send `RequestAdmissionToAuction` and receive an *admission confirmation* (`AllowToAuctionMessage`) or a *denial* in the form of a `DenyAccessToAuction` message.

Periodically, bidders post bids (`Bid`) to the `AuctionHost` until they receive `AuctionEndNotificationB` messages.

Inside of the `AuctionHost` pool a similar activity is taking place (`processBid`). It consists of receiving bids and processing them. A bid processing ends up by sending a `BidStatus` message containing the `currentHighestPrice` of the auction, so the agents could raise the bid, in the next cycle.

An auction is finished when the auction time expires and all the participants are notified. After the auction end, the `AuctionHost` has no other tasks to perform. The auction notification messages are different from seller and bidders. A seller auction end notification message (`AuctionEndNottificationS`) contains the `auctionStatus`,

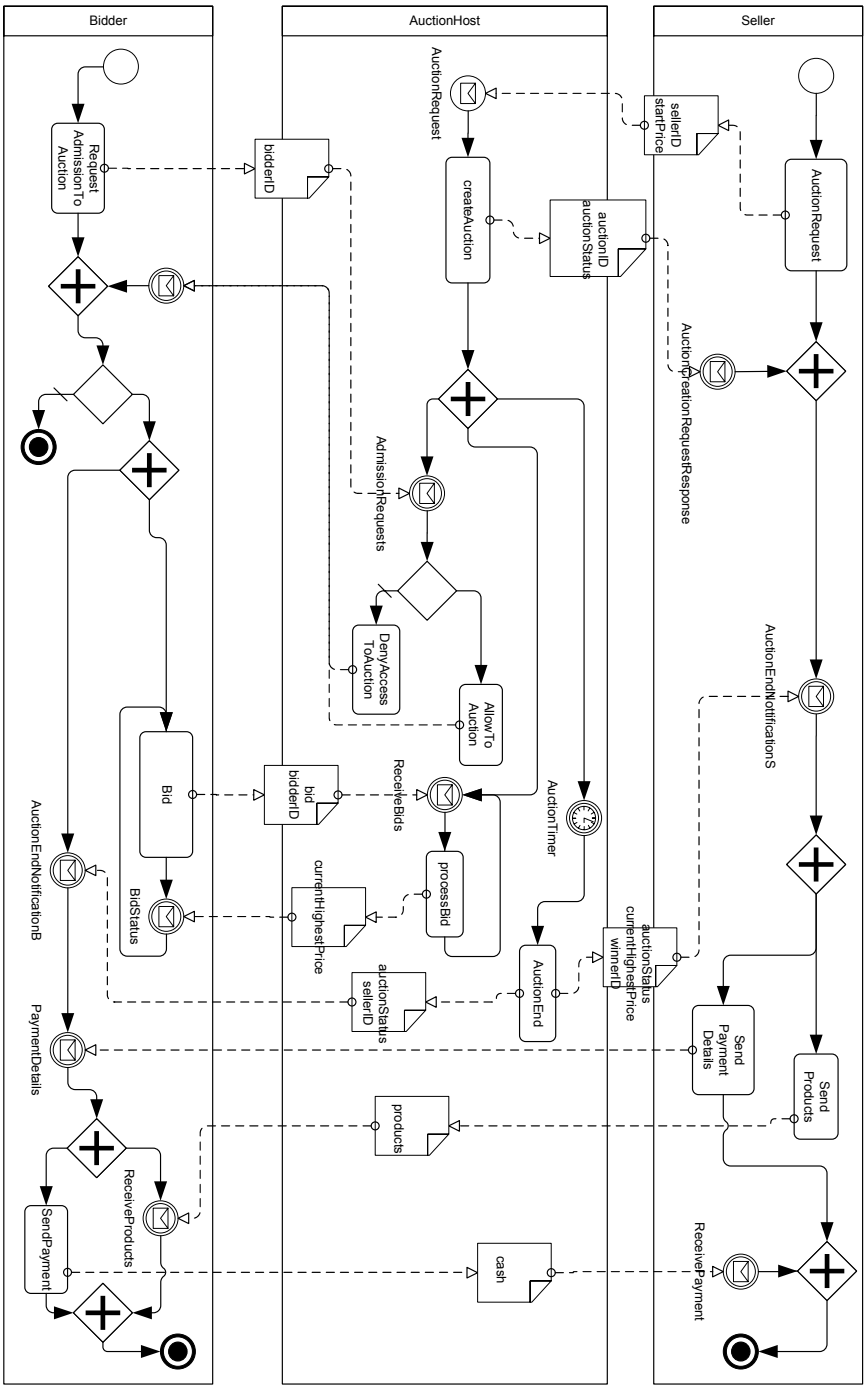


Figure 1: BPMN model of Single Item English Auction

currentHighestPrice and winnerID. The bidders' auction end notification message (AuctionEndNottificationB) content is slightly different since it transports only the auctionStatus and sellerID.

In the seller pool the auction end notification message precedes 2 tasks, namely the SendPaymentDetails and SendProducts. The whole process ends up after the winning bidder receives its products and performs the payment, and the sellers receives the money.

The reader may notice that for modeling English auctions was enough to use the service interaction patterns as they are described in [Wes07] (page 249) i.e. *Send pattern*, *Receive pattern*, *Send and Receive pattern*, *One-to-many send pattern*, *One-from-many receive pattern*.

2.2 Agent-Object-Relationship Model

In [Wag03] was argued that *"semantics of business processes may be more adequately captured if the specific business agents associated with the involved events and actions are explicitly represented in the information systems in addition to passive business objects"* (page 1). In addition, Agent-Object Relationship offers high-level abstraction that facilitate modeling, expressing and actually simulating concepts and interaction between agents. According to [Wag03] an entity is either an agent, an event, an action, a claim, a commitment, or an ordinary object. AOR models mainly reactive agents having the state represented by a knowledge base and its behavior modeled by means of actions and reaction rules. The visual language represents agents by using a modified UML package representation. Yet there is an obvious difference, emphasized by two rectangles that are below the agent name. The left rectangle comprises the self belief properties of an agent, or it's subjective properties. The right rectangle comprises the agent's objective properties, those properties characterized by physical attributes, those that follow causality rules. The body of the agent can encapsulate beliefs about other entities, events, actions and rules. The reaction rules (or Event-Condition-Action Rules) are rules of the form

```
ON <Event> IF <logical-condition> THEN DO <actions>
```

They are represented graphically as a circle with an internal label RR and a rule identifier attached to it. There are two kinds of incoming arrows (condition arrows and event arrows) and two kinds of outgoing arrows (action arrows and postcondition arrows).

2.2.1 The Seller

Single Item English Auctions have one seller as agent. The Seller (see Figure 2) is a reactive agent having the following internal properties: startPrice, endPrice, inventory, cash, winnerID, auctionID and auctionStatus. The agent behavior is modeled by reactions rules as following:

```
RULE "AR"
```

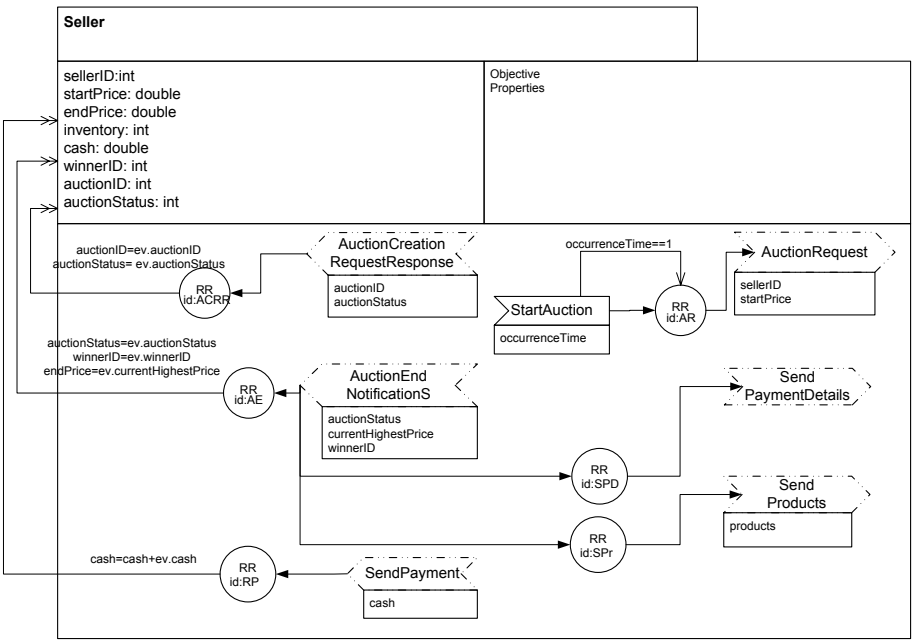


Figure 2: The Seller Agent Model

```
ON StartAuction(?ev)
IF Seller(?Seller) AND ?ev.occurenceTime == 1
THEN
DO AuctionRequest(?Seller.sellerID, ?Seller.startPrice)


RULE "ACRR"
ON AuctionCreationRequestResponse(?ev)
IF Seller(?Seller)
THEN DO (?Seller.sellerID = ?ev.auctionID,
         ?Seller.auctionStatus = ?ev.auctionStatus
         )

RULE "AE"
ON AuctionEndNotificationS(?ev)
IF Seller(?Seller)
THEN DO (?Seller.auctionStatus = ev.auctionStatus,
         ?Seller.winnerID = ?ev.winnerID,
         ?Seller.endPrice = ?ev.currentHighestPrice
         )

RULE "SPD"
ON AuctionEndNotificationS(?ev)
IF Seller(?Seller)
THEN DO  SendPaymentDetails()

RULE "SPr"
ON AuctionEndNotificationS(?ev)
```

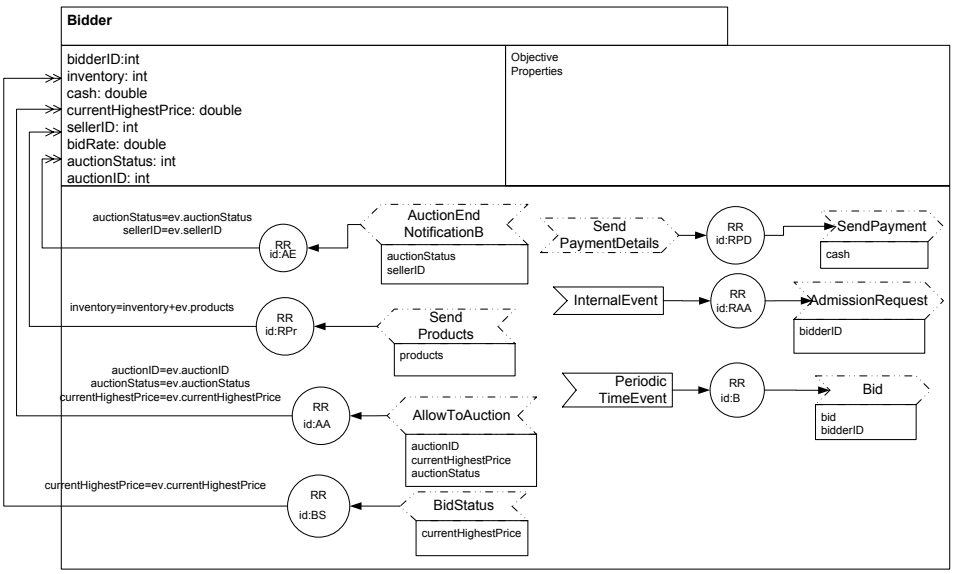


Figure 3: The Bidder Agent Model

```
IF Seller(?Seller)
THEN DO  SendProducts(?Seller.inventory)

RULE "RP"
ON SendPayment(?ev)
IF Seller(?Seller)
THEN DO  ?Seller.cash = ?Seller.cash + ?ev.cash
```

2.2.2 The Bidder

Single Item English Auctions have one or more bidders as agents. The `Bidder` (see Figure 3) is a reactive agent having the following internal properties: `inventory`, `cash`, `currentHighestPrice`, `sellerID`, `bidRate`, `auctionID`, and `auctionStatus`. Its behavior is governed by the following ECA rules:

```
RULE "RAA"
ON InternalEvent(?ev)
IF Bidder(?Bidder)
THEN DO  AdmissionRequest(?Bidder.bidderID)

RULE "AA"
ON AllowToAuction(?ev)
IF Bidder(?Bidder)
THEN DO (?Bidder.auctionID = ?ev.auctionID,
         ?Bidder.auctionStatus =?ev.auctionStatus,
         ?Bidder.currentHighestPrice = ?ev.currentHighestPrice
        )

RULE "B"
```

```
ON PeriodicTimeEvent(?ev)
IF Bidder(?Bidder)
     AND
   ?bid = ?Bidder.currentHighestPrice + ?Bidder.bidRate
THEN DO Bid(?Bidder.bidderID, ?bid)

RULE "BS"
ON BidStatus(?ev)
IF Bidder(?Bidder)
THEN DO (?Bidder.currentHighestPrice = ?ev.currentHighestPrice)

RULE "AE"
ON AuctionEndNotificationB(?ev)
IF Bidder(?Bidder)
THEN DO (?Bidder.auctionStatus =?ev.auctionStatus,
         ?Bidder.sellerID = ?ev.sellerID
         )

RULE "RPr"
ON SendProducts(?ev)
IF Bidder(?Bidder)
THEN DO (?Bidder.inventory = ?Bidder.inventory + ?ev.products)

RULE "RPD"
ON SendPaymentDetails(?ev)
IF Bidder(?Bidder)
THEN DO SendPayment(?Bidder.cash)
```

2.2.3 The Environment

According with [Wag04] the state of an Agent-based Discrete Event Simulation system consists of:

- The simulated time,
- The environment state representing:
  - The non-agentive environment (as a collection of objects) and
  - The external states of all agents (e.g., their physical state, their geographic position etc.),
- The internal agent states (e.g., representing perceptions, beliefs, memory, goals),
- A (possibly empty) list of future events.

In Single Item English Auctions the host is responsible with the bidders and seller coordination therefore it is clear that the `AuctionHost` is suitable to be modeled as the agent environment. The environment model is depicted in Figure 4 and consists of: (i) The Seller agent (see Figure 2), (ii) The Bidder agents (see Figure 3), (iii) The Auction, (iv) The environment behavior rules:

```
RULE "CA"
ON AuctionRequest(?ev)
```

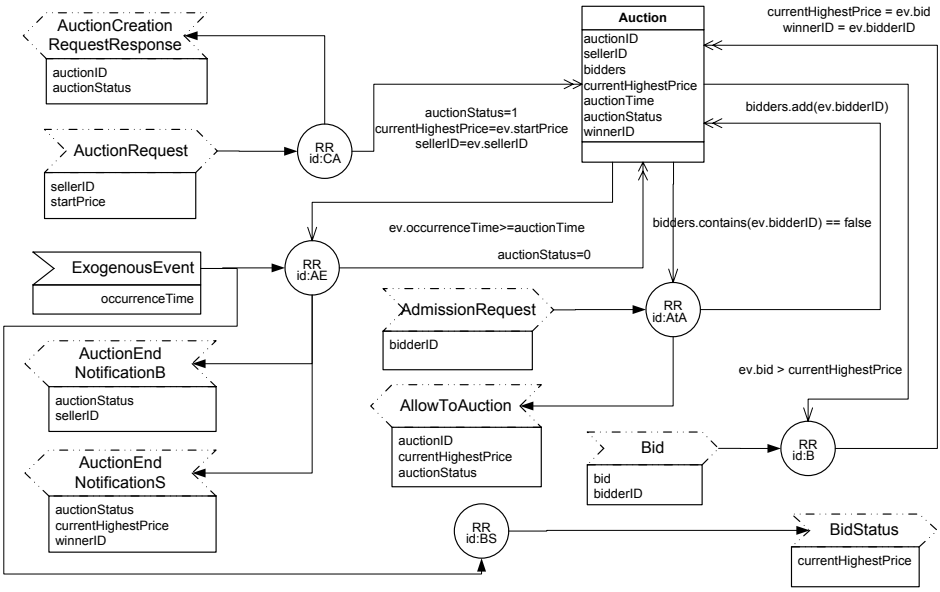


Figure 4: The Environment Model

```
IF Auction(?A)
THEN DO (?A.auctionStatus = 1,
         ?A.currentHighestPrice = ?ev.startPrice,
         ?A.sellerID = ?ev.sellerID
         )
     AuctionCreationRequestResponse(?A.auctionID,
                                    ?A.auctionStatus)

RULE "AtA"
ON AdmissionRequest(?ev)
IF Auction(?A) AND NOT(?A.bidders.contains(?ev.bidderID))
THEN DO (?A.bidders.add(?ev.bidderID))
        AllowToAuction(?A.auctionID,
                       ?A.currentHighestPrice,
                       ?A.auctionStatus)

RULE "B"
ON Bid(?ev)
IF Auction(?A) AND ?ev.bid > ?A.currentHighestPrice
THEN DO (?A.currentHighestPrice = ?ev.bid
         ?A.winnerID = ?ev.bidderID
         )

RULE "BS"
ON ExogenousEvent(?ev)
IF Auction(?A)
THEN DO BidStatus(?A.currentHighestPrice)

RULE "AE"
```

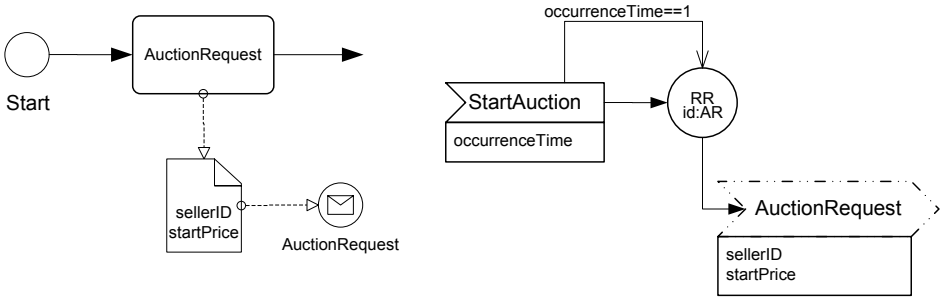


Figure 5: Mapping Activities to Reaction Rules

```
ON ExogenousEvent(?ev)
IF Auction(?A) AND ?ev.occurrenceTime >= ?A.auctionTime
THEN DO (?A.auctionStatus = 0)
        AuctionEndNotificationB(?A.bidders,
                                ?A.auctionStatus,
                                ?A.sellerID)
        AuctionEndNotificationS(?A.auctionStatus,
                                ?A.currentHighestPrice,
                                ?A.winnerID)
```

3 Mapping Auction BPMN Model to AOR Model

The validation of a BPMN process model with the help of multi agent system simulations requires a mapping between BPMN models and multi agent systems models. Some steps towards were already done in our previous work [PGW09] where the Agent-Object Relationship simulation has been modeled as a BPMN model.

In an agent based simulation the agents are the main entities of the system. The BPMN 1.1 specification [OMG08] states that ”a pool represents a participant in a process” (page 43). It may also contain lanes to partition activities. Lanes are sub-partitions of pools. For the simple case a pool should naturally be mapped to an agent. It is the case for `Seller` and `Bidder`.

The agent properties are artifacts for storing data The BPMN models offer only `Data Object` and `Annotations` as artifacts. The mapping investigates how can be used such artifacts to model agents properties. The central point of agents behavior is declaratively modeled by means of agent reaction rules. By contrast BPMN models express flows by means of events, activities and gateways.

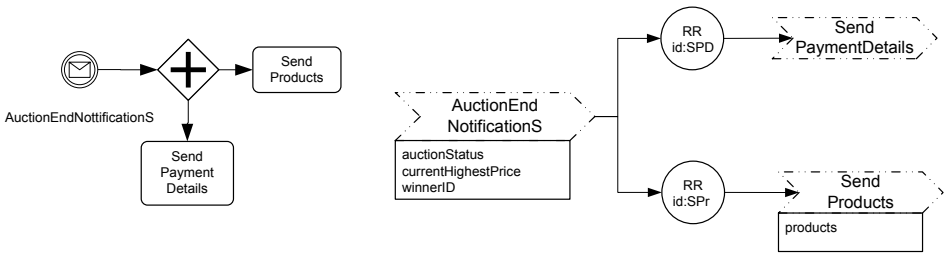


Figure 6: Mapping Gateways to Reaction Rules

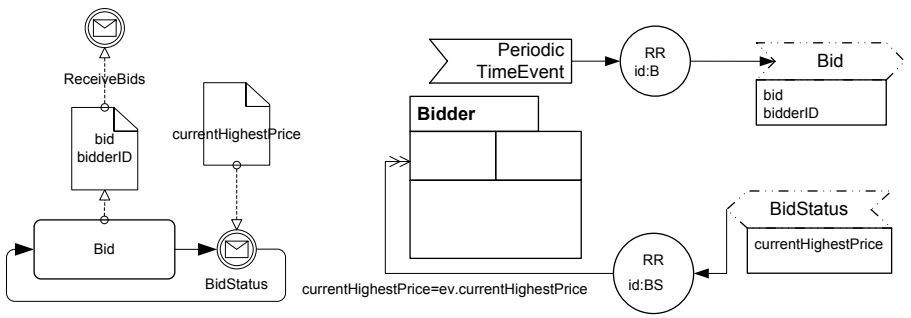


Figure 7: Mapping loops

3.1 Mapping the Seller and the Bidders

The `Seller` properties are derived from BPMN objects available in the `Seller` lane. For example since `sellerID` and `startPrice` are sent by the Seller to AuctionHost they are suitable candidates as agent properties (see Figure 2). This is similar with `endPrice`, `inventory`, `cash`, `winnerID`, `auctionID`, and `auctionStatus`.

Much more complex issues are raised when we map the flow to agent reaction rules. Suitable candidates to identify agent rules are BPMN activities and gateways. It is obvious to assume that any BPMN activity is started by an event and ends with an event, even these events might not be explicitly modeled in the BPMN model.

For example the `AuctionRequest` activity is the origin of the `Seller` agent reaction rule `"AR"` (see Section 2.2.1). The Figure 5 show us the intuitive mapping.

Another significant mapping example involves BPMN gateways. The Figure 6 illustrate this mapping. We can see the need of a much better specification of `AuctionEndNotifiationS` (such as specifying its properties) to be encoded in a proper AOR event.

The `Bidder` is very similar with the `Seller` and follows the same mapping principles. Recall, in the auction can be many bidders but all of them maps to the same agent type.

The `Bidder` has a loop construction that comprises the `Bid` activity and the incoming

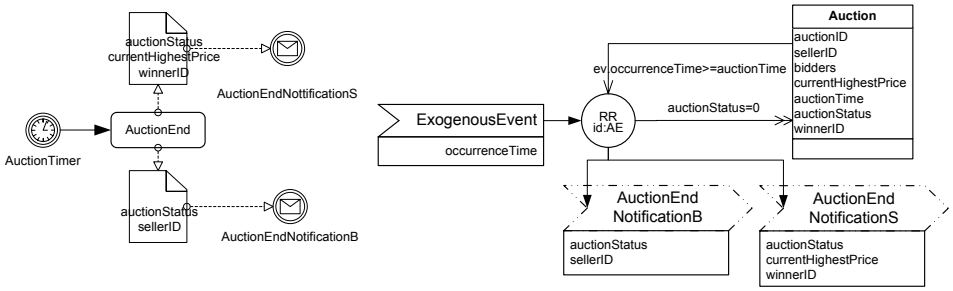


Figure 8: Mapping timers

message `BidStatus`. The mapping is depicted in Figure 7. It must be underlined the fact that in AOR simulations the `Bidder` loop as well as the `AuctionHost` loop can not exist without each other. The `Bidder` posts bids periodically and receives `BidStatus` messages periodically because these are sent periodically by the `AuctionHost`. Now the `AuctionHost` sends periodically because itself receives bids periodically. So the two loops are maintaining each other.

3.2 Mapping the AuctionHost

Any agent-based simulation system has an environment where the agents perform. The main question is how to distinguish between pools that have to be mapped to agents and pools that have to be mapped to environment. In our use case `AuctionHost` is suitable to be modeled as the environment. However since the mapping should be done automatically, we have to underline another question for further research: Is the graphical syntactical representation in the form of a BPMN model enough so that a computer program be able to make the right choice? If not what would be the necessary annotations that have to be added to the BPMN models in order to facilitate this distinction?

The two loops from the `AuctionHost` and `Bidder` are modeled in AOR model by means of periodic time events, as already explained in Section 3.1.

Mapping `AuctionHost` introduces two time-related problems: BPMN timers, and timing for events. Time is very important to multi agent systems. Usually agent based systems are discrete with respect to time. However is not completely clear how BPMN deals with time events. In AOR, which is an agent based discrete event simulation, system timers can be modeled either as periodic time events (time events internal to agents), or as exogenous events (when such timers refer to the environment). In our use case the timer event from the `AuctionHost` is mapped into an `ExogenousEvent` (Figure 8).

3.3 A Discussion on Patterns

Recall that the Single Item English Auction use the following patterns: *Send pattern*, *Receive pattern*, *Send and Receive pattern*, *One-to-many send pattern*, and *One-from-many receive pattern*.

We will discuss only the *Send and Receive pattern* since is the correlated combination of *Send pattern*, *Receive pattern*. This pattern involves in the both pools that are part of the conversation an activity and a message. However in different order. For the auction creation situation, the Seller performs activity `AuctionRequest`. This activity is followed by the `AuctionCreationRequestResponse` incoming message from the `AuctionHost`. Opposed the `AuctionHost` first receives the `AuctionRequest` incoming message and then performs activity `createAuction`. Notice that in both cases activities are started by an event. In the AOR model this pattern comprises 3 rules. Two rules (`AR`, `ACRR`) belonging to the seller, and one rule in the environment (`CA`). The `environment` rule CA is started by the incoming message `AuctionRequest`. The rule ends up by sending the message `AuctionCreationRequestResponse` (see Figure 4). While in the BMPN model the activity `AuctionRequest` is preceded by a statring event in the AOR model we have the rule `AR` belonging to the `Seller` triggered by the `StartAuction` event. This rule actually performs the action of posting the message `AuctionRequest`. The second rule belonging to the `Seller` that completes the pattern is `ACRR`. This one is triggered by the incoming `AuctionCreationRequestResponse` message from the environment. Notice that the order of the rules is `Seller:AR -> environment:CA -> Seller:ACRR`. While this order in the BPMN model is imposed by the flow arrows, in the agent based simulation this is imposed by the order of the messages.

The *One-to-many send pattern* and *One-from-many receive pattern* mapping process is similar. The difference occurs by the fact that actually several messages are sent or received from/to different agents.

4 Conclusion and Future Work

The paper here underlined facts and open issues about execution and validation of BPMN process models as agent based simulations. It explained why such an approach is suitable and which are the gains and identified some open issues. The open questions identified such as (a) *"How to distinguish between different categories of pools i.e. to be mapped to agents or to environment?*", (b) *"What are properties/metadata that help the distinction?"*, or (c) *"How complete is BPMN such that a computer program be able to make the right choice?"*, as well as mapping other BPMN identified patterns to agent based constructs and, in addition, the timing problems are starting points for future research.

References

[BGW06] Costin Badica, Adrian Giurca, and Gerd Wagner. Using Rules and R2ML for Modeling Negotiation Mechanisms in E-Commerce Agent Systems. In Dirk Draheim and Gerald Weber, editors, *Proceedings of the 2nd International Conference on Trends in Enterprise Application Architecture, TEAA2006*, volume 4473 of *Lecture Notes in Computer Science*, pages 84–99. Springer, November 2006.

[BPJ02] Claudio Bartolini, Chris Preist, and Nicholas R. Jennings. A Generic Framework for Automated Negotiation. Technical report, January 2002.

[BPJ03] Claudio Bartolini, Chris Preist, and Nicholas R. Jennings. Architecting for Reuse: A Software Framework for Automated Negotiation. In *Proceedings of the 3nd International Workshop Agent-Oriented Software Engineering, Bologna, Italy, AOSE02*, volume 2585 of *Lecture Notes in Computer Science*, pages 88–100. Springer Berlin / Heidelberg, 2003.

[DASK02] Esther David, Rina Azoulay-Schwartz, and Sarit Kraus. An English Auction Protocol for Multi-Attribute Items. In *Proceedings of the Workshop on Workshop on Agent Mediated Electronic Commerce on Agent-Mediated Electronic Commerce IV, Designing Mechanisms and Systems*, volume 2531 of *Lecture Notes in Computer Science*, pages 52–68. Springer-Verlag London, UK, 2002.

[DRS+05] Esther David, Alex Rogers, Jeremy Schiff, Sarit Kraus, and Nicholas R. Jennings. Optimal Design Of English Auctions With Discrete Bid Levels. In *Proceedings of the 6th ACM conference on Electronic commerce, Vancouver, BC, Canada*, pages 98–107. ACM New York, NY, USA, 2005.

[DvdAtH05] Marlon Dumas, Wil M. van der Aalst, and Arthur H. ter Hofstede. *Process-Aware Information Systems: Bridging People and Software Through Process Technology*. Wiley, 2005.

[EHKA07] Holger Endert, Benjamin Hirsch, Tobias Küster, and Sahin Albayrak. Towards a Mapping from BPMN to Agents. In Aalst Wil, Hofstede Arthur, and Weske Mathias, editors, *Proceedings of Service-Oriented Computing: Agents, Semantics, and Engineering 2007*, volume 4504 of *Lecture Notes in Computer Science*, pages 92–106. Springer Berlin / Heidelberg, 2007.

[EKHA07] Holger Endert, Tobias Küster, Benjamin Hirsch, and Sahin Albayrak. Mapping BPMN to Agents: An Analysis. In *Proceedings of Workshop MALLOW-AWESOME'007 Durham, September 6th7th, 2007*, pages 43–58, 2007. http://awesome007.disi.unige.it/EndertEtAl-AWESOME007.pdf.

[HTvdAW01] Verbeek H, Basten T, and van der Aalst W. Diagnosing workflow processes using Woflan. *The Computer Journal*, 44(4):246–279, 2001. `http://comjnl.oxfordjournals.org/cgi/reprint/44/4/246`.

[JFL+01] Nicholas R. Jennings, Peyman Faratin, A. R. Lomuscio, Simon Parsons, Michael Wooldridge, and Carles Sierra. Automated Negotiation: Prospects, Methods and Challenges. *Group Decision and Negotiation*, 10(2):199–215, March 2001.

[Kra97] Sarit Kraus. Negotiation and cooperation in multi-agent environments. *Special issue on economic principles of multi-agent systems*, 94(1-2):79–97, 1997.

[OMG08] OMG. Business Process Modeling Notation, V1.1. `http://www.omg.org/spec/BPMN/1.1/PDF`, January 2008.

[PGW09] Emilian Pascalau, Adrian Giurca, and Gerd Wagner. *Handbook of Research on Emerging Rule-Based Languages and Technologies: Open Solutions and Approaches*, chapter The Agent Object Relationship Simulation as a Business Process. IGI Publishing, 2009. accepted.

[RE05] Daniel Rolli and Andreas Eberhart. An Auction Reference Model for Describing and Running Auctions. *Wirtschaftsinformatik 2005*, pages 289–308, 2005.

[vdAdMW06] Wil M. P. van der Aalst, A. K. Alves de Medeiros, and A. J. M. M. Weijters. Process Equivalence: Comparing Two Process Models Based on Observed Behavior. In *Proceedings of Business Process Management Conference*, volume 4102 of *Lecture Notes in Computer Science*, pages 129–144. Springer Berlin / Heidelberg, 2006. `http://is.tm.tue.nl/staff/aweijters/ComparePM.pdf`, 21 July 2008.

[vdAtHW03] Wil M. P. van der Aalst, Arthur H. M. ter Hofstede, and Mathias Weske. Business Process Management: A Survey. In Aalst Wil, Hofstede Arthur, and Weske Mathias, editors, *Proceedings of Business Process Management Conference*, volume 2678 of *Lecture Notes in Computer Science*, pages 1–12. Springer Berlin / Heidelberg, 2003. `http://is.tm.tue.nl/staff/wvdaalst/publications/p183.pdf`, 21 July 2008.

[Wag03] Gerd Wagner. The Agent-Object-Relationship Meta-Model: Towards a Unified View of State and Behavior. *Information Systems*, 28(5):475–504, 2003.

[Wag04] Gerd Wagner. *Agent-Oriented Information Systems*, volume 3030 of *LNAI*, chapter AOR Modelling and Simulation – Towards a General Architecture for Agent-Based Discrete Event Simulation, pages 174–188. Springer-Verlag, 2004.

[Wes07] Mathias Weske. *Business Process Management: Concepts, Languages, Architectures* . Springer-Verlag Berlin Heidelberg, 2007.

[WG08] Peter Y.H. Wong and Jeremy Gibbons. A Process Semantics for BPMN. In *Proceedings of 10th International Conference on Formal Engineering Methods.*, LNCS, October 2008. To appear. Extended version available at `http://web.comlab.ox.ac.uk/oucl/work/peter.wong/pub/bpmnsem.pdf`.

[Woo02] Michael Wooldridge. *Introduction to MultiAgent Systems*. John Wiley & Sons, 2002.

[WWW01] Peter R. Wurman, Michael P. Wellman, and William E. Walsh. A Parametrization of the Auction Design Space. *Games and Economic Behavior*, 35:304–338, 2001.

[WWW02] Peter R. Wurman, Michael P. Wellman, and William E. Walsh. Specifying Rules for Electronic Auctions. *AI Magazine*, 23:15–23, 2002.

On Application of Structural Decomposition for Process Model Abstraction

Artem Polyvyanyy, Sergey Smirnov, and Mathias Weske
Business Process Technology Group
Hasso Plattner Institute at the University of Potsdam
D-14482 Potsdam, Germany
{Artem.Polyvyanyy,Sergey.Smirnov,Mathias.Weske}@hpi.uni-potsdam.de

Abstract: Real world business process models may consist of hundreds of elements and have sophisticated structure. Although there are tasks where such models are valuable and appreciated, in general complexity has a negative influence on model comprehension and analysis. Thus, means for managing the complexity of process models are needed. One approach is abstraction of business process models—creation of a process model which preserves the main features of the initial elaborate process model, but leaves out insignificant details. In this paper we study the structural aspects of process model abstraction and introduce an abstraction approach based on process structure trees (PST). The developed approach assures that the abstracted process model preserves the ordering constraints of the initial model. It surpasses pattern-based process model abstraction approaches, allowing to handle graph-structured process models of arbitrary structure. We also provide an evaluation of the proposed approach.

1 Introduction

Business process management is an important approach to represent and improve the way companies work in dynamic and competitive settings [HC94]. In most cases one model for one business process is not enough, since different types of business process analysis require different perspectives of one process. Certain analysis tasks imply exhaustive process description, while others need only a process overview. Therefore, there is a demand for several models of the same process, but with different levels of abstraction. Meanwhile, without a formal relation between these models, consistency between them cannot be guaranteed. Model maintenance becomes a pricey and error-prone task.

To cope with this problem, modeling notations, like Business Process Modeling Notation (BPMN) [BPM08] or Event-driven Process Chains (EPC) [KNS92, STA05], allow hierarchical model structuring. Such structuring gives a user a possibility to organize model details putting them to the appropriate level in the model hierarchy. However, hierarchical structuring requires a user to decide to which abstraction level an element should be related.

Business process model abstraction addresses the outlined problem. Abstraction generalizes the model, leaving out insignificant details. Under the assumption that a company already possesses a repository of detailed process models, abstraction derives a set of

coarse-granular process representations.

There exists a number of approaches which address the abstraction task. However, they have certain limitations. One common drawback is handling of block-structured process models only [EG08, PSW08b]. This limitation becomes crucial in practical tasks where process models have arbitrary structure. Other approaches are generic, but cannot guarantee preservation of the ordering constraints of the initial model [BRB07, GA07].

In this paper we propose an approach for business process model abstraction based on the construction and analysis of process structure trees (PST) [VVL07]. The proposed approach allows to handle arbitrary graph-structured process models. Furthermore, it preserves the ordering constraints of the initial model. The developed process model abstraction focuses on the structural aspects of the abstraction, rather than on the semantics of the model. This means that the method tells how model elements can be correctly abstracted, but does not tell which elements should be abstracted. We evaluate efficiency of the presented technique, i.e., its capability to leave out process details gradually.

This paper has the following structure. Section 2 explains the concept of process model abstraction. In section 3 we identify requirements a business process model abstraction should meet and formulate key assumptions of this work. Section 4 describes a method of PST construction and the abstraction mechanism. An evaluation of the proposed abstraction mechanism concludes the section. In section 5 an outline of the related work is given. Finally, the conclusions and the future work are provided.

2 Business Process Model Abstraction

Process model abstraction is generalization of a model which leaves out insignificant process details in order to reduce complexity of the model and retain information relevant for a particular purpose. Therefore, information loss is the fundamental property of abstraction and is its desirable outcome. When business process model abstraction is discussed, one can imagine various details to be omitted: activities, events, or even whole execution paths. The choice of subjects to be abstracted is usually dictated by user needs. In [PSW08a] we identified several use cases for process model abstraction. For each use case it was shown which elements of the model are subject for abstraction.

When business users talk about process model abstraction, they often imply abstraction of activities, requesting a transition from low level steps to high level tasks. Several research projects in this area (see [BRB07, EG08]), as well as our research experience [PSW08a, PSW08b], prove that activities are often in the focus of abstraction. Therefore, we use activities as the abstraction subject.

In this paper we introduce a simplified process modeling notation, rather than using notations like EPC or BPMN. We aim at choosing the simplest process model formalism that enables our task. In the model we consider activities, which are abstraction subjects, and gateways, which define control flow logic. We do not address events. Obviously, events are the core elements of many modeling notations, like EPC or BPMN. Nevertheless, without loss of generality we neglect events in the developed abstraction approach. This design de-

cision allows us to adapt the approach to various modeling notations, where semantics of events may vary from one notation to another. Accordingly, we define a business process model.

Definition 1 $(N, E, type)$ is a *business process model* where:

- $N = N_A \cup N_G$ is a set of nodes, where $N_A \neq \emptyset$ is a set of activities and N_G is a set of gateways; the sets are disjoint
- $E \subseteq N \times N$ is a set of directed edges between nodes representing control flow
- (N, E) is a connected graph
- every activity has at most one incoming and at most one outgoing edge
- there is at least one activity which has no incoming edges—a start activity, and at least one activity which has no outgoing edges—an end activity
- $type : N_G \rightarrow \{and, xor, or\}$ is a function that assigns to each gateway a control flow construct
- every gateway is either a split or a join; splits have exactly one incoming edge and at least two outgoing; joins have at least two incoming edges and exactly one outgoing.

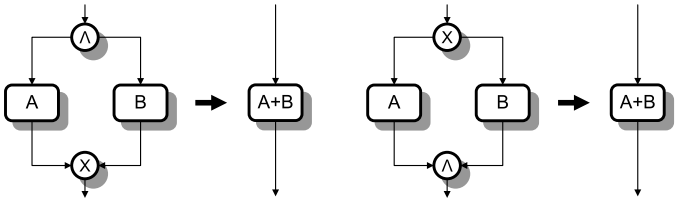
According to this definition an activity has no more than one incoming and one outgoing edge. Some modeling notations impose this restriction (e.g., EPC), while others (e.g., BPMN) allow activities to have multiple incoming/outgoing edges. However, by introducing gateways, it is always possible to transform the model in the way that every activity has exactly one incoming edge and one outgoing edge.

We propose to implement process model abstraction in several steps. In every *abstraction step* one activity from a set of insignificant activities is processed. The output of the current abstraction step is a new process model, which is the input for the next abstraction step. Abstraction evolves until every insignificant activity is handled.

Every abstraction step reduces the number of model elements. We distinguish two methods to abstract an element: *aggregation* or *elimination*. Aggregation replaces several elements with one aggregating element. The properties of an aggregating element are derived from the properties of the elements which are aggregated. Therefore, aggregation preserves information about the abstraction subject in a model. Elimination, on the other hand, does not preserve information. Elimination simply omits the element in the abstracted process model.

3 Assumptions and Requirements

In this section we discuss the underlying assumptions and the requirements of the process model abstraction approach to be proposed. We argue why the formulated assumptions are



(a) Abstraction of a model fragment with lack of synchronization

(b) Abstraction of a model fragment containing a deadlock

Figure 1: Abstraction of process models that are not sound

important. Following in this section we introduce two requirements: *order preservation* and *smoothness*.

3.1 Assumptions on Process Models

Business process modeling is a creative task that allows humans to represent process knowledge in a formal way. However, modeling practices permit people to end up with wrong models, i.e., models that are not sound or not safe. Safeness assures that no activity is enabled more than once at one point in time. The idea of soundness is to make sure that all tasks can participate in a process instance, every process instance eventually terminates, and when it terminates there is no running activities in a process [Wes07]. An abstraction step performs generalization of a *process model fragment*—a connected subgraph of a graph representing a process model. As a result, the problems existing in the model might be unintentionally concealed. Thus, the abstracted process model might become correct, while the original model was not. In Figure 1(a) an example of a process model fragment with multi-merge control flow pattern [RHAM06] is shown (i.e., model is not safe). After fragment generalization the problem is hidden. Similarly, an abstraction step might generalize a process model with a deadlock to a sound one (see Figure 1(b)). The given examples clearly illustrate that abstraction can substantially change the process logic, not only because insignificant fragments are generalized, but also because modeling errors can get hidden. To avoid confusing situations we allow abstraction of only correct, i.e., sound process models.

The explicit assumption on process soundness further imply that every process model can have only one distinguished start node and only one distinguished end node. This observation is important for the future discussion of the abstraction approach, since it is based on the decomposition algorithm that assumes models to have exactly one start node and one end node.

3.2 Requirements

The developed technique of process model abstraction should be order preserving and smooth. We realize process model abstraction as a sequential application of abstraction steps. Both of the stated requirements can be discussed within a scope of a single abstraction step.

Essentially, process model abstraction should preserve the ordering constraints of an initial model. For instance, if an original process model specifies to execute either activity A or B, it should not be the case that in the abstracted model two activities appear in sequence. Assume that activity A should be abstracted in the current abstraction step. Let f_A be a process fragment affected by this abstraction step (f_A contains A). As a result of abstraction, f_A gets replaced by activity F. If activity B also belongs to f_A, information about the ordering constraints between activities A and B is lost. However, the order preserving abstraction should assure that for any pair of activities not in f_A, e.g., activities C and D, the ordering constraints between them are preserved. Furthermore, the order preserving abstraction must guarantee that the ordering constraints between any activity not in f_A, e.g., activity E, and any activity in f_A, activities A or B in our example, are the same as between activities E and F. In the end, the order preserving abstraction secures the overall process logic to be reflected in abstracted model.

The fundamental characteristic of abstraction is that it leads to information loss. If a sequence of activities is abstracted to one activity, loss of information about generalized activities and their relations is intended. Abstraction technique should provide effective mechanisms to achieve (and not to under- or overachieve) the desired level of information loss and available abstraction steps might allow or not allow this. In general, the "smaller" the abstraction step (the less process information it generalizes), the better it suits for achievement of a precise model information level. In order to quantitatively measure the precision of the abstraction technique we introduce a notion of *abstraction smoothness*. Abstraction smoothness quantitatively estimates the information loss produced by one abstraction step. In case of an abstraction which is based on activity generalization, the abstraction smoothness reflects the difference between the number of activities in the process model before and after one abstraction step. The less activities are generalized in a single step, the smoother is the abstraction. From the user perspective it is important to have a smooth abstraction which allows reaching required model information level and forbids undesired side effects. The smoothness of application of an abstraction technique can be obtained as the mean of smoothness for every abstraction step.

4 Abstraction Approach

In this section we present the algorithm of PST decomposition of a process model. Afterwards, we show how PST can be used for the purpose of process abstraction. Finally, we evaluate the approach.

4.1 Process Model Decomposition

As we have argued, the abstraction algorithm transforms a process model stepwise, affecting one model fragment at a time. We propose to use a well established algorithm to derive fragments from a process model. The algorithm enables decomposition of a process model into special kind of process fragments called *canonical single entry single exit (SESE) fragments*. Informally, a SESE fragment is a fragment which has exactly one incoming edge and exactly one outgoing edge. From the perspective of the abstraction task SESE fragments are very handy: structurally every SESE fragment can be replaced with one aggregating activity. The semantics of this new aggregating activity corresponds to the semantics of the replaced process fragment.

To formalize the concept of canonical SESE fragments, auxiliary concepts have to be introduced. We assume a process model to have one start activity and one end activity. This assumption aligns with the discussion of the assumptions and requirements in section 3. We say that node x *dominates* node y in a process model graph if every path from start activity to y includes x. Node x *postdominates* node y in a process model graph if every path from y to end activity includes x. The concepts of dominance and postdominance can be transferred to edges. Thus, SESE fragment and canonical SESE fragment can be defined in the following way.

Definition 2 *A SESE fragment in graph G is a process model fragment defined by an ordered edge pair (a, b) of distinct control flow edges a and b, where:*

1. *a dominates b,*
2. *b postdominates a,*
3. *every cycle containing a also contains b and vice versa.*

Edge c belongs to the SESE fragment defined by (a, b), if c postdominates a and c dominates b. We say that node n belongs to the node set of a SESE fragment if all the incident edges of this node belong to the fragment.

SESE fragment defined by (a, b) is canonical if b dominates b' for any SESE fragment defined by (a, b') and a postdominates a' for any SESE fragment defined by (a', b).

Definition 1 distinguishes two node types: activities and gateways. In the decomposition algorithm we do not make use of it, since activities and gateways are treated simply as nodes of a graph. In Figure 2 an example of a process model is shown. Canonical SESE fragments are marked with a dashed line; those which contain more than one activity are named X, Y, and Z.

We define two types of relations between canonical SESE fragments: parent-child and predecessor-successor. From Definition 2 it follows that the node sets of two canonical SESE fragments are either disjoint or one contains the other. That is why a parent-child relation can be introduced for canonical SESE fragments. If the node set of SESE fragment s_1 is the subset of the node set of SESE fragment s_2, then s_1 is the *child* of s_2 and s_2 is

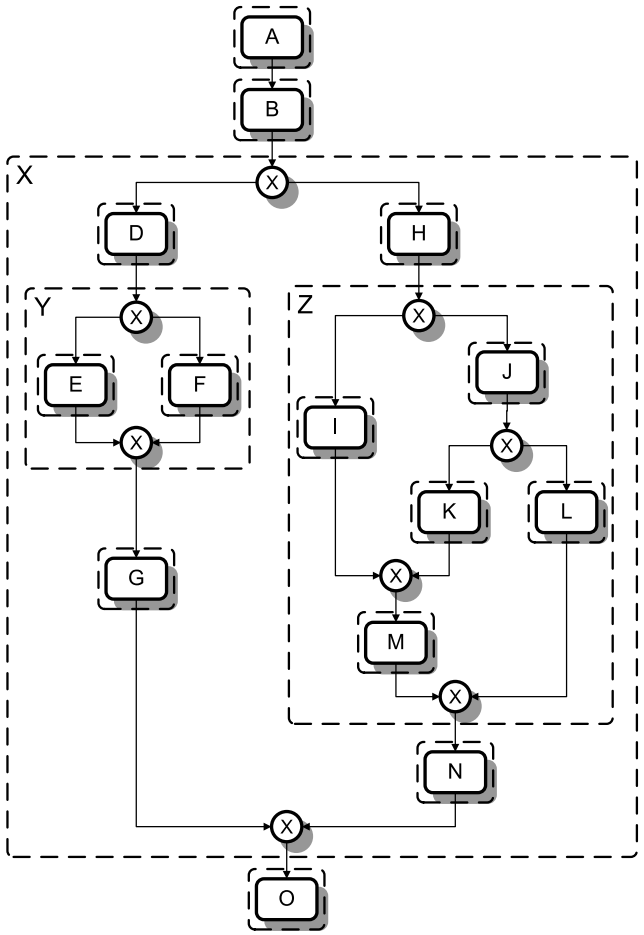


Figure 2: Process model decomposed into canonical SESE fragments

the *parent* of s_1. If s_1 is the child of s_2 and there is no s_3, such that s_3 is the child of s_2 and s_3 is the parent of s_1, s_1 is the *direct child* of s_2. Canonical SESE fragments can be organized into a hierarchy according to the parent-child relation. The hierarchy is represented with a directed tree called *process structure tree*. The tree nodes represent canonical SESE fragments. Let tree nodes n_1 and n_2 correspond to SESE fragments s_1 and s_2 respectively. An edge leads from tree node n_1 to n_2 if SESE fragment s_1 is the direct parent of s_2. Figure 3 presents the PST for the process model from Figure 2. Node R is the root and corresponds to the whole process model. Canonical SESE fragment K is the direct child of Z, therefore, there is a directed edge between the corresponding nodes in the tree.

Two canonical SESE fragment can be in the predecessor-successor relation. We say that s_1 precedes s_2 (and s_2 succeeds s_1) if the outgoing edge of s_1 is the incoming edge of s_2.

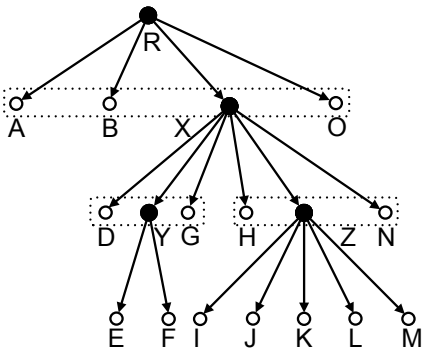


Figure 3: PST corresponding to the process model from Figure 2

One can observe that only the sibling nodes can be in the predecessor-successor relation. In the PST we visualize sequences of nodes which are in predecessor-successor relation using dotted border rectangles. For instance, canonical SESE fragments H, Z, and N are put in the rectangle.

4.2 Abstraction Mechanism

Once a process model is decomposed into canonical SESE fragments and the corresponding PST is built, abstraction can be applied. In the abstraction approach we rely solely on aggregation of activities. This means that in every abstraction step two or more activities are aggregated. Let A be an activity to be abstracted in the current step. We aim to find the minimal canonical SESE fragment $sese_{min}$, containing A and at least one more activity in order to perform generalization. Every activity has one incoming edge and one outgoing edge. Thus, it constitutes a canonical SESE fragment, represented by a leaf in the PST. Hence, we traverse all the leaves in the PST and find the one containing A. Let us call it $sese_A$. The discovered fragment contains only A and is of no use for the abstraction; $sese_A$ cannot be used as $sese_{min}$. There are two options for the selection of $sese_{min}$:

1. There is a canonical SESE fragment $sese_{A'}$ which is in the predecessor-successor relation with $sese_A$. Then $sese_{min}$ is a SESE fragment with the incoming edge of the predecessor and the outgoing edge of the successor in the pair $sese_A$, $sese_{A'}$.
2. If there is no canonical SESE fragment, which is in the predecessor-successor relation with $sese_A$, then $sese_{min}$ is a SESE fragment which is the parent of $sese_A$.

Once $sese_{min}$ is identified, it is replaced with one aggregating activity in the process model. The incoming edge of the aggregating activity is the incoming edge of $sese_{min}$, while its outgoing edge is the outgoing edge of $sese_{min}$.

After the abstraction mechanism was discussed, we would like to summarize its properties. The developed process model abstraction preserves the ordering constraints in the sense

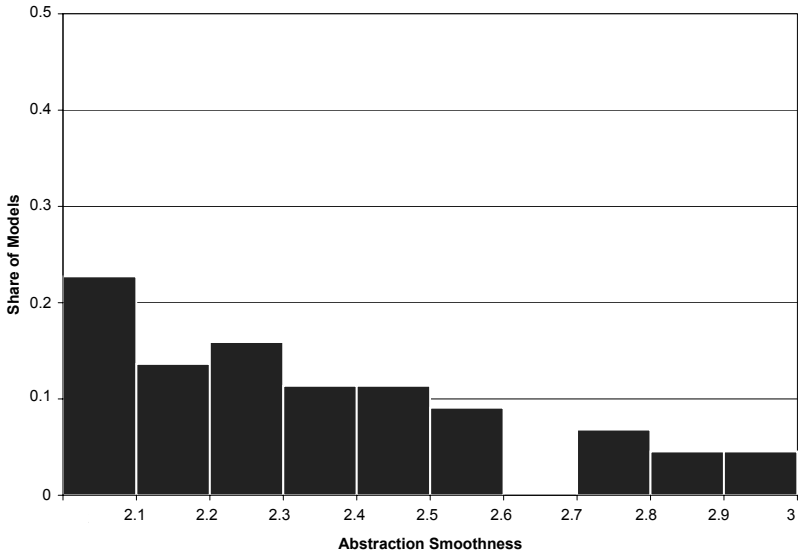
described in section 3. This can be shown with the following reasoning. Assume we abstract activity E contained in canonical SESE fragment Y (see Figure 2). According to the algorithm, SESE fragment Y is replaced with one activity EF. Information about the control flow within Y is lost. Ordering constraints between any two activities, which do not belong to Y (e.g., D and H), are preserved. This holds, since Y has exactly one incoming and one outgoing edge and all the transformations are localized within fragment Y. Control flow relations between any activity, which does not belong to Y, (e.g., D) and the aggregating activity EF are the same as between D and every activity which is contained in Y. Again, this is true, since Y has exactly one incoming and one outgoing edge.

4.3 Smoothness Evaluation

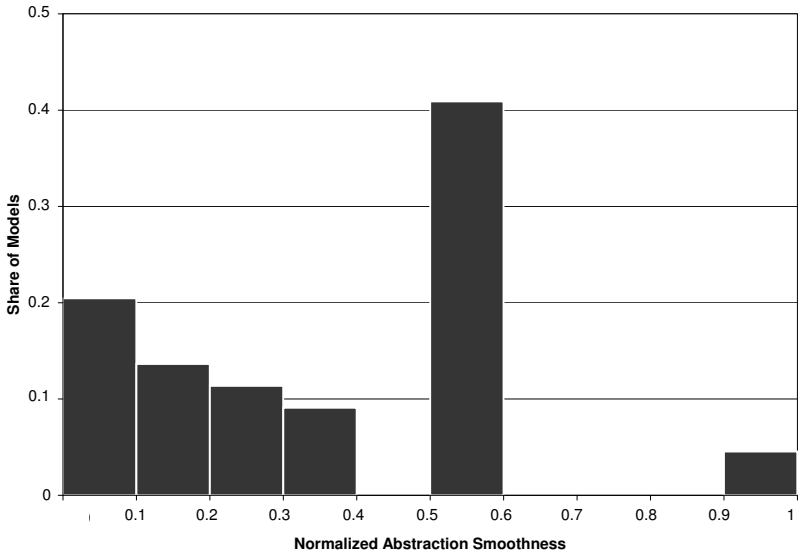
The abstraction smoothness of the presented approach can be measured as the average number of nodes aggregated in one abstraction step. Theoretically the approach demonstrates the best smoothness if in every abstraction step only two activities are aggregated. However, this optimal condition may not hold: process model structure may lead to aggregation of more than two nodes at once. For instance, if activity K has to be abstracted (see Figure 2), activities I, J, L, and M are affected as well, since they are contained in canonical SESE fragment Z, and Z is $sese_{min}$ for K. To evaluate the smoothness of the developed abstraction approach we conduct an experiment and statistically analyze the results. Initially, we select a collection of process models to be abstracted. Afterwards, each model is abstracted to one activity. While models are being abstracted, information about the smoothness is collected.

In the experiment we use a set of 50 real world process models, capturing business processes of a large German health insurance company. The models vary in size from 50 to 204 nodes. The experiment goal is to design strategies representing the "optimistic" and "pessimistic" abstraction scenarios and evaluate their smoothness. For both scenarios we have employed greedy algorithms. In the optimistic scenario the algorithm abstracts a process model in the way that the minimal number of activities is reduced in every abstraction step. In the pessimistic scenario we use the algorithm that abstracts the maximal number of activities per step. Then, the smoothness of model abstraction is found as the mean value of activities reduced at every abstraction step.

Figure 4 presents the distribution of abstraction smoothness obtained in the experiment. Results for the optimistic scenario are shown in Figure 4(a) and for pessimistic—in Figure 4(b). In the optimistic scenario all the models were abstracted with the smoothness between 2.0 and 3.0; more than a half—with the smoothness under 2.5. This means that very often only two activities were aggregated, which is close to the best theoretically possible result. Pessimistic strategy aims to abstract the maximal number of activities in every step. Since models vary in size, in this scenario we use normalized smoothness, dividing abstraction smoothness by the number of nodes in a model. According to the diagram, around 40% of the models were abstracted in huge steps—about half of the model per step, while a few were abstracted even in one step. This statistics proves that the smoothness of the approach relying only on activity aggregation can be quite poor.



(a) Optimistic case



(b) Pessimistic case

Figure 4: Evaluation of abstraction smoothness

Introduction of activity elimination improves smoothness of the abstraction. If the size of a SESE fragment to be aggregated is too large, aggregation of all the activities constituting the fragment leads to high information loss. Not to loose valuable process details, abstraction can be realized through elimination. Instead of replacing the canonical SESE fragment with one aggregating activity, the abstracted activity is eliminated. The choice between aggregation and elimination depends on what operation leads to smaller information loss: elimination of one activity or aggregation of the whole SESE fragment. If the abstraction is performed in semiautomatic manner the user can make this decision. In case of fully automatic abstraction the decision should rely on a criterion. Identification of such criteria is a very interesting problem. However, it is out of scope of this paper and is the subject for the future work.

5 Related Work

The abstraction technique proposed in this paper is based on the PST construction proposed in [VVL07]. In that work the authors showed how control flow graph analysis techniques developed in [JPP94] can be applied for the analysis of business process models. The prototype of the PST for business process models was discussed in [HFKV06], where fragment with multiple entry and exit nodes were addressed. Finally, in [VVK08] the authors elaborate on the idea of using fragments having single entry single exit nodes. This results in more fine granular tree—refined process structure tree.

Alternatively, process model abstraction can be realized by means of patterns. In this case the abstraction approach defines a set of patterns to be recognized in the process model and the rules how these patterns should be handled. Usually the transformation aims to simplify the model through minimization of the nodes number. In literature there is a number of works which employed transformation rules for analysis of process models. The examples are [SO00] and [MVD+08, DJVVA07]. Abstraction methods can be based on such rules: in [LS03] the authors show how process views can be constructed using techniques from [SO00]. In [PSW08b] four abstraction patterns were introduced: sequential, block, loop and dead end. The patterns describe not only structural transformations, but define how to derive properties of new process elements from the original ones. In [BRB07] the authors propose a comprehensive approach to construction of process views. The approach relies upon the wide set of elementary operations, enabling stepwise construction of a view. The defined operations can be used on model and instance levels. The given operations are very generic and include, for instance, those which are not order preserving. The main limitation of the approaches based on patterns is that they cannot handle arbitrary models. In the real life, however, users tend to capture processes in sophisticated models.

The authors of [EG08] introduce a two step approach for creation of process views, which targets cross-organizational collaboration. In the first step the process owner can hide private or irrelevant details, while in the second step elements which are not in the focus of the process consumer are omitted. The views are constructed basing on the sets of nodes to be eliminated and does not rely on patterns. On the other hand, only block structured process models are handled.

6 Conclusions and Future Work

In this paper we have proposed the new approach to process model abstraction. The approach exploits decomposition of a process model into a hierarchy of SESE fragments called process structure tree. Since SESE fragments have arbitrary inner structure, the approach can successfully abstract graph-structured process models. This is one of the main advantages of the solution. This approach allows us to handle fragments with higher flexibility than the technique based on patterns [PSW08b]. However, there are algorithms enabling more fine-granular decomposition of process models, for instance, SPQR [DBT89, DBT96] and RPST [VVK08], which are based on triconnected graph decomposition technique [TV80]. Therefore, the fundamental value of the approach proposed in this paper is the idea of using process structure tree for process model abstraction. With the example of SESE decomposition we have illustrated and evaluated how PST can be employed for the abstraction task. The direct continuation of this paper is the study of methods of more fine-granular model decomposition.

It should be noticed that PST addresses only the structural aspect of abstraction. This means that it splits a model into elements to be abstracted, but does not tell which elements should be abstracted. Therefore, another direction of future work is the search for criteria to judge about significance of model elements. Here, analysis of semantics and non-functional properties of a process model should be taken into account.

References

[BPM08] BPMI. *Business Process Modeling Notation*, 1.1 edition, February 2008.

[BRB07] R. Bobrik, M. Reichert, and Th. Bauer. View-Based Process Visualization. In *BPM*, volume 4714 of *LNCS*, pages 88–95. Springer, 2007.

[DBT89] G. Di Battista and R. Tamassia. Incremental Planarity Testing. In *FOCS*, pages 436–441, 1989.

[DBT96] G. Di Battista and R. Tamassia. On-Line Maintenance of Triconnected Components with SPQR-Trees. *Algorithmica*, 15(4):302–318, 1996.

[DJVVA07] B. Dongen, M. Jansen-Vullers, H. Verbeek, and W. Aalst. Verification of the SAP Reference Models Using EPC Reduction, State-space Analysis, and Invariants. *Comput. Ind.*, 58(6):578–601, 2007.

[EG08] R. Eshuis and P. Grefen. Constructing Customized Process Views. *Data Knowl. Eng.*, 64(2):419–438, 2008.

[GA07] C. Günther and W. Aalst. Fuzzy Mining–Adaptive Process Simplification Based on Multi-perspectiveMetrics. In Gustavo Alonso, Peter Dadam, and Michael Rosemann, editors, *Business Process Management, 5th International Conference, BPM 2007, Brisbane, Australia, September, 2007, Proceedings*, volume 4714 of *Lecture Notes in Computer Science*, pages 328–343. Springer, 2007.

[HC94] M. Hammer and J. Champy. *Reengineering the Corporation: A Manifesto for Business Revolution*. HarperBusiness, April 1994.

[HFKV06] R. Hauser, M. Friess, J. M. Kuster, and J. Vanhatalo. Combining Analysis of Unstructured Workflows with Transformation to Structured Workflows. pages 129–140, Los Alamitos, CA, USA, 2006. IEEE Computer Society.

[JPP94] R. Johnson, D. Pearson, and K. Pingali. The Program Structure Tree: Computing Control Regions in Linear Time. pages 171–185. ACM Press, 1994.

[KNS92] G. Keller, M. Nüttgens, and A. Scheer. Semantische Prozessmodellierung auf der Grundlage "Ereignisgesteuerter Prozessketten (EPK)". Technical Report Heft 89, Veröffentlichungen des Instituts für Wirtschaftsinformatik University of Saarland, 1992.

[LS03] D. Liu and M. Shen. Workflow Modeling for Virtual Processes: an Order-preserving Process-view Approach. *Information Systems*, 28(6):505–532, 2003.

[MVD+08] J. Mendling, H. Verbeek, B. Dongen, W. Aalst, and G. Neumann. Detection and Prediction of Errors in EPCs of the SAP Reference Model. *Data Knowl. Eng.*, 64(1):312–329, 2008.

[PSW08a] A. Polyvyanyy, S. Smirnov, and M. Weske. Process Model Abstraction: A Slider Approach. In *Proceedings of the 12th International Conference on Enterprise Distributed Object Computing (EDOC)*, 2008.

[PSW08b] A. Polyvyanyy, S. Smirnov, and M. Weske. Reducing Complexity of Large EPCs. In *EPK'08 GI-Workshop*, Saarbruecken, Germany, 11 2008.

[RHAM06] Nick Russell, A.H.M. ter Hofstede, W.M.P. van der Aalst, and Natalya Mulyar. Workflow Control-Flow Patterns: A Revised View. Technical report, BPMcenter.org, 2006.

[SO00] W. Sadiq and M. E. Orlowska. Analyzing Process Models Using Graph Reduction Techniques. *Information Systems*, 25(2):117–134, 2000.

[STA05] A. Scheer, O. Thomas, and O. Adam. *Process Aware Information Systems: Bridging People and Software through Process Technology*, chapter Process Modeling Using Event-Driven Process Chains, pages 119–145. John Wiley & Sons, 2005.

[TV80] R.E. Tarjan and J. Valdes. Prime Subprogram Parsing of a Program. In *POPL '80: Proceedings of the 7th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, pages 95–105, New York, NY, USA, 1980. ACM.

[VVK08] J. Vanhatalo, H. Völzer, and J. Koehler. The Refined Process Structure Tree. In *Business Process Management, 6th International Conference, BPM 2008 Milan, Italy, September, 2008, Proceedings*, volume 5240 of *Lecture Notes in Computer Science*. Springer, 2008.

[VVL07] J. Vanhatalo, H. Völzer, and F. Leymann. Faster and More Focused Control-Flow Analysis for Business Process Models Through SESE Decomposition. In Bernd J. Krämer, Kwei-Jay Lin, and Priya Narasimhan, editors, *Service-Oriented Computing - ICSOC 2007, Fifth International Conference, Vienna, Austria, September 17-20, 2007, Proceedings*, volume 4749 of *Lecture Notes in Computer Science*, pages 43–55. Springer, 2007.

[Wes07] M. Weske. *Business Process Management: Concepts, Languages, Architectures*. Springer Verlag, 2007.

On Modeling "Web-Service-Based" Processes for Healthcare

Amir Afrasiabi Rad, Morad Benyoucef, Craig E. Kuziemsky

Telfer School of Management
University of Ottawa
55 Laurier Ave. E, Ottawa, Ontario, K1N 6N5, Canada
a.afrasiabi@uottawa.ca
{Benyoucef, Kuziemsky}@telfer.uottawa.ca

Abstract: In terms of business process-modeling, healthcare is a rather complex sector of activity. Indeed, modeling healthcare processes presents some special requirements dictated by the complex and dynamic nature of these processes as well as by the specificity and diversity of the actors involved in these processes. We discuss these requirements and propose a framework for evaluating process-modeling languages based on such requirements. The proposed evaluation framework is tested using BPEL and BPMN to model a complex healthcare process and the results of the evaluation are highlighted.

1 Introduction

Web services foster the integration of disparate systems despite being developed at different times by different people. Efforts on standardization of web service composition and modeling have resulted in the release of different standards, namely PDL (Process Definition Language), XPDL (XML Process Definition Language), BPSS (Business Process Schema Specification), BPML (Business Process-modeling Language), WSDL (Web Services Definition Language), WSCI (Web Services Choreography Interface), ebXML (Electronic Business using eXtensible Markup Language), BPMN(Business Process-modeling Notation), and BPEL4WS (Business Process Execution language for Web Services). The increase in the number of modeling standards and the diversity of business sectors raised the question of identifying suitable standards for specific business sectors. Many frameworks have been proposed for evaluating process-modeling standards; most of them focusing on one or two aspects of business process languages. Furthermore, evaluation results represented overlaps, limitations, inconsistencies, and ambiguities. These limitations are more observable on the modeling of complex systems.

Healthcare business processes are complex owing to integration of workflows, collaborations, and data transactions between different units, and also the existence of different medical data transaction standards (HL7 [He08]) and DICOM [Ne08]) contribute to the complexity of healthcare processes. They are profiled in the IHE (Integrating the Healthcare Enterprise) technical framework. Moreover, an increase in the number of medical disciplines and the dynamic nature of healthcare delivery call for change tolerant process models [AD04].

A process model is not just a graphical representation but rather it should also serve as a communication base both for communicating domain details between stakeholders and for communicating domain details to system designers [Ge04]. The ability to use a process model for communication becomes difficult if the modeling specialists are the only people who understand the models. Therefore the models should be representative enough to be also understandable by healthcare administrative or clinical stakeholders. A further issue is integration. If each healthcare unit uses a modeling language that best fits its requirements, the whole system's model will be a combination of several standards making the integration of models and their understandability difficult.

Scheduled workflow (SWF) is one of the most complex IHE workflow. It is representative of healthcare processes and provides a complete test base for the evaluation of our results because of its complexity, large amount of domain communication, and service like behavior. Most research initiatives on healthcare process-modeling within a web service environment have chosen BPEL as their modeling standard, and there has been limited research done on the suitability of business process-modeling languages in healthcare, particularly in complex work processes.

This paper reports on our ongoing research in process-modeling in healthcare by identifying the complexities of healthcare and proposing a methodological approach for applying modeling languages to healthcare. The paper is organized as follows. Section 2 presents background and related work. Section 3 investigates the applicability and suitability of process-modeling languages for the healthcare domain, Section 4 proposes an evaluation framework for modeling languages based on requirements specific to healthcare, and Section 5 applies the framework in a case study. The paper concludes with a discussion of the main findings and future research arising from these findings.

2 Background and Related Work

Modeling languages have fundamental differences with regard to Expressiveness, Flexibility, Adaptability, Dynamism, and complexity aspects [LS07]. A consequence is that different classes of modeling languages suit different business sectors. Explorations of common factors of modeling languages concluded that the understandability and complexity of models have a positive relationship [GL06]. However, to date, the focus has been on common features of all modeling languages while unique features have been neglected.

Existing research has developed a quality framework for evaluation of modeling languages [NK05]. However the quality framework does not suggest any definite metrics for evaluating languages and the comparison of modeling languages is only case-based and tightly related to the domain appropriateness quality area. Other research [NRC07] evaluated the understandability of models, providing clear metrics that make the measurement of understandability more accurate. Moreover, Luo and Tung [LT99] classified the characteristics of modeling languages and defined a set of steps for their selection. However, their proposed framework is general while healthcare has some specific requirements, such as permanent evolution and high value of communication, which are not considered in the framework in question.

Workflows and data flows are an important factor in an evaluation framework. Researches like [Wo02, Wo06, Va02, Va03] extracted Workflow, Data flow, and Control flow patterns, and provided detailed evaluations of modeling languages based on formerly mentioned patterns. Green et. al [Gr07] provided an ontological framework for evaluating process-modeling languages. They evaluated languages based on their ability to model ontological constructs. Results show that the number of constructs that each business sector implements is different, and some modeling standards are unable to represent some ontological constructs, or they are too complex in representation. In our research, we apply these frameworks to healthcare.

Anzbock and Dustdar [AD04] explored IHE workflows to study healthcare process-modeling in a web service based environment. Their work is limited to the evaluation of BPEL in test cases and the identification of considerations for modeling healthcare web services in BPEL. They defined transport, security and reliability as healthcare process model requirements and provided solutions for them in BPEL. Since their work is limited to the applicability of BPEL, and they have not analyzed the resulting models based on an evaluation framework, we believe further research should be done on the appropriateness of modeling languages besides their applicability. Furthermore, the authors argue that some healthcare process-modeling challenges, which come from the immaturity of the web service stack standards, remain unsolved.

3 Suitability of Process-modeling Languages for Healthcare

Healthcare processes are complex since modern health services involve care delivery by members of multiple groups with different knowledge levels and often residing in different physical locations. The complexity of a process can be measured by the number of units and the number of transactions between collaborating parts. For instance, the radiology sub system in the IHE framework contains 34 units and more than 65 transactions, and each of them contains several message types. Meanwhile, the fact that departments communicate and interact with each other adds to the complexity of healthcare processes. In this section we investigate the complexity of healthcare processes from a modeling perspective. We then elaborate on some features that should be included in modeling languages to support for modeling of healthcare systems.

Factors contributing to the complexity of processes include the fact that healthcare systems are constantly evolving and dynamic in nature. New expertise emerges and processes change in order to adopt new services. Healthcare actors' tasks are interconnected in a way that a change in one agent's task may affect others and departmental processes. Meanwhile, due to the dynamic nature of healthcare, agent's actions do not always follow normal routines since many emergency situations happen in healthcare systems. In emergency situations speed has the more important role, so agents do not exactly follow the routines, and sometimes their actions conflicts with others' tasks. Modeling languages should be able to represent the dynamic nature of healthcare processes through exception handling.

Influence of change in complex systems can go further if the inter-departmental interaction is tight. A change in an agent's task or a non-routine action can easily affect other agents, so changes can transmitted to other units in an incremental manner due to tight interconnections between healthcare units. Consequently, with regard to interconnection, a change in routines makes a huge change in the system. A simple example would be an emergency surgery which needs fast collaboration of many departments such as pathology laboratory, radiology, pharmacy, admission, and so on. Modeling languages should be able to represent this situation, so models flexibly represent routine and non-routine exceptional actions. Meanwhile, Models should always be loosely designed to accommodate independent actions of agents and their successive units.

Different agents in complex systems may simultaneously be member of more than a unit and memberships may change frequently due to unforeseen events. A small change in the processes of a unit in the healthcare system may create conflict in departmental business models. Nested healthcare processes and agents create inherently non-linear relationships and remove the boundaries between sub-processes. This fact complicates the modeling task and increase difficulty of change in models as internal rule sets are not very strict, coordination of units in complex systems cannot be represented as a rigid model. For the long lasting models, modeling languages should be able to represent healthcare's fuzzy departmental boundaries, and provide features that make the maintenance process easier. Modeling languages, also, should be able to model this high level of communication. Meanwhile, a modeling language should be able to adapt to wide range of changes in models without needing extensive remodeling. Modularity and support for abstraction in models are key concepts for change adoption. Completeness and extensibility of the language provide means of modifying the model to adapt it to changing processes.

The behavior of a complex system emerges from the interaction among the agents, departments, and even systems. Different sub systems may have dissimilar features thus divergent modeling requirements. In order to accommodate all requirements in models, sometimes, it is essential to use diverse modeling language for different departments. These models should be able to communications between departments and it is not possible if modeling languages do not have matching features and ability to be mapped to each other.

Security and privacy are crucial issues in healthcare systems, particularly when the communications are done over the Internet and the structure is service-oriented. Thus the models need the ability to represent security and privacy considerations.

To deal with the complexity of healthcare processes, process-modeling languages should allow the modeler to create optimized models with different numbers of specifications for different purposes. For models used by healthcare staff, for instance, more detailed models are confusing; and thus should only represent the high level graphical representation of processes. However, models used by IT department staff require detailed specifications. To comply with the need for different views, flexibility is required in the level of details and representation.

4 Evaluation Framework

Process-modeling languages are innovatively designed or developed from existing modeling approaches by unifying several methods or adding features to an existing one [POB00]. We developed a framework (Fig.1) for evaluating modeling languages for service based healthcare environments. We consider languages' unique features in addition to their common features since the unique features are key for specific uses. We also need to consider general modeling qualities, which are required by all systems, in addition to healthcare specific requirements. We identified important features that allow us to measure the quality of modeling languages. From the work performed in [LM04], we understand that models should be representative, understandable, easy to use, support abstraction, and be optimized in the level of details. In the following paragraphs we explain our framework and the metrics used for measuring each evaluation criterion.

Security: Security is important when dealing with service oriented architectures and communications over the internet. For all the various uses of the models (training, development, documentation, etc.) security and privacy issues must be considered. For healthcare processes, where security has a special importance, modeling languages must represent inter-departmental, general access privileges and secure interactions.

Pattern representation: The understandability of a modeling language is a fundamental requirement for increasing its usability, and is tightly related to the process modeler's capabilities in addition to the modeling language's features. Factors such as the modeler's level of expertise, creativity, and familiarity with the business, and tool, are inevitable. Pattern representation capability of a language for data flow, communication, and control flow patterns is an important criterion for the understandability of languages. Studies in [Va02, Va03] showed that some patterns cannot be modeled by any modeling language, and some patterns can only be modeled using a limited number of languages. Thus the understandability of modeling languages depends on the ability and methods of representing patterns. If there is a construct in the language that directly represents a pattern, the understandability of the language will increase. However, combining several constructs in order to represent a pattern decreases the understandability and makes the language ambiguous.

Fig 1. Evaluation Framework

Ontological constructs: Language concepts are evaluated using ontological constructs. The BWW (Bunge-Wand-Weber) ontological framework is proposed in [WW93] for evaluating modeling grammars. The authors argue that modeling notations which are not able to represent all of the ontological constructs are incomplete. Language incompleteness reduces understandability because incomplete language features and structures force the modeler to ignore some features of models as there is no construct supporting those specific needs. Nevertheless, while there may be other ways to represent the needed constructs in order to compensate the languages' incompleteness, these ways are diverse and cause ambiguity.

Extendibility: Extensions help to support different technologies and different business sectors, and also to overcome modeling language deficiencies. However, since model users are familiar with basic model notations, extensions in the language can create confusion for those who are not familiar with the extended features. Consequently, extendibility is a compromise between increased capability and general understandability.

Notations: Modeling languages are categorized, based on notations, into graph based, rule based languages, and their combinations [LS07]. Language notations can be measured by the way they follow the standards e.g. standard element size and colors for graphical notations. Textual notations can be measured based on the same concept. The keywords representing concepts and descriptive words should follow their existing standards. For languages that provide execution features, separation of computation and representation of processes differentiate two diverse aspects of modeling, so each actor

can access a part of the model which addresses his needs. Meanwhile, uniformity (use of the same set of notation with unique and unified meanings) and formality (choosing commonly accepted graphical notations for language concepts) of notation improve the language's ease of use.

Modularity: The modularity of modeling languages can be measured by considering the support for abstract processes and sub-processes. Abstraction prevents from revealing the underlying layers of a process, hence improving the understandability of process models for non-developer actors. Also, the understandability of developers and designers from models will improve when the unimportant data is hidden. Reusability is an important factor and it is tightly related to modularity. The ability to use previously modeled processes increases the speed and accuracy of modeling. It also reduces the time required for understanding models. Maintenance and modification of models with reused parts is easier. Reusability has a positive connection with understandability in a way that more reused sections in models help decrease the required effort for learning the process. However, it is not necessarily true that languages that support sub-processes also support reusability.

Level of detail: The level of detail provided by modeling languages should be optimized for different modeling proposes. This metric has a close positive relationship with other aspects of a modeling language such as notation, abstraction and ability of the language to execute processes. Detailed documentation of processes improves the level of understanding for the designer and developer actors. However, the end user, who uses the models for training, does not need detailed information, and more detailed documentation may impair his understanding. Consequently, the languages should be flexible in the level of details, and provide facilities to support both detailed documentation and high level representation of processes. This criterion can be measured by support for abstract processes. Also, languages should not force the modelers to create detailed models. Languages that support execution should be able to separate execution and representation in order to hide unnecessary details.

Exception handling: Exception handling is an important area of evaluation. Exceptions have been an indispensable part of business process-modeling for more than two decades [Cu03]. Exception handling features increase modeling difficulty but also increase the adaptability of models to exceptional circumstances. Improved adaptability helps model users to predict all model behavior in the time that exceptions occur, so the models are more understandable. For healthcare processes with a high possibility of exception occurrence, this criterion has high importance. Investigations in modeling languages show that not all modeling languages contain cancelation functionalities. In the case of those languages, the processes end naturally; the modeler does not have the ability to terminate a process at a special time. Being able to define the termination conditions and terminate processes at a certain moment improves the understanding of process timelines.

The evaluation framework we propose here provides a starting point to evaluate different modeling languages in the healthcare domain. .The strength of the framework is its extensiveness with regard to covering unique features of languages, which makes it an unbiased test-bed for comparing different language families. Implications for general modeling are also considered.

5 Case Study

The *Radiology Admission Process* scheduled workflow (SWF) is one of the most complex IHE processes in healthcare. Consisting of twenty transactions and eight different IHE Actors, SWF is representative of healthcare processes and provides a complete test base for the evaluation of our framework because of its complication, large amount of domain communication, and service like behavior. As security, privacy, change, and exceptions are indispensable parts of healthcare processes, this SWF represents all features of healthcare processes and all requirements of healthcare process models.

In this paper we use the *Radiology Admission Process* SWF to evaluate two modeling languages (BPMN and BPEL) using our proposed framework. Note that our ongoing research is not limited to these two languages, and that other modeling languages are being studied. BPMN is a graphical workflow modeling language that does not focus on any technology; it is able to model processes focusing on service oriented architectures. BPEL is a process-modeling language within web service environments. The collection of these two different languages (different from a notation and architectural perspectives) makes a unique test bed for our framework at this stage of our research. We also evaluate the applicability of the two languages to healthcare processes.

Security: None of the languages has constructs to represent security in the models, and security representations are provided in the extensions to the languages. The two languages lack the fundamental security feature without use of extensions. However, it should be noted that some extensions make a significant alteration in the language. BPMN's security enabled version is developed in [RFP07]. In this version the security extension provided additional graphical notations which compromised the standard notation of the language.

Pattern support: Based on our evaluation criteria, one of the measurements is the level of support for the workflow and data flow patterns involved in healthcare processes. Patterns are documented in [Wo08]. Evaluating the patterns using our evaluation framework and the findings in [Wo02, Wo06] are shown in Table 1. The capability of modeling languages in representing patterns and the understandability of their method in representing patterns are separated by a slash (/) in the table. For instance, BPEL is able to represent "Structured Loop" directly using while, but there is no direct representation for the *repeat until* structure. In another example, BPMN supports "Deferred Choice", but there are two different options; either by combining event-based exclusive gateway and either intermediate events or receive tasks.

Ontological Completeness: Findings in [Gr07] on ontological mapping of modeling languages and Bunge-Wand-Weber (BWW) ontological base model show that modeling languages are not able to represent all the ontological constructs documented in [WW93, We97]. We studied healthcare processes for their constructs and mapped them to BWW models. The findings show that not all the ontological constructs are used in healthcare processes. Table 2 illustrates the constructs used in healthcare and the findings in [Gr07] are used in order to show the support of languages for the ontological constructs. For BPMN, we used the same method that is used in [Gr07] to map it with BWW.

Extendibility: Both BPEL and BPMN are extendable and can accept new features to the language. For healthcare modeling both languages need a security extension and BPEL also needs an extension to support sub-processes.

Notations: Both BPEL and BPMN follow an accepted standard for their notations. From the graphical notations viewpoint both languages' representational components are unique. However, BPEL's notation is not understandable for non-modeler actors since its textual and graphical notations are entwined, so a model user must understand the BPEL's background coding as well as the graphical notation. Moreover, this inseparability makes changes to the graphical notation for the non-expert user rather cumbersome.

Modularity: The overload of transmitting information, the high amount of communication and the need for reusability make sub-processes an indispensable feature in healthcare process-modeling. BPMN supports sup-processes in the modeling procedure without extensions to the modeling languages. By contrast, BPEL does not provide any support for sub-processes as standard. IHE transactions take place between IHE actors (healthcare unit), and each transaction contains a set of HL7 and/or DICOM messages. Each actor is assigned an internal process determining the type of messages that will be communicated in a transaction, and creates the information in the way that is necessary for the message type. Consequently, there is a process running inside each IHE actor. Using BPEL, the goal is to model processes in a web services environment. For the whole IHE model, the central process will regulate communication between services, if IHE actors are considered as service providers. In this system, however, actors need to perform internal processes. There are two ways to represent this system in BPEL; the first way is to provide an abstract central process with conceptual transactions between units in addition to actor's internal processes modeled in different BPEL processes with a reference to a central process (see Fig 2). This makes the process simple and understandable without sacrificing the model's execution feature, each sub-process can be executed separately. The second way is to use extensions. While extensions to BPEL increase its understandability to users experienced with extensions, understandability decreases for those who are unfamiliar with new elements. Moreover, the use of extensions decreases the portability of models. On the other hand, BPMN is able to represent sub-processes and IHE's internal and external one-to-many relationships (see Fig 3).

Level of Detail: Different healthcare model users require different views of the processes. Different levels of detail specify the intended use of models. To a large extent, BPEL is inflexible in the level of detail that can be provided to the end user. BPEL process models are associated with coding which contains a high volume of details. BPEL's limitation is the inseparability of graphical notation from background execution code. Healthcare managerial staffs do not need to see detailed code while it is necessary for IT actors. The inflexibility of BPEL at the detail level makes the modeling, on some levels, cumbersome and decreases the understandability of models to specific non-IT actors by confusing them with additional components on models. On the other hand, BPMN is more flexible on the detail level. Although it does not allow the model designer to add coding to the model, additional detail for special model users can be added in the form of comments. BPMN does not force the user to a pre-specified level of comments as opposed to BPEL in which the model designer is forced to create fully detailed models.

Exception Handling: Both BPMN and BPEL provide a complete set of exception handling and compensation features. While these features are more important in BPEL, healthcare processes demand adequate support for exception handling features to increase the understandability of processes. Fig 2 and Fig 3 illustrate the complexity of exceptions and the way that BPMN and BPEL represent them. However, from the representational point of view BPEL is more limited in dealing with exceptions since there is no cancellation element. The modeler cannot decide to end a process instantly; the process continues until it gets to the ending point. This limitation results in larger models with more connections however the flow of model finishing only at the end point makes it easier to follow the process workflow.

Pattern	Standard (capability / understandability)	
	BPEL	**BPMN**
Sequence	+/ +	+/+
Parallel Split	+/(+/-)	+/+
Synchronization	+/(+/-)	+/+
Exclusive Choice	+/+	+/+
Simple Merge	+/+	+/+
Structured Discriminator	-/-	(+/-)/-
Arbitrary Cycles	-/-	+/+
Implicit Termination	+/+	+/+
Multiple Instances without Synchronization	+/(+/-)	+/+
Multiple Instances with a Priori Design-Time Knowledge	-/-	+/+
Multiple Instances with a Priori Run-Time Knowledge	-/-	+/+
Deferred Choice	+/+	+/(+/-)
Interleaved Parallel Routing	(+/-)/-	-/-
Cancel Activity	+/+	+/+
Cancel Case	+/+	+/+
Structured Loop	+/(+/-)	+/+
Recursion	-/-	-/-
Transient Trigger	-/-	-/-
Persistent Trigger	+/+	+/+
Cancel Region	(+/-)/-	(+/-)/-
Blocking Discriminator	-/-	(+/-)/-
Cancelling Discriminator	-/-	+/(+/-)
Structured Partial Join	-/-	(+/-)/-
Blocking Partial Join	-/-	(+/-)/-
Cancelling Partial Join	-/-	(+/-)/-
Critical Section	+/+	-/-
Interleaved Routing	+/+	(+/-)/-
Thread Merge	(+/-)/-	+/+
Thread Split	(+/-)/-	+/+
Explicit Termination	-/-	+/+

Table 1. Pattern support of modeling languages for healthcare

Constructs	Standard (capability / understandability)	
	BPEL	**BPMN**
Property	+	+
Class	+	-
State	+	+
Conceivable state space	-	-
State Law	-	-
Lawful State Space	-	-
Event	+	+
Conceivable Event Space	-	+
Transformation	+	+
Lawful Transformation	+	+
Lawful Event Space	-	+
Coupling	+	+
System	+	+
System Composition	+	+
System Environment	-	-
System Structure	+	+
Subsystem	-	+
System Decomposition	-	+
Level Structure	-	+
External Event	+	+
Stable State	-	-
Unstable State	-	-
Internal Event	+	+

Table 2. Ontological representation of modeling languages for healthcare

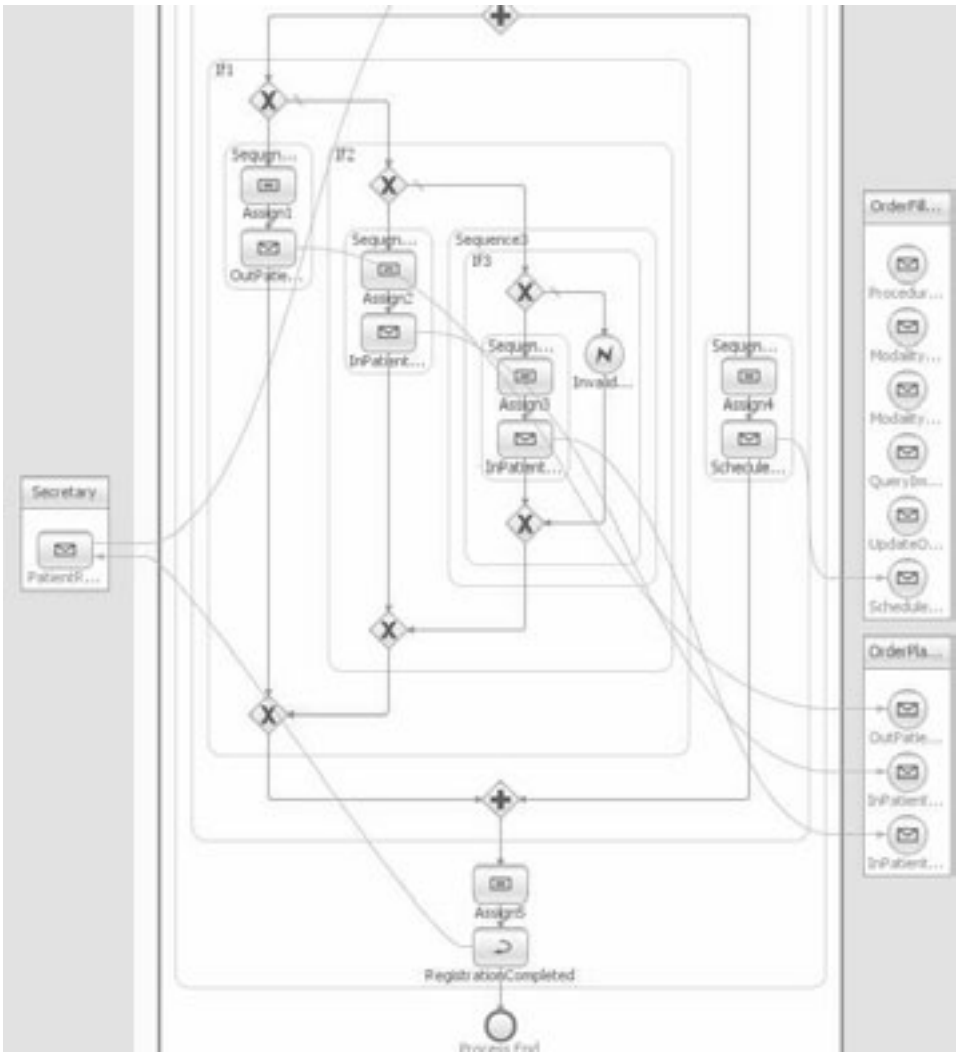


Fig 2. ADT sub-Process and exception handling - BPEL

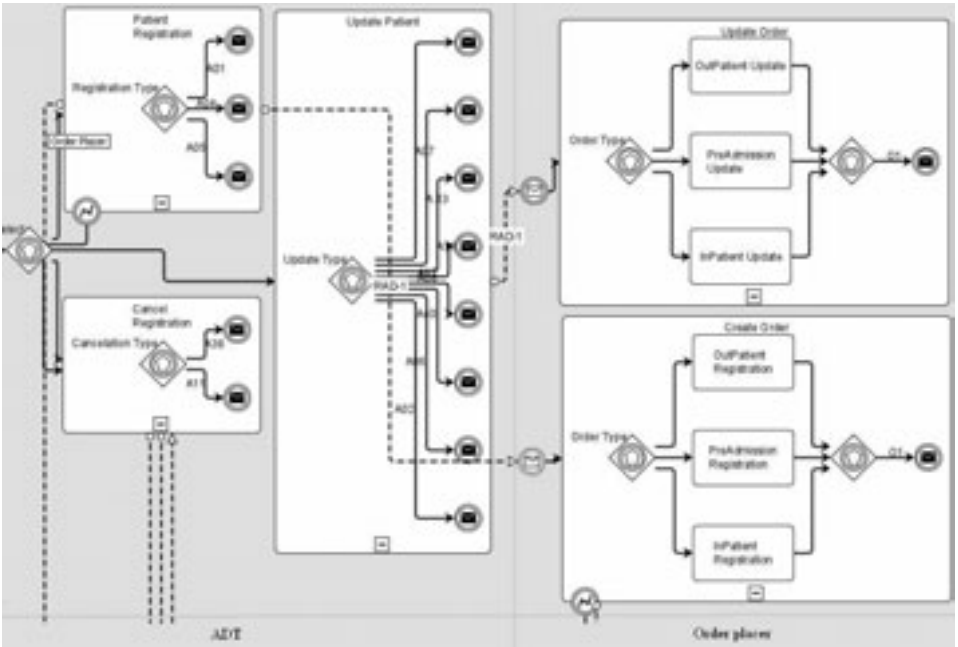


Fig 3. Sub-processes and exception handling - BPMN

6 Conclusion

Many research initiatives have been focused on the ambiguities and difficulties on normalization of models and modeling languages in general business sector focusing on the goal of improve our ability to design information and communication technologies. As specifications of different business sectors and variety of processes imposes the utilization of a subset of modeling features and complex and simple business sectors are fundamentally different, modeling languages should be seperately evaluated on each business sector. This paper presents the preliminary results of our research on modeling healthcare processes within a service oriented environment, taking into account the specificities of the healthcare sector as well as its actors in order to enhance business process-modeling and improve to design information and communication technologies to support complex domains such as healthcare.

We introduced a framework for evaluating business process-modeling languages in healthcare, defined a set of requirements for healthcare process models, and performed a case study on a complex IHE transaction using BPEL and BPMN. So far, results show that neither of the studied languages satisfies healthcare modeling requirements, and that extensions to existing languages are necessary in order to generate models that can suit healthcare modeling requirements.

Most complex systems share a similar set of requirements and specifications. As a result, although healthcare was your domain of focus the findings can support process-modeling in other domains that have complex requirements.

The next step in our research is to study different suitable modeling methodologies for healthcare, merge languages, and possible extensions to selected modeling languages. In addition to BPMN and BPEL, other modeling languages are being studied for their suitability for the healthcare sector.

Bibliography

[AD04] Anzbock, R.; Dustdar, S.: Modeling medical e-services. Lecture Notes in Computer Science - Business Process-modeling Springer Berlin, Heidelberg, 2004; pp. 49-65.

[Cu03] Curbera, F. et. al.: Exception Handling in the BPEL4WS Language. BPM 2003; pp. 276.

[Ge04] Gemino A, Wand Y. A framework for empirical evaluation of conceptual modeling techniques. Requirements Eng, 2004, 9:248-260

[Gr07] Green, P. et. al.: Candidate interoperability standards: An ontological overlap analysis. Data & knowledge engineering, vol. 62, no. 2, 2007; pp. 274-291.

[GL06] Gruhn, V.; Laue, R.: Complexity Metrics for Business Process Models. 9th international conference on business information systems, 2006; pp. 1.

[Hl08] Health Level Seven, Inc.: Health Level Seven, Inc. Available: http://www.hl7.org/ 2008.

[LM04] Laguna, M.; Marklund, J.: Business Process-modeling, Simulation, and Design, Pearson Printice Hall, USA, 2004.

[LS07] Lu, R.; Sadiq, S.: A Survey of Comparative Business Process-modeling. BIS 2007, pp. 82.

[LT99] Luo, W.; Tung, Y.A.: A framework for selecting business process-modeling methods. Industrial Management & Data Systems, vol. 99, no. 7, 1999; pp. 312-319.

[MRG07] Mendling, J.; Reigers, H.A.; Cardoso, J.: What Makes Process Models Understandable?. Business Process ManagementSpringer, Springer Berlin, 2007; pp. 48.

[Ne08] NEMA, O.: DICOM. Available: http://medical.nema.org/, 2008.

[NK05] Nysetvold, A.G.; Krogstie, J.: Assessing Business Processing Modeling Languages Using a Generic Quality Framework. Workshop on Exploring Modeling Methods in Systems Analysis, 2005; pp. 545.

[POB00] Paigea, R.F.; Ostroffa, J.S.; Brookeb, P.J.: Principles for modeling language design. Information and Software Technology, vol. 42, no. 10, 2000; pp. 665-675.

[RFP07] Rodriguez, A.; Fernandez-Medina, E.; Piattini, M.: A BPMN Extension for the Modeling of Security Requirements in Business Processes. IEICE Transactions on Information and Systems, vol. E90-D, no. 4, 2007; pp. 745-752.

[RT06] Rossi, D.; Turrini, E.: What your next workflow language should look like. 2nd InternationalWorkshop on Coordination and Organization. 2006.

[Va02] Van Der Aalst, W.M.P.; Dumas, M.; Ter Hofstede, A.H.M.; Wohed, P.: Pattern based analysis of BPML (and WSCI), Queensland University of Technology, QUT, 2002.

[Va03] Van Der Aalst, W.M.P., Ter Hofstede, A.H.M.; Kiepuszewski, B.; Barros, A.: Workflow Patterns. Distributed and Parallel Databases, vol. 14, no. 1, 2003; pp. 5-51.

[WW93] Wand, Y.; Weber, R.: On the ontological expressiveness of information systems analysis and design grammars. Information Systems Journal, vol. 3, no. 4, 2003; pp. 217-237.

[We97] Weber, R.: Ontological Foundations of Information Systems, Coopers & Lybrand and the Accounting Association of Australia and New Zealand, Melbourne, Australia, 1997.

[Wo02] Wohed, P.; Van Der Aalst, W.M.P.; Dumas, M.; Ter Hofstede, A.H.M.: Pattern Based Analysis of BPEL4WS, Queensland University of Technology, QUT, 2002.

[Wo06] Wohed, P.; Van Der Aalst, W.M.P.; Dumas, M.; Ter Hofstede, A.H.M.; Russell, N.: On the sustainability of BPMN for business process-modeling" in Lecture Notes in Computer Science - Business Process Management Springer Berlin, Heidelberg, 2006; pp. 161-176.

[Wo08] Workflow Patterns Initiative: Workflow Patterns [Homepage of Workflow Patterns Initiative], [Online]. Available: http://www.workflowpatterns.com/ 2008;

Human Activities in Distributed BPM

Yoichi Takayama, Ernie Ghiglione, Scott Wilson*, James Dalziel

Macquarie E-Learning Centre Of Excellence (MELCOE)
Macquarie University
E6A-248 Eastern Road
Macquarie University, NSW 2109
Australia
*Centre for Educational Technology & Interoperability Standards (CETIS)
The University of Bolton
Deane Road
Bolton, BL3 5AB
United Kingdom
{yoichi, ernieg, james}@melcoe.mq.edu.au
scott.bradley.wilson@gmail.com

Abstract: The general concept of inter-workflow system communications has been proposed by the WfMC in 1995. However, there has been little study or use case on general inter-workflow system communications, except for business massage exchange-based protocols. Since BPEL allows a local system to invoke remote BPEL Processes via Web Service interface, this can be used as a mechanism for inter-workflow communications for BPEL Processes. Currently, however, it causes a problem if the remote BPEL Processes use BPEL4Peole extension and include People Activities and Human Tasks. This is because BPEL4People has not anticipated Processes to be called remotely and there is no provision for the remote invocation of the user interfaces. This paper studies a possible mechanism in which a local system can invoke a remote BPEL Process with Human Tasks and let local users perform human activities via user interfaces that are defined in the remote Human Tasks.

1 Introduction

The aim of business process management system (BPMS) is to aid business processes with a good balance of manual and automated resources. The computerization is expected to contribute to the better organization and management of entire system by transferring some of the functionalities and activities to automated resources. The manual resources, i.e. human activities, have been recognized as one of the essential elements that drive the system [Ha05].

Although it is not the only model, it is thought that one of the natural models for business processes is a workflow model. To allow componentized and modular architectural design and to allow different vendors to integrate easily, it is also considered advantageous to implement the workflow model with Service Oriented Architecture (SOA) [Oa07].

One of such models, Web Services Business Process Execution Language (BPEL), was designed mainly for automated processes and Web Services were used [Oa07]. Scientific Workflows are also designed for automated processing of data [De08]. One of the Scientific Workflows, Taverna Workflow [Oi04], also resembles BPEL to some extent, probably because its workflow language was originally derived from the IBM Web Service Flow Language (WSFL), a predecessor for BPEL. A recent trend is that these automated workflows are now incorporating human interactions into the workflow designs and the systems. WS-BPEL Extension for People (BPEL4People) and Web Services Human Task (WS-HumanTask) extend the BPEL with a People Activity a Human Task, respectively [Ag07a, Ag07b]. In this paper, we refer them as BPEL4People unless we need to refer them separately. The Taverna Workflow now can have so-called Interaction Service nodes in its workflows [LO08]. The Interaction Service typically sends an email notification (a) with a simple question and a return URL or (b) with a URL for a Web application that provides interactions. The user may respond with a simple answer or use the Web user interface on a remote Interaction Service server to prepare more complex response, respectively.

Our system (Research Activity Management System, RAMS) also uses human activities in workflows [Da06]. It uses human-flow workflows (or group workflows) as it is loosely based on IMS Learning Design model [Im03]. Since this workflow is different from other types of workflows, we have initiated an integration project to incorporate Scientific Workflows and BPEL-based business processes into RAMS workflows.

There are different ways by which different types of workflows may integrate or communicate. This has been documented by the Workflow Management Coalition (WfMC) [Ho95] and we also produced a research document considering options in integrating human-flow workflows and Scientific Workflows [Ta07]. In Business Processes, inter-business communications as messages or choreography have been actively studied but there has been little study on more general inter-workflow communications.

One of the ways to achieve inter-workflow communications is that some workflows and BPEL-based business Processes can be invoked remotely. In RAMS, this can be easily achieved by providing a new Tool Activity [GD07]. We may name such an activity as Remote Process Tool Activity. Alternatively, a remote workflow may be integrated by converting it to a Web application with a Remote Web Application Tool Activity. The first of such an attempt was to integrate Pegasus Workflow Management System [Le08] by using a Web application interface. (The details will be published elsewhere). If the remote workflows or processes contain human activities, however, this poses a problem.

For example, a workflow definition of Taverna Workflow can be invoked from the Remote Workflow Tool Activity. If the execution point of the workflow encounters a human activity (Interaction Service), it should request a manual input from the invoker of the remote workflow who is on our system. Currently, each Interaction Service (representing a notification server) of a Taverna Workflow can be configured to send email notices of the required human activities to a fixed email address declared in its configuration. There is no way to allow the remote workflow server to accept dynamic email addresses for different invokers or to call back our workflow system to present the notification, human interface, or to return the results.

Likewise, RAMS may invoke a remote BPEL4People-compliant process. However, the BPEL4People syntax allows the human activities to be performed by only the remote users on the remote host. It does not allow local RAMS users to perform the Human Tasks defined in the remote Processes [Ag07a, Ag07b]. For example, imagine a travel plan business process hosted on a remote host by some travel agency. When we invoke it from our system, we want the local invoker to fill in all the interactions required and to receive the output when it is completed. The current syntax allows human users on the remote host to perform the human activity, but not users on the local host.

Also, the nature of the processes and workflows containing human activities or of the Scientific Workflows is that they are long-standing. Most likely, the invoker will have to proceed with other Activities on the current workflow. The user may either return to check the results, receive a notification to retrieve the results, or use another Tool Activity that shows the pending jobs or returned results. In other words, asynchronous invoking arrangement should be in place.

For these reasons, invoking remote Business Processes or workflows that contain human interfaces poses a unique technical and research problem. BPEL4People V. 1.0 has been proposed only recently (June, 2008) and also no one seems to have considered the complication of invoking BPEL Processes with BPEL4People extensions as remote Processes. In our knowledge, there is no previous study that examined the full implications of invoking remote Processes or workflows that have to provide human interfaces. This paper, therefore, addresses these problems and presents a possible mechanism to achieve such interactions.

This paper first explains how a remote BPEL Process may be invoked by another workflow system, then points out the problem in case human activities are included in the Process, and presents a solution. Also, although the paper mainly focuses on BPEL and BPEL4People, it also compares it with Taverna Workflow in which an inter-workflow communication has been achieved by a simpler mean. Although the initial examples also mentions RAMS, the principle presented is general and not limited to any particular implementation. The figures used for illustrative purposes are loosely related to the UML Component Diagrams and Activity Diagrams (OMG07). A strict adherence to the UML formalism, however, is not observed.

2 Workflow Integration via Invocation

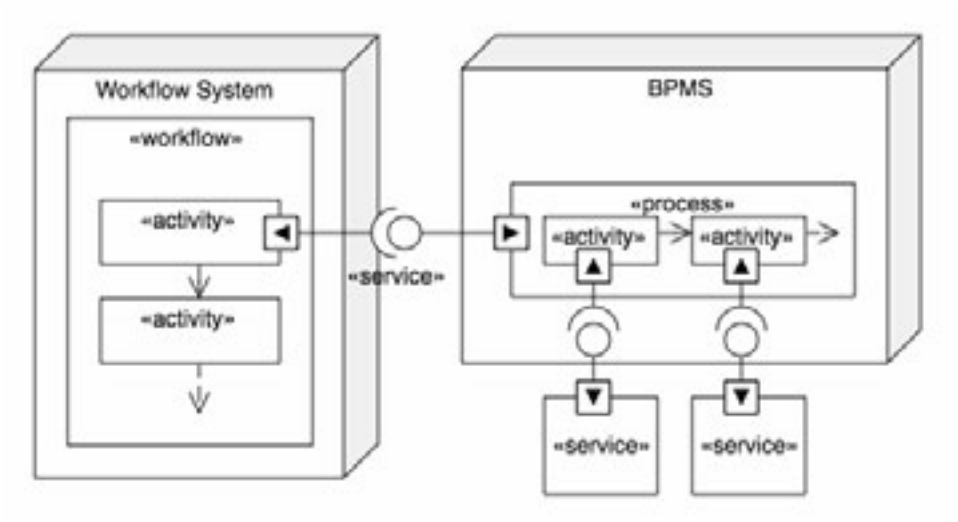


Fig. 1. Process Integration by Invocating a Process from an Activity.

One of the ways to integrate a workflow (e.g. a RAMS workflow) with a remote workflow (e.g. a BPEL Process) is to give a RAMS Activity an ability to invoke a remote Process. Our user Activities are called Tool Activities, because the activities use Tools that implement various tasks the end users can perform [GD07]. Tools are plug-ins and it is easy to extend the system by adding a new Tool that is capable of performing a new activity. Therefore, RAMS system can provide a newly created "Remote Process Tool" that can invoke a remote BPEL Process via Web Interface. The remote BPEL Process may further invoke other Web Services and return results to RAMS workflow that invoked it (Fig. 1).

3 Human Tasks in BPEL4People

In case of remote Processes with People Activity and Human Tasks, BPEL4People syntax allows only the remote users to work on the Human Tasks (Fig. 2).

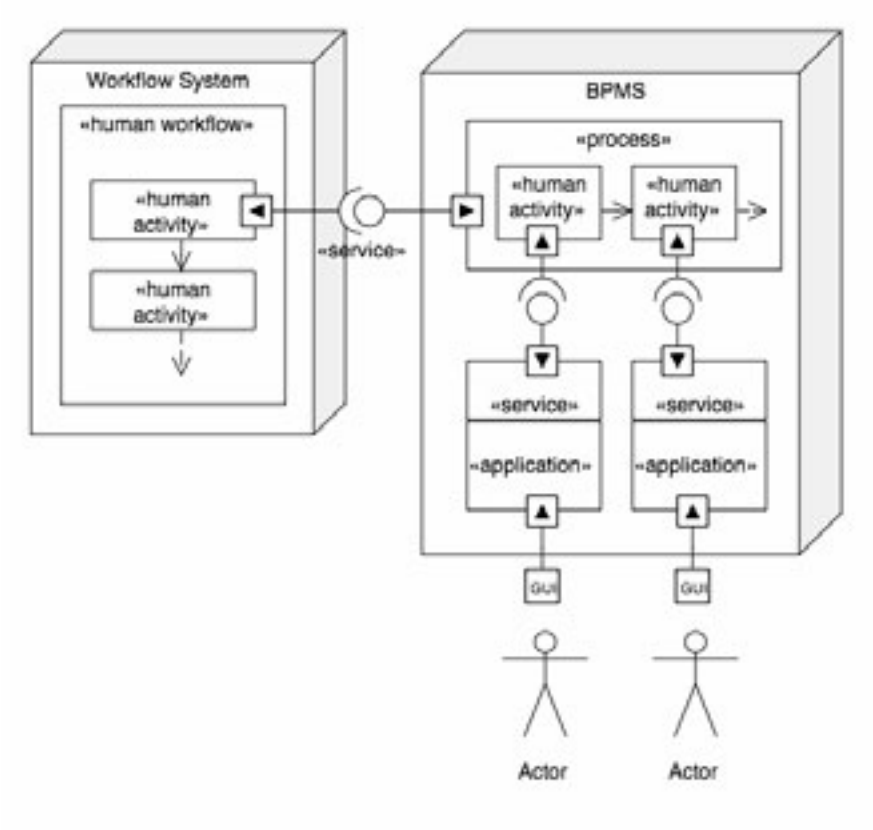


Fig. 2. Human Tasks in Process.

The Human Tasks are presumably managed by a Task Manager, which creates Task instances by reading the Process design and manages its life-cycle and provides a user interface, Task List, for users to work with life-cycle of Tasks (Fig. 3). (A Task Manager is not defined in BPEL4People, but it is a presumed functionality of collection of Task life-cycle management activities suggested by the standard). An end user can claim and start a Task from the Task List. Starting a Task will take the user to the application interface of the Web Service, which has implemented the actual work of this Task as an application. The Web Service has a triple interface, Web Services interface, Task life-cycle management interface and the application interface. Fig. 3 has omitted the life-cycle management interface since that is not relevant to the subject of this paper. BPEL4People also uses a term People Activity, but here it is called a human activity.

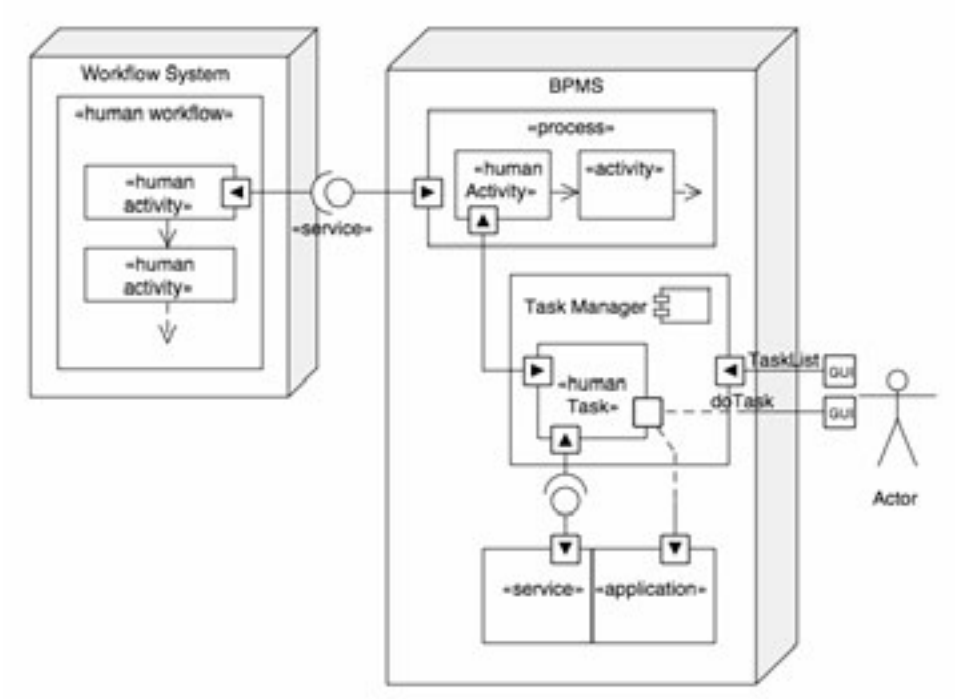


Fig. 3. Task Manager and Human Task.

4. Modifications Required for BPEL4People to Allow Local Users

RAMS may invoke a remote Process that requires remote users to provide some manual work (e.g Figs. 2 and 3). What we would like to achieve, however, is for the RAMS users, i.e. local users, to be able to perform remote Human Tasks defined in the remote Process.

The presumed Task Manager provides the Task List as a way for end users to access and manage Tasks as Fig. 3. The BPEL4People indicates that a remote Task List Client may access the Task List in a standardized manner. As Fig. 4, therefore, the Task List could be accessed by a local Task List Client on RAMS system. This looks as though the local RAMS user could do the Human Tasks. The only users, however, who can access the Task List Client (the users for that Task), are the users of the Process host, as it is defined in the BPEL4People syntax. It is not poossible for the local system to let local users to work with the remote Human Tasks.

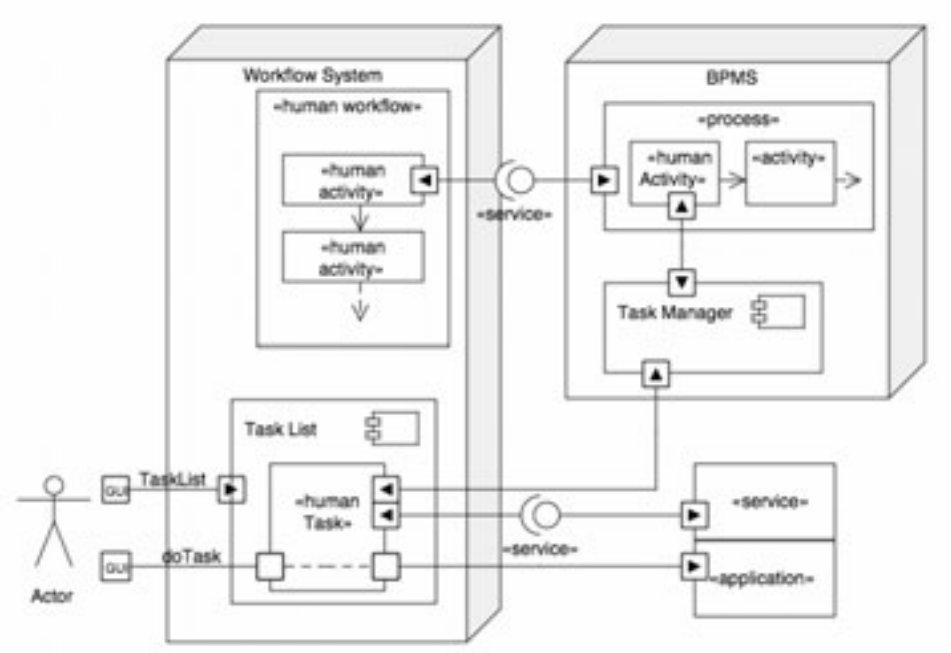


Fig. 4. Remote Task List Client.

BEPL4People allows the people information to be defined in the Process Input and a Human Task can use this information to determine the performer(s) of the Task. These are, however, remote host users. As in Fig. 5, what we need is (1) a user descriptor which describes the users of RAMS system and (2) the reference of a Task List Client to which these users belong and to which the Task Manager can redirect the Task information. In practice, probably a RAMS Task List Client will connect to the Task Manager and the Task Manager provides the List appropriate for that Task List Client. By selecting a Task on the Task List and telling it to Start (a Task life-cycle message), the end user will get the Task implemented by the Web Service.

The user descriptor has to be in a form both the Task List Client and the hosting system (RAMS) can coordinate to allow those users to access the intended Tasks. The details of a Task List Client and its working with the user authentication and access policies in BPEL4People (although they are implementation dependent and not defined in the BPEL4People specifications in the first place) have to be extended in order to allow this integration of the remote Process invocation.

As to the Task List Client reference, both the Task List Client and the Task Manager must recognise it in order to allow connections and to recognize sub-domains of the Tasks according the invoking hosts.

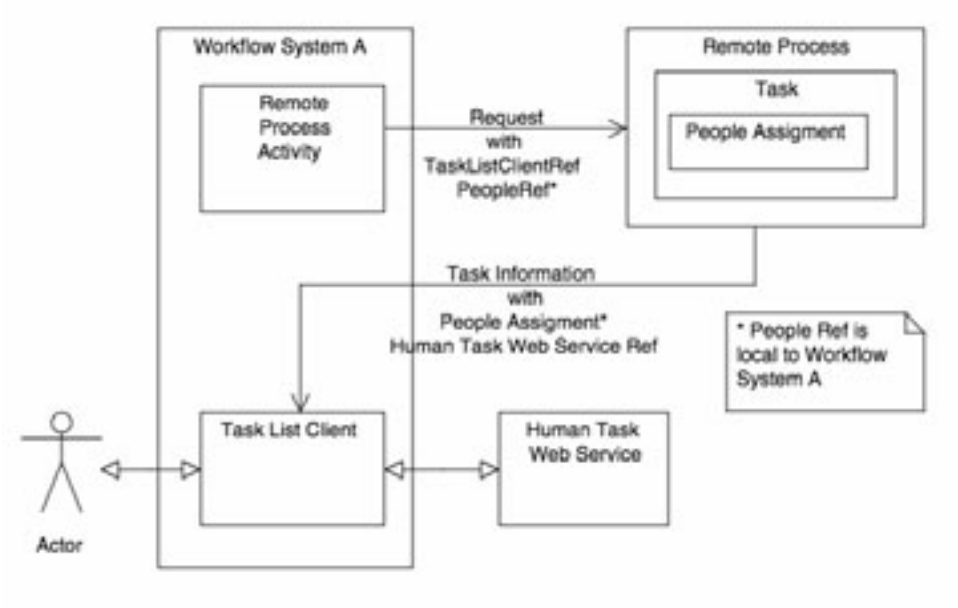


Fig. 5. Passing Task List Client and People References.

5. Asynchronous Call and Tool Activity

As it has been mentioned, in case of RAMS, Tool Activities are used to invoke remote processes. In fact, the Tools assigned to Activities give different abilities to Activities as Fig. 6. Therefore, in case of RAMS, for example, the ability to invoke remote Processes can be actually implemented with a Remote Process Tool.

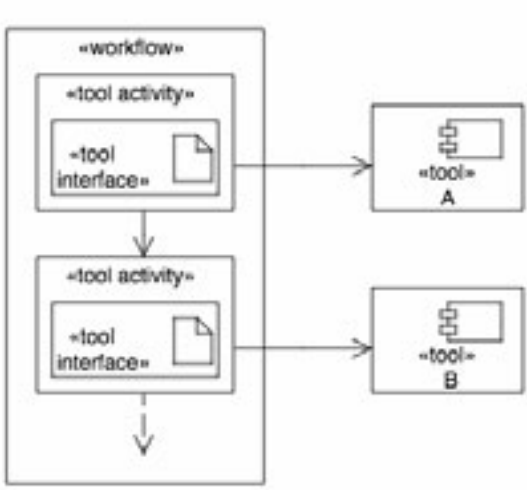


Fig.6. Tool Activities and Tools.

The Processes with Human Tasks may provide synchronous Process Web Service interface as Fig. 7 (A). Since it is most likely that they contain asynchronous calls to Human Task Web Services, it may be natural to postulate that those Processes are offered with asynchronous Web Service interface as Fig. 7 (B).

In these cases, however, unless we call the Tool from the Activity as asynchronous call as Fig 7 (C), the progress of the main workflow will be blocked, that is, it must keep the end user waiting at the Remote Process Activity. In case of RMAS, the system allows the user to log out at the Activity and to come back at the exact point to continue, this does not pose a strong problem. This is different from automated Processes (There is no user, and the automated Process blocks).

If we allow asynchronous calls from the Remote Process Activity to the Tool as Fig. 7 (C), the user will be able to move ahead on the workflow without waiting for the response. The user, then, must come back to the Activity to retrieve the results later.

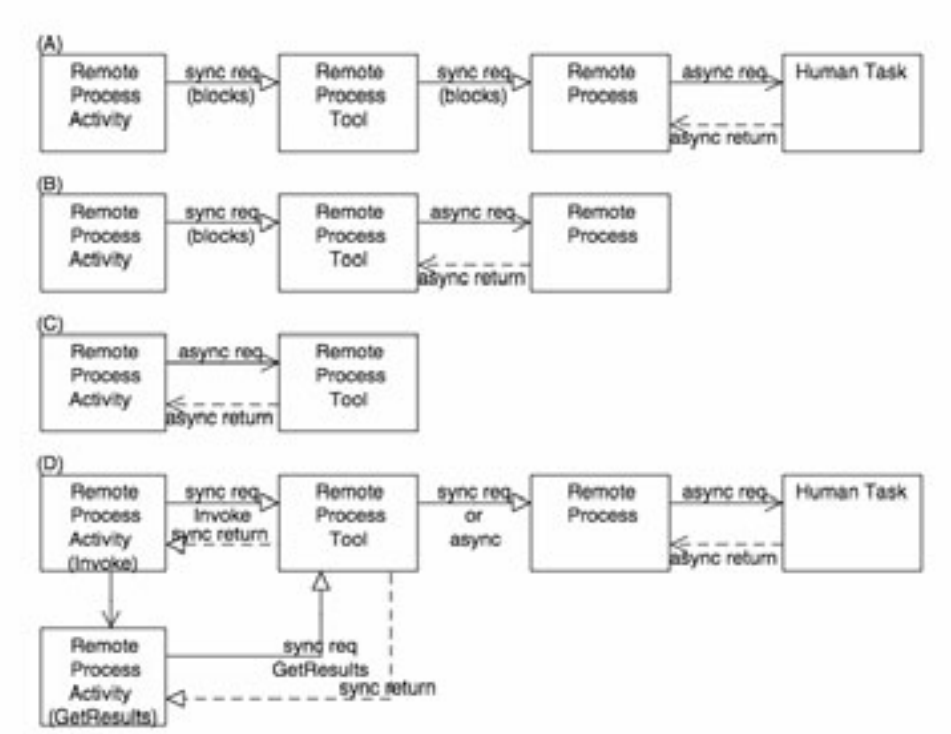


Fig. 7. Asynchronous Web Services Interface and Tools. With (B) and (C), the full details are omitted to draw attention to the asynchronous calls only.

Fig. 7 (D) is an alternative arrangement that a Remote Process Tool with two Activities. The first Activity submits the Remote Process invocation request. It returns immediately if the invocation request is successful at the Tool level, although the Tool itself my block

on the Remote Process if it sends a synchronous request to Remote Process. Later in the same workflow or in another workflow, another Remote Process Activity (e.g. GetResults) may be configured to send a synchronous request to the same instance of the Remote Process Tool. This will return either with the results of the Process or with "Remote Process still pending" message. This type of arrangement may be useful. Or, we may put a waiting Activity on a workflow with a periodic notification loop to check on the return of the asynchronous call. This would allow us to keep the job instance for the Tool maintained by the waiting loop.

6 Security for Remote Process Tool

Although the security is often out of scope of BPEL and BEP4People, we need to consider the outline of the security concerns.

Our local users have logged on the RAMS first, and we would like to see a Single Sign On (SSO) mechanism implemented between all the clients and servers involved. We assume that the Web Services Security is employed. Web Services Security uses a security token for authentication when a client connects to a Web Service. To do an SSO, this security token must be shared with all the secondary connections and with the clients.

First, a user authenticates with RAMS and the security token must be passed from RAMS to the Web Service interface of the remote Process. Then, the Task List Client must be able to securely connect to the Process host. Most likely, the Task Client's security context for the Task Manager is not shared on the remote BPMS, and the security token, which was already passed to the Web Service interface, must be propagated and correlated with the Task List Client.

Thirdly, the Human Task Web Services that will be called by the Task List Client via the Task Manager also requires the security token to be passed. Finally, the security token must be propagated from the Web Service context to Human Task application implemented by the Web Service and correlated with the Task Client which must log on to the Human Task Application.

6. Taverna Interaction Service

In case of Taverna, it has a simple Notification mechanism. As shown in Fig. 8, if we could send a remote workflow with an email address of the RAMS user, who would like to receive the notification when human interaction is required, it seems to solve the problem of integration.

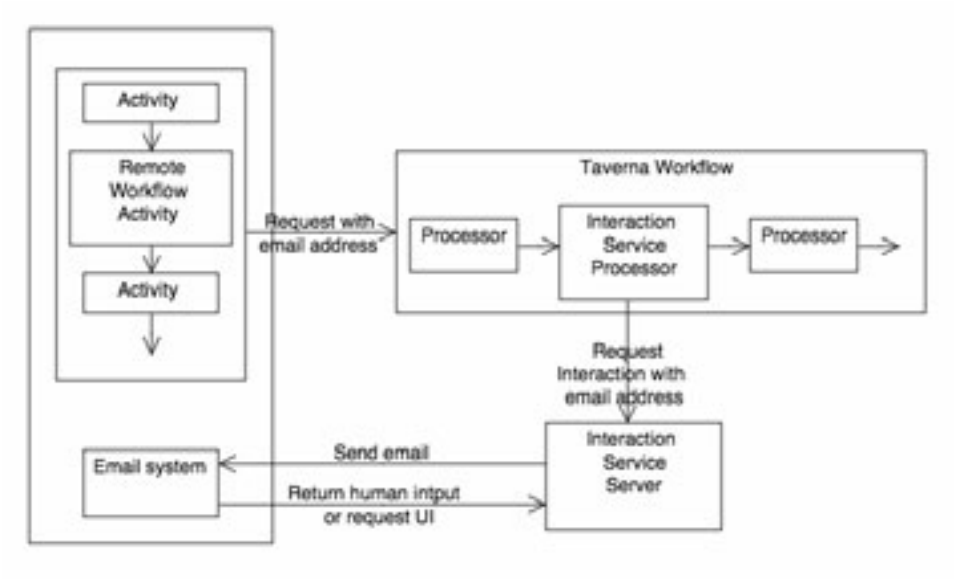


Fig. 8. Using a Dynamic Email Address with Remote Taverna Workflow.

8 Discussions and Conclusions

In order to be able to invoke remote Processes with human activities, this paper suggested modifications to BPEL4People to add the people reference and the Task List Client reference to its syntax. It also suggested that accessing the Task Manager of the remote system from the local system with its own Task List Client to handle the life-cycle of remote human activities. In case of Taverna Workflow, it suggested to send email addresses of local system users to the remote host. Both are the measures to allow remote systems to be accessed by the local system users.

In principle, the proposed models should be sufficient examples to allow the local system to work with remote human activities. The models also indicate the general principle that the local user information must be relayed from the local system to the remote system for this type of cooperation to take place.

When generalised, the models do not necessarily assume Web connections. When the Web connections are presumed, however, they are not always reliable and fault-tolerance ability of the Web connection model should be investigated.

The particular BPEL4People example presented relied on the availability of the Task List Client on the local system as defined in BPEL4People. It is relatively a simple element that can be provided by BPEL4People developers to the local system as a pluggable element, since the main logic resides on the remote server. If that is not the case, this approach may not be feasible. As the BPEL4People-based systems may

become more common, we expect that a sharable Task List Client, which can be deployed by external systems, may become readily available.

To demonstrate the usage of the Task List Client with RAMS, we have depicted it as it is installed alongside RAMS on the host. For better integration, however, it may be possible to make a Tool that uses the Task List Client inside the main system, e.g. RAMS. Then, it can handle remote human activities inside the local workflow activities.

Even if the Task List Client does not become readily available, the idea of integrating human workflows and Business Processes/Scientific workflows is still a valid cause, and we should consider other possible ways for integrations [Ho95, Ta07]. As we indicated, we are not aware of similar work that focuses on inter-workflow integrations, other than inter-workflow messaging. We hope that our work will provide a starting point for increasing similar studies.

Acknowledgements

This work was funded by the Australian Government Department of Education, Science and Training.

Bibliography

[Ag07a] Agrawal, A., Amend, M., Das, M., Ford, M., Keller, C., Kloppman, M., König, D., Leymann, F., Müller, R., Pfau, G., Plösser, K., Rangaswamy, R., Rickayzen, A., Rowley, M., Schmidt, P., Ivana Trickovic, I., Alex Yiu, A., Zeller, M.: WS-BPEL Extension for People (BPEL4People) Version 1.0, 2007. Available from http://xml.coverpages.org/bpel4people.html

[Ag07b] Agrawal, A., Amend, M., Das, M., Ford, M., Keller, C., Kloppman, M., König, D., Leymann, F., Müller, R., Pfau, G., Plösser, K., Rangaswamy, R., Rickayzen, A., Rowley, M., Schmidt, P., Ivana Trickovic, I., Alex Yiu, A., Zeller, M.: Web Services Human Task (WS-HumanTask) Version 1.0, 2007. Available from http://xml.coverpages.org/bpel4people.html

[Da06] Dalziel, J.: Research Activityflow and Middleware Priorities project proposal. 2006 Available from http://www.ramp.org.au/RAMP.MainBodyPublic.doc

[De08] Deelman, E., Gannon, D., Shields, M., Taylor, I.: Workflows and e-Science: An overview of workflow system features and capabilities. Future Generation Computer Systems, In Press. Corrected proof available on line.

[GD07] Ghiglione, E., Dalziel, J.: Design Principles for LAMS Version 2 and the LAMS "Tools Contract". Current Research on IMS Learning Design and Lifelong Competence Development Infrastructures, 2007. Available from http://www.tencompetence.org/files/Barcelona/presentations/session1/Ghiglione.pdf

[Ha05] Harrison-Broninski, K.: Human Interactions: The Heart and Soul of Business Process Management. Tampa: FL, Meghan-Kiffer Press, 2005.

[Ho95] Hollingsworth, D.: Workflow Management Coalition. The Workflow Reference Model, 1995. Available from http://www.wfmc.org/standards/docs/tc003v11.pdf

[Im03] IMS Learning Design specification V1.0, 2003. Available from http://www.imsglobal.org/learningdesign/index.cfm

[Le08] Lee, K., Paton, N.W., Sakellariou, R., Deelman, E., Fernandes, A.A.A., Mehta, G.: Adaptive Workflow Processing and Execution in Pegasus. Proceedings of the Third International Conference on Grid and Pervasive Computing Symposia/Workshops, pp 99-106, 2008

[LO08] Lanze´n, A., Oinn, T.: The Taverna Interaction Service: enabling manual interaction in workflows. Bioinformatics Applications Note, 24(8), pp. 1118–1120, 2008.

[Oa07] Web Services Business Process Execution Language Version 2.0. Committee Specification. OASIS Web Services Business Process Execution Language (WSBPEL) TC. 2007. Available from http://docs.oasis-open.org/wsbpel/2.0/

[Oi04] Oinn, T., Addis, M., Ferris, J., Marvin, D., Senger, M., Greenwood, M., Carver, T., Glover, K., Pocock, M., Wipat, A., Li, P.: Taverna: a tool for the composition and enactment of bioinformatics workflows. Bioinformatics, 20(17), pp. 3045-3054, 2004.

[OMG07] Object Management Group: UML Version 2.1.2. Available from http://www.omg.org/spec/UML/2.1.2/

[Ta07] Takayama, Y., Ghiglione, E., Wilson, S., Dalziel, J.: Collaborative Activity Human Workflow (CAHW) in eResearch. Available from https://spaces.internet2.edu/download/attachments/4788/Collaborative+Activity+Human+Workflow+in+eResearch.Final.pdf?version=1

Intelligent Service Management
ISM 2009

A conceptual information model for service management dimensions

Roman Belter, Rolf Kluge, Thomas Hering, Holger Müller

Information Systems Institute
University of Leipzig
Marschnerstr. 31
04109 Leipzig, Germany
{rbelter, rkluge, hering, hmueller}@wifa.uni-leipzig.de

Abstract: Service-oriented Computing (SOC) is known as the leading paradigm for the creation of agile and flexible enterprise IT infrastructures. The implementation of enterprise-wide Service-oriented Architectures (SOA) is a complex task. In most cases, more evolutionary approaches are used to deal with the arising complexity. However most of the design methodologies and implementation strategies focus on more technical, service realization specific aspects. Challenges regarding the definition and the management of related service artifacts throughout the whole service lifecycle are neglected. Also the implementation of a lifecycle-encompassing information management infrastructure is not addressed adequately in research and industry. In this paper we introduce different stakeholder roles and their information requirements as well as their influence on services within the lifecycle. Furthermore, this paper proposes a common service management information model (coSIM) that builds a foundation for the management of services and service infrastructures during design-, run- , and change-time.

Keywords: SOA, Service Management, Information Model, Governance, SOA Dimensions

1 Introduction

Service-oriented Computing (SOC) is a promising approach to face the increasing demand for business-aligned applications that provide the ability to react quickly to new requirements of continuously changing business environments. SOA is a design paradigm for IT systems that evolved from distributed computing and offers a way of designing a software system that provides service-based functionalities to end-users, applications, or other service systems [PTDL+06]. Today, an increasing amount of business applications are built as composed service applications [RBHS07]. Enterprises identified the tremendous advantage of SOC and continuously transform their applications and resources to services. However, since more IT-infrastructures follow the SOC paradigm, the number of services that are deployed and utilized, either from internal or external sources increases dramatically. Thus, management solutions that control available services and their service lifecycles and assist enterprises in managing the complexity of a service infrastructure, which grows exponentially with the number and type of services deployed, are needed. Especially in service infrastructures that span multiple enterprises, service providers are confronted with arising complexity in ensuring requirements such as security, policy, and governance additionally to other crosscutting business concerns such as manageability, scalability, and dependability [ADEF+07]. Enterprises will soon require solutions and infrastructure tools that fully support the management of SOC.

The remainder of this paper is structured as follows. Section 2 introduces the challenges in service management whilst section 3 outlines the dimensions that need to be taken into account for a common service management information model (coSIM). Section 4 describes the structure of the proposed coSIM. In section 5 we discuss related work and open issues. The following section 6 presents our summary and outlook to future work.

2 Challenges in Service Management

The problems which drive the management of service infrastructures are manifold. Infrastructure systems evolve from silos of technologies supporting only discrete applications and specific business processes to distributed service resources used in service compositions and various business processes. Infrastructure management must shift from focusing on the support of single applications, for specific user groups or business functions, to the support of complex infrastructure systems and service compositions supporting the needs of multiple groups utilizing service resources within different business functions across the whole enterprise.

The capability to encapsulate business functionality into separate computational services has led to a new way of how IT is used for business integration. The possibility to compose coarse-grained business functionality from finer grained service components is utilized to link different business units within an enterprise. On the down side, the overall management of distributed components, including their orchestration and tracing, as well as the integration of suppliers, partners, and customers through services poses new problems, including the need to handle a higher-level of infrastructural complexity and harder to maintain infrastructures [VS08]. Therefore a common information grounding providing specific, stakeholder tailored views of different aspects of the service infrastructure is a key requirement for an enterprise-scale SOA. Another challenge in service infrastructure management is the variable, hardly-predictable workload for service applications. One of the benefits of SOC is flexibility in supporting new business requirements by composing services to new business functions. Therefore, workload of a service is not predictable at the beginning of its lifecycle.

More and more services are exposed to customers, suppliers, and partners outside the company, leading to nearly unpredictable and traceable service utilization. To ensure the service availability recurring provisioning, configuration and optimization tasks need to be fulfilled, which require a complete understanding of the services provision requirements and dependencies. Such kind of service provisioning specific information is based and influenced on different decisions of various stakeholders during the service lifecycle and can only be provided by a service related coSIM. SOC forces infrastructure management to evolve towards service management. New paradigms need to be introduced within a service management solution, which turn management itself to a service that utilizes all available service related information to control and manage other services. One of the first steps to handle the increasing complexity of service management task, information and governance requirements as well as acting stakeholders will be to identify relevant information dimensions, analyze their associated requirements, information structure, artifacts, and dependencies. A second step will be to support the requirements and characteristics of these dimensions in a coSIM.

3 Dimensions of Service Management

This section outlines different aspects of a service computing infrastructure that need to be taken into account in developing a service information model. The following dimensions as well as the coSIM focus information model requirements only from a service provider perspective. Although service providers are generally supposed to act as service consumers too, this paper firstly aims on supporting the service lifecycle management issues on the provider side as service providers are due to meet QoS requirements issued by their consumers.

3.1 Service Lifecycle

A typical lifecycle of a service can be divided into the following different stages [SG05].

Idea is the first phase which indicates the general business-driven intention to create a new service which provides business functionality. Although, at this stage, service requirements are yet undefined, some information and management functions are already needed. These are the initial registration of a service entry in a service repository and the setup of a service information space to store all relevant service information and artifacts (created during the service lifecycle). During the service registration information like a rudimentary description of the planned service, service responsibilities, and information about the current service lifecycle stage are stored [SSOA07].

Creation, as the second stage, involves service definition, design, implementation, and testing tasks. It also includes the discovery and incorporation of already deployed and available services as one of the major advantages of SOA.

In the **deployment phase**, the service is introduced into its target environment. This involves acquiring of system resources and the provisioning of the service instance and all required components. From a management and service information perspective the relevant tasks at this stage are, e. g. the update of the lifecycle state and all service specific registry metadata [SSOA07], the subscription as new service consumer to all utilized operational consumed or supporting services as well as the setup of the state monitoring of this service instance during operation.

The **operational phase** of the service lifecycle is the main stage. The service is in use and provides its capabilities to different kinds of consumers like other applications or services. In the operational phase of the service lifecycle, a wide range of service management information are required and used to ensure the service functionality and operation. These include service consumer related information requirements like service description, service behavior and performance as well as service provider related information requirements like business activity monitoring metrics, service level definition coverage, or failure rate.

Service upgrades or changes are done within the **maintenance stage**, especially those tasks which cannot be carried out in operation stage. Service relevant tasks and information aspects are aligned with (re)configuration functionalities and resulting meta-information updates to the service registry.

The last stages at the end of the service lifecycle are **phase-out** and **archiving**. Both of them create service related artifacts and update existing service related information to ensure, that a service is not longer promoted through service registries nor intended to be used in available applications. In the achieved stage, to which services finally transit, the service provider should promote the information regarding the service retirement and available service substitutes.

This short section outlines that the maintenance of service lifecycles in a SOA can become very challenging and extensive, especially in complex and highly distributed service infrastructures. To handle this complexity a strong information and governance infrastructure with a coSIM needs to be established [SSOA07].

3.2 Stakeholder and Artifacts

A significant challenge in the establishment of a SOA in the area of Business/IT alignment is to deliver true value to the enterprises requirements. To achieve this, a widespread control and governance system needs to be introduced. SOA governance is such an approach that integrates an organization's people, processes, information, assets and artifacts [Bi08]. It ensures the creation, communication, maintenance and enforcement and adaptation of policies across the SOA lifecycle of design time, runtime and change time as well as measuring the overall effectiveness, SOA Governance mitigates many of the business risks inherent in SOA adoption [Er07]. Tasks of SOA Governance are e.g. the management of the service portfolio, documentation of connections and dependencies, planning for a continuous improvement and gradual modernization of the overall service infrastructure and landscape. A major and initially part in the establishment of governance structures is the definition of (SOA-specific) roles and persons in charge. The following roles are proposed as relevant stakeholders that produce or operate on artifacts within a SOA and therefore represent a service management dimension that needs to be taken into account (cf. [Ti07, Du08]).

The owner of the role **domain architect** (DA) is responsible for the overall coordination of the service landscape either in a specialized domain or for the whole enterprise-wide SOA. Therefore the DA has to ensure that the work of the individual service engineers (SE) is aligned in a global context in the meaning of coherency, quality and business strategy. The DA is the leading design authority regarding the service portfolio, service granularity, service dependencies, flexibility and service reuse. In this context the DA is the key role to deliver the business value of SOA.

The owner of the role **service engineer** (SE) is responsible for “good design” of a service in its context. This means the SE needs to find the optimal balance between suitability for the functional problem and re-use potential to ensure appropriate levels of reuse for other enterprise service applications. The SE defines the operations and therefore the granularity of a service. The decisions regarding communication styles and mandatory non-functional policies are also resided with the SE.

The **service developer** (SD) role is slightly similar to the well known role of the application developer. Generally this role is responsible for the implementation of the services operations-functionality based on the specifications from the SE and the business service owner (BSO). A reason to encompass the SD in the SOA-roles is twofold. Firstly the SD directly creates and operates on service related artifacts like service and implementation documentation, code documents, or deployment information. Secondly the SD is the “executing unit” of the SE and applies the platform and policies of the platform architect, which assigns an important feedback operation to the SD.

The **platform architect** (PA) defines the technical infrastructure stack and guidelines to make sure that all service implementations are compliant. This includes decisions for the "right" (combination of) technology (e.g. Web services, CORBA, J2EE/JMS [KBS04], REST [Fi00], etc.), supported standards (e.g. WS-I compliance, WS-Policy, WS-Security) as well as aspects of platform and standards maintenance, evolution, and enforcement. Furthermore the DA is responsible for all decisions related to the technical service landscape. For technical services like authorization, transformation, resource provisioning, or metadata management the PA is comparable to the DA.

The **business service owner** (BSO) is the person (or group of people) who will drive the strategy and decisions for the service from a business perspective. The owner of this role will be the final point for any business related issues to be resolved [Jo06]. Role owners are obliged to define the functional requirements for the service implementation and direct and control the service evolution and retirement. They own the functional scope of the service, the Service Level Agreements. The BSO is the information hub for all business related aspects of the services lifecycle and behavior and responsible for augmenting the business value the service delivers as well as the cost model to deliver that value.

The **technical service owner** (TSO) is responsible for all deployed services within an infrastructure. This includes business related (eventually composed) services as well as technical services that support other services in the SOA infrastructure and generally relate not directly to a business function (e.g. authentication, monitoring, transformation, etc.). The TSO is in charge for service related deployment and maintenance tasks, the management of operation level agreements in terms of availability, performance, and security as well as the maintenance of the overall enterprise structure including the technical implementation and enforcement of specifications and policies defined by the PA [Er07].

3.3 Infrastructure Maturity

One of the key learning's from enterprise-SOA projects is that SOA is an individual, evolutionary step-by-step approach to a new computing paradigm and infrastructure organization rather than a piece of software that is installed and provided in an over-night buy and provisioning process [Er07]. Since one accepted the evolutionary characteristic of SOC it is clear that we need metrics to define the actual level of the SOA evolution process of an enterprise. Such metrics a provided by SOA maturity models which deduce different maturity levels based on various technological as well as organizational criteria [RG08]. Important in the context of a coSIM is the fact that maturity models not only allow to detect the current evolution level, they also offer the opportunity to define general set of required SOA-roles, (associated) artifacts, and most important information requirements and governance processes that are necessary but also sufficient for the current level of evolution. The maturity of an enterprise SOA has a deep influence on the way service management should be organized and therefore is an important SOA dimension that should be represented in- and influence the characteristics and structure of an coSIM.

4 Information Model

The following section presents an initial structure of the proposed coSIM. In Fig. 1 we present a simplified view of the set of top-level elements, addressing the most relevant categories of service management artifacts. Each category (*BaseElement*) contains a complex and extensible structure of related elements, which either contain or reference properties and artifacts.

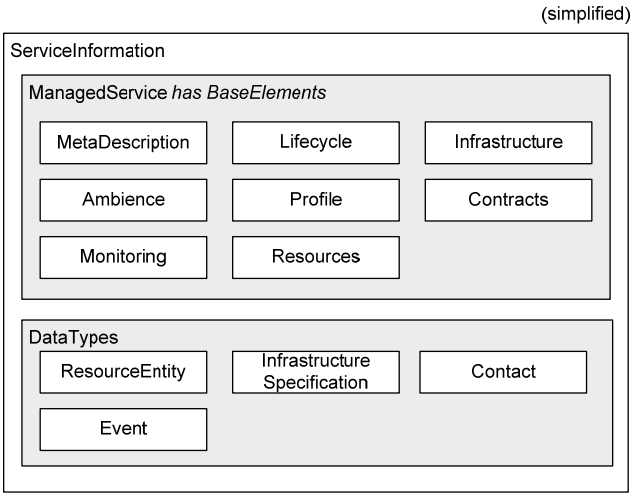


Fig. 1. Top-level elements of the coSIM

The entities of the coSIM structure can be divided into BaseElements which are designated to encompass information elements and are represented either by references to any arbitrary data type or as *DataTypes* defined by the coSIM model.

MetaDescription contains any kind of meta information of the service resource (comparable to the common information in service registries) like e.g. textual descriptions of the service functionality, address information, registry location, business context, associated organizational units, persons in charge (BSO, SD, TSO in stored as *Contact* data type), etc.

Lifecycle contains and archives information regarding the service-lifecycle history, the current lifecycle-stage as well as processed and planned stage transitions.

The *Infrastructure* element contains predominantly information of hard- and software related prerequisites, required technical services as well as deployment specifications (*InfrastructureSpecification* data type). This includes detailed information to version and configuration of used application- and web containers, dependency specification to backend systems (like databases) or other supporting infrastructure components, and any kind of technical condition this service resource needs for its operation.

Ambience is designed to hold information about the current services context with respect to its associated business/technical workflows. Furthermore it contains information about dependencies to and from other services. The primary information sources for ambience are e.g. references to BPEL documents created during the lifecycle-phases creation and deployment as well as monitoring information from the services operation phase.

Profile contains aggregated information that can be classified as key performance indicators on various behavioral aspects of the service. Service profiles are an up-to-date description of non-functional properties of the service. Profiles can be utilized for service comparison, service selection and measuring of business requirement alignment and fulfillment [AKMZ06]. The primary information sources are business activity monitoring (BAM) services within the infrastructure and log files of the service, which combined with specific calculus defined in the profile section of the coSIM, prepare and feed the indicators.

Contract contains instances or references to service level agreements and contracts that define the service related business-driven non-functional requirements. Depending on the architectural style of the service (atomic or composed) either a single or a composed service level agreement resource is referenced and monitored [LF08].

Monitoring primarily contains references to related logging resources of different types. Such types can be e.g. references to structured text document, BAM services or specific management information services that present aggregated historical logging information as outcome of management processes or event correlation systems [MHSV08, Lu01] e.g. as *Event* data type.

The information model element *Resources* is a general store for instances or references (*ResourceEntity* data type) to any kind of resources of the services lifecycle that can be not mapped to any of the above named categories. This includes e.g. different kind of design- and specification documents from the initial service design phases as well as change requests and change documentations from the operational phase.

Artifacts and information elements within the coSIM are represented according to the following rules:

1. All artifacts either if their structure is defined by a schema or any arbitrary binary data are stored in an external, infrastructure-wide reachable information store (e.g. as WS-Resources) and referenced by the coSIM instance.

2. Every information element based on defined DataElement-Types (properties, references, meta information) in the coSIM structure is directly stored in the coSIM instance.

The infrastructure maturity dimension is not directly addressed with the information model. Nevertheless maturity influences the complexity and the required elements which differ on each maturity level. The elements *ambience, profile* and *contracts* for example can be neglected on an early level. Other elements like *lifecycle, infrastructure*, and *monitoring* will be cut to a subset of required properties.

5 Service Management Research

Since the presented approach of a service management infrastructure and information model is based on research of a variety of domains this section presents an overview of the related work in the involved areas.

Papazoglou et al. presented the current state-of-the art and forthcoming challenges in the area of service infrastructure management [PTDL+06]. They clarify that service management is essential in SOC and encompass the control and monitoring of service applications through their complete lifecycle. They propose a management architecture concept and define the grand challenges for forthcoming SOC management infrastructures like self-configuring, self-adapting, and self-optimizing services.

A Web Service Manager introduced by Casati et al. focuses on involving business driven requirements in the management of service infrastructures by creating automatically controlled services driven by actual or predicted values of business metrics that possibly can be represented in the coSIM.

In the area of performance monitoring and anomaly detection Ghezzi et al. [GG07] present a monitor modeling language to define service workflow objects of interest which in combination with a proposed monitoring manager offer possibilities to monitor the execution of BPEL processes in detail and react in defined circumstances.

Berbner et al. [BGRH+05] propose an infrastructure that enhances the SOA concept with management functionalities and presents a service proxy-based monitoring prototype. Their work is concentrated on aspects of service discovery, service rating, composition, and monitoring of SLA requirements. Beside a valuable management approach, a service information model instance for each managed service is not in the scope of their paper. Furthermore, the management of the service lifecycle is not directly addressed.

Sahai et. al. [SG05] discuss required management functionalities for service infrastructures as well as possible tasks and metrics for service management. They propose approaches to align the business context to service management. However, they do not present a distributed solution lifecycle spanning information management solution to support the management of services as well as the business/IT alignment.

An interesting approach in the area of service description, extending the service description with semantic information is presented by Schröpfer et al. [SSOA07]. Their article proposes an OWL-S based service description that extends the regular UDDI service repository which possibly can be utilized in semantic annotations in a forthcoming version of the coSIM.

Müller et al. propose a conceptual framework for SOA management. Their technical-driven work combines a plug-in architecture with the concept of management workflows and primarily addresses the technical implementation of a service management system with management workflows that possibly can utilize the coSIM as source for required management processes according to the current lifecycle stage as well as common information storage for the resulting artifacts [MHSV08].

Rathfelder and Groenda [RG08] introduce a independent SOA Maturity Model (iSOAMM) which describes possible challenges, benefits and risks associated with different SOA maturity levels. It also reflects the implications on organizational structures and governance requirements and therefore provides an interesting source for relevant information requirements, management processes and involved stakeholders the coSIM should support in each maturity level.

6 Summary and Future Work

Current work in industry and research in the area of service management is primarily focused on technical aspects and solutions to support a service management infrastructure but the foundation for such supporting system – a common foundation for relevant information – is neglected. We proposed an initial conceptual approach in establishing a coSIM which can build the foundation to support the governance of a service infrastructure including different dimensions, stakeholder informational requirements and types of artifacts. Our future work on this concept will concentrate on a more detailed description of the information categories and their possible extensions, too. Furthermore we will evalutate technical opportunites for information collection during the lifecycle phases and will investigate possible technical representation alternatives that build the coSIM instances either as a decentralized federated information resource system based on e.g. WS-Resource Framework [CFFF+04, FFGT+04] or alternatively by facilitating existing repository systems to support the described service information model [So08, St07].

References

[ADEF+07] Arrott, M., Demchak, B., Errnagan, V., Farcas, C., Farcas, E., Kriiger, I.H., Menarini, M.: Rich Services: The Integration Piece of the SOA Puzzle. IEEE International Conference on Web Services (ICWS 2007), 2007, pp. 176-183

[AKMZ06] Abramowicz, W., Kaczmarek, M., Kowalkiewicz, M., Zyskowski, D.: Architecture for Service Profiling. Modeling, Design, and Analysis for Service-oriented Architecture Workshop: (mda4soa'06) in conjunction with the 2006 IEEE International Conferences on Services Computing (SCC 2006) and Web Services (ICWS 2006), Vol. 0. IEEE, Chicago, 2006, pp. 121-130

[BGRH+05] Berbner, R., Grollius, T., Repp, N., Heckmann, O., Ortner, E., Steinmetz, R.: An approach for the Management of Service-oriented Architecture (SoA) based Application Systems. Enterprise Modelling and Information Systems Architectures, Proceedings of the Workshop in Klagenfurt. GI, 2005, pp. 208-221

[Bi08] Biske, T.: Implementing SOA Governance, 2008

[CFFF+04] Czajkowski, K., Ferguson, D.F., Foster, I., Frey, J., Graham, S., Sedukhin, I., Snelling, D., Tuecke, S., Vambenepe, W.: The WS-Resource Framework, 2004

[Du08] Dubray, J.-J.: Establishing a Service Governance Organization. InfoQ.com, 2008

[Er07] Erl, T.: SOA Principles of Service Design. Prentice Hall, 2007

[FFGT+04] Foster, I., Frey, J., Graham, S., Tuecke, S., Czajkowski, K., Ferguson, D., Leymann, F., Nally, M., Sedukhin, I., Snelling, D., Storey, T., Vambenepe, W., Weerawarana, S.: Modeling Stateful Resources with Web Services, 2004

[Fi00] Fielding, R.: Architectural styles and the design of network-based software architectures. 2000

[GG07] Ghezzi, C., Guinea, S.: Run-Time Monitoring in Service-Oriented Architectures. Test and Analysis of Web Services, 2007, pp. 237-264

[Jo06] Jones, S.: Enterprise SOA Adoption Strategies. C4Media Inc., 2006

[KBS04] Krafzig, D., Banke, K., Slama, D.: Enterprise SOA: Service-Oriented Architecture Best Practices (The Coad Series). Prentice Hall PTR, 2004

[LF08] Ludwig, A.; Franczyk, B.: COSMA - An Approach for Managing SLAs in Composite Services. In: Bouguettaya, A., Krueger, I., Margaria, T. (eds.): 6th International Conference on Service Oriented Computing (ICSOC 2008). LNCS, vol. 5364, Springer, Berlin, Heidelberg, 2008, pp. 626–632

[Lu01] Luckham, D.C.: The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems. Addison-Wesley Longman Publishing Co. Inc., 2001

[MHSV08] Mueller, I., Han, J., Schneider, J.-G., Versteeg, S.: A Conceptual Framework for Unified and Comprehensive SOA Management, 6th International Conference on Service Oriented Computing (ICSOC 08), Sydney, 2008

[PTDL+06] Papazoglou, M.P., Traverso, P., Dustdar, S., Leymann, F., Krämer, B.J.: Service-Oriented Computing: A Research Roadmap. In: Cubera, F., Krämer B., Papazoglou M. (Ed.): Service Oriented Computing (SOC). Dragstuhl Seminar Proceedings, Dagstuhl 2006, pp. 1-29

[RBHS07] Repp, N., Berbner, R., Heckmann, O., Steinmetz, R.: A Cross-Layer Approach to Performance Monitoring of Web Services. Emerging Web Services Technology, 2007, pp. 21-32

[RG08] Rathfelder, C., Groenda, H.: iSOAMM: An Independent SOA Maturity Model. Distributed Applications and Interoperable Systems, 2008, pp. 1-15

[SG05] Sahai, A., Graupner, S.: Web services in the enterprise: Concepts, standards, solutions, and management. Springer, New York, 2005

[So08] Software AG: The SOALink Cookbook: A guide to extending and integrating with CentraSite, 2008

[SSOA07] Schröpfer, C., Schönherr, M., Offermann, P., Ahrens, M.: A Flexible Approach to Service Management-Related Service Description in SOAs. Emerging Web Services Technology, 2007, pp. 47-64

[St07] Strnadl, C.: Bridging Architectural Boundaries Design and Implementation of a Semantic BPM and SOA Governance Tool. Service-Oriented Computing – ICSOC 2007, 2007, pp. 518-529

[Ti07] Tilkov, S.: Roles in SOA Governance. InfoQ.com, 2007

[VS08] Virili, F., Sorrentino, M.: The enabling role of Web services in information system development practices: a grounded theory study. Information Systems and E-Business Management, 2008

FinGrid Accounting and Billing

Houssam Haitof[1], Hans-Dieter Wehle[2], Michael Gerndt[1]

[1]Institut für Informatik
Technische Universtität München
Boltzmannstr. 3
85748 Garching, Germany
{haitof, gerndt}@in.tum.de

[2]IBM Deutschland Research
& Development GmbH
Schönaicher Str. 220
71032 Böblingen, Germany
hdwehle@de.ibm.com

Abstract: For a commercial entity to entrust the Grid for its business operations either as a consumer or a provider of resources, mechanisms that would guarantee its interests need to be implemented. Especially resource usage tracking and billing. In this paper, we present our experience with designing and building an autonomic accounting and billing system for the Financial Grid (FinGrid). We used a service oriented architecture for FinGrid, we relied on open standards and recommendations for our accounting system and on knowledge representation and reasoning to model our billing infrastructure.

1 Introduction

Grids consist of a virtual platform for computation and data management using a heterogeneous cluster of computer resources [BHF03]. It enables users and applications seamless access to vast IT capabilities. In his reference paper [Fos02], Foster provides a three-points checklist that a system has to fulfill before it could be identified as a Grid. The first is that [the Grid] *coordinates resources that are not subject to centralized control*. Meaning that not only the Grid resources are geographically distributed but that those resources belong to different administrative domains, i.e. different institutions or departments. Issues like security, usage policies, accounting and billing arise. Grids are often based on the "best-effort" principal that does not guarantee a sophisticated level of quality assurance. This may be quite satisfactory for an academic usage, however, Grids are more and more used and adapted for a commercial usage. And for a commercial entity to entrust the Grid for its business operations either as a consumer or a provider of resources, mechanisms that would guarantee its interests need to be implemented. In the frame of the FinGrid project, we built a general purpose, standard-compliant, Grid accounting system that tracks resources usage for tasks, like the management, SLA enforcement or billing. We also built a rule-based billing system, seamlessly integrated with the accounting module. The billing system allows to generate bills from the the resource usage according to a set of billing policies. All the components of FinGrid are composed within a SOA model.

The Financial Service Grid (FinGrid) is a project funded by the German Federal Ministry of Education and Research, to develop a Grid architecture to virtualize services and processes in the financial sector and to build banking Grid services based on an accounting

and pricing infrastructure through the development of several prototypes. In this context, we pursue research on the necessary components for a financial Grid to better model an industrialization and pricing scheme. We draw the architecture and implemented the resulting accounting and billing services. Sections 2 is about FinGrid SOA model, sections 3 and 4 will present FinGrid accounting and FinGrid billing respectively. Section 5 introduces our system architecture. In section 6 we will talk about the related work in the field and in section 7 we present our future work.

2 FinGrid SOA Model

Service Oriented Architecture (SOA) guidelines and web services technologies can be used to construct a solution for a flexible model for Grid management that would tackle part of the Grid management complexity challenges. According to OASIS [MLM+06], SOA is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. Some of the main drivers for SOA are to *facilitate the manageable growth of large-scale enterprise systems and to facilitate Internet-scale provisioning and use of services*. It revolves around the concept that needs or requirements of one party are met by the capabilities of another. The parties at either ends can be person or a software agent. Applying SOA principles to the Grid seems to be a natural process. The Grid is composed of a set of distributed resources under the control of different administrative domains, and SOA is a model for organizing such system. In fact, the standardization efforts for the Grid are channeled towards the adoption of web services technologies that lay the necessary infrastructure and building blocks for a Service Oriented Grid Architecture.

In a SOA, the central mechanisms for coupling needs and capabilities are services, defined by the capability of performing work for another, specifying the work offered for another and the offer to perform work for another. The first step to meet SOA objectives is done through decomposition or factoring of complex systems into smaller chunks for more convenient design, implementation and maintenance. Those smaller chunks are what services are supposed to be: small independent components easier to manage and control. The SOA model does not preconize the specific use of web services, however they constitute the used de-facto standard.

FinGrid accounting and billing contains several small services composed together to constitute an autonomic management layer whose function is to automatically collect usage information from the different Grid nodes using collector agents and produce usage bills. Every service is independent and can be recomposed in a different context. The main FinGrid services are the Grid accounting, Grid billing, Grid rules Manager, job submission and collector agents. In this paper we present the accounting and billing architecture and thus concentrate on the accounting, billing and collectors services.

3 FinGrid Accounting

The purpose of this component is to provide an interface for collecting (upload) and accessing (download) accounting information of the Grid resources consumption and provision. Accounting is the collection of information and data on the usage of Grid resources resulting in a report of the resource consumption and/or provision. The resulting report is generally used for capacity planning, trend analysis, auditing, billing and/or cost allocation. Gathered accounting information is in the form of an XML document complying with the Usage Record (UR) Recommendation[MLMJM07] proposed by the Open Grid Forum (OGF).

3.1 Usage Record

The UR recommendation specifies a common format for representing resource consumption data. It contains accounting and usage information gathered at the local Grid sites. The usage metrics are divided into three categories: base properties, differentiated properties, and extensions. The base properties define the most common metrics necessary for proper accounting such as user and job identification. The differentiated properties are job-related measurement metrics that every Grid site can accommodate to its particular needs. The last category is the extensions, which are a set of metrics specific to the Grid site and jobs that can be added to the UR specification. The set of required items to be accounted is, of course, site and situation dependent. The listing below shows a sample UR document generated in our testing environment.

```
 1 <urwg:UsageRecord>
 2 <urwg:RecordIdentity urwg:createTime="2007-11-20T10:
    59:30Z" urwg:recordId="2007-11-20T10:59:28Zfingrid.
    boeblingen.de.ibm.com@griduser"/>
 3 <urwg:JobIdentity>
 4 <urwg:GlobalJobId>https://9.152.4.12:8443/wsrf/
    services/ManagedExecutableJobService?4499c150-974f-
    11dc-ab80-852cb8646fdc</urwg:GlobalJobId>
 5 <urwg:LocalJobId>fingrid.boeblingen.de.ibm.com#119555
    2763#147.0</urwg:LocalJobId>
 6 </urwg:JobIdentity>
 7 <urwg:UserIdentity>
 8 <urwg:LocalUserId>griduser</urwg:LocalUserId>
 9 <ds:KeyInfo>
10 <ds:KeyName>/O=Grid/OU=GlobusTest/OU=simpleCA-fingrid
    .boeblingen.de.ibm.com/OU=boeblingen.de.ibm.com/
    CN=Grid User</ds:KeyName>
11 </ds:KeyInfo>
12 </urwg:UserIdentity>
```

```
13 <urwg:JobName urwg:description="">UR test
14 </urwg:JobName>
15 <urwg:Status urwg:description="">Done</urwg:Status>
16 <urwg:Host urwg:description="">fingrid.boeblingen.de.
    ibm.com</urwg:Host>
17 <urwg:CpuDuration urwg:description="">PT0S
18 </urwg:CpuDuration>
19 <urwg:WallDuration urwg:description="">PT0S
20 </urwg:WallDuration>
21 <urwg:StartTime urwg:description="">
    2007-11-20T10:59:28Z</urwg:StartTime>
22 <urwg:EndTime urwg:description="">
    2007-11-20T10:59:28Z</urwg:EndTime>
23 <urwg:SubmitHost urwg:description="">fingrid.
    boeblingen.de.ibm.com</urwg:SubmitHost>
24 <urwg:Queue urwg:description="">Condor</urwg:Queue>
25 <urwg:Disk urwg:description="" urwg:storageUnit="KB">
    10000</urwg:Disk>
26 </urwg:UsageRecord>
27 </urwg:UsageRecords>
```

3.2 Metrics

Metrics are the measurable values gathered from the resources and represented in the UR document. For the purpose of our implementation, we used the standard metrics of the UR recommendation as well as a set of custom metrics as extensions. The custom metrics offer capabilities that were necessary to our use cases but that are not provided by the recommendation. They relate to the categorization of clients and resources and to the representation of the cost for the resource usages.

4 FinGrid Rule-based Billing

The purpose of this component is to provide an interface for generating usage bills. Billing is the process of generating bills from the resource usage data using generally a set of predefined billing policies. The bill can be in real money or it can use more abstract notions depending on the Grid site general policies. We should note here that the billing service does not preconize the use of a specific economic model. In fact it is independent from the economic model to be used. FinGrid Billing permits the definition, storage and manipulation of billing rules through a set of services. The rules are expressed in either a declarative rule language or simpler (but less expressive) *business* languages. They can be updated and deployed at run-time without any need to recompile or restart any part of the application.

4.1 Representation

We use knowledge representation to describe the different business rules. By using a knowledge based representation, we are constrained to use a logic-based language. Knowledge-based systems can be viewed at a symbolic level, or a knowledge level[New82]. Representation language formalism lies at the knowledge level, where we are concerned with the expressive adequacy of the language and its entailment relation. Logic being the study of entailment and rules of inference, the tools of formal symbolic logic are ideally suited for a knowledge representation system[BL04]. The choice for a logic rule language is dictated by the following criteria: entailment characteristics, expressiveness, Objects representation, Computational complexity, adequacy, extensibility, standard compliance.

4.2 Inference Engine

The inference engine is a reasoning system that keeps its actual knowledge in a database like structure called the *working memory*. The working memory gets updated in real-time with the changes in the system state. The inference engine's task is a three steps cycle[BL04]:

- recognize the rules that are applicable, i.e. rules whom prerequisites are a bill computation for instance.
- resolve conflict among the resulting rules.
- act accordingly by changing the working memory and firing the appropriate actions.

It is called inference engine because it matches facts with the rules to *infere* actions. The facts are stored in the working memory whereas the rules are stored in what is called *production memory*. Facts maybe added or removed from the working memory at run-time depending on the data received. A system with a large number of rules and facts may find at a certain time several rules to be true for a specific working memory state. Chances are that some of those rules would conflict (shutdown the database server vs. keep a high availability for premium clients to access the database). The inference engine needs then to implement conflict resolution strategies.

There are three types of inference engines. Forward-chaining types, backward-chaining types, and hybrid inference engines. Forward-chaining is *data-driven*. Facts are added in the working memory, and the inference engine looks for applicable rules with a conditional part being true, then adding the result to the working memory and evaluating again the rules against the facts until no new rule is fired. Backward-chaining is *goal-driven*. The inference engine is given a goal to reach and looks for rules with results matching the goal. Our business rules are deductive rules and foreward-chaining engines are the most appropriate for this type of rules. We are using Drools[Dro] as our inference engine. Drools is production rule system using an enhanced implementation of the Rete[For82] algorithm. Rules can be written in Drools Rule Language (DRL) or, using *expanders*, Drools provides the possibility to use Domain Specific Languages (DSL) by defining the

language semantics. We defined two DSL (technical DSL and natural DSL) that we are using in conjunction with the more complex DRL.

4.3 Billing Rules

We support two types of billing schemes: duration-specific and one-off cost. The duration-specific type applies the billing rule based on the usage duration whereas the one-off cost type is concerned with the proper usage of the resources. The following is an example of a duration specific rule written in the technical domain-specific language, it specifies 0.0002 unit (Euro, Pounds...), for every second of usage of the described resource and user type:

```
when EVENT = "VM Assignment", CLIENT_TYPE = "Platinum",
     RESOURCE_TYPE = "BLADE Type 4",
     RESOURCE_AGE < 240 * 60 * 60,
     SERVICE_LEVEL = "Platinum" then
      COST_PER_SECOND = 0.0002
```

A similar one-off cost rule in the natural DSL:

```
when the event is "VM Assignment" using a resource type
      "BLADE Type 4" and
     the user type is "Platinum" then the one of cost
      is 15
```

The second rule written in DRL:

```
1 rule "rule_2"
2 when
3 e:BillingEvent(event="VM Assignment",
   resourceType="BLADE Type 4", clientType="Platinum");
4 then
5 OperationResult fact = new OperationResult();
6 fact.setCost(10);
7 fact.setMessage("rule_2 asserted successfully");
8 e.setOperationResult(fact);
9 update(e);
```

The technical DSL as well as the natural DSL have very limited semantics and are intended to rules managers that are not programmers. The DRL is much more powerful and offer more expressiveness.

5 Architecture

5.1 FinGrid Accounting Architecture

Figure 1 shows our general architecture. Using the FinGrid portal, the user can submit jobs through the Community Scheduler Framework (CSF) or directly to a specific resource using GRAM. The collectors take care of gathering usage data and storing it in the records repository through the exposed FinGrid Accounting Interface.

5.1.1 Accounting Portal

The accounting portal is a web application (Figure 2) built using AJAX. It serves as a front-end to the FinGrid accounting service. It offers various management functions and operations on the usage records as well as the support for XPath for advanced queries. The accounting portal is integrated with the billing interface. Users can mark records for billing from the accounting portal.

5.1.2 FinGrid Accounting Service

For our accounting interface, we implemented OGF's Record Usage Service (RUS) [ANM05]. RUS is a stable OGF draft defining a basic infrastructure for accounting and auditing, it specifies the service interface to normalize operations on the accounting information of the resource usage as described by the Usage Record specification. The UR document can be maintained either centrally or in a distributed fashion. Our accounting service interface permits the upload of usage records or the extraction of necessary information and possibly the aggregation of resource usage data. It normalizes the operations on Usage Records documents that are stored in a persistent XML database.

5.1.3 Collector

We use collectors to gather accounting information from the local resources. A Collector is resource environment-specific agent that collects accounting data generated at the level of the Grid resources to which it belongs. This data is then submitted to the Grid Accounting Service in the form of UR documents. There is no proper standard for usage logs at the level of the resource manager. The collectors' task is to extract the relevant data from the generated logs and transform it into compliant OGF UR-WG document. For every Grid node, a collector agent instance is created whenever there is a job assignment, and the usage data is automatically collected and dumped to the RUS server. Currently we support Unix fork and OpenPBS. We should note here that some resource manager such as Platform LSF[LSF] provide logs directly in the UR format.

5.1.4 UR Repository

The RUS standard does not specify how the UR should be physically stored and leaves this decision to the implementers. We opted for the most natural way to store XML doc-

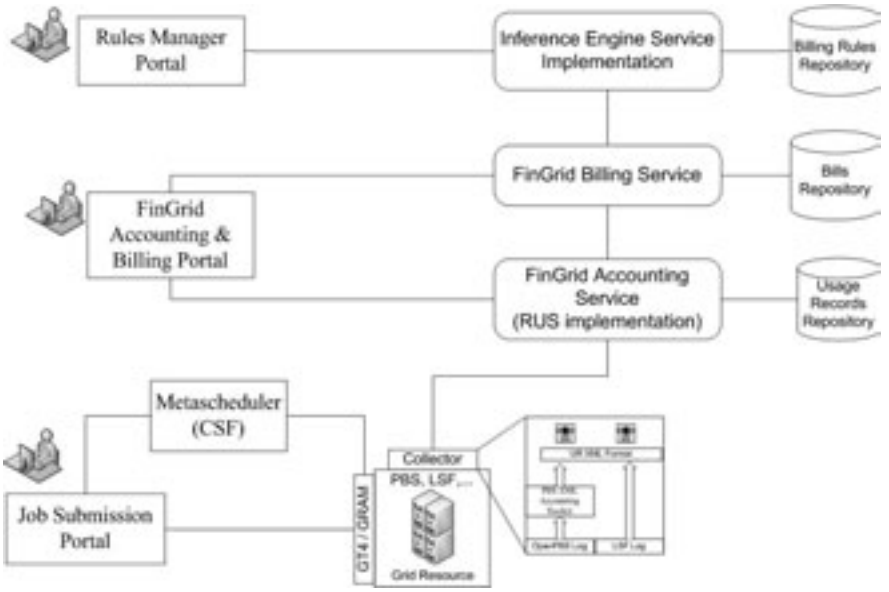


Figure 1: FinGrid general architecture

uments which is a native XML database. Our choice settled on Xindice[Xin] for the persistent storage. The advantage of using a native XML database is that we don't have to worry about mapping our XML document to a specific data structure to store in a normal RDBMS for instance. We store documents as XML and we retrieve them as XML. Xindice supports XPath 1.0 for querying XML documents which is very handful for aggregating results from different stored Usage Records, and XUpdate 1.0 for updating XML documents. Our implementation supports WS-Security as security layer for authentication and authorization.

5.2 FinGrid Billing Architecture

Our billing model separates the logic of the billing service from the data or the rules and uses a *hot-deployment* scheme where users can manage the billing rules and deploy them without restarting the billing service. This separation along with the possibility to write rules in more user-friendly languages, make our services accessible to a wide audience of users. The kind of users that would generally specify the billing policies but would not be necessarily a programmer or a computer savvy.

5.2.1 Billing Portal

In the billing portal we can mark single or multiple usage records for billing, generate usage bills and export the bills as XML files. However, sometimes we would need a higher

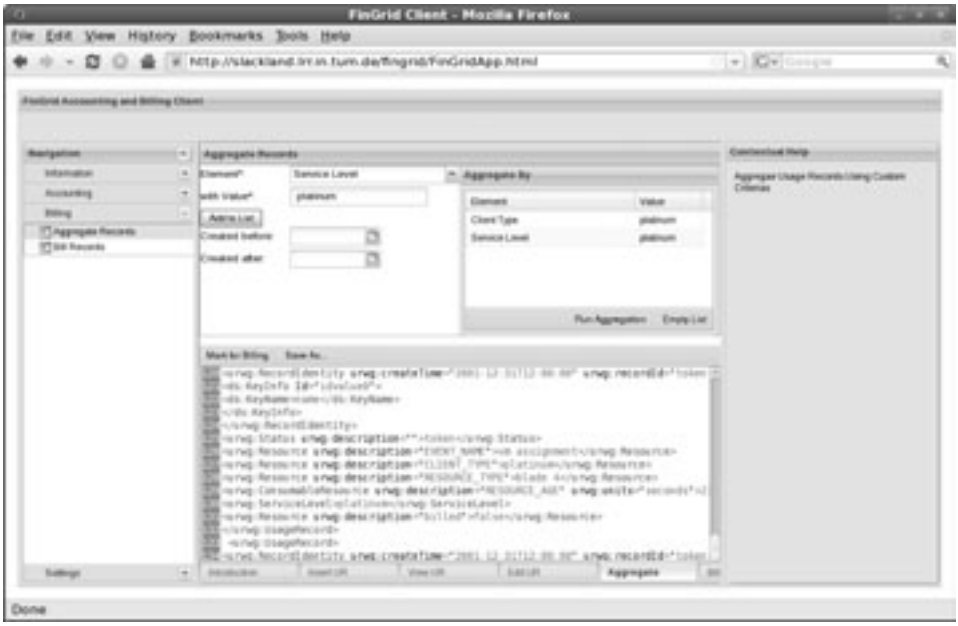


Figure 2: Aggregation support in the accounting portal

degree of control over which data to be billed. Our implementation is powerful enough to support aggregation of usage records over any combination of usage metrics elements as well as ranges of times. It also supports aggregation of usage data using complex XPath queries, giving the user a high degree of liberty in composing usage records. Records of virtual organizations or some specific parts of job can be then created using aggregation for a detailed billing document.

5.2.2 FinGrid Billing Service

The billing service provides port types to generate bills from usage data according to predefined billing rules. The records can then be marked as billed. We can also bill records created from aggregation results and save the aggregation for later reference. Our billing service implementation is a front end to the Drools inference engine. It parses the usages records and extract the relevant usage metrics for the billing process. It feeds this data to the inference engine working memory and fires the evaluation process. The bills generation process is not automated and needs to be initiated. However, we do have a command line version of our billing service that can be used with cron for instance for automated and scheduled bills generation.

5.2.3 Inference Engine and the Rules Manager Portal

As mentioned earlier, we are using Drools as our inference engine and we extended Drools BRMS to offer a management portal for the FinGrid billing rules. Users can write or edit rules written in any of the supported languages and deploy them without the need to recompile or restart the application, thanks to the separation between the application logic and the billing rules.

6 Related Work

Various work has been done in the field of Grid accounting and billing. The Distributed Grid Accounting System (DGAS)[ABG+] intends to implement resource usage metering, accounting and resource pricing in a distributed Grid environment. It supports decentralized banking structure where the billing occurs before the job submission. It also supports billing using various billing metrics. However, they tend to use proprietary solution and protocols for representing and exchanging accounting data. The Grid Accounting Service Architecture (GASA)[BB03] developed within the context of the Australian GRIDBUS project is another related work that supports different payment strategies (post-payment, pre-payment and pay as you go). It also supports billing using different billing metrics. It has, however, a centralized billing server and does not also adhere to the use of standards. The last example is the Swedish SweGrid Accounting System (SGAS)[SGE+04], an OGSA-based accounting architecture with decentralized banking structure. It supports a service-oriented architecture with an implementation of the UR recommendation. However, it only supports one metric (clock time per node) for the billing and is unable to track usage data on heterogeneous resources.

7 Future Work

In FinGrid, everything is a service but the Grid node themselves. As future work, we intend to change that by representing the Grid resources as services for ease of composition and management. We already identified the important points to achieve this and the approach that we should follow. It is clear to us that we need to describe the resources as services and provide a way to easily manage those services.

7.1 Describing Resources as Services

Web services are stateless application components, which are not suitable to describe and interact with Grid resources being logical or physical (servers, storage media,...) that need to maintain a state. Web services need therefore to define custom means to preserve state, discover other resources and interact with them. Standardization efforts were made to tackle this issue. The most prominent is the Web Services Resource Framework (WSRF)

set of specifications by OASIS [Ban06]. The WSRF provides a general solution using web services to an originally specific problem: describing and representing Grid resources. Another relevant feature of WSRF is that it brings a solution for management of a resource lifetime, faults and properties. A resource that is described in this way is called a WS-Resource. Rendering resources as WS-Resource and decomposing software component into services is the first step toward an SOA enabled management architecture with all the advantages that it can bring such as ease of management, adaptability and automation.

7.2 Managing Services

Considerable work has been done to define an architecture to manage web services, the most notorious are WSDM [BVWS06] and WS-Management [MMR08]. WSDM stands for Web Services Distributed Management and is composed of two sets of specifications: WSDM-MUWS, Management Using Web Services (MUWS) and WSDM-MOWS, Management Of Web Services (MOWS) specifications. MUWS specification defines how can we expose any resource as a manageable resource and is built on top of WSRF and WS-Notification [GHM06]. Seeing the importance of those new specifications, the consortia behind them decided to reconcile WSDM and WS-Management specifications into a single standard for management of IT resources using Web services [ea06].

8 Conclusion

We have presented the architecture and implementation of the FinGrid accounting and billing services based on open standards and recommendations. Our architecture is serviceoriented and modular. We used a rule-based approach for the billing service. This approach permits to detach the service control from the actual code enabling the business users to change the service behavior without the intervention of an IT staff, thus, enhancing greatly the application adaptability in a world where business rules may change on a daily basis. Our system is currently deployed at our academic and industrial partners for evaluation and testing.

References

[ABG+] C. Anglano, S. Barale, L. Gaido, A. Guarise, G. Patania, R. Piro, F. Rosso, and A. Werbrouck. The Distributed Grid Accounting System (DGAS). *http://www.to.infn.it/grid/accounting*.

[ANM05] J. Ainsworth, S. Newhouse, and J. MacLaren. Resource Usage Service (RUS) based on WS-I Basic Profile 1.0. *UR*, August 2005.

[Ban06] T. Banks. Web Services Resource Framework(WSRF) - Primer v1.2. *Official Committee Specification*, May 2006.

[BB03] A. Barmouta and R. Buyya. GridBank: a Grid Accounting Services Architecture (GASA) for distributed systems sharing and integration. *Parallel and Distributed Processing Symposium*, April 2003.

[BHF03] F. Berman, A. J. G. Hey, and G. Fox. *Grid Computing: Making the Global Infrastructure a Reality*. John Wiley and Sons Ltd, 2003.

[BL04] R. J. Brachman and H. J. Levesque. *Knowledge Representation and Reasoning*. Morgan Kaufman, 2004.

[BVWS06] V. Bullard, W. Vambenepe, K. Wilson, and I. Sedukhin. Web Services Distributed Management. *Official Committee Specification*, August 2006.

[Dro] JBoss Drools. http://www.jboss.org/drools/.

[ea06] J. Antony et al. WSDM/WS-Man Reconciliation. *IBM Report*, August 2006.

[For82] Charles Forgy. Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem. *Artificial Intelligence*, 19:17–37, 1982.

[Fos02] I. Foster. What is the Grid? A Three Point Checklist. *GRIDToday*, July 2002.

[GHM06] S. Graham, D. Hull, and B. Murray. Web Services Base Notification 1.3 (WS-BaseNotification). *Official Committee Specification*, October 2006.

[LSF] Platform LSF. http://www.platform.com/.

[MLM+06] C. Matthew MacKenzie, Ken Laskey, Francis McCabe, Peter F. Brown, and Rebekah Metz. OASIS Reference Model for Service Oriented Architecture V 1.0. *Official Committee Specification*, August 2006.

[MLMJM07] R. Mach, R. Lepro-Metz, S. Jackson, and L. McGinnis. Usage record - Format Recommendation. *UR*, 2007.

[MMR08] R. McCollum, B. Murray, and B. Reistad. Web Services for Management (WS-Management) Specification. *DMTF*, 2008.

[New82] A. Newell. The knowledge level. *Artificial Intelligence*, 18(1):87–127, 1982.

[SGE+04] T. Sandholm, P. Gardfjell, E. Elmroth, L. Johnsson, and O. Mulmo. An OGSA-Based Accounting System for Allocation Enforcement across HPC Centers. *Proceedings of the 2nd International Conference on Service Oriented Computing*, pages 15–19, November 2004.

[Xin] Apache Xindice. http://xml.apache.org/xindice/.

A Medical Diagnosis System based on MAS Technology and Neural Networks

Christina Klüver, Jürgen Klüver, Rainer Unland

COBASC Research Group
Department of Economy
Universitätsstr. 12
45117 Essen, Germany

Institute for Computer Science and
Business Information Systems (ICB)
University of Duisburg-Essen
Schützenbahn 70
45117 Essen,Germany

{c.stoica-kluever, juergen.kluever}@uni-due.de
rainer.unland@icb.uni-due.de

Abstract: Reliable, cost-efficient, and fast medical diagnosis is still a challenge in today's world. This paper presents a medical diagnosis system that combines the advantages of multi-agent system technologies and neural networks in order to realize a highly reliable, adaptive, scalable, flexible, and robust diagnosis system for diseases. The medical diagnosis system consists of a structured alliance of medical experts - realized by agents - that collaborate in order to provide a viable medical diagnosis. Each agent has a certain responsibility. The agents rely on a reactive pattern-based matching process. Their power is substantially augmented by standard kinds of neural networks (interactive neural nets and self organizing maps), which represent the diagnosis capacity of the agents.

1 Introduction

In today's global world a fast and reliable medical diagnosis generation is of eminent importance as can be seen, for example, from the problems with SARS or the bird flu. Such highly contagious and lethal diseases can threaten the world if they are not fought immediately and with high efficiency and reliability. However, to do so, it is, first of all, necessary to quickly and surely diagnose this disease in case a person is suffering from it - and this regardless of where this person currently may stay in the world. While, after a short while, the identification of the disease at its hot spots may become routine its diagnosis at more remote/unlikely places will remain the challenge.

Research on medical diagnosis has been an important topic in computer science for quite a while, especially in the field of artificial intelligence. The biggest drawback of most proposals published so far is that they implement monolithic systems that cannot be extended easily and that are difficult to maintain. In contrast to that a multi-agent system-based architecture is proposed that relies on simple neural networks to implement the reactive behavior of the proposed agents. Thus, the medical diagnosis system exhibits many of the key properties of such paradigms as self-organization, emergent behavior, simplicity, the capability of fast and unbounded learning, flexibility, robustness, adaptivity, self-sustainability, and easy extensibility.

This paper is structured as follows. The next section will discuss the state-of-the-art. Section 3 introduces the basic concepts and terminology. Section 4 will give a deeper insight in how the agents are realized. Since the idea is to build a complex diagnosis system from simple components the agents rely on several types of simple neural networks. Finally, section 5 concludes the paper.

2 State of the Art

Medical diagnosis has been intensively discussed as an application of artificial intelligence (AI) techniques, especially of expert systems (cf., e.g., [Po82]). The probably best-known expert system in this direction is MYCIN (cf. [Sh76], [BS84]). It is a program for advising physicians on treating bacterial infections of the blood and meningitis. Even though MYCIN does not exhibit common sense it does reasonably well, provided the user has common sense and understands the limitations of MYCIN. The problem with MYCIN, as well as with expert systems in general, is its lack of flexibility, adaptability and extensibility. As long as rule bases are relatively small in size, expert systems can be highly effective. However, the extension of rule-bases, e.g., as part of a learning process, can cause serious problems since it can easily happen that the overall semantics and behavior of the rule base gets out of control. Moreover, the extension to different or new areas of diagnosis is everything but straightforward. It easily exhibits the limitations of these systems. For these reasons, expert systems had success mainly in specific, well-defined fields of expertise. As soon as expert systems are meant to cover broader areas of expertise, they run into serious problems (cf. [Po82]). Multi agent systems (MAS) are much more flexible since each agent is supposed to cover only a very limited but concise area of knowledge. Complex and broad areas of expertise are dealt with by a common effort of a number of cooperating agents. Due to this "natural" modularization the overall system is highly flexible and adaptable. By adding new units (agents) the system can naturally extend its capabilities. It is a gradual, controlled growth (adding of some new knowledge) instead of the necessity to extend a monolithic knowledge base. We do not want to discuss the different approaches that have evolved in the area of AI. Instead, we will concentrate on approaches that have emerged in the area of distributed AI, especially multi-agent systems.

Today, agent technology is on the brink to penetrate health related applications. However, given the high potential of this technology in this area it is surprising how few projects deploy this technology yet. Applications can mainly be found in the areas of patient monitoring, health care and patient management. A good overview of applications of agents in the health care domain can also be found in [MN03]. The book identifies a number of fields of application within health care in which agent technology has been applied. Examples are patient scheduling, organ and tissue transplant management, community care, information access, decision support systems, training, and internal hospital tasks and senior citizen care. Nealon and Moreno [NM03] give a good overview about the applications described in the book. The other articles deal with the above application areas in some detail. However, none of the articles is especially related to medical diagnosis.

3 (Intelligent) Agents and Multi-Agent Systems

Intelligent agents can be regarded as autonomous, problem-solving computational entities with social abilities that are capable of effective pro-active behavior in open and dynamic environments. While there are a number of definitions of intelligent agents (cf., e.g., [WJ95], [Wo02], [Wo01], [We99], [OMG04], [RG95]) the most relevant properties in the context of this paper are the following:

1. *Autonomy*: An intelligent agent has control over its behavior, i.e., it operates without the direct intervention of human beings or other agents, and has control over its internal state and its goals.

2. *Responsiveness/Reactivity*: An intelligent agent perceives its environment, and responds in a timely fashion to changes that occur in it in order to satisfy its design objectives.

3. *Pro-activeness*: An intelligent agent is goal directed, deliberative, opportunistic, and initiative. Due to its goal-directed behavior the agent takes initiative whenever there is the opportunity to satisfy its goals. It especially may react pro-actively to changes in its environment; i.e., it responds to it without being explicitly asked for it from the outside.

4. *Social ability*: An intelligent agent is capable of interacting with other agents (and possibly humans) in order to satisfy its design objectives as well as combined or organizational goals.

5. *Intelligence*: an agent has specific expertise and knowledge. Thus, it is capable to solve problems that fall into its domain of expertise.

There are quite a number of different types of agents. For a good overview have a look at [OMG04]. This paper will only concentrate on one type, namely reactive agents. A *reactive* agent does not contain a representation of a central symbolic world model. Nor does it utilize complex symbolic reasoning. Reactive behavior implies that the agent immediately responds to stimuli from its environment. Such stimuli can either be perceived changes in the external world or received communications from other agents. Based on this received information as input they produce output by simple situation-action associations, often implemented by pattern matching. The behavior of a reactive agent is in general infinite since it continuously senses for and responds to stimuli from its environment. Such a pattern of behavior is called stimulus-response behavior. Reactive agents are sometimes also called behavior-based or situated agents [MFP01].

A *multi-agent system* (MAS) consists of a collection of individual (intelligent) agents (problem solvers). Either hierarchical, heterarchical, partially or flat structured communities of agents perform joint operations or decision making. Their underlying means are communication, collaboration, negotiation, and responsibility delegation, all of which are based on individual rationality and social intelligence of the involved agents (cf. [MFP01], [UU04], [Un03]). The capability of a MAS surpasses the capabilities of each individual agent. Reduction of complexity is achieved by decomposing the overall task into a number of well-defined sub-tasks, each of which being solved by a specific agent. To solve collaboratively complex problems that exceed the capabilities of each individual agent a set of agents may temporarily join forces. However, unlike hard-wired cooperation domains, these coalitions or teams are very flexible. Depending on the organizational structure agents may autonomously join and leave the coalition whenever they feel like provided their commitments are fulfilled.

4 Structure and Operations of the Agents

The previous sections discussed the general possibilities MAS offer for the construction of a medical diagnosis system. This section will concentrate on the architecture and design of the individual agent types. In order to fulfill the requirements for the medical diagnosis system the agent types have to exhibit, first of all, a simple pattern based behavior. Such kind of behavior is especially relevant on the way down; i.e., when a diagnosis request infiltrates the medical diagnosis system and traverses down along the hierarchy. On the way up, more sophisticated tasks are to be performed:

- All proposed diagnoses are to be evaluated, compared and classified.
- In case that the data base did not contain enough information for a solid diagnosis proper further measures are to be suggested in order to verify the diagnosis.
- In case that a diagnosis was reached with sufficient probability an explanation for the outcome and possible treatments to tackle the disease are to be proposed.

To deal with such complex and demanding tasks an agent has to reveal deliberative behavior. In order to be able to come to reasonable results within the project it was decided to only tackle the basic requirements. Thus, the decision was, for the time being, to concentrate first of all on the reactive features of agents only and leave the deliberative behavior for future work if it will turn out that it is needed. Decision about the quality of an answer to a request is done on a simple basis of how many symptoms of those that were identified by the doctor were involved in the diagnosis finding process of the delivering agents and how many symptoms an agent was expecting from the data base in order to have all information available.

Another important requirement is learning of the individual agents by increasing and adapting their knowledge and its accuracy. In the diagnosis system this is supposed to take place by a feedback circle. After the diagnosis was sent to the receiver some feedback is to be returned as soon as the receiver is capable to do so, e.g., immediately if the receiver excludes the diagnosis or after a while, as soon as the diagnosis is verified or refuted by the course of the disease. This feedback needs to be as specific as possible in order to allow the agents to learn from it.

Currently, we are still in the stage of carefully investigating how exactly an agent architecture has to look like in order to fulfill the above requirements best. Built on experiences we made with neural networks in other projects [KS04], [KK07] we believe that agents that rely totally or at least partially on simple neural networks are the best candidates. Since the agents need to be able to learn from one another and, in particular, to evaluate the information obtained from other agents, we are experimenting with different types of simple neural nets. One advantage of differently structured agents is that the agents may discover artifacts generated by agents with a different structure. Two types of neural nets turned out to be the most promising ones, namely Kohonen Feature Maps (SOM) and Interactive Nets (IN). In contrast to multi-layered nets or many ad hoc rule-based expert systems of the MYCIN-type these types operate with a simple architecture. In order to compare the different results we gave single IN and SOM the same task; a MAS was constructed by the combination of different IN that specialized on different diseases.

Kohonen Feature Map or Self-organizing Map (SOM)

In contrast to supervised learning nets like, e.g., perceptrons, SOMs belong to the type of non supervised learning networks. They operate according to the learning rule "winner takes all", i.e., only neurons with the highest activation values pass their activation on to other neurons. The activation function is mainly a sigmoid function. The result of a training process of SOMs is a clustering of information. In other words, SOMs generate an explicit order of data that is only implicitly given.

There exist several types of SOMs. In the proposed diagnosis system a Ritter-Kohonen type is used (cf. [RK89]) where the data is ordered according to a *semantic matrix*. A Ritter-Kohonen SOM, therefore, contains two matrices: a variable weight matrix like usual learning neural networks and the semantic matrix. Only the latter needs to be constructed by the user.

The structure of the semantic matrix can be illustrated by the following example: The information given in implicit order refers to particular diseases and contains symptoms like *fever, cough, headache, muscle pain* and so on. This information has to be given to a SOM in form of a matrix like the one in Table 1.

Disease concepts / Symptoms	Angina Pectoris	Influenza	Bronchitis	Arthrosis	Pneumonia	Arthritis	Myocardial attack
Fever	-1	1	1	-1	-1	-1	-1
Cough	-1	1	1	-1	1	-1	-1
Headache	-1	1	-1	-1	-1	-1	-1
Muscle pain	-1	1	-1	1	-1	1	-1
Swelling	-1	-1	-1	1	-1	-1	-1
Vomitus	1	1	-1	-1	-1	-1	1

Table 1: Example of a semantic matrix

The task of the SOM is to cluster the disease concepts according to the provided information (symptoms). It performs this task according to the so-called Winner-takes-all principle, i.e., combining certain concepts with respect to the neurons with the highest degree of activation. On the basis of this example, the SOM will, for example, combine *Angina Pectoris* and *Myocardial attack* because their semantic vectors have the greatest similarities[1].

SOMs have been used only seldom in medical diagnosis (e.g. [TF02]), however, as far as we know never Ritter-Kohonen types. The advantage of this type is the easy construction of the semantic matrix. It can be enlarged easily if new knowledge is acquired.

[1] Since we have only taken a small portion from the set of possible symptoms these combinations do not reflect the reality. However, our purpose here is to show how a semantic matrix is constructed.

A medical diagnosis Ritter-Kohonen SOM will contain a semantic matrix with the attachment of symptoms to the according diseases.

Construction of a medical Ritter-Kohonen SOM

Table 1 shows a relevant portion of the semantic matrix (altogether, this SOM contains seven diseases and 23 symptoms):

Operations:

On the basis of the provided data basis those symptoms are activated that were identified by the doctor. In the first example (see Fig. 1) the first three and the fifth symptoms are activated. The SOM is clustering the diseases according to the semantic matrix and stabilizes after about 1000 learning cycles. In the visualization the clustering is generated according to spatial neighborhoods, i.e., the diseases that belong together with respect to the activated symptoms are grouped in the same spatial region.

The result is satisfactory in the sense that the diseases are correctly identified. The Ritter-Kohonen SOM is as good in performing medical diagnosis as the standard diagnostic systems; however, it can be extended and cloned much easier. Moreover, it can learn from other agents by simply enlarging its semantic matrix.

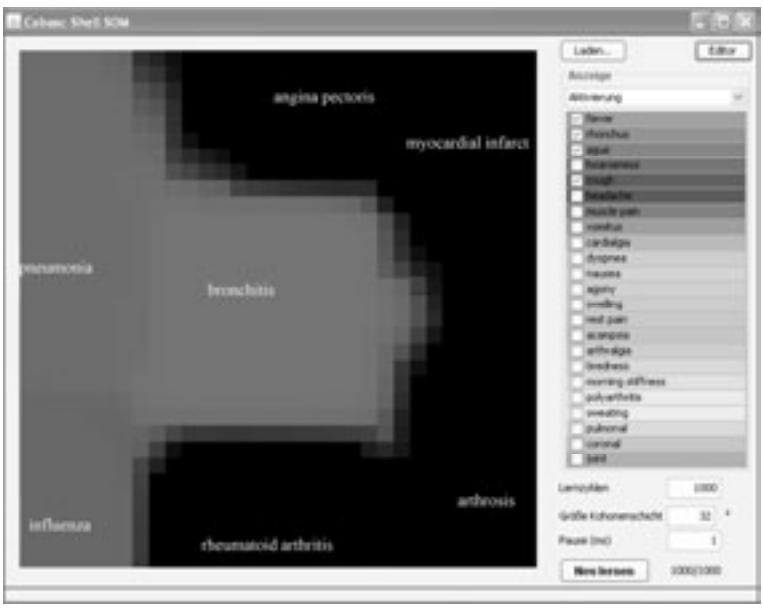


Fig. 1: Medical diagnosis based on symptoms

In the second example (see Fig. 2) the regions of different diseases are activated. The question is which diseases belong to certain corporeal regions. The result shows that the different diseases are clustered according to their occurrences in the respective corporeal regions.

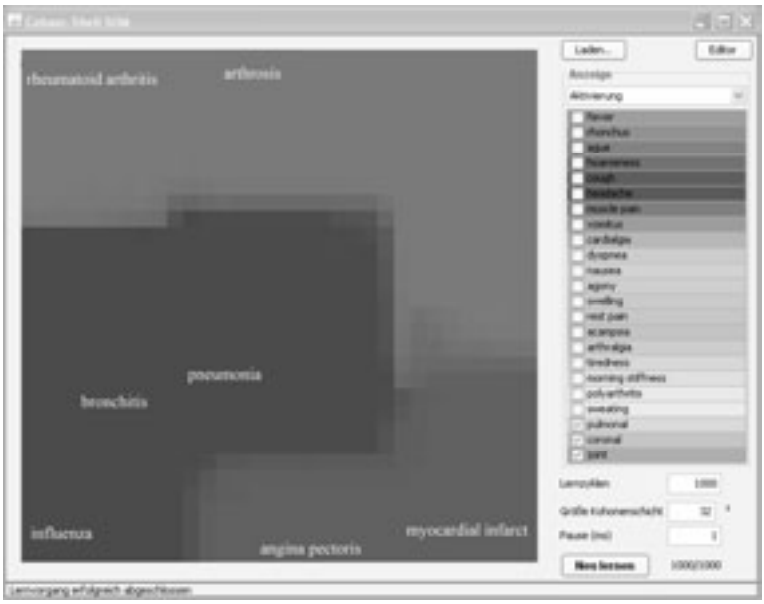


Fig. 2: Regions of different diseases

Interactive neural nets (IN)

IN are recurrent networks that are usually not trained. The user has to construct the weight matrix according to his/her particular problems. It is, of course, possible to generate suited weight matrices by optimization algorithms, e.g., genetic algorithms. Recurrent means that, in principle, all neurons can be connected with all others in a direct manner, i.e., the weight matrix may contain no zeroes. INs are particularly well suited for analyzing logical and semantic relations (cf. [WP85]). In a strict sense medical diagnosis is basically the semantic combination of certain symptoms with certain diseases. That is why INs are used as a second type of agent.

As a first testbed for medical diagnosis a 30 ∗ 30 weight matrix as maximum was chosen. The columns and rows of the matrix consist of seven diseases and 23 symptoms. The weight values are real numbers from the interval [–1, 1].

When using the IN one starts by "externally activating" the symptoms of the particular case, i.e., these neurons are activated with a particular real number. The size of the number depends on the intensity of the symptoms. All other neurons have the activation value of zero. The IN uses the well-known linear activation function and, if the weight matrix is adequately constructed, it reaches a point attractor. In the first example the attractor was reached after 16 runs.

Example 1:

The IN in example 1 starts with an external activation of the first five symptom neurons with a value of 0.15. The attractor consists of the strong activation of the bronchitis neuron (0.66) and the weak activation of the pulmonal neuron. The activation value of all other diseases remains zero. Therefore, it can securely be concluded that the respective disease is Bronchitis (see Fig. 3).

Example 1:

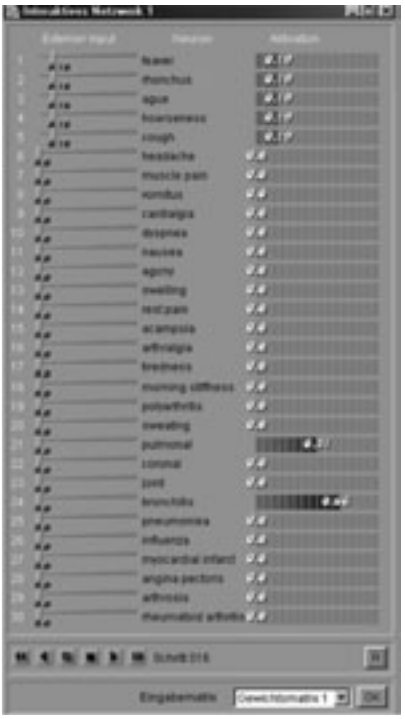


Fig. 3: Unambiguous diagnosis of bronchitis

Example 2:

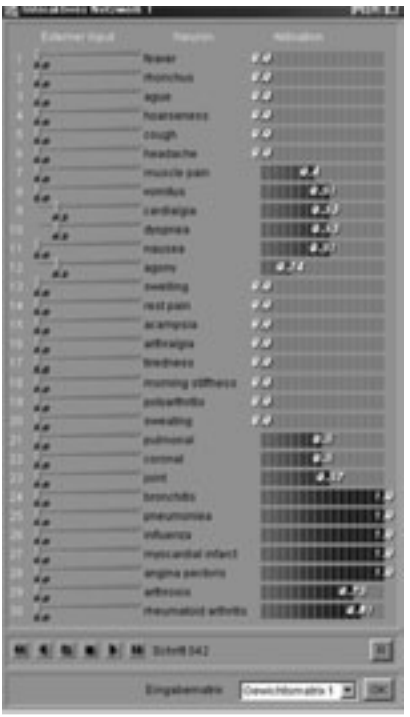


Fig. 4: Indifferent diagnosis possible

Example 2:

The IN in example 2 has the external activation of heartache, dyspnea and mortal agony. In this case the symptoms apparently are too general; therefore, the IN is not able to uniquely identify a disease (see Fig. 4). This can be concluded because five diseases are activated with values of 1 and two other diseases are also activated, although with lower values. It can be concluded that the IN needs to get more input.

Both the SOM and the IN are simple systems that cannot be compared with advanced ES like the later MYCIN-types with respect to their diagnostic capabilities. But they are only a small part of a much bigger system and, in particular, they will interact with other agents of the same type. Learning from other agents, for example, consists in the correction or extension of the semantic matrix in the case of SOM-type agents or in the correction and extension of the weight matrix in the case of IN-type agents. Because these are quite simple operations they can easily be integrated into a system of many different agents. It is even possible to combine these types with, e.g., evolutionary algorithms in order to optimize their learning capabilities.

An IN-MAS

One serious problem with such usage of INs as diagnosis system is the size that an IN needs to store a lot of information. Already the rather simple prototype that is shown in Fig. 3 and 4 contains 30 units and an according weight matrix of 900 values. If such formal systems become enlarged they become quickly rather unintelligible and difficult to understand, although it is possible to generate them in an automatic way. For this reason we constructed a diagnosis system that consists not of one IN but of several networks that interact. This extended model is shown in Fig. 5:

Fig. 5: Extended model consisting of different networks

When a user inserts an input containing a list of symptoms the administrative instance selects that particular IN that contains most of the symptoms in its list. Then this IN externally activates those symptoms that it has as a subset of the input symptoms. If the IN generates a unique solution it is given to the user with the question if the user considers the solution as adequate. If that is the case the process is finished.

If the IN does not generate an unique solution and/or if the user is not satisfied the networks begin to interact, i.e., the first IN checks which other IN has also some of the symptoms the user inserted and which the first IN had not. The first IN takes over the other symptoms – and also other diseases if the new symptoms are also related to these diseases – and adds them to its own lists. Subsequently the first IN tries again, i.e. it externally activates the completed list of symptoms and searches for a unique solution. If that is possible the new solution is given to the user. If no unambiguous solution can be obtained even after the first IN has "learned" in that way from all other INs that contained some of the input symptoms then the user will be informed that the whole system is not able to answer the question. The user will be requested to gather more information about the medical case in question and to add the new information to the system.

Learning new information from outside is done in a similar way. If new lists of symptoms and the diseases related to them are inserted into the system then either one or several of the already existing networks will be enlarged in the manner described above or, if these networks become too large, new networks will be generated that "specialize" on the new information.

One technical challenge of the IN and the SOM is the construction of the necessary matrix, i.e., the weight matrix in the case of the IN and the semantic matrix of the SOM, have to be (manually) constructed, respectively enlarged for each particular (set of/instances of) disease(s) and the according set of symptoms. Therefore, the nets of the agents get their matrices with the help of a so-called *categorizing algorithm* (CaA). The CaA roughly operates as follows:

A particular set D of diseases d and an according set S of symptoms s is given to the net. The net fills its respective columns and rows of its matrix as follows:

1. A 1 is inserted at each place in the underlying (semantic or weight) matrix where a $d \in D$ and an $s \in S$ intersect (these places are called w_{sd}).

2. For all other units w_{dx}[2] of the matrix CaA inserts a 0, if there is no confirmation about possible or probable relationships to other diseases or symptoms.

In this way the whole systems automatically generates its own agents by transforming the information into adequately structured different nets. The whole system, therefore, is self-organizing and self-enlarging.

[2] All places for which $x \notin S$ is true.

The advantages of this type of diagnosis system with respect to other forms are rather obvious: They are generated via a comparatively simple construction technique; the different networks can be automatically generated and in the same way the system can be enlarged by adding new networks. In addition the solutions the system gives to a user are rather easily to understand because the system can always explain which IN with which lists of symptoms and diseases has generated the output solution. It is well known from the practical experiences with diagnosis systems that this criterion is very important for laymen as users (laymen in computer science). Because IN are simpler in their architecture and their operational logic than SOM our further work on network MAS will concentrate on IN as agents. One of the next steps will be to make our MAS web based.

First runs with real cases from the practice of an ophthalmologist obtained rather satisfying results although the system is still in a stage as a prototype. In 18 from 20 cases the ophthalmologist agreed with the diagnosis of our MAS; that is in 90%. With respect to the two other cases the doctor admitted that the diagnosis of our system could be correct too. He got another opinion because of additional facts with respect to the patients not known to the system Hence, the performance of our system is at least comparable to that of other known diagnosis systems. Further practical results with this diagnosis system will be reported in due time.

6 Conclusion

The main advantages of our system are a) the simple architecture, b) its robustness, c) simple possibilities to enlarge the knowledge of the whole system, and, last but not least, d) an easy usage by laymen, i.e., medical doctors. If the system will become web based the enlargement of the knowledge can be done by interested medical experts and by this way the system will grow via its users.

Currently we are implementing the first prototype of the proposed system with the help of a FIPA-based multi-agent system platform. The proposed will use it as a test bed to find out what types of neural networks will work best to implement the proposed agent types. From experiments and simulations with the proposed system we expect a deeper understanding of those parameters that influence the quality of the outcome of our system.

References

[Po82] Pople, H. E.: Heuristic Methods for Imposing Structure on Ill-Structured Problems: The Structuring of Medical Diagnostics. Chapter 5 in Szolovits, P (Ed.): Artificial Intelligence in Medicine. Westview Press, Boulder, Colorado, 1982

[Sh76] Shortliffe, E. H.: Computer-Based Medical Consultations: MYCIN. Elsevier, New York, 1976

[BS84] Buchanan, R. G., Shortliffe, E. H.: Rule-Based Expert Systems. Addison Wesley, 1984

[MN03] Moreno, A., Nealon, J. L.: Applications of Software Agent Technology in the Health Care Domain, Birkhäuser Verlag, Berlin, 2003

[NM03] Nealon, J. L., Moreno, A.: Agent-Based Applications in Health Care. In: Moreno, A., Nealon, J. L. (eds): Applications of Software Agent Technology in the Health Care Domain, Birkhäuser Verlag, Berlin, 2003

[WJ95] Wooldridge, M., Jennings, N. R.: Intelligent agents: Theory and practice. The Knowledge Engineering Review, 10 (2), 1995, pp. 115-152

[Wo02] Wooldridge, M.: An Introduction to Multiagent Systems. John Wiley & Sons, 2002

[Wo01] Wooldridge, M.: Intelligent Agents: The Key Concepts. In: Maík, V., Stepánková, O., Krautwurmová, H., Luck, M. (Eds.): Multi-Agent-Systems and Applications II: 9th ECCAI-ACAI/EASSS 2001, AEMAS 2001, HoloMAS 2001, Springer Publishing Company; Lecture Notes in Artificial Intelligence, Vol. 2322, 2002, pp. 3-43

[We99] Weiss, G.: Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence. MIT Press, 1999

[OMG04] Object Management Group, Agent Working Group (2000) Agent Technology Green Paper; Document ec/2000-03-01; http://www.objs.com/isig/ec2000-03-01.pdf, 2004

[RG95] Rao, A., George, M.: BDI agents: From theory to practice; Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95), San Francisco, 1995, pp. 312-319

[MFP01] Marík, V., Fletcher, M., Pechoucek, M.: Holons & Agents: Recent Developments and Mutual Impacts. In: V. Maík, O. Stepánková, H. Krautwurmová, M. Luck (Eds.): Multi-Agent-Systems and Applications II: 9th ECCAI-ACAI/EASSS 2001, AEMAS 2001, HoloMAS 2001, LNAI 2322, Springer-Verlag, Berlin, Heidelberg, 2002, pp. 233-267

[UU04] Ulieru, M., Unland, R: Enabling Technologies for the Creation and Restructuring Process of Emergent enterprise alliances; International Journal of Information Technology and Decision Making, World Scientific Publishing Co., Inc.; Vol. 3, No. 1, 2004, pp. 33-60

[Un03] Unland, R.: A holonic multi-agent system for robust, flexible and reliable medical diagnosis. In: Proceedings: On The Move Federated Conferences (OTM '03): Tenth International Conference on Cooperative Information Systems (CoopIS), International Symposium on Distributed Objects and Applications (DOA), International Conference on Ontologies, Databases and Applications of SEmantics (ODBASE); Workshop on Reliable and Secure Middleware; Catania, 2003

[KS04] Klüver, J., Stoica, C.: The Communicative Generation of Cultural Systems. In: Proceedings of the Conference on Cybernetics and Systems Analysis, Vienna, 2004

[KK07] Klüver, J., Klüver, C.: On Communication. An Interdisciplinary and Mathematical Approach. Springer, Dordrecht, 2007

[RK89] Ritter, H., Kohonen, T.: Self-organizing semantic maps. In: Biological Cybernetics Vol. 61, 1989, pp. 241 - 254

[TF02] Túlio, A., Filho, A.: Segmentation of Digitized Mammograms Using Self-Organizing Maps in a Breast Cancer Computer Aided Diagnosis System; Proceedings of the IEEE VII Brazilian Symposium on Neural Networks (SBRN'02), 2002

[WP85] Waltz, D. L., Pollack, J. B.: Massively parallel parsing. A strongly interactive model of natural language interpretation. In: Cognitive Science. Vol. 9, 1985, pp. 51–74

A Case Study on Managing SLAs in Composite Services with COSMA

André Ludwig, Thomas Hering, Rolf Kluge, Bogdan Franczyk

Information Systems Institute
University of Leipzig
Marschnerstr. 31
04109 Leipzig, Germany
{ludwig,hering,kluge,franczyk}@wifa.uni-leipzig.de

Abstract: COSMA proposes a novel SLA management approach for composite services. It supports a composite service provider in managing SLAs with providers of atomic services, in managing SLAs with requesters of composite services, and aligning both SLA management activities with each other. On this basis, a composite service provider can control and optimize its composite SLA management activities during the entire SLA lifecycle. This includes, in particular, planning and negotiating SLAs, monitoring and evaluating SLAs. In this paper, a case study on managing SLAs in composite services with the COSMA approach is presented in detail.

1 Introduction

Service-oriented computing (SOC) has emerged as the most promising design paradigm for next-generation distributed information systems. The vision that goes along with SOC is that once standards have established themselves and become widely adopted by service providers and requesters, a globally available infrastructure for hosting and accessing services will be created [Pap07]. This infrastructure will allow service providers to offer multiple services with individually adapted service capabilities to their changing customers that can dynamically and on-demand bind these services into their own applications; thus forming a market of services and an Internet of Services, respectively [Rug07]. The advent of service markets on the basis of the SOC paradigm will pave the way for a service-oriented business model which is referred to as composite service provider (CSP) [ACKM04]. A composite service provider requests (atomic) services from atomic service providers (ASP) and provides these services according to a process flow as (composite) service to service requesters (SR). In this constellation, a composite service provider acts as an independent, self-interested business entity, motivated to fulfill own goals, i.e. be profitable and maximize customer satisfaction.

In order to control the interface between service requesters and providers, a contractual basis in form of service level agreements (SLA) is needed. Current SLA management approaches applicable for SOC environments, i.e. WSLA [KL03], WS-Agreement

[ACD+], or WSOL [TPP02], provide extensive SLA language formalizations and management frameworks. However, they focus on bi-lateral service requester/provider constellations neglecting the SLA management requirements of composite service providers, i.e. managing SLAs with atomic service providers and with composite service requesters and aligning both with each other. A SLA management solution for composite services has to consider the contribution of sourced services - formalized in their (atomic) SLAs (ASLA) - in the management of the provided service - formalized in its respective (composite) SLA (CSLA). Since composite services are created on-the-fly also their SLA management must be realized on-the-fly. Manual SLA management is not appropriate for CSPs and automation support is required, i.e. for creation, monitoring and evaluation of SLAs.

The COmposite Sla MAnagement (COSMA) approach provides a novel solution for CSPs in managing ASLAs, CSLAs and their alignment. On this basis, a CSP can control and optimize its composite SLA management activities during the entire SLA lifecycle. COSMA and its constitutional elements COSMAdoc, COSMAframe, and COSMAlife are presented in detail in [LF08a]. Complementary to this presentation, this paper illustrates the application of the COSMA approach on a case study (based on a use case presented in [MGL+07]). With this motivation, it aims at contributing to the overall understanding and enhancement of the approach.

The paper is organized as follows: section 2 gives a brief overview of COSMA and its constitutional elements. Section 3 introduces a general use case and demonstrates the application of COSMA throughout the main phases of the SLA lifecycle. Section 4 refers to a prototypical implementation of the COSMA approach and section 5 concludes the paper and gives an outlook to next steps.

2 Composite SLA Management Approach (COSMA)

The central idea behind the COSMA approach is the integration of all SLAs a composite service provider has to deal with into one composite SLA management document. This composite SLA management document, which is defined as COSMAdoc, contains all contractual information of all involved SLAs and in addition the relationships and dependencies that exist between the different aspects of atomic and composite SLAs. On the basis of a COSMAdoc, the SLA management system of a CSP is able to map atomic SLA contents to contents of the composite SLA. This mapping enables a CSP to control and optimize its SLA management activities in providing a composite service. This includes, in particular, planning and negotiating SLAs, monitoring and evaluating SLAs (cf. SLA lifecycle as outlined in [LBKF05]). The COSMA approach consists of the following three parts:

- COSMAdoc: A generic information model that integrates contractual data, SLA management data, and elements for the expression of dependencies and relationships between SLA elements.

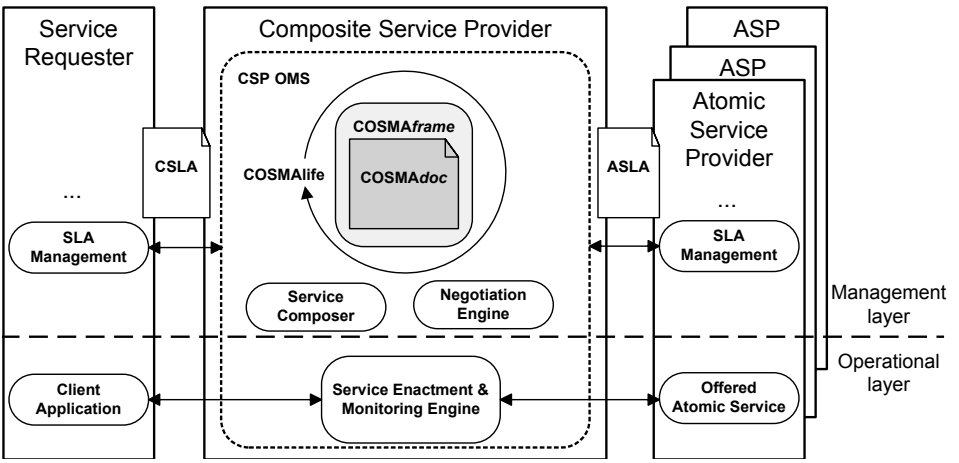


Figure 1: High-level perspective onto a CSP using the COSMA approach

- COSMAframe: A conceptual framework that outlines components that are necessary for the management of the composite SLA lifecycle on the basis of a COSMAdoc instance.
- COSMAlife: An integrated set of SLA management practices that use COSMAdoc instances to cover the phases of the SLA lifecycle.

COSMAframe, COSMAdoc, and COSMAlife are embedded into an operation and management system (OMS) of a CSP. In particular, the OMS provides components for automated run-time service composition (service composer), agent-based negotiation of SLAs (negotiation engine), enactment and monitoring of composite services (service enactment and monitoring engine), service registration, and interaction with service requesters and atomic service providers. In the Adaptive Services Grid project [ASG], all of these components including prototypical implementations were developed. Fig.1 presents the high-level perspective onto the business model composite service provider utilizing the COSMA approach.

3 Use Case

The use case presented in this paper is based on a business-to-business wholesale model of an Internet service provider (ISP). The ISP offers the whole spectrum of Internet services for Webspace provisioning. The ISP acts as a composite service provider. It integrates externally provided atomic services at run-time into end-to-end composite services.

On the supplier side, the ISP uses several atomic services, like domain name checking and registration, Web hosting configuration, operation and maintaining of Domain Name

Server information, handling of payments bundled to end customer products, messaging, and so forth. All of these services are provided by external atomic service providers, i.e. Web hosting ASP or payment ASP. Each service can be provided by different ASPs in different implementations with different functional and non-functional characteristics. It is also possible that an ASP provides the same service implementation with different characteristics depending on the individual requirements to its customers. For example, the service used for domain registration may be provided by several service providers in diverse implementations for different top-level domains (functional characteristic) at different service qualities (non-functional characteristics).

On the sales side, customers of the ISP are business units which request Internet service packages. These requested Internet services are further sold to end customers. Thus, customers of the ISP are resellers of Internet services again. These resellers bundle their services with Internet services sourced from the ISP. Typical examples of reselling customers of the ISP are portal providers like newspapers, TV-stations, or information portals. Portal providers bundle their core products, such as newspapers, with Internet services in order to gain higher profits and customer binding.

The ISP as an integrator of externally provided atomic services into end-to-end composite services has to determine at runtime which atomic service implementations are used in the composite service and has to manage the service provision in an automated fashion. It uses components such as a service composer, a negotiation engine, and a service enactment and monitoring engine. For the management of SLAs involved in the composite service it uses a component utilizing the COSMA approach which will be described in detail below.

The concrete composite service that is presented in the use case is a dynamic supply chain for automated Domain Name registration and provisioning of Webspace (later on referred to as DSC service). Atomic services included in the DSC service are:

- CheckDomain: checks whether a certain domain is available for registration
- CreateWebhostingAccount: creates a Web hosting account to access the Webspace
- CreateDomain: creates a domain for the Webspace
- SetupPhysicalHosting: sets up the physical Web hosting/Webspace which hosts the displayed files of a Website owner
- Nameserver: updates the name server with the new domain connected to the created Webspace
- RegisterDomain: registers a certain domain
- CreditCardPayment: executes a complete credit card authorization and payment process

The DSC service and all atomic services are characterized by service parameters which are defined in equal measures. They include quality parameters such as response time, availability, throughput, and encryption level, and financial terms such as reward. Fig.2

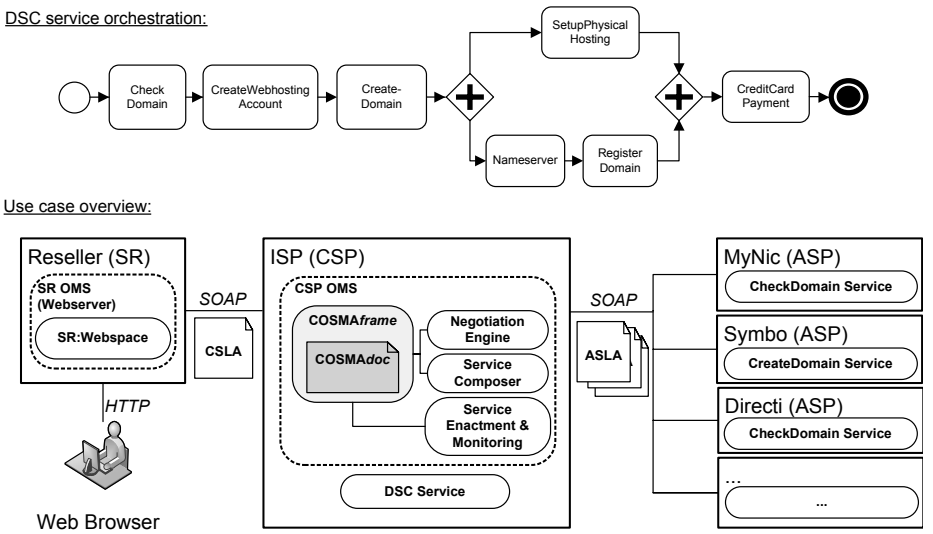


Figure 2: High-level use case perspective

shows the service orchestration composed by the service composer and illustrates the high-level perspective onto the use case.

The use case is based on a number of assumptions and restrictions. Firstly, all parties need to set up a negotiation environment which allows the creation and configuration of negotiation agents that negotiate SLA documents dynamically and that are allowed to sign resulting SLAs. Questions of interoperable negotiation environments, the ability to communicate and understand each other and so on are not tackled. Secondly, it is assumed that all negotiated SLA parameters are defined equally or can at least mapped to a common definition. Thirdly, the whole area of semantic service descriptions, creating them, processing them and using them for registering atomic services or composing service orchestration scripts is blinded out.

3.1 Creation and Integration of a COSMAdoc Instance

The COSMAframe component COSMAdoc Creator is responsible for the creation of a composition-specific instance of the COSMAdoc template. On the basis of a given generic service orchestration script provided by the service composer, the COSMAdoc Creator uses a composition decomposer to atomize the script into its atomic service types. For each atomic service, e.g. CheckDomain, CreateWebhostingAccount, etc., and for the DSC service an empty SLA element is created and embedded into the SlaSetAssembly of the COSMAdoc instance. Initial parameters are set for all SLA documents, e.g. service names, etc.

Afterwards, the COSMAdoc Integrator integrates the COSMAdoc instance. First, pre-

defined settings such as SLA parameters to be used, identifiers for the service requester and the composite service provider are taken from a database. Second, the SlaSetUsageValidation section of the COSMAdoc instance needs to be integrated. The directives for which elements are restricted in which way are taken from a configuration file. Available predicates defined in COSMA include but are not limited to makeMandatory, makeOptional, excludeElement, setNegotiable, or setMask. Exemplary, the following usage validation restrictions are created for the CheckDomain service:

- The predicate makeMandatory enforces that i.e. an end point reference of an atomic service implementation is specified in the SLA document.

```
<constraint action="makeMandatory(//Sla[@SlaId='2']
/.../ServiceReference/anyRef)"'>
```

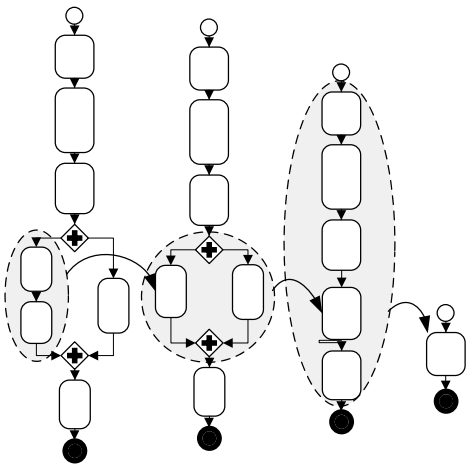
- The predicate excludeElement excludes i.e. the possibility to specify penalties in the SLA with the service requester. The predicate is bound to an equal condition (=). Thus, the predicate is only used if the atomic service provider of the SLA 1 is "MyNic". This restriction may be caused by the fact that MyNic does not accept the specification of penalties.

```
<conditional>
  <context condition="//Sla[@SlaId='2']/.../
    ServiceProvider='MyNic'">
    <constraint action="excludeElement(//
     Sla[@SlaId='1']/.../FinancialTerms/Penalty)"/>
  </context>
</conditional>
```

- The predicate setNegotiable is used i.e. to explicitly state that the reward for invocation of the CheckDomain service is negotiable.

```
<constraint action="setNegotiable(//Sla[@SlaId='2']
/.../ClearingUnitPrice/ValueExpr)">
```

Finally, the SlaSetDataValidation section needs to be integrated. Values for static data validation restrictions are taken from a database similar to usage restrictions above, i.e. availability must always be below hundred percent or a certain service requires a minimal response time retrieved from experiences. Possible predicates include i.e. setValue, setMaxValue, setMinValue, or setValueRange. Before dynamic data validation restrictions can be integrated, the COSMA Integrator analyzes the service orchestration script and used SLA parameters of the SlaSetAssembly. Dynamic data validation restrictions usually result from the structure of the service orchestration script and depend on the actual service parameters. Thus, the COSMA Integrator decomposes the service orchestration script into its atomic composition patterns, i.e. sequence, parallel split, and loop. Afterwards,



Aggregation formulas (extract)

Pattern	CSLA parameter	Aggregation formula
Sequence	Response time	$RT_{CS} = \Sigma(RT_{S\text{-}i})$
	Availability	$A_{CS} = \Pi(A_{S\text{-}i})$
	Throughput	$T_{CS} = \min\{T_{S\text{-}i}\}$
	Encryption Level	$E_{CS} = \min\{E_{S\text{-}i}\}$
Parallel split	Response time	$RT_{CS} = \max(RT_{S\text{-}i})$
	Availability	$A_{CS} = \min(A_{S\text{-}i})$
	Throughput	$T_{CS} = \min\{T_{S\text{-}i}\}$
	Encryption Level	$E_{CS} = \min\{E_{S\text{-}i}\}$
Loop	Response time	$RT_{CS} = k\ RT_S$
	Availability	$A_{CS} = A_S$
	Throughput	$T_{CS} = T_S$
	Encryption Level	$E_{CS} = E_S$

Notation:
CS = composite service
S-i = atomic services I
k = assumed number of repetitions in a loop

Composition-specific aggregation formula for RT_{CS} in the DSC service:

$$RT_{CS} = \Sigma(RT_{checkDomain}, RT_{createWebhostingAccount}, RT_{createDomain}, \max(RT_{setupPhysicalHosting}, \Sigma(RT_{nameServer}, RT_{registerDomain})), RT_{creditCardPayment})$$

Figure 3: Decomposition of the DSC service and creation of composition-specific aggregation formulas

composition-specific aggregation formulas are created from generic aggregation formulas in a reverse order, i.e. as proposed in [JRGM04]. Fig.3 illustrates the decomposition of the DSC service orchestration script, summarizes generic aggregation formulas for different SLA parameters and shows the resulting composition-specific aggregation formula for response time of the DSC service.

The composition-specific aggregation formulas are then embedded into the Aggregation-Formulas section of the COSMAdoc instance, given a unique identifier and referenced by data validation predicates of the SlaSetDataValidation section. For example, the created aggregation formula for composite service response time would be embedded as follows:

```
<AggregationFormulas>
  <GuaranteeTerms>
    <Formula Id="789">
    '//Sla[@SlaId='2']/Terms/GuaranteeTerms/.../
    ServiceLevelObjective/Value'+
    '//Sla[@SlaId='3']/Terms/GuaranteeTerms/.../
    ServiceLevelObjective/Value'+
    '//Sla[@SlaId='4']/Terms/GuaranteeTerms/.../
    ServiceLevelObjective/Value'+ max(
    '//Sla[@SlaId='5']/Terms/GuaranteeTerms/.../
    ServiceLevelObjective/Value',
    '//Sla[@SlaId='6']/Terms/GuaranteeTerms/.../
    ServiceLevelObjective/Value'+
```

```
      '//Sla[@SlaId='7']/Terms/GuaranteeTerms/.../
      ServiceLevelObjective/Value') +
      '//Sla[@SlaId='8']/Terms/GuaranteeTerms/.../
      ServiceLevelObjective/Value'
      </Formula>...
    </GuaranteeTerms>
    <FinancialTerms>...
    <MonitoringTerms>...
</AggregationFormulas>
```

The according data validation restriction that connects the aggregation formula with a SLA parameter specified in GuaranteeTerms of the CSLA would be stored in the SlaSetDataValidation section as follows:

```
<constraint action="setMinValue(//Sla[@SlaId='1']/.../
  ServiceParameter[@Name='ResponseTime']/
  ServiceLevelObjective/Value,//AggregationFormulas/.../
  Formula[@Id='789'])"/>
```

Analogously, aggregation formulas and data validation restrictions are created for all existing SLA parameters. They will be used in the following to express relationships and dependencies between ASLA and CSLA contents.

3.2 Negotiation of a COSMAdoc Instance

The negotiation of SLAs involved in the COSMAdoc instance's SlaSetAssembly is executed in three steps: preparation of the negotiation process, iterative SlaSetAssembly negotiation, and final ASLA/CSLA conclusion. The high-level perspective on the negotiation of the DSC service is illustrated in Fig.4.

For the preparation of the negotiation process, a negotiation engine needs to be configured. For each atomic service involved in the service orchestration script and the composite service itself, a negotiation agent is created that is responsible for negotiating the according SLA (agents 1-8 in Fig.4). In the use case, seven atomic SLAs and the resulting DSC CSLA need to be negotiated; hence, eight negotiation agents are created in the negotiation engine. The configuration of each negotiation agent in terms of negotiation protocols and negotiating parties is made according to the NegotiationTerms individually specified in respective SLAs. The negotiation strategy of each negotiation agent depends on policies and decision rules that were given to a negotiation engine for its agents by the ISP.

After the preparation, an iterative negotiation process is executed by the negotiation engine in combination with the COSMAdoc instance. For this, first, an initial validation of all ASLAs is executed by the COSMA Manager on the validation interface. Since no negotiations were made so far, all usage and data validation restrictions are violated and according errors are set into all SLA documents. Each negotiation agent receives only the

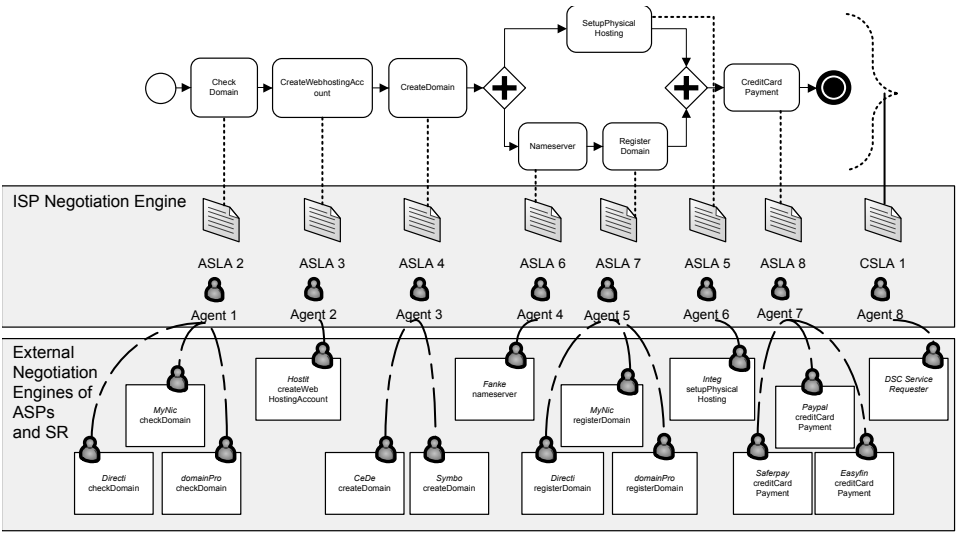


Figure 4: Setting of agent-based SLA negotiations for services involved in the DSC service

SLA document it negotiates. Thus, the agent does not control how its individual negotiation results influence the overall CSLA negotiation. This mapping is controlled by the COSMA Manager using data validation restrictions and aggregation formulas. Negotiation agents must only ensure that all usage and data validation errors in their individual SLA document are eliminated. Example usage and data validation restrictions on an ASLA are as follows:

```
<Sla SlaId="2" ServiceName="CheckDomain"> ...
  <AnyRef usage-error="makeMandatory"/> ...
</Sla>

<Sla SlaId="7" ServiceName="RegisterDomain"> ...
  <ValueExpr dataValidation-error=
   "recommendValues(10.00)"> ...
  <Value dataValidation-error="setMinValue(128)"/> ...
</Sla>
```

The produced error messages are used by negotiation agents for negotiation of ASLAs. Each negotiation agent (agents 1-7) is now able to negotiate with its opponent the contents of the ASLA considering the initially created data and usage validation errors. Based on the negotiation agent's decision making strategy, the agent selects an atomic service implementation for each service and fills the respective ASLA with the required contents.

On this basis, an initial CSLA validation can be executed on the validation interface of the COSMA Manager and usage and data validation errors are set into the DSC service's SLA. Data errors are created using the aggregation formulas specified for the DSC service. One

data validation error corresponding to the data validation restriction presented above is:

```
<Sla SlaId="1" ServiceName="DSCService"> ...
  <ServiceParameterSet>
    <ServiceParameter Name="ResponseTime">
      <ServiceLevelObjective>
        <Value dataValidation-error=
         "setMinValue(80000)"/> ...
      </ServiceLevelObjective>
    </ServiceParameter> ...
  </ServiceParameterSet> ...
</Sla>
```

The value of 80,000 milliseconds is a dynamically calculated value (Formula Id="789") based on the negotiation results of the ASLAs. The negotiation agent (agent 8) can then execute the negotiation of the CSLA using the validation errors as support. Once the CSLA is filled, the validation interface is invoked to validate the CSLA contents and add remaining validation errors. For example, if the service level objective of response time was 60,000 milliseconds, then the dataValidation-error="setMinValue(80000)" would remain in the CSLA and would need to be fixed by the agent.

Depending on the used negotiation protocol, the process of ASLA validation-negotiation and CSLA validation-negotiation is continued in order to optimize conditions of service consumption and provision. For every CSLA validation, the data validation restrictions change dynamically depending on the ASLA negotiation outcomes. Thus, a stepwise optimization can be realized with adaptive restrictions and harmonized SLAs.

If no validation errors remain in ASLAs and the CSLA and if all negotiating parties can accept proposed SLAs, their representing negotiation agents are allowed to conclude their SLA. Signing SLAs is only possible if the validation process does not produce any validation errors. This ensures that all SLAs are harmonized with each other and comply with the goals of the CSP.

A description of SLA negotiations based on the COSMA approach is published in [LF08b].

3.3 Monitoring and Evaluation of a COSMAdoc Instance

Monitoring and evaluation of the DSC service COSMAdoc instance include (1) determining the actual service levels of atomic services, (2) evaluation of measured service levels with regard to the atomic and composite service, (3) determination of corrective actions that support the composite service provision according to the CSLA, and (4) explanation of service level violations and reasoning for future composite service provisions.

In order to determine actual service levels of atomic services, the MonitoringTerms defined for every service and every service parameter in a COSMAdoc instance need to be processed by the COSMA Manager. The MonitoringTerms defined for the service Directi:CheckDomain and the service parameter response time are as follows:

```
<MonitoringTerms>
  <ServiceParameterSet>
    <ServiceParameter Name="ResponseTime">
      <Obligated>Directi</Obligated>
      <AccessPattern>
        <AccessMechanism>Pull</AccessMechanism>
        <DataSource>rdbms.directi.com:4040</DataSource>
        <AssessmentInterval>
          <TimeIntervall>
            <StartTime>2007-10-15T14:00:00.000+01:00
            </StartTime>
            <Duration>2008-01-05T00:00:00.000+01:00
            </Duration>
          </TimeIntervall>
          <Count>02:30:00.000</Count>
        </AssessmentInterval>
      </AccessPattern>
      <ServiceLevelMeasurement>2000
      </ServiceLevelMeasurement>
    </ServiceParameter> ...
  </ServiceParameterSet>
</MonitoringTerms>
```

The example shows that the ASP Directi is obligated to monitor the response time and provide the aggregated monitoring data in the defined database interface which can be pulled by the ISP between the specified dates every 2.5 hours. The last service level measurement for response time which is stored in the ServiceLevelMeasurement element is 2,000 milliseconds.

The monitoring process is executed for all service parameters and all atomic services of the DSC service COSMAdoc instance. The results of these parallel monitoring processes are stored in the service level measurement elements. The evaluation of the measured service levels is executed by the COSMA Validator and Violation Detector and can be invoked on its validateCOSMAdoc interface. First, measured service levels are compared with service level objectives. For example, for the Directi:checkDomain service, the measured response time is 2,000 milliseconds whereas the service level objective and qualifying condition could be “LESSTHAN” and “3000” milliseconds. Hence, the service level objective is fulfilled and not violated. This result is stored in the StateTerms of the according SLA.

After the atomic service level evaluation, service levels of the DSC service must be evaluated. Therefore, first, the anticipated service levels of the composite service must be calculated and compared with the service level objectives of the CSLA. The calculation is based on an aggregation formula that was created during COSMAdoc integration and that is stored in the AggregationFormulas section. The formula for calculating the service level measurement of response time is almost the same as the one presented in Fig.3. The only difference is that the terms of the formula (paths to ASLA elements) are the service level measurements of the ASLAs for response time rather than the service level objectives.

The predicate defined in the data validation section for the service level measurement of response time is setValue (not setMinValue) since the response time of the DSC service results directly from response times of atomic services.

In general, there are three possible situations. First, all ASLAs are fulfilled and no violation of the CSLA occurs. Second, one or more ASLAs are violated but do not cause a violation of the CSLA. Third, one or more ASLAs are violated and cause a violation of the CSLA. It should be noted that managing the evaluation process for more than one service level and for constantly changing service level measurements is a complex task. The elements of COSMA enable a full automation of this evaluation process. The aspect of dealing with CSLA violations in terms of claiming penalties, re-negotiating ASLAs, changing service implementations, or even re-planning the service orchestration are not covered in this paper. For the explanation of service level violations and reasoning for future composite service provisions, the service orchestration script and sources of violation need to be analyzed. Dynamic service profiles are a useful mechanism to store service fulfillment or violation experiences [AKKZ06].

4 Prototypical Implementation

To prove the applicability of the COSMA approach in managing SLAs of composite services, a prototypical implementation was developed. This prototypical implementation serves as a demonstrator for the realization of the key elements of COSMA. The demonstrator is far from being restrictive. It can be adapted and extended depending on a certain usage area or scenario. The use case scenario presented above is implemented in a demonstrator. Although the COSMA approach represents an automated approach towards composite SLA management and does not require human intervention in executing the management steps, the demonstrator provides a graphical user interface for human interaction (see screenshot in Fig.5). This graphical user interface allows a human user to manually trigger individual COSMA management tasks and view the results of every single operation (creation, negotiation, validation, etc.). Additionally, the results of single operations can be changed to allow testing and probing alternatives, i.e. in restricting values and altering aggregation formulas.

5 Conclusions and Outlook

The paper presented a use case for managing SLAs in composite services with the COSMA approach. It aimed at contributing to the overall understanding and enhancement of COSMA. It was shown how relationships and dependencies between SLAs can be maintained at a central point of information. On this basis, a CSP can control and optimize its SLA management activities, pro-actively plan financial consequences, and dynamically calculate the expected service level objectives from dynamically varying service orchestration scripts. By using the SlaSetDataValidation restrictions, a CSP can dynamically calculate

Figure 5: Screenshot of the COSMA demonstrator during COSMAdoc monitoring and evaluation

the expected service level objectives of a composite service and limit the values contracted in all SLAs to meaningful levels. A CSP can restrict the penalty payments that are due in case of service level violations. Aggregation formulas can be used to split up imposed penalty functions (of the composite service) to penalty functions that a CSP imposes to the atomic service provider. These penalty functions can then be used to define minimal penalties claimed from the atomic service provider. On this basis, a CSP can evaluate the impact of service level variations on composite service levels and trigger corrective actions if necessary and possible.

COSMA and the presented use case are to be interpreted as a starting point that must be extended, adapted to individual requirements of further usage scenarios, and tested on them. This may lead to a widespread evaluation and assessment of the approach and will result in a broader range of practical experiences.

Currently, COSMAlife describes mechanisms for the most important SLA lifecycle phases: creation, integration and negotiation of SLAs and monitoring, evaluation and termination of SLAs. For an overall SLA lifecycle management, support of all phases of the lifecycle would be necessary. In the preliminary phase of finding a potential agreement partner, this includes mechanisms to advertise provided services and their SLAs and find suitable service providers based on certain criteria. After the termination of SLAs involved in a managed composite service, the experienced SLA behaviour should be stored in dynamic service profiles which can be used during SLA negotiations and creation of usage and data restrictions. These and other steps which would iteratively improve the behavior of a COSMAframe implementation and connect multiple COSMAlife cycles with each other need to be added and integrated in the proposed approach.

Another important aspect regards the more detailed investigation on penalties and bonuses in the management of SLAs. Instead of aggregating quality of service parameters and the financial parameter reward, penalties would require a decomposition of a received (multi-dimensional) penalty from a service requester and the allocation of sub-penalties to atomic SLAs. With different penalties on service parameters and complex service composition patterns, this task is an extremely complicated task which is however vital for a broad adoption of the COSMA approach by composite service providers.

References

[ACD+] A. Andrieux, K. Czajkowski, A. Dan, K. Keahey, H. Ludwig, T. Nakata, J. Pruyne, J. Rofrano, S. Tuecke, and M. Xu. Web Service Agreement Specification (WS-Agreement). `http://www.gridforum.org/documents/GFD.107.pdf`.

[ACKM04] G. Alonso, F. Casati, H. Kuno, and V. Machiraju. *Web services: concepts, architectures and applications*. Springer, Berlin, New York, 2004.

[AKKZ06] W. Abramowicz, M. Kaczmarek, M. Kowalkiewicz, and D. Zyskowski. *Architecture for Service Profiling. In: Modeling, Design, and Analysis for Service-oriented Architecture Workshop (MDA4SOA 2006), pp. 121–130*. Chicago, 2006.

[ASG] ASG. Integrated Project Adaptive Services Grid (ASG). `http://www.asg-platform.org`.

[JRGM04] M.C. Jaeger, G. Rojec-Goldmann, and G. Muehl. *QoS aggregation for Web service composition using workflow patterns. In: 8th International IEEE Enterprise Distributed Object Computing Conference (EDOC 2004), pp. 149-159*. Monterey, 2004.

[KL03] A. Keller and H. Ludwig. *The WSLA Framework: Specifying and Monitoring Service Level Agreements for Web Services, pp. 57-81*. J. of Network and Systems Management 11, 2003.

[LBKF05] A. Ludwig, P. Braun, R. Kowalczyk, and B. Franczyk. *A Framework for Automated Negotiation of Service Level Agreements in Service Grids. In: Bussler, C., Haller, A. (eds.) Business Process Management Workshops. LNCS, vol. 3812, pp. 89–101*. Springer, Berlin, Heidelberg, 2005.

[LF08a] A. Ludwig and B. Franczyk. *COSMA - An Approach for Managing SLAs in Composite Services. In: Bouguettaya, A., Krueger, I., Margaria, T. (eds.) 6th International Conference on Service Oriented Computing (ICSOC 2008). LNCS, vol. 5364, pp. 626-632*. Springer, Berlin, Heidelberg, 2008.

[LF08b] A. Ludwig and B. Franczyk. *Managing Dynamics of Composite Service Level Agreements with COSMA. In: 5th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2008)*. Jinan, 2008.

[MGL+07] M. Momotko, M. Gajewski, A. Ludwig, R. Kowalczyk, M. Kowalkiewicz, and J.Y. Zhang. *Towards adaptive management of QoS-aware service compositions. J. of Multiagent and Grid Systems 3, pp. 299-312*. J. of Multiagent and Grid Systems 3, 2007.

[Pap07] M.P. Papazoglou. *Web Services: Principles and Technology*. Prentice Hall, Essex, 2007.

[Rug07] R. Ruggaber. *Internet of Services SAP Research Vision. In: 16th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 2007), pp. 3*. Rome, 2007.

[TPP02] V. Tosic, K. Patel, and B. Pagurek. *WSOL - Web Service Offerings Language. In: Web Services, E-Business, and the Semantic Web. LNCS. pp. 57-67*, volume 2512. Springer, Berlin, Heidelberg, 2002.

Towards enabling SaaS for Business Rules

Emilian Pascalau[1], Adrian Giurca[2]

[1]Hasso Plattner Institute
Prof.-Dr.-Helmert-Str. 2-3
14482 Potsdam, Germany
emilian.pascalau@hpi.uni-potsdam.de

[2]Brandenburg University of Technology
Walther-Pauer-Str. 3
03046 Cottbus, Germany
giurca@tu-cottbus.de

Abstract: The actual trends concerning enterprise development take heavily into account the Software as a Service paradigm. Business rules are widely used to define business behavior. Therefore, to be able to access business rules in this context a proper way of discovery and invocation is required. This paper describes the *architecture*, and *business processes* of a Semantic Web Service based registry for business rules.

1 Introduction and Motivation

The problematic of discovering and differentiating data is certainly not new in the software community. In order for data to be meaningful, the context of that data must be understood. As such, service discovery infrastructures are essential. Providing the ability to structure and model data and metadata to solve this problem is at the heart of any registry design, so as to prevent data from being undiscoverable or misunderstood. Nowadays, business rules are widely used in software applications. In the last 10 years business rules were employed to declaratively describe policies, business processes and practices inside enterprise. Rules are becoming increasingly important in business modeling and requirements engineering, as well as in Semantic Web Applications. There are several rule platforms and rule languages notably Jess, FLORA 2, Drools, Jena Rules, JSON Rules [GP08] but there is no standard way for defining business rules. The standardization is a concern for both OMG and W3C. The first produces OMG PRR Proposal[OMG07] and the second RIF-BLD [BK08]. The actual trend concerning aggregate enterprise that follows the Software as a Service [BLB+00] paradigm envisions a continuous growth (Market Trends: Software as a Service, Worldwide, 2007-2012, Gartner). An obvious necessity and requirement in order to be able to use rules in SaaS based applications is to able to find and access rules in a service oriented manner. Besides the already well known registries and repositories such as UDDI [OAS04] and ebXML [OAS01b, OAS01a] that enable enterprises with the ability to conduct businesses based on standards for business process collaboration, core data components, collaboration protocol agreements, messaging, there are also other types of registries such as metadata registries: NSDL Registry, Metadata Project. The founding principles of metadata registries were introduced in [HW02]. The authors argue in [HW02] that the approach of declaring schemas in metadata registries advance the W3C vision of

enabling a 'cooperative' Web where machines and humans can exchange electronic content that has clear-cut, unambiguous meaning, by providing a common approach for the discovery, understanding, and exchange of semantics. Therefore a registry to describe and discover business rulesets remains actual and necessary. First steps towards a business rules registry have been already done. While [GDPW08] introduced the entry information model for a business rules registry, this paper introduces the architecture of such a registry foreseeing also the SaaS demand. Besides the architecture also the Client/Registry interaction is depicted with the help of BPMN [OMG08] models. Moreover the registry will facilitate discovering of business rules by using Semantic Web techniques.

2 The Registry Architecture

There is a need of being capable to change software easily to meet evolving business requirements. More and more nowadays software development requires a shift towards a demand-centric orientation. This trail foresees the point were software will be delivered as a service within the framework of an open marketplace.

As stated in [BLB+00] "*the term software as a service is beginning to gain acceptance in the market-place; however the notion of service-based software extends beyond these emerging concepts*".

In such an approach a service conforms with the much accepted definition stating that a service is "*an act or performance offered by one party to another. Although the process may be tied to a physical product, the performance is essentially intangible and does not normally result in ownership of any of the factors of production*" [LVL96].

The authors in [BLB+00] argued that the "*service-based model of software is one in which services are configured to meet a specific set of requirements at a point in time*". Components may be bound instantly, just at the time they are needed - and then the binding may be discarded.

Based on these facts service based architecture is the normal consequence.

2.1 The Information Model

One proposal towards a ruleset entry model has been already introduced in [GDPW08]. However, we are going to use a simplified version of it. Therefore, the following information is required for a *web-based rule registry* entry:

Information related to the ruleset

1. An URI reference to the specific ruleset implementation, encoded by using *rulesetID* property;

2. An URI reference to the ruleset representation language, encoded by using Dublin Core `dc:type`;

3. A literal representing the code of the business addressed by this ruleset. It is a code (e.g. Naics, UNSPSC) of the corresponding business part (`dc:related`);

4. An URI reference to the format of the ruleset representation (`dc:format`);

Information related to the ruleset vocabulary

1. An URI reference to the specific vocabulary implementation, expressed by using *vocabularyID* property;

2. An URI reference to the vocabulary representation language (encoded with the Dublin Core `dc:type`);

3. An URI reference to the format of the vocabulary representation (`dc:format`);

All the above parts are required, and represent the minimal request of representing a rule set entry into the registry.

In addition to the above required information, users can provide optional information such as: (a) An URI reference to the natural description of the ruleset, (b) metadata of the ruleset creator, (c) the release date and others as they were discussed in [GDPW08].

As already been argued at the beginning of this section, the registry architecture is based on Web Services (See also [CMRW07, GHM+07] for more arguments on such solution).

2.2 The Architecture

The registry general architecture[1] is depicted in Figure 1 and at the first look follows the general principles of enterprise architecture. One notable distinction is that the Data Access Object Layer is based on SPARQL and not SQL. This is due to the fact that the Platform Specific Language of our registry is RDF.

An instantiation of the general architecture depicted in Figure 1 into an implementation specific one is illustrated in Figure 2.

The presentation layer from the general architecture (Figure 1) is represented in the instantiated architecture view (Figure 2) by two possible representation technologies JSP and Web Service. Same the business layer of the general architecture comprises two possible representations in the instantiation architecture: EJB and/or Web Service annotated EJB.

[1] Modeled with the Fundamental Modeling Concepts by using the ORYX browser based editor. The ORYX editor is a web-based editor (developed at the BPT chair, Hasso Plattner Institute) for modeling business processes in BPMN.

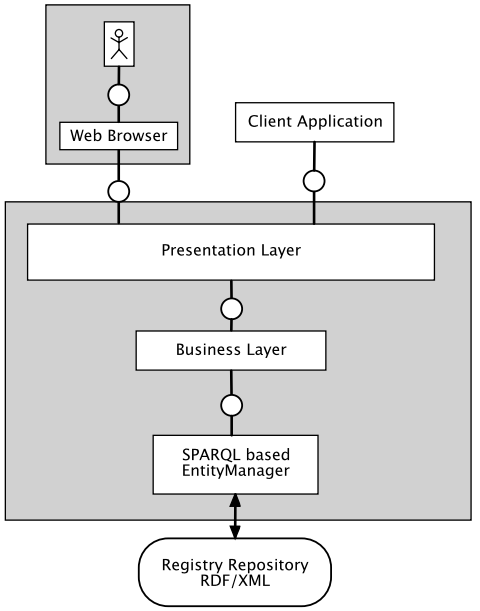


Figure 1: Registry's General Architecture

The instantiation architecture we propose is based upon JEE5 technologies and particularly EJB3 [Mic07]. As deployment environment we experiment with JBoss Application Server but any JEE5 compliant application server can be used as well. JBoss AS meets all of our platform requirements particularly it is JEE5 compliant, is open source, and is one of the most used application servers.

Recall that registry entries are encoded by using RDF based technologies. In addition, since the amount of data is likely to be low space consuming, our solution proposes to store data into RDF/XML files.

This architecture provides the user with a wide range of connectivity possibilities including: (a) either through a JSP web based interface easy to use for human users and (b) for machine clients in two flavors: (b1) SOAP Web Service based and (b2) a programming API. The JSP interface and the SOAP one are comprised inside the Web Server (Tomcat). The programming API is likely to be compatible with the JAXR[Mic02] Sun specifications. The specification provides a uniform and standard Java API for accessing different kinds of XML Registries. Also the API based approach could be useful for those applications that don't support Web Services.

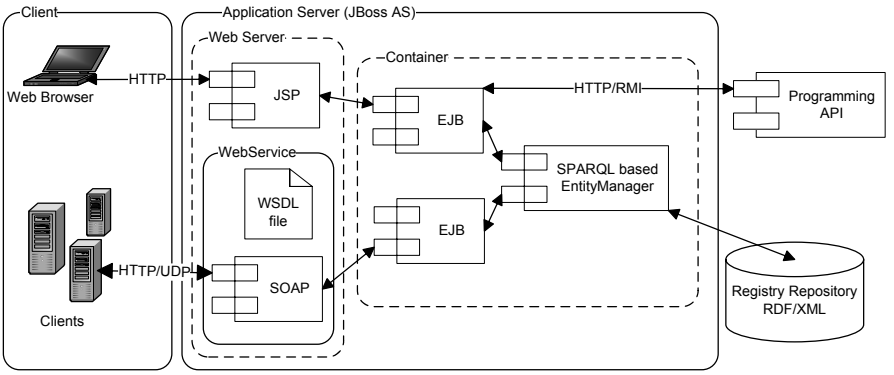


Figure 2: The Instantiation of the Registry's General Architecture

2.3 The Implementation

The web service is implemented as an annotated EJB3 and also the JSP interface communicates through EJBs by including SEAM framework. An important component is the *SPARQL based* `EntityManager` [Mic07]. The EJB3 technology comes with an already DAO layer in the form of the `EntityManager`. However, this EntityManager is SQL based and is suited for relational databases and not for RDF/XML repositories as in our case. Our approach is based on a Semantic Query Language SPARQL which is suited for our RDF/XML repository. SPARQL is a W3C standard and its purpose is to query, in principle the Web, but in general data models following semantic connections that exists between the models. The implementation of such an `EntityManager` will also enrich the JBoss Application Server with a new semantic query language.

3 Client-Registry Interaction

The Business Process Management Initiative (BPMI) has developed a standard Business Process Modeling Notation (BPMN) [OMG08]. The primary goal of BPMN is to provide a notation that is readily understandable by all business users such that business processes can be illustrated using a standard notation understandable also to business analysts that create the initial drafts of the processes, continuing towards technical developers responsible for implementing the technology that will perform those processes, and finally going on, towards the business people who will manage and monitor those processes.

As defined in [Wes07] "*a business process model consists of a set of activity models and execution constraints between them*". In today's business scenarios, service composition to provide added-value products to the market seems to be general trend. This section illustrates with the help of BPMN models the interaction between registry and its clients.

The registry interacts with the client through four processes: *publish*, *update*, *delete* and *query*. Architectural speaking there can be different types of clients (see Figure 2). However the registry itself interacts with these different types of clients based on the four operation types, using *the same workflow*. All processes models depicted in the following subsections comprise the `Client` pool and the `RuleRegistry` pool.

3.1 RuleSet Publish Process

The rule publish process is depicted in Figure 3. The process of publishing a `RuleSet` is started by the `Client` with a `ruleSetEntryPublishRequest` activity. The `Rule Registry` publish process is triggered by the message event of `RuleSet` publishing from a client. The process continues with the `Publish Sub-Process`. The Publish Sub-Process must end within a predefined time frame otherwise a `Timeout` exception will be raised. In case of a Timeout the client is notified by the activity `notifyTimeout`, and the `RuleRegistry` process terminates. The `Publish Sub-Process` comprises two activities: `verifySubmittedRuleSetEntry` and `saveSubmittedRuleSet Entry`. If the `Publish Sub-Process` has successfully ended then the client is notified of the activity success, and the process ends.

While the `Publish Sub-Process` is active the user can cancel its request. This is depicted with the help of the `cancelRuleSetEntryPublishRequest` activity, in the `Client` pool.

In the `RuleRegistry` pool this is modeled with some complex constructs. First the `RuleRegistry` receives the cancellation message. The cancelation request is handled by the `handelCancelPublishSub-Process` activity, followed by the `Cancel PublishSub-Process` event. An event with the same name is also attached to the `Publish Sub-Process`. In fact this is one and the same event. This pattern is called *Cancel Case* according to [WvdAD+06]. Besides the *Cancel Case* pattern, this model takes use also of *Send Pattern* and *Receive Pattern*.

3.2 RuleSet Update Process

The process of updating a registry `RuleSetEntry` (See Figure 4) starts with a `Client updateRuleSetEntryRequest`. In this case the message sent to the `RuleRegistry` contains the `ruleSetEntryId` and the `clientCredentials`. The registry receives the request and tries to find the `ruleSetEntry` requested by the client. If the entry is not found then the registry informs the client and the update process is finished for both parties. If the entry is found then the registry validates client's credentials. Only the client that published the `ruleSetEntry` is allowed to update the registry entry. When these requirements are met then the registry sends the complete `ruleSetEntry` to the client. In such a way the client is able to update the entry's content and sends it back to the registry. As for the `publish` operation, the `update` operation is time based, so it has to

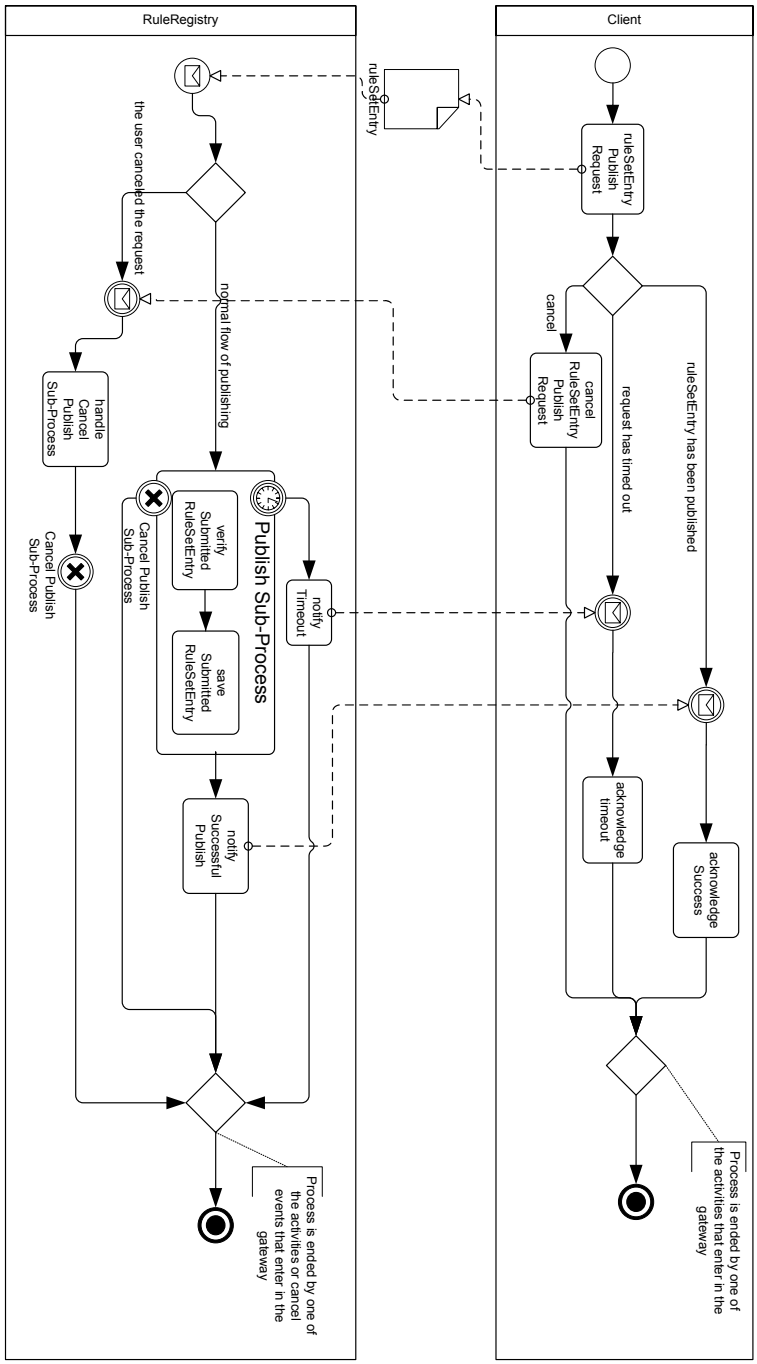


Figure 3: RuleSet Publish Process

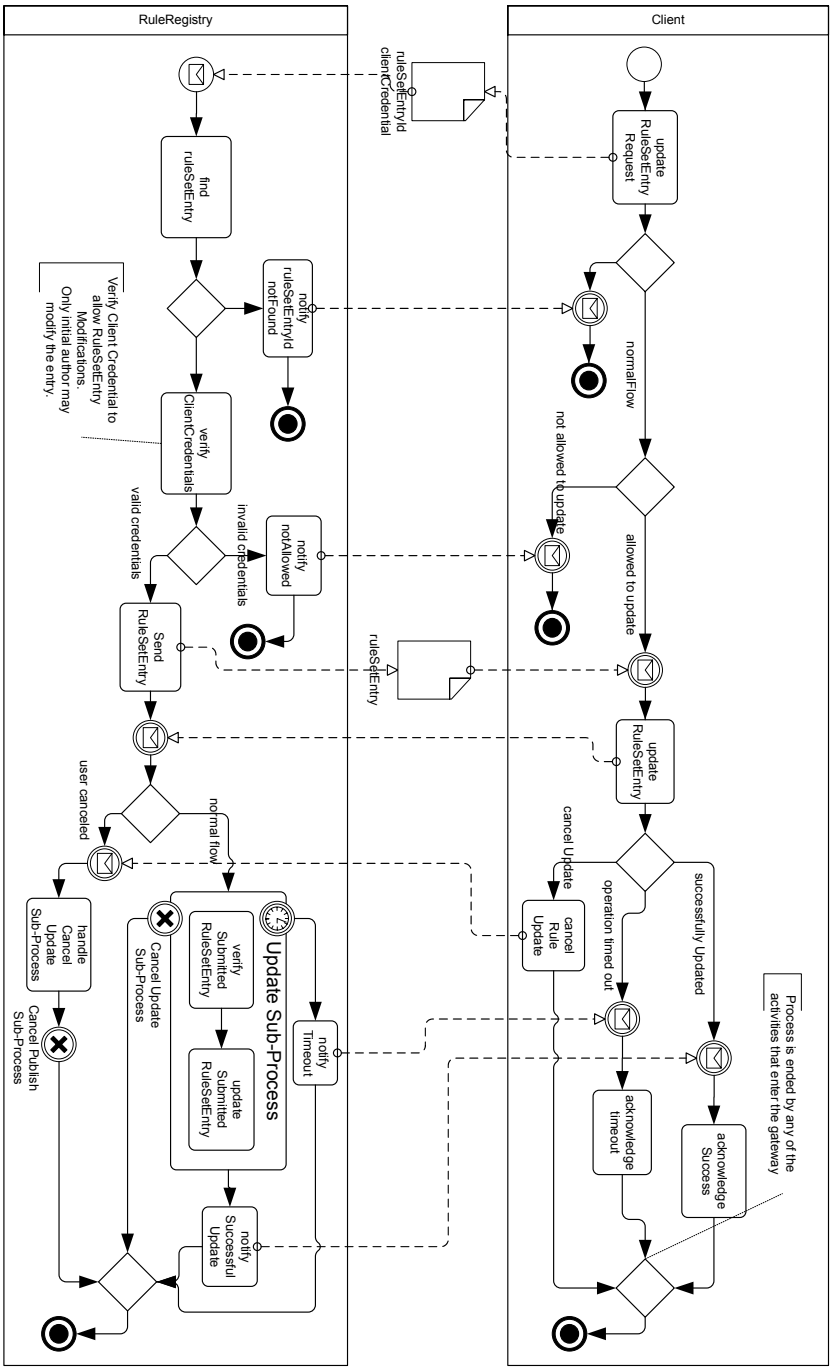


Figure 4: RuleSet Update Process

be fulfilled in a specific time. During this time the client is allowed to cancel the process.

3.3 RuleSet Delete Process

The `delete` process is illustrated in Figure 5 and is similar with the update process. What it brings new is that in this case the client is requested to confirm the operation. Again only the initial publisher is allowed to delete the `ruleSetEntry`. After delete confirmation is received the registry's process continues with atomic activities. No sub-processes are involved.

3.4 RuleSet Query Process

The `query` process (Figure 6) compared to the other three processes already discussed looks as being the simplest. The `Client` requests a query by sending a `QueryRequest` message to the `RuleRegistry`. This message is the starting event for the `RuleRegistry` query process. The registry verifies the query's content. If query's content is invalid then informs the `Client` about the invalidity of the request and ends its own process. Receiving a message stating that the initial request was invalid, makes the `Client` able to choose from two flows: either acknowledging the invalidity of the request and then terminating the process or acknowledging the invalidity of the request but issuing a new query towards the registry.

If the request is valid then the registry searches for the `RuleSetEntry` fulfilling the query content and, if a `RuleSetEntry` is found then the registry forwards it towards the `Client`. In case that no `RuleSetEntry` has been found the `Client` can either issue another query or terminate its process.

This process takes advantage of the *Arbitrary cycles* pattern described in detail in [Wes07, WvdAD+06]. A small excerpt of Figure 6, depicting only the loop is presented in Figure 7.

4 Conclusion and Future Work

This paper proposes architecture for a business rule registry. The architecture is service based and foresees the general actual trend of developing new software in the context of software as a service (SaaS). One core part is the BPMN modeling of registry business processes. Such approach is well suited for both business administration professionals and also for software engineers and software architects that must provide the concrete implementation. Future work envisions concrete implementation, and how the registry comes in handy in real life business use cases. Besides all, the need for providing registry services also in REST based style has to be investigated.

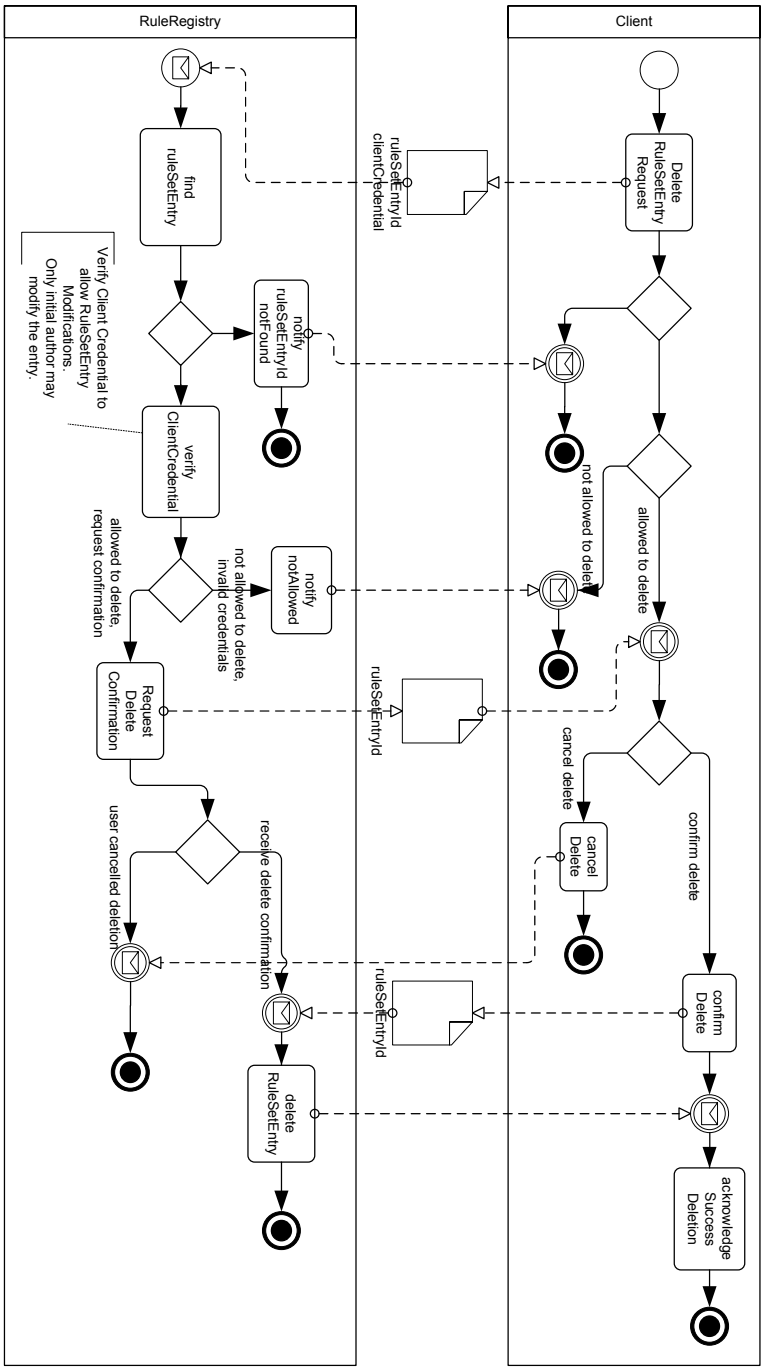


Figure 5: RuleSet Delete Process

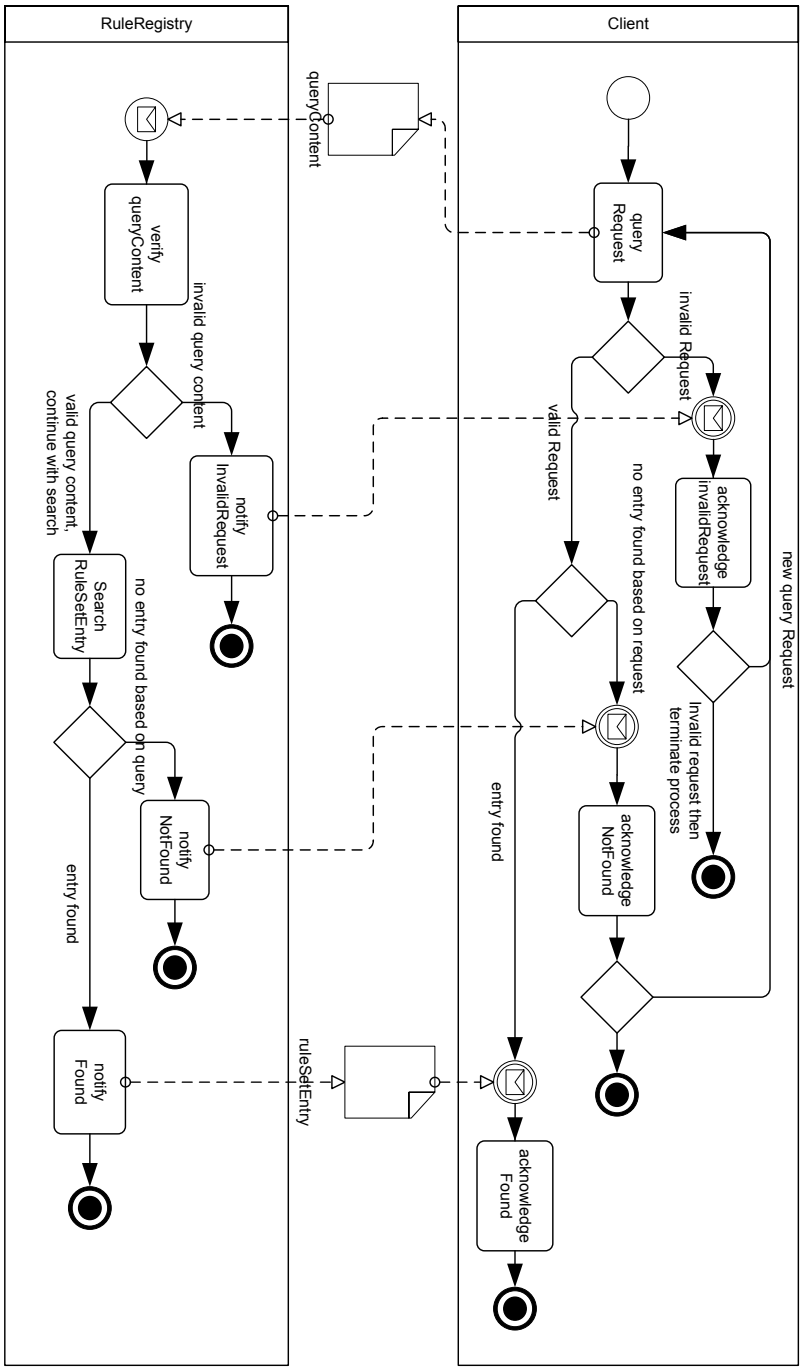


Figure 6: Query Process

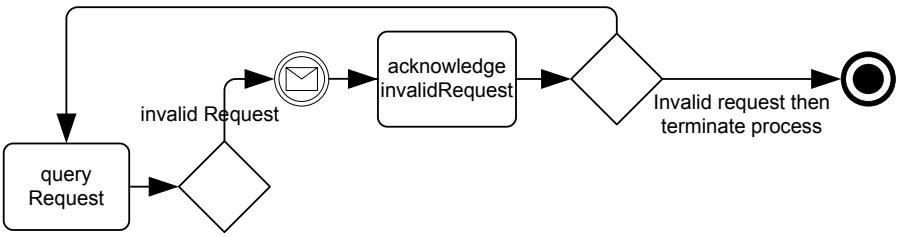


Figure 7: Loop

References

[BK08] H. Boley and M. Kifer. RIF Basic Logic Dialect. http://www.w3.org/2005/rules/wiki/BLD, 2008.

[BLB+00] K. Bennett, P. Layzell, D. Budgen, P. Brereton, L. Macaulay, and M. Munro. Service-Based Software: The Future for Flexible Software. In *Proceedings of the Seventh Asia-Pacific Software Engineering Conference (APSEC2000)*, pages 214 – 221. IEEE Computer Society, 2000. http://www.bds.ie/Pdf/ServiceOriented1.pdf.

[CMRW07] R. Chinnici, J. Moreau, A. Ryman, and S. Weerawarana. *Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language*. W3C, 2 edition, June 2007. http://www.w3.org/TR/wsdl20/, retrieved Novemeber 2008.

[GDPW08] A. Giurca, I. Diaconescu, E. Pascalau, and G. Wagner. On the Foundations of Web-based Registries for Business Rules. In *Proceedings of the 2nd International Symposium on Intelligent Distributed Computing, IDC2008*, volume 162 of *Studies in Computational Intelligence*, pages 251 – 255. Springer Berlin / Heidelberg, 2008. http://dx.doi.org/10.1007/978-3-540-85257-5_26.

[GHM+07] M. Gudgin, M. Hadley, N. Mendelsohn, J. Moreau, H. Frystyk Nielsen, A. Karmarkar, and Y. Lafon. *SOAP Version 1.2 Part 1: Messaging Framework (Second Edition)*. W3C, 2 edition, April 2007. http://www.w3.org/TR/soap12/, retrieved Novemeber 2008.

[GP08] A. Giurca and E. Pascalau. JSON Rules. In *Proceedings of the Proceedings of 4th Knowledge Engineering and Software Engineering, KESE 2008, collocated with KI 2008*. CEUR Workshop Proceedings, 2008.

[HW02] R. Heery and H. Wagner. A Metadata Registry for the Semantic Web. *D-Lib Magazine*, 8(5), May 2002. http://dlib.org/dlib/may02/wagner/05wagner.html.

[LVL96] C. Lovelock, S. Vandermerwe, and B. Lewis. *Services Marketing* . Prentice Hall Europe, 1996.

[Mic02] Sun Microsystems. JSR 93: JavaTM API for XML Registries 1.0 (JAXR). http://www.jcp.org/en/jsr/detail?id=93, June 2002.

[Mic07] Sun Microsystems. JSR 220: Enterprise JavaBeansTM 3.0. http://jcp.org/en/jsr/detail?id=220, November 2007.

[OAS01a] OASIS. ebXML Business Process Specification Schema Version 1.01. http://www.ebxml.org/specs/ebBPSS.pdf, May 2001.

[OAS01b] OASIS. ebXML Technical Architecture Specification v1.0.4. http://www.ebxml.org/specs/ebTA.pdf, February 2001.

[OAS04] OASIS. UDDI Version 3.0.2, UDDI Spec Technical Committee Draft, Dated 20041019. http://uddi.org/pubs/uddi-v3.0.2-20041019.pdf, October 2004.

[OMG07] OMG. Production Rule Representation (PRR), Beta 1. Technical report, OMG, November 2007.

[OMG08] OMG. Business Process Modeling Notation, V1.1. http://www.omg.org/spec/BPMN/1.1/PDF, January 2008.

[Wes07] M. Weske. *Business Process Management: Concepts, Languages, Architectures* . Springer-Verlag Berlin Heidelberg, 2007.

[WvdAD+06] P. Wohed, W.M.P. van der Aalst, M. Dumas, A.H.M. ter Hofstede, and N. Russell. On the Suitability of BPMN for Business Process Modelling . In *Business Process Management*, volume 4102 of *Lecture Notes in Computer Science*, pages 161–176. Springer Berlin / Heidelberg, 2006. http://dx.doi.org/10.1007/11841760_12, http://www.workflowpatterns.com/documentation/documents/BPMN-eval-BPM06.pdf.

The Young Researchers Workshop on Modeling and Management of Business Processes YRW-MBP 2009

Detective Information Flow Analysis for Business Processes*

Rafael Accorsi and Claus Wonnemann
Department of Telematics, University of Freiburg
{accorsi|wonnemann}@iig.uni-freiburg.de

Abstract: We report on ongoing work towards *a posteriori* detection of illegal information flows for business processes, focusing on the challenges involved in doing so. Resembling a forensic investigation, our approach aims at analyzing the audit trails resultant from the execution of the business processes, locating informations flows that violate the (non-functional) requirements stipulated by security policies. The goal is to obtain fine-grained evidence of policy compliance with respect to information flows.

Information flow (IF) characterizes the transfer of information from a classified container h to a public container l during the execution of a process. A "container" can be a logical or physical device, such as a process instance, network socket, or variable. An IF is labeled "illegal" whenever it violates the security policies expressing the non-functional requirements placed on the execution of the process, in particular the confidentiality and non-interferability of pieces of information.

Assuring that the executions of business processes do not allow illegal IF is essential in the context of regulatory compliance, which is largely automated by business processes deployed over service-oriented architectures. Most of the compliance requirements, and hence security policies, are concerned with the propagation of sensitive data, such as personally identifiable information, credit card numbers and the like.

However, only the minority of these policies, namely those denoting *safety properties*, can be enforced with access control mechanisms based on execution monitors. The majority of the security policies, in particular those expressing non-interference, denote *hyperproperties* for which mechanisms for runtime enforcement do not exist, nor are there techniques for *a posteriori* analysis of process executions tailored to the detection of illegal IF.

As a result of lacking techniques for IF control (IFC), illegal IF arising from *covert channels* – information channels whose primary purpose is not the transmission of information, but which are misused for this purpose – and *information interference* – the extraction of sensitive information from a set of accumulated data items or events – may go undetected. This leads to a situation in which the executions of a process, and the process itself, may be thought as complying with the security policies, whereas a thorough analysis for illegal IF could prove the opposite: IF leads to policy violations and non-compliance.

We investigate approaches for the a posteriori analysis of IF in business processes. Resembling a forensic investigation and building on authentic log files recorded during the

*See http://www.informatik.uni-freiburg.de/~accorsi/publications.html for a version containing the references.

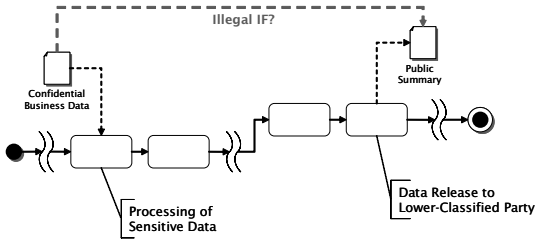


Figure 1: Confinement of data in business processes.

execution of processes, our goal is to advance IFC by developing approaches for the analysis of audit trails to detect illegal IF. Put another way, we investigate an approach for *a posteriori* IFC. In doing so, we do not prevent illegal IF; instead, we support the detection of illegal IF, considerably improving the state of the art audit mechanisms.

The threat of illicit information disclosure arises, for instance, when medical records are released in (assumedly) anonymized form, or when a company releases public statements that are based on confidential business data, as schematically depicted in Figure 1. Here, it must be ensured that data made public does not allow conclusions on secret information to be drawn, such as the identity of patients. Eventually, our approach either returns a proof of adherence to the policies, or gives evidence on violations and their circumstances.

To do so, we develop notions of IF for this setting and corresponding analysis techniques. Specifically, we currently focus on the following research issues:

- *Formalizations of IF properties for business processes.* We evaluate different formalizations of non-interference for their adequacy in a business process context. Further, we search for IF definitions apart from non-interference that capture additional covert channels (e.g. probabilistic and timing channels) and inference of data. A medium-term goal is to devise a language for the expression of hyperproperties, allowing the specification of a multitude of IF properties for business processes.

- *Data selection.* The basis for the analysis is log data recorded by the business process execution engine. While mechanisms for secure logging exist, it is unclear to date which pieces of information are in fact relevant for the detection of illegal IF. One of our efforts is thus to select the log data to be collected for the analysis.

Upcoming research challenges include, among others, the development of appropriate *analysis algorithms* and *accuracy measurement*. As for analysis algorithms, in considering hyperproperties, an analysis must look at *sets* of traces, interconnecting the events therein according to the IF policy. While this can happen by event correlation, we believe that data mining techniques, in particular those based on mixture models, allow for a more precise analysis of audit trails. Accuracy measurement subsumes *precision assessment* and *error estimation*: while false negatives must be ruled out, false positives are to be minimized and their occurrence quantified in a formal way.

Maintaining WS-BPEL Workflows Using Aspects

Connie Haoying Bao, Nicolas Gold, Mark Harman
King's College London, CREST
Department of Computer Science,
Strand, London, WC2R 2LS, UK.
{Haoying.Bao; Nicolas.Gold; Mark.Harman }@kcl.ac.uk

Abstract: In Service Oriented systems organisational processes are represented as WS-BPEL workflows, WS-BPEL is different from traditional workflow languages as a hybrid of block-based and graph-based language; it also has limited support for separation of concerns. Changes to such processes usually impact many places in the underlying system, without separating such cross-cutting concerns system maintenance is therefore difficult. This work proposed an Aspect Oriented solution to maintaining WS-BPEL workflows using meta-model transformation.

1 Introduction

Workflow language is the enabling technology in Business Process Management; it represent real world process that can be used to document organisational processes or carry out the process execution in its underlying management systems. In Service Oriented architecture where software system is more tightly aligned with organisational processes, service workflows represent these processes as well as organisational and non-functional requirements.

WS-BPEL has since become the emerging standard for representing workflows in Service Oriented systems. WS-BPEL is different from traditional workflow languages in that it is used by developers largely as a block-based structured language, as its origin is partly in XLANG, however, BPEL is also derived from WSFL, and therefore it is also partly graph-based. Because of these features traditional workflow techniques do not apply fully to such a hybrid language. Another limitation of BPEL is the lack of support for separation of concerns. Organisational processes are subject to constant change, maintaining the underlying system is to maintain the workflow they support. Changes usually affect many places in a process; without the support for cross-cutting concerns as first class entities, changes need to be directly integrated and implemented [CM07]. Maintaining Service Oriented workflows is therefore complexity and difficult.

Research in Aspect Oriented Software Development (AOSD) has introduced an powerful abstraction termed Aspects [KZ01], each aspect represents a cross-cutting concern and defines its implementation in respect of the base process. AOSD techniques are known to improve on modelling cross-cutting concerns as first class entities, such as changes to workflows. Model Driven Architecture (MDA) is a contemporary approach to describing and manageing meta-models. In particular executable models using imperative languages specify how models should behave at run time.

2 Proposed Approach

The contribution of this paper is in two fold, first it provides a model drive approach to weave executable aspects models into BPEL workflow model. Secondly the method uses a standard technology approach that preserves the service technology stack from aspectual language extensions. Previous AOSD approaches introduced aspectual constructs to existing languages[CF04][CM07] however there are risks involved in adopting closed solutions.

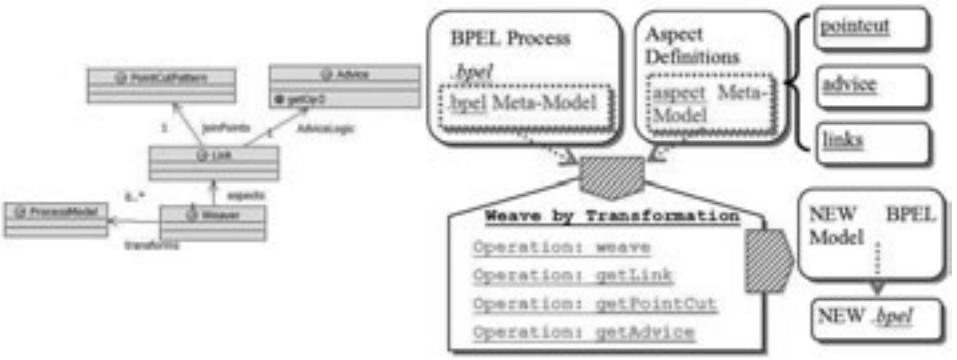


Figure 1. Design of Meta Weaver and the Weaving Framework

Using executable models as the weaving technique, the approach first propose a meta model for BPEL process. The meta model contains workflow constructs as well as point-cuts abstractions which represents possible cross-cutting concerns in BPEL workflow. Deriving from this meta model, aspect models are proposed, as shown on the left in Figure 1, with point-cut patterns, advice logic and a Link component that contains the associations between point-cuts and advice. A Weaver is modelled and implemented that contains the instructions for the executable aspect models to be woven into a BPEL process model. The diagram on the right of Figure 1 shows how the design is implemented. Instance of BPEL process and aspects are transformed into meta objects, by iternating the weaving rules on these meta objects a new process is created. The proposed approach is based on an aspect oriented modeling approach and shares similarities with reflective approach to meta-modelling.

Reference:

[CF04] Courbis, C, and Finkelstein, A. Towards an Aspect Weaving BPEL engine. In Proceedings of The Third AOSD Workshop on Aspects, Components and Patterns for infrastruction Software 2004

[CM07] Charfi, A. and Mezini, M.:" AO4BPEL: An Aspect-Oriented Extension to BPEL".World Wide Web Journal: Recent Advances on Web Services (special issue), to appear, Springer, 2007.

[KM05] Kiczales, G., Mezini, M.: Aspect-oriented programming and modular reasoning. In: Proceedings of the 27th International Conference on Software Engineering (AOSD), 2005.

[Kz01] Kiczales, G.; et.al: An Overview of AspectJ.Lecture Notes in Computer Science, 2001, 2072, 327-355

Declarative Workflow Modeling with UML class diagrams and OCL

Jens Brüning

University of Rostock, Department of Computer Science,
A.-Einstein-Str. 21, 18059 Rostock, Germany
jens.bruening@uni-rostock.de

Abstract: This paper describes an approach of modeling workflows with UML class diagrams and OCL constraints [OCL06] in a declarative way. These are modeled in the UML tool USE [USE08] that can generate object diagrams (snapshots) out of UML class diagrams. USE checks specified OCL constraints against the generated snapshots. With the declarative workflow modeling approach presented here, activity model states are integrated in the object model states. By analyzing these snapshots the model is validated against requirements.

In figure 1 a UML class diagram for modeling flat workflows is presented. The class *Process* has an attribute *name* and contains a set of activities which is described by the association *belongs* between *Process* and *Activity*. Atomic actions in the process that are executed in the workflow are expressed by the class *Activity*. Activities have also names that describe the actions and a state in which the activity is just in the time, the snapshot of the system is taken. Possible states for the activities are described in the enumeration *State*. A derived state of the process depends on the states of the included activities and is delivered by the operation *getState* in the class *Process*. Further on, the operation *getActivity* is needed by the subsequent OCL constraints and returns the requested activity instance. The described operations are specified in OCL expressions and will be interpreted by USE.

Figure 1: UML model for flat workflow specifications modeled in USE

Actual states of an activity instance can be changed by invoking the methods *skip()*, *start()* or *done()*.The functionality of the operations are expressed by OCL pre- and post-conditions. Operation *enabled()* proves on basis of the current object model state and the inner state of the activity itself, if it is enabled and thus can be started. This method is also coded in an OCL expression.

Without constraints, the model of figure 1 is insufficient to express concrete workflow definitions. OCL invariants are used to get process definitions with its containing activities. For example, the following invariant guarantees that the process “processing” consists of the activities “generate invoice”, “send invoice”, “debit” and “send goods”.

```
context Process inv OrderProcessing:
  self.name='processing' implies
    self.activity.name = Bag{'generate invoice', 'send invoice','debit','send goods'}
```

Declarative workflow models are flexible because all execution paths of the modelled activities are allowed if they are not forbidden explicitly [PA07]. All activities are in an interleaved relationship by default. Other classical temporal relations like sequences or alternatives can be modelled by constraints. In the example presented above there should be a sequence relation between “generate invoice” and “send invoice” and an alternative between “debit” and “generate invoice”. These relationships can be expressed by the following OCL invariants.

```
context Process inv Accounting_Generate_Send_Sequence:
  self.name='processing' implies
    (self.getActivity('send invoice').state=#running implies self.getActivity('generate invoice').state=#done)

context Process inv Accounting_Invoice_Debit_Alternative:
  self.name='processing' implies
    ((self.getActivity('generate invoice').state=#running implies  self.getActivity('debit').state=#skipped) and
     (self.getActivity('debit').state=#running implies self.getActivity('generate invoice').state=#skipped))
```

Further on, declarative workflow models can express additional relations between activities that are not possible to model in the traditional workflow modelling languages like BPMN or UML Activity Diagrams. For example, simply expressing that two activities must not occur at the same time [PA07]. This is modelled in the next OCL invariant where the activities “send goods” and “debit” are not allowed to be both in the state running at the same time.

```
context Process inv Debit_SendGoods_Entangled:
  self.name='processing' implies
    not (self.getActivity('generate invoice').state=#running and self.getActivity('send goods').state=#running)
```

The next working step for this modelling approach is, to express additional temporal relations in the declarative way on basis of the workflow patterns. Furthermore, it can be extended to hierarchical workflow models.

References

[OCL06] Object Constraint Language (OCL) Specification 2.0, OMG, http://www.omg.org/docs/formal/06-05-01.pdf, 2006.

[PA07] Pesic, M., Aalst, W., et.al.: Constraint-Based Workflow Models: Change Made Easy. In: LNCS 4103, pp. 77–94, Berlin, Springer, 2007.

[USE08] A UML-based Specification Environment, University of Bremen, http://www.db.informatik.uni-bremen.de/projects/use/, Bremen, 2008.

An Approach for Semantic Checks of Process Models

Sven Feja
Department of Computer Science, Christian-Albrechts-University Kiel
svfe@informatik.uni-kiel.de

Abstract: This paper presents an approach for using model checking for process models and workflows. The approach allows it to graphically state semantic validation rules on the level of processes. This enables the process modeler to use model checking techniques to get higher quality models for the further software development process.

1 Introduction

The beginning of a software development process is the collection of requirements for the new software solution. Often these requirements are collected in numerous specification documents. The upcoming Model-Driven Software Development (MDSD) aims to use different kinds of models in addition to these specification documents. The type of models varies from high level business processes to implementation near component models. But both, the specification documents and the models, cannot be checked easily automatic for correctness.

The check of models has to be divided into two parts – the syntax and semantic check. Automatic syntax checks are often provided by the modeling tools. But especially, the automatic semantic check of high level models like business processes has not been solved satisfactorily. This results from the lack that it is not possible to state formalized requirements on the level of business processes. In hardware related development areas the formalization of requirements is widespread. But in software development processes the requirements mostly are only defined unformalized and implicit in the requirement documents and models.

A possible technique to verify the semantically correctness of a model is Model Checking. This technique allows the automatic formal verification of models. But Model Checking requires despite a formal model a formalized textual rule as input. In the sense of MDSD this is not a satisfying solution. Therefore, an approach for the definiton of rules on the level of models is needed.

2 Temporal Logics Visualization Framework

The use of Model Checking for e. g. business processes requires two main tasks. First, the process model needs to be transformed into an appropriate format for a model checker. Second, the formal requirements have to be definable on the level of the business process. The first task is realizable with a transformation of the graphical model (often represented

as XML) to the input format of the model checker. The second task could be solved by the in [FF08] proposed Temporal Logics Visualization Framework (TLVF). This framework allows the definition of requirements as graphical validation rules. Thus, it is possible to translate the graphical rules (although represented as XML) into the input format of the model checker. Furthermore, the framework handles required model transformation tasks.

Figure 1 shows the main concept of the use of the TLVF in a MDSD process (MDSD is symbolized by the pictures of the MDA framework [The03] and the cover of a MDSD book [VS06]). The figure although depicts a small example graphical rule in EPC-G-CTL notation. The name of the notation results from the used components. This notation allows it to specify **G**raphical rules in the temporal language **CTL** for the business process model **EPC**. The structure and all essential components of the TLVF are described in [FF08].

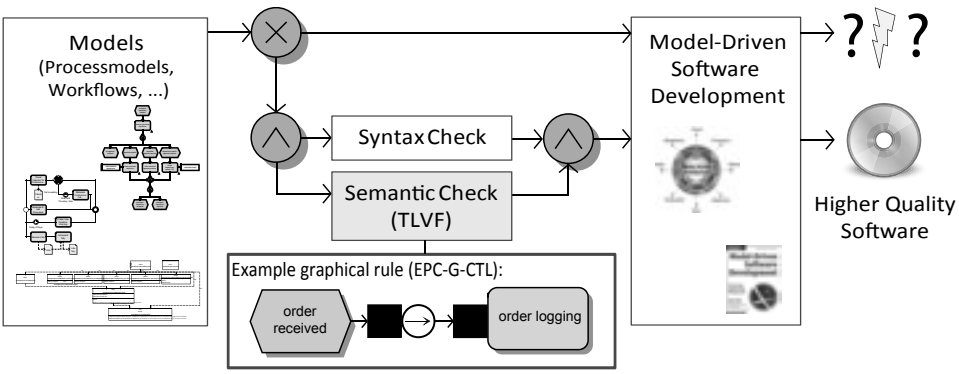


Figure 1: The main concept.

The presented TLVF provides a concept for using model checking for process models and workflows. In further steps the creation of graphical rules could be simplified. This can be achieved by providing rule patterns which represents often used rule constructs. Alternatively, the definition of rules should be supported by dialog-based wizards. Next to the creation of rules the reuse of models and their related rules is an important point. In a first attempt different types of rules need to be defined. Though, it will be possible to use rules for a specific szenario, a specific domain or all types of models. The resulting rules could be stored in a database and are usable for the appropriate type of modeling task.

References

[FF08] Sven Feja and Daniel Fötsch. Model Checking with Graphical Validation Rules. In *15th IEEE International Conference on the Engineering of Computer-Based Systems (ECBS 2008), Belfast, NI, GB*, pages 117–125. IEEE Computer Society, April 2008.

[The03] The Object Management Group (OMG). MDA Guide Version 1.0.1. June 2003.

[VS06] Markus Völter and Thomas Stahl. *Model-Driven Software Development : Technology, Engineering, Management*. John Wiley & Sons, June 2006.

Process Modeling as a Basis for Auditing Information Privacy

Ralph Herkenhöner, Hermann de Meer
Faculty of Informatics and Mathematics
University of Passau, Germany
{rhk|hdm}@fim.uni-passau.de

Abstract: Information privacy has become an important task for every data processing organization. To meet its demands, organizations apply privacy-enhancing technologies and identity management to their business processes. But the increasing number of privacy breaches shows that this task is complex and not well understood.

In this position paper, a formal method for modeling an proving information privacy within a process model is envisioned. Such a model would allow an integration at process design, increase the understanding and effectiveness of the privacy protection mechanisms, and enable compliance checks and data protection auditing.

Acknowledgment: This research work was funded by the Federal Ministry of Education and Research of the Federal Republic of Germany (Funding-Id: 01|S08016B).

1 Information Privacy in Business Processes

With the shift from a paper-based to a digital information society, collection and processing of data have become much easier and very fast. Without the proper carefulness, information easily can be used for purposes they are neither intended for nor permitted to. This problem is documented by the increasing number of incidents regarding data leakages, missuses of personal data and even identity thefts. Like in the latest incident, where a secret file of the US-Army was found by a civilian on an second-hand mp3-player[1], often the reason is the missing awareness in using non-confidential communication channels. This results in loss of control on information flow and enables data misuse and theft. A predictable and comprehensible information flow could avoid such incidents or even aids investigation. We call this information privacy.

Reasonable information privacy requires at least reconcilement of privacy and accountability of a data subject's electronic interaction (empowering the data subject in its self-determination) [CSS+05] and a privacy-aware identity and information lifecycle management within the data processing organizations (including compliance, data protection, and access control) [MB07]. Both depend on the underlying business processes and the involved information privacy techniques.

[1] In January 2009, the New Zealand news channel One News gets possession of a 60 page long file containing name and contact information of US-Soldiers, details on equipment, and other secret information. The file was stored on an mp3-player bought at a second-hand shop by a civilian.

2 Privacy-Aware Process Modeling

As presented in a case study on biobanks [Her08], modeling business processes can help to gain a better understanding on information privacy within the data processing organization using the UML extension UMLsec.

The idea is to integrate security characteristics within the process model granting information privacy . Such an integration enables the additional benefit that the interaction of control flow, information flow, and information privacy is visible within a single model. With proper levels of abstraction—avoiding an information overload during examination—such a model increases the understanding of the privacy mechanisms and, for example, allows data protection auditing [Aud01].

For that purpose, the modeled security characteristics must cover the major security target *confidentiality*, including the *non-propagation* of protected information and the non-observability of the processes by a non-authorized third. To assure the processing of correct data subjects' information and to avoid incorrect alterations, *integrity* and *authenticity* must be taken into account. Nevertheless, as the protection of the data subjects' privacy is the main target, anonymization mechanisms have to be clearly specified within the model. This also includes the pseudonym management in a case of pseudonymization. Last but not least, there must be an *accountability* and *non-repudiation* for every access on and processing of the protected data. Therefore, access and processing have to be able to be audited.

3 Proving Information Privacy

The goal of data protection auditing is to comprehensibly prove information privacy. Therefore, it is helpful to have a formal argumentation to prove correctness and effectiveness for a given privacy protection concept. For this purpose, using UMLsec for process modeling, has an additional benefit. As shown by Jürjens [Jür03], UMLsec provides a formal basis to prove the fulfillment of security targets within the process model . For utilization of these methods to prove information privacy, it is necessary to refine information privacy from provable security characteristics. This goal is located within the research field of the authors.

References

[Aud01] Data Protection - Complete Audit Guide. Technical report, The Information Commissioner's Office, UK, 2001.

[CSS+05] J. Camenisch, A. Shelat, D. Sommer, S. Fischer-Hübner, M. Hansen, H. Krasemann, G. Lacoste, R. Leenes, and J. Tseng. Privacy and Identity Management for Everyone. In *DIM '05: Proceedings of the 2005 Workshop on Digital Identity Management*, pages 20–27, New York, NY, USA, 2005. ACM.

[Her08] Ralph Herkenhöner. Process Modeling for Privacy-conformant Biobanking: Case Studies on Modeling in UMLsec. In *Proceedings of the 6th International Workshop on Security Information Systems*, pages 3–12, Portugal, 2008. INSTICC Press.

[Jür03] Jan Jürjens. *Secure Systems Development with UML.* SpringerVerlag, 2003.

[MB07] Marco Casassa Mont and Filipe Beato. On Parametric Obligation Policies: Enabling Privacy-Aware Information Lifecycle Management in Enterprises. In *POLICY*, pages 51–55. IEEE Computer Society, 2007.

Generating WS-SecurityPolicy Documents via Security Model Transformation

Meiko Jensen
Horst Görtz Institute for IT-Security, Ruhr University Bochum
Meiko.Jensen@ruhr-uni-bochum.de

Abstract: When SOA-based business processes are to be enhanced with security properties, the model-driven business process development approach enables an easier and more reliable security definition compared to manually crafting the security realizations afterwards. In this paper, we outline an appropriate security model definition and transformation approach, targeting the WS-SecurityPolicy and WS-BPEL specifications, in order to enable a Web-Service-based secure business process development.

A crucial part of SOA-based business process design is the ability to define security properties for the service invocations. Thus, a business process modeling tool must provide appropriate models and capabilities to specify these properties on a semantical level in order to automatically create technical realizations at the implementation layer.

In this paper, we outline a model-driven approach for adding security properties to a business process model. The approach consists of a security model definition, an appropriate model transformation, and a model-specific technical realization based on the Web Services technology (i.e. WS-BPEL and WS-SecurityPolicy).

The Security Model Approach

When it comes to the issue of specifying security properties on a process level, the conjunction of the WS-BPEL and WS-SecurityPolicy specifications is lacking completeness. Though both specifications rely on the Web Service description (WSDL), there is no possibility for specifying security properties at the process level, e.g. end-to-end encryption of particular data items within the complete business process execution. Thus, it is necessary to investigate the requirements of enabling security policy assertions to directly annotate the process document without intermediating the WSDL. This can be addressed by using the model-driven approach. The idea is to create a separate *security model view* for the process model, which directly annotates the communication parts of the underlying process model. Thus, the process model can be defined by a business-semantics-aware process developer, and can then be annotated with appropriate security properties by a separate *security architect* (cmp. Figure 1). Then, both the process model and the security model can be tranformed into appropriate process and security descriptions (based on WS-BPEL and WS-SecurityPolicy), which are related in that the security description directly annotates the process description. Once these descriptions are deployed to an appropriate Web Ser-

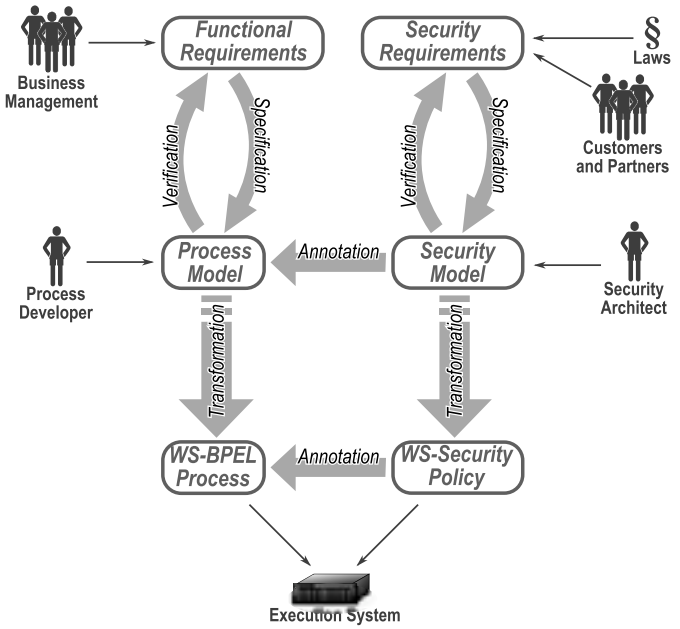


Figure 1: Conceptional view on the full security model approach

vice execution engine, the resulting business process is ready to run, with the appropriate security properties enabled (see also [JF09]).

Future Work

A major part of the presented approach consists in specifying appropriate transformations of security model properties to WS-SecurityPolicy equivalents. Thus, for every security property considered in the security model (which by now are confidentiality, data integrity, and user authentication) it is necessary to specify an appropriate WS-SecurityPolicy fragment that properly reflects the particular security aspect. Then, these fragments must be completed with some more details (algorithms to use, data items to consider, Web Services involved etc.), and the results must be joined into WS-SecurityPolicy documents for each Web Service endpoint of the workflow.

References

[JF09] Meiko Jensen and Sven Feja. A Security Modeling Approach for Web-Service-based Business Processes. In *Proceedings of the 7th IEEE Workshop on Model-Based Development for Computer-Based Systems (MBD)*, 2009.

Semantic Compliance Management in Business Process Management

Marwane El Kharbili

ARIS Research, IDS Scheer AG, Altenkesselerstrasse 17, 66115, Saarbrücken, Germany. Marwane.Elkharbili@ids-scheer.com

Abstract: Processes count to the most important assets of companies. Ensuring the compliance of processes to legal regulations, governance guidelines, and strategic business requirements is indispensable to controlling business behavior. Implementing business process compliance requires means for modeling and enforcing compliance measures. In this work, we motivate the need for automation in compliance management and introduce the role of policies. We propose a policy-based framework for business process compliance management and detail its architecture as part of the SUPER research project on semantic business process management.

1 Problem Definition & Research Motivation

Business Process Management (BPM) is the discipline of capturing, modeling, implementing, and controlling all activities taking place in an environment defining the enterprise, and this, in an integrated manner. Several languages, frameworks, and tools that support one or many of the listed aspects are available. Organizations do not only own business processes, they are also subject to regulations. Not being compliant to regulations diminishes the added-value that business processes represent for a given organization, e.g. through non-optimal alignment with, among other aspects, (i) quality standards, (ii) business partner service agreements or (iii) non-identified security flaws[1].

Non-compliance to regulations can also be the cause of judiciary pursuits as many financial scandals in recent years have shown. Consequently, non-compliance has both short-term (e.g. cost savings, reduced governance effort) and long-term (e.g. judiciary pursuits, market confidence) consequences. Compliance management is the term referring to the definition of means to avoid such harming outcomes by controlling an enterprise's activities. By extension, compliance management also refers to standards, frameworks, and software used to ensure a company's observance of legal texts and behavior according to pre-defined business requirements. In the context of BPM, compliance management applies on business processes and all related resources like data, people and systems.

[1] Non-identified security leaks can drop the overall quality of the business processes by making them vulnerable to malicious attacks.

2 Problem Domain & Preliminary Results

One possible approach to formalizing regulations in order to be able to verify and validate them is the use of policies and rules. Placed in the context of semantic business process management (i.e. modelling, configuring, executing and analyzing semantically enriched business processes as defined by the SUPER project[2]), our work tackles compliance management proposing the use of ontological representations of regulations as semantic policies and semantic rules.

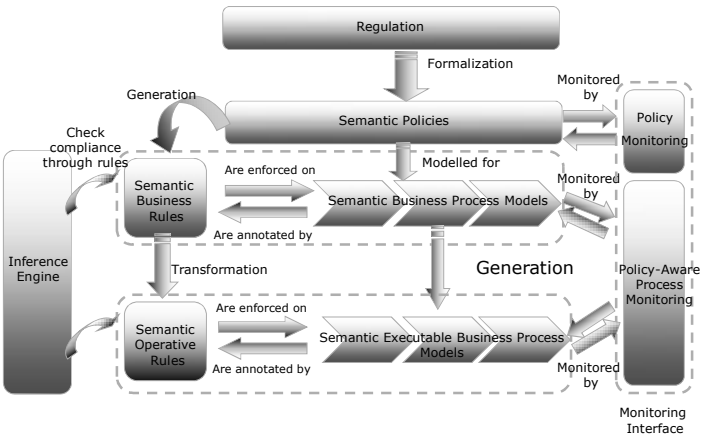


Figure 1: Overall approach to semantic compliance management in BPM

The main rationale here is that policies can model decision-making while business rules model action-taking. Another rationale is that different levels of description of business processes and services require different representations of these policies and rules. The overall approach we propose (Fig.1) has been introduced in [Kha08]. An ontology stack called the BPRO (Business Policy and Rules Ontology) [Kha09] has been designed for this purpose. Using the BPRO, BP descriptions can be extended to contain semantic information necessary to conduct compliance checking. A case study and a concrete rule language for serialization of regulation ontologies are the next steps in showcasing the use of our framework for compliance checking.

References

[Kha08] El Kharbili, M.; Stein, S.; Markovic, I. & Pulvermüller, E. Towards a Framework for Semantic Business Process Compliance Management *Proceedings of the workshop on Governance, Risk and Compliance for Information Systems (GRCIS 2008) CEUR Proceedings, Vol. 339*, 1-15.

[Kha09] El Kharbili, M.; & Pulvermüller, E. A Semantic Framework for Compliance Management in Business Process Management. Proceedings of the international conference on business process and services computing. March 2003. To appear.

[2] www.ip-super.org

Modelling and Solving Configuration Problems on Business Processes Using a Multi-Level Constraint Satisfaction Approach

Wolfgang Runte
Institute of Computer Science
University of Osnabrueck
D-49069 Osnabrueck, Germany
woru@informatik.uni-osnabrueck.de

Abstract: In this paper we present our ideas to apply constraint satisfaction on business processes. We propose a multi-level constraint satisfaction approach to handle different levels of abstraction in business process modelling.

1 Motivation

A main problem in the area of business process modelling is the management of the dependencies between processes. A lack of attention can easily result in inconsistent process models. Inconsistencies should be discovered in an early stage of modelling to reduce the amount of time and costs. The ordering of business processes depends in most cases on complex relations between the processes. Commonly there are no single business processes with no relations to other processes. Usually processes depend on each other and their input/output data respectively. While this scenario describes a sequential dependency there are hierarchical dependencies as well: One or more processes can be the sub-item(s) of a higher-ordered process. In this paper we propose an approach to employ constraint techniques to ensure the sequential and hierarchical consistency of business processes.

2 Consistent Configurations through Constraint Satisfaction

We aim at developing a solution process for handling different levels of nested business processes. Flexibility is an important criteria, because different kinds of problem and/or sub-problem dependencies require the flexibility to define different solution strategies and the application of problem specific solving algorithms.

During the modelling process of business processes there are static dependencies that can be checked to ensure a consistent configuration. Dependencies relate the data flow and the input/output interface of each process. We propose the application of AI methods out of the area of *knowledge based configuration* [Stu97] to build consistent configurations

of business processes. The *constraint satisfaction problem* (CSP) is an adequate way to model complex relations between components [Dec03]. CSPs have been in the focus of intensive research and experiences for decades which lead to efficient algorithms and heuristics. Main benefits are the reduction of problem size, the efficient generation of problem solutions and so the guarantee that specific relations hold.

Accordingly constraints have to be satisfied in order that processes are allowed to follow each other and represent a sequence of business processes. Another sort of constraints have to be defined to specify processes to be allowed to be nested sub-items of upper processes, in order to satisfy all requirements of super- and sub-processes. Different layers of processes in hierarchies define different sub-problems. For each sub-problem another solution strategy can be applied depending on the value domain of the involved variables and the problem structure. For the emerging *multi-level constraint problem* the integration of local solutions of sub-processes has to be done on the higher-ordered level leading to global solutions and hence consistent configurations.

3 Related Work

The paper proposes the application of constraint satisfaction to ensure the consistency in sequential and hierarchical relations of business processes. Related work to this approach therefore may be found in the domain of constraint satisfaction [Dec03]. Existing constraint satisfaction approaches may be employed to achieve to goals outlined in this paper. Hierarchies of constraint problems can be described by the *composite CSP* whose main application is in the configuration domain [SF96]. An interesting field is the research on the coordination of cooperative constraint solvers [AM98]. Planning algorithms (from the field of AI) would be an alternative to describe the orderings of sequences of business processes [RN02]. However CSP is more appropriate to handle hierarchies and provide more flexibility expressing consistency requirements.

References

[AM98] Farhad Arbab and Eric Monfroy. Coordination of Heterogeneous Distributed Cooperative Constraint Solving. *ACM SIGAPP Applied Computing Review*, 6(2):4–17, 1998.

[Dec03] Rina Dechter. *Constraint Processing*. The Morgan Kaufmann Series in Artificial Intelligence. Morgan Kaufmann Publishers, San Francisco, California, USA, 2003.

[RN02] Stuart J. Russel and Peter Norvig. *Artificial Intelligence: A Modern Approach (The Intelligent Agent Book)*. Prentice Hall, Upper Saddle River, New Jersey, USA, 2002.

[SF96] Daniel Sabin and Eugene C. Freuder. Configuration as Composite Constraint Satisfaction. In Boi V. Faltings and Eugene C. Freuder, editors, *Configuration – Papers from the AAAI Fall Symposium*, pages 28–36, Menlo Park, California, USA, 1996. AAAI Press.

[Stu97] Markus Stumptner. An Overview of Knowledge-Based Configuration. *AI Communications (AICOM)*, 10(2):111–125, July 1997.

Modelling Critical Success Factors *in mCommerce-Programs*

Early Research/ Concept Paper, January 2009

Andreas Rusnjak
Institut für Informatik/ AG Angewandte Informatik (Wirtschaftsinformatik), Uni Kiel
aru@informatik.uni-kiel.de

Abstract: Actual developments and trends on Mobile Commerce are prompting more and more companies or startups in dislocating or positioning their business model in this attractive market. To be successful, it is necessary to know about the own strengths and weaknesses, about customers, competitors and critical success factors. The goal of this thesis is to provide a contribution about considering pertinent critical success factors during the modeling of business processes or verifying existing business processes to the existence of critical success factors.

1 Initial Situation

Increasingly powerful mobile devices, more favorable data-rates, an increasing number of multimedia-based offerings and a better crosslinking of applications among each other are driving a highly dynamic and rapid development on the markets of mCommerce. Currently consumers are able to buy parking-/ event-tickets, employ online-banking, define their position and make use of additional services like timetables of public transport with their devices. In addition to this the mobile device is also used as a multimedia player to watch videos, movies or playing music (cf. [AnRa08]; [HuLi+07]). Many User having a very intensive relation with their mobile devices and perceive it as a familiar object. In this respect it isn't very surprisingly that wireless and other mobile technologies is not only changing the social cooperation but also playing a big role in revolutionizing fabrication, trade and distribution of services [Baue+08].

2 Problem

This trend will continue in the following years, accompanied with new technologies (e.g. 4G), increasing market volumes and competition. Companies need to monitoring actual developments and future trends to gain profit from the market for mobile services, the so-called Mobile Business [Baue+08]. They should be able to answer basic questions about essential strategic-competitive instruments, competitive potential, possibilities for differentiation or possibilities for reducing costs via mCommerce-Applications [Link03]. Adequate strategies to deal with the problem require the knowledge about critical success factors: Factors with sustainable influence to the company's business success and factors which are describing potential competitive advantages. Since these factors mainly affect the business processes, e.g. online communication with customers, integration of additional (virtual community) elements or the facilitation of operations via newest technology [Böin01], they must be captured by business process modeling technologies.

3 Goals & Procedure

The goal is to build technology-driven business models of companies which point them to critical success factors during modeling business processes and help them to map this factors and allowing them to check the availability in existing business processes.

mBusiness, mCommerce as well as the mobile internet and associated critical success factors are marking a very young area of research. Although there are increasing number of internet sources and studies, the amount of literature is relatively low [Bern08]. For this reason we will use complementary literature from related disciplines as an overview about actual developments and results of relevant studies.

The identified critical success factors have to be integrated in consequential business processes and will be modeled with e.g. ARIS, one of the leading Tools for business process modeling. Referring to the Target-Performance-Comparison checking the economic adaptability an evaluation will showing the goal of this thesis and references to b2b-business-processes.

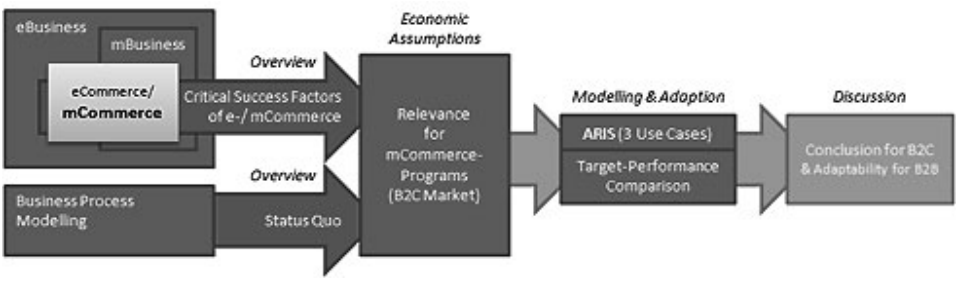


Figure 1: Procedure

Bibliography

[AnRa08] Anderson, Janna; Ranie, Lee: *The Future of the Internet III*; online from: http://www.pewinternet.org/pdfs/PIP_FutureInternet3.pdf; Access Date: 23.01.2009

[Baue+08] Bauer, Hans H.: *Erfolgsfaktoren des Mobile Marketing*; Berlin: Springer, 2008

[Bern08] Bernauer, Dominik: *Mobile Internet*; Saarbrücken: VDM Verlag Dr. Müller AG, 2008

[Böin01] Böing, Christian: *Erfolgsfaktoren im Business-to-Consumer-E-Commerce*; Wiesbaden: Gabler, 2001

[HuLi+07] Huang, Hao; Liu Lu; Wang, Jianjun: *Diffusion of Mobile Commerce Application in the Market*; online from: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4428127&isnumber=4427647; Access Date: 23.01.2009

[Link03] Link, Jörg: *Mobile Commerce. Gewinnpotenziale einer stillen Revolution*; Berlin: Springer, 2003

Concept-Driven Engineering for Supporting Different Views of Models

Peggy Schmidt
Christian-Albrechts-University Kiel, Department of Computer Science
pesc@is.informatik.uni-kiel.de

Abstract: This paper investigates the the development and evolution of concepts and the management of transformers, which adds semantics to the concepts. We illustrate how concepts, their variants and transformers can be developed via cooperation.

In order to meet increasing user demands for more various software systems, we are revising the software developing process to accommodate mass customisation based upon Concept-Driven Engineering (CDE). CDE is a strategy to application specification and generation of new concepts via transformers. The concept and its transformation rather than the implementation is central to the development process. It allows automation for specification from early stages to executable specification and code generation. CDE continue to be a challenges in building complex software systems that have several variations and options. Software development is based upon a lot of specifications and implementations such as feature models, UML models and code, which are in different formats but share a certain amount of information. CDE is similar to the ideas of Model Driven Engineering (MDE) and Software Product Line Engineering, whereby we use the term concept instead model. A concept can be a model or a specification and is defined on a concept structure and a set of transformers. Concepts are assigned to a set of transformers, which generates new concepts. In this sense also a software system is a concept. One main remark is the management of the evolution of concepts and its transformers. Concept-Driven Engineering supports that requirements of the software do not remain constant during its development time and therefore the specification has to evolved and refined in order to fulfil the new requirements. One remark should lie an the management of the transformers that can be effectively be used on specifications. Against MDE CDE focus on the transformers which carry out the semantics of the specification, resp. the model. It is needful to study how transformers will affect the development process, that means how easy is a transformer to use, resp. to reuse. CDE abilities rely on the detailed transformer designer's knowledge of concept structure and development work flow while considering software system knowledge, software engineering techniques and methods. The aim of CDE is to avoid the development concepts and transformers which are in downstream development incapable to use.

When developers change one concept simultaneously, we need to propagate these changes across all concepts to guarantee them consistent. The process of synchronization propagates changes among specifications in different stages to all involved participants. Exchange of models between local platforms is still a challenging issue. The exchange and

Abbildung 1: ConceptManger for Supporting Concept-Driven Engineering

the synchronization of different local copies between local development systems is so far a tough problem. Tools like subversion support the developers by parallel development, global revisioning, create working copy. The main problem is how to work in parallel and obviate one from overwriting the work of another. Tools like subversion use therefore mechanisms like copy-modify-merge or lock-modify-unlock. We have been developing a tool for collaborative management of concepts, called ConceptManager. The ConceptManager is based upon concept structures. Developers can create concepts. After the activation of a concept all other developers with access rights on this concept can use a concept. When developer works together on a concept we transmit only these parts which were modified and where the other developers are involved. To supports this feature we assign each development task of one developer to a component. Overlayed parts of different developers are connected via ports. Thus a component consists only of a partial copy of the developing software system. This problem faces how to reflect the changes that have occurred in one concept to involved parts of the other developers and in the transformed concepts . An other important feature is that we may have a set of possible modifications on concepts (think on different realizations). The ConceptManger will support the negotiation process between developers and is based upon the ideas of a earlier merge of overlayed concept parts. Additionally we have to pay attention that a transformation can be run on concepts, provided the concepts have compatible concept definitions.

Figure 1 represents a piece of our tool ConceptManager. It shows the concept Person, which consists of the concepts Name and Adress. Later we want to build a database and a java class for this concept. But a database table and a Java class itself are concepts. To get a database and a Java class we have to merge to concepts together (DBPersonMerge and JavaClassPersonMerge). For a given concepts we may have a set of transformers, e.g. to get a database definition script we need a transformer what generates it. Only now the transformer adds semantics to a concept. Other useful feature are the reuse of concepts through the derivation possibility of new concepts from older ones, the search engine, and the definition of constraints on concept structures.

GI-Edition Lecture Notes in Informatics

P-1 Gregor Engels, Andreas Oberweis, Albert Zündorf (Hrsg.): Modellierung 2001.

P-2 Mikhail Godlevsky, Heinrich C. Mayr (Hrsg.): Information Systems Technology and its Applications, ISTA'2001.

P-3 Ana M. Moreno, Reind P. van de Riet (Hrsg.): Applications of Natural Language to Information Systems, NLDB'2001.

P-4 H. Wörn, J. Mühling, C. Vahl, H.-P. Meinzer (Hrsg.): Rechner- und sensorgestützte Chirurgie; Workshop des SFB 414.

P-5 Andy Schürr (Hg.): OMER – Object-Oriented Modeling of Embedded Real-Time Systems.

P-6 Hans-Jürgen Appelrath, Rolf Beyer, Uwe Marquardt, Heinrich C. Mayr, Claudia Steinberger (Hrsg.): Unternehmen Hochschule, UH'2001.

P-7 Andy Evans, Robert France, Ana Moreira, Bernhard Rumpe (Hrsg.): Practical UML-Based Rigorous Development Methods – Countering or Integrating the extremists, pUML'2001.

P-8 Reinhard Keil-Slawik, Johannes Magenheim (Hrsg.): Informatikunterricht und Medienbildung, INFOS'2001.

P-9 Jan von Knop, Wilhelm Haverkamp (Hrsg.): Innovative Anwendungen in Kommunikationsnetzen, 15. DFN Arbeitstagung.

P-10 Mirjam Minor, Steffen Staab (Hrsg.): 1st German Workshop on Experience Management: Sharing Experiences about the Sharing Experience.

P-11 Michael Weber, Frank Kargl (Hrsg.): Mobile Ad-Hoc Netzwerke, WMAN 2002.

P-12 Martin Glinz, Günther Müller-Luschnat (Hrsg.): Modellierung 2002.

P-13 Jan von Knop, Peter Schirmbacher and Viljan Mahni_ (Hrsg.): The Changing Universities – The Role of Technology.

P-14 Robert Tolksdorf, Rainer Eckstein (Hrsg.): XML-Technologien für das Semantic Web – XSW 2002.

P-15 Hans-Bernd Bludau, Andreas Koop (Hrsg.): Mobile Computing in Medicine.

P-16 J. Felix Hampe, Gerhard Schwabe (Hrsg.): Mobile and Collaborative Busi-ness 2002.

P-17 Jan von Knop, Wilhelm Haverkamp (Hrsg.): Zukunft der Netze –Die Verletzbarkeit meistern, 16. DFN Arbeitstagung.

P-18 Elmar J. Sinz, Markus Plaha (Hrsg.): Modellierung betrieblicher Informationssysteme – MobIS 2002.

P-19 Sigrid Schubert, Bernd Reusch, Norbert Jesse (Hrsg.): Informatik bewegt – Informatik 2002 – 32. Jahrestagung der Gesellschaft für Informatik e.V. (GI) 30.Sept.-3.Okt. 2002 in Dortmund.

P-20 Sigrid Schubert, Bernd Reusch, Norbert Jesse (Hrsg.): Informatik bewegt – Informatik 2002 – 32. Jahrestagung der Gesellschaft für Informatik e.V. (GI) 30.Sept.-3.Okt. 2002 in Dortmund (Ergänzungsband).

P-21 Jörg Desel, Mathias Weske (Hrsg.): Promise 2002: Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen.

P-22 Sigrid Schubert, Johannes Magenheim, Peter Hubwieser, Torsten Brinda (Hrsg.): Forschungsbeiträge zur "Didaktik der Informatik" – Theorie, Praxis, Evaluation.

P-23 Thorsten Spitta, Jens Borchers, Harry M. Sneed (Hrsg.): Software Management 2002 – Fortschritt durch Beständigkeit

P-24 Rainer Eckstein, Robert Tolksdorf (Hrsg.): XMIDX 2003 – XML-Technologien für Middleware – Middleware für XML-Anwendungen

P-25 Key Pousttchi, Klaus Turowski (Hrsg.): Mobile Commerce – Anwendungen und Perspektiven – 3. Workshop Mobile Commerce, Universität Augsburg, 04.02.2003

P-26 Gerhard Weikum, Harald Schöning, Erhard Rahm (Hrsg.): BTW 2003: Datenbanksysteme für Business, Technologie und Web

P-27 Michael Kroll, Hans-Gerd Lipinski, Kay Melzer (Hrsg.): Mobiles Computing in der Medizin

P-28 Ulrich Reimer, Andreas Abecker, Steffen Staab, Gerd Stumme (Hrsg.): WM 2003: Professionelles Wissensmanagement – Erfahrungen und Visionen

P-29 Antje Düsterhöft, Bernhard Thalheim (Eds.): NLDB'2003: Natural Language Processing and Information Systems

P-30 Mikhail Godlevsky, Stephen Liddle, Heinrich C. Mayr (Eds.): Information Systems Technology and its Applications

P-31 Arslan Brömme, Christoph Busch (Eds.): BIOSIG 2003: Biometric and Electronic Signatures

P-32 Peter Hubwieser (Hrsg.): Informatische Fachkonzepte im Unterricht – INFOS 2003

P-33 Andreas Geyer-Schulz, Alfred Taudes (Hrsg.): Informationswirtschaft: Ein Sektor mit Zukunft

P-34 Klaus Dittrich, Wolfgang König, Andreas Oberweis, Kai Rannenberg, Wolfgang Wahlster (Hrsg.): Informatik 2003 – Innovative Informatikanwendungen (Band 1)

P-35 Klaus Dittrich, Wolfgang König, Andreas Oberweis, Kai Rannenberg, Wolfgang Wahlster (Hrsg.): Informatik 2003 – Innovative Informatikanwendungen (Band 2)

P-36 Rüdiger Grimm, Hubert B. Keller, Kai Rannenberg (Hrsg.): Informatik 2003 – Mit Sicherheit Informatik

P-37 Arndt Bode, Jörg Desel, Sabine Rathmayer, Martin Wessner (Hrsg.): DeLFI 2003: e-Learning Fachtagung Informatik

P-38 E.J. Sinz, M. Plaha, P. Neckel (Hrsg.): Modellierung betrieblicher Informationssysteme – MobIS 2003

P-39 Jens Nedon, Sandra Frings, Oliver Göbel (Hrsg.): IT-Incident Management & IT-Forensics – IMF 2003

P-40 Michael Rebstock (Hrsg.): Modellierung betrieblicher Informationssysteme – MobIS 2004

P-41 Uwe Brinkschulte, Jürgen Becker, Dietmar Fey, Karl-Erwin Großpietsch, Christian Hochberger, Erik Maehle, Thomas Runkler (Edts.): ARCS 2004 – Organic and Pervasive Computing

P-42 Key Pousttchi, Klaus Turowski (Hrsg.): Mobile Economy – Transaktionen und Prozesse, Anwendungen und Dienste

P-43 Birgitta König-Ries, Michael Klein, Philipp Obreiter (Hrsg.): Persistance, Scalability, Transactions – Database Mechanisms for Mobile Applications

P-44 Jan von Knop, Wilhelm Haverkamp, Eike Jessen (Hrsg.): Security, E-Learning. E-Services

P-45 Bernhard Rumpe, Wofgang Hesse (Hrsg.): Modellierung 2004

P-46 Ulrich Flegel, Michael Meier (Hrsg.): Detection of Intrusions of Malware & Vulnerability Assessment

P-47 Alexander Prosser, Robert Krimmer (Hrsg.): Electronic Voting in Europe – Technology, Law, Politics and Society

P-48 Anatoly Doroshenko, Terry Halpin, Stephen W. Liddle, Heinrich C. Mayr (Hrsg.): Information Systems Technology and its Applications

P-49 G. Schiefer, P. Wagner, M. Morgenstern, U. Rickert (Hrsg.): Integration und Datensicherheit – Anforderungen, Konflikte und Perspektiven

P-50 Peter Dadam, Manfred Reichert (Hrsg.): INFORMATIK 2004 – Informatik verbindet (Band 1) Beiträge der 34. Jahrestagung der Gesellschaft für Informatik e.V. (GI), 20.-24. September 2004 in Ulm

P-51 Peter Dadam, Manfred Reichert (Hrsg.): INFORMATIK 2004 – Informatik verbindet (Band 2) Beiträge der 34. Jahrestagung der Gesellschaft für Informatik e.V. (GI), 20.-24. September 2004 in Ulm

P-52 Gregor Engels, Silke Seehusen (Hrsg.): DELFI 2004 – Tagungsband der 2. e-Learning Fachtagung Informatik

P-53 Robert Giegerich, Jens Stoye (Hrsg.): German Conference on Bioinformatics – GCB 2004

P-54 Jens Borchers, Ralf Kneuper (Hrsg.): Softwaremanagement 2004 – Outsourcing und Integration

P-55 Jan von Knop, Wilhelm Haverkamp, Eike Jessen (Hrsg.): E-Science und Grid Adhoc-Netze Medienintegration

P-56 Fernand Feltz, Andreas Oberweis, Benoit Otjacques (Hrsg.): EMISA 2004 – Informationssysteme im E-Business und E-Government

P-57 Klaus Turowski (Hrsg.): Architekturen, Komponenten, Anwendungen

P-58 Sami Beydeda, Volker Gruhn, Johannes Mayer, Ralf Reussner, Franz Schweiggert (Hrsg.): Testing of Component-Based Systems and Software Quality

P-59 J. Felix Hampe, Franz Lehner, Key Pousttchi, Kai Ranneberg, Klaus Turowski (Hrsg.): Mobile Business – Processes, Platforms, Payments

P-60 Steffen Friedrich (Hrsg.): Unterrichtskonzepte für inforrmatische Bildung

P-61 Paul Müller, Reinhard Gotzhein, Jens B. Schmitt (Hrsg.): Kommunikation in verteilten Systemen

P-62 Federrath, Hannes (Hrsg.): „Sicherheit 2005“ – Sicherheit – Schutz und Zuverlässigkeit

P-63 Roland Kaschek, Heinrich C. Mayr, Stephen Liddle (Hrsg.): Information Systems – Technology and ist Applications

P-64 Peter Liggesmeyer, Klaus Pohl, Michael Goedicke (Hrsg.): Software Engineering 2005

P-65 Gottfried Vossen, Frank Leymann, Peter Lockemann, Wolffried Stucky (Hrsg.): Datenbanksysteme in Business, Technologie und Web

P-66 Jörg M. Haake, Ulrike Lucke, Djamshid Tavangarian (Hrsg.): DeLFI 2005: 3. deutsche e-Learning Fachtagung Informatik

P-67 Armin B. Cremers, Rainer Manthey, Peter Martini, Volker Steinhage (Hrsg.): INFORMATIK 2005 – Informatik LIVE (Band 1)

P-68 Armin B. Cremers, Rainer Manthey, Peter Martini, Volker Steinhage (Hrsg.): INFORMATIK 2005 – Informatik LIVE (Band 2)

P-69 Robert Hirschfeld, Ryszard Kowalcyk, Andreas Polze, Matthias Weske (Hrsg.): NODe 2005, GSEM 2005

P-70 Klaus Turowski, Johannes-Maria Zaha (Hrsg.): Component-oriented Enterprise Application (COAE 2005)

P-71 Andrew Torda, Stefan Kurz, Matthias Rarey (Hrsg.): German Conference on Bioinformatics 2005

P-72 Klaus P. Jantke, Klaus-Peter Fähnrich, Wolfgang S. Wittig (Hrsg.): Marktplatz Internet: Von e-Learning bis e-Payment

P-73 Jan von Knop, Wilhelm Haverkamp, Eike Jessen (Hrsg.): “Heute schon das Morgen sehen“

P-74 Christopher Wolf, Stefan Lucks, Po-Wah Yau (Hrsg.): WEWoRC 2005 – Western European Workshop on Research in Cryptology

P-75 Jörg Desel, Ulrich Frank (Hrsg.): Enterprise Modelling and Information Systems Architecture

P-76 Thomas Kirste, Birgitta König-Riess, Key Pousttchi, Klaus Turowski (Hrsg.): Mobile Informationssysteme – Potentiale, Hindernisse, Einsatz

P-77 Jana Dittmann (Hrsg.): SICHERHEIT 2006

P-78 K.-O. Wenkel, P. Wagner, M. Morgenstern, K. Luzi, P. Eisermann (Hrsg.): Land- und Ernährungswirtschaft im Wandel

P-79 Bettina Biel, Matthias Book, Volker Gruhn (Hrsg.): Softwareengineering 2006

P-80 Mareike Schoop, Christian Huemer, Michael Rebstock, Martin Bichler (Hrsg.): Service-Oriented Electronic Commerce

P-81 Wolfgang Karl, Jürgen Becker, Karl-Erwin Großpietsch, Christian Hochberger, Erik Maehle (Hrsg.): ARCS´06

P-82 Heinrich C. Mayr, Ruth Breu (Hrsg.): Modellierung 2006

P-83 Daniel Huson, Oliver Kohlbacher, Andrei Lupas, Kay Nieselt and Andreas Zell (eds.): German Conference on Bioinformatics

P-84 Dimitris Karagiannis, Heinrich C. Mayr, (Hrsg.): Information Systems Technology and its Applications

P-85 Witold Abramowicz, Heinrich C. Mayr, (Hrsg.): Business Information Systems

P-86 Robert Krimmer (Ed.): Electronic Voting 2006

P-87 Max Mühlhäuser, Guido Rößling, Ralf Steinmetz (Hrsg.): DELFI 2006: 4. e-Learning Fachtagung Informatik

P-88 Robert Hirschfeld, Andreas Polze, Ryszard Kowalczyk (Hrsg.): NODe 2006, GSEM 2006

P-90 Joachim Schelp, Robert Winter, Ulrich Frank, Bodo Rieger, Klaus Turowski (Hrsg.): Integration, Informationslogistik und Architektur

P-91 Henrik Stormer, Andreas Meier, Michael Schumacher (Eds.): European Conference on eHealth 2006

P-92 Fernand Feltz, Benoît Otjacques, Andreas Oberweis, Nicolas Poussing (Eds.): AIM 2006

P-93 Christian Hochberger, Rüdiger Liskowsky (Eds.): INFORMATIK 2006 – Informatik für Menschen, Band 1

P-94 Christian Hochberger, Rüdiger Liskowsky (Eds.): INFORMATIK 2006 – Informatik für Menschen, Band 2

P-95 Matthias Weske, Markus Nüttgens (Eds.): EMISA 2005: Methoden, Konzepte und Technologien für die Entwicklung von dienstbasierten Informationssystemen

P-96 Saartje Brockmans, Jürgen Jung, York Sure (Eds.): Meta-Modelling and Ontologies

P-97 Oliver Göbel, Dirk Schadt, Sandra Frings, Hardo Hase, Detlef Günther, Jens Nedon (Eds.): IT-Incident Mangament & IT-Forensics – IMF 2006

P-98 Hans Brandt-Pook, Werner Simonsmeier und Thorsten Spitta (Hrsg.): Beratung in der Softwareentwicklung – Modelle, Methoden, Best Practices

P-99 Andreas Schwill, Carsten Schulte, Marco Thomas (Hrsg.): Didaktik der Informatik

P-100 Peter Forbrig, Günter Siegel, Markus Schneider (Hrsg.): HDI 2006: Hochschuldidaktik der Informatik

P-101 Stefan Böttinger, Ludwig Theuvsen, Susanne Rank, Marlies Morgenstern (Hrsg.): Agrarinformatik im Spannungsfeld zwischen Regionalisierung und globalen Wertschöpfungsketten

P-102 Otto Spaniol (Eds.): Mobile Services and Personalized Environments

P-103 Alfons Kemper, Harald Schöning, Thomas Rose, Matthias Jarke, Thomas Seidl, Christoph Quix, Christoph Brochhaus (Hrsg.): Datenbanksysteme in Business, Technologie und Web (BTW 2007)

P-104 Birgitta König-Ries, Franz Lehner, Rainer Malaka, Can Türker (Hrsg.) MMS 2007: Mobilität und mobile Informationssysteme

P-105 Wolf-Gideon Bleek, Jörg Raasch, Heinz Züllighoven (Hrsg.) Software Engineering 2007

P-106 Wolf-Gideon Bleek, Henning Schwentner, Heinz Züllighoven (Hrsg.) Software Engineering 2007 – Beiträge zu den Workshops

P-107 Heinrich C. Mayr, Dimitris Karagiannis (eds.) Information Systems Technology and its Applications

P-108 Arslan Brömme, Christoph Busch, Detlef Hühnlein (eds.) BIOSIG 2007: Biometrics and Electronic Signatures

P-109 Rainer Koschke, Otthein Herzog, Karl-Heinz Rödiger, Marc Ronthaler (Hrsg.) INFORMATIK 2007 Informatik trifft Logistik Band 1

P-110 Rainer Koschke, Otthein Herzog, Karl-Heinz Rödiger, Marc Ronthaler (Hrsg.) INFORMATIK 2007 Informatik trifft Logistik Band 2

P-111 Christian Eibl, Johannes Magenheim, Sigrid Schubert, Martin Wessner (Hrsg.) DeLFI 2007: 5. e-Learning Fachtagung Informatik

P-112 Sigrid Schubert (Hrsg.) Didaktik der Informatik in Theorie und Praxis

P-113 Sören Auer, Christian Bizer, Claudia Müller, Anna V. Zhdanova (Eds.) The Social Semantic Web 2007 Proceedings of the 1st Conference on Social Semantic Web (CSSW)

P-114 Sandra Frings, Oliver Göbel, Detlef Günther, Hardo G. Hase, Jens Nedon, Dirk Schadt, Arslan Brömme (Eds.) IMF2007 IT-incident management & IT-forensics Proceedings of the 3rd International Conference on IT-Incident Management & IT-Forensics

P-115 Claudia Falter, Alexander Schliep, Joachim Selbig, Martin Vingron and Dirk Walther (Eds.) German conference on bioinformatics GCB 2007

P-116 Witold Abramowicz, Leszek Maciszek (Eds.) Business Process and Services Computing 1st International Working Conference on Business Process and Services Computing BPSC 2007

P-117 Ryszard Kowalczyk (Ed.) Grid service engineering and manegement The 4th International Conference on Grid Service Engineering and Management GSEM 2007

P-118 Andreas Hein, Wilfried Thoben, Hans-Jürgen Appelrath, Peter Jensch (Eds.) European Conference on ehealth 2007

P-119 Manfred Reichert, Stefan Strecker, Klaus Turowski (Eds.) Enterprise Modelling and Information Systems Architectures Concepts and Applications

P-120 Adam Pawlak, Kurt Sandkuhl, Wojciech Cholewa, Leandro Soares Indrusiak (Eds.) Coordination of Collaborative Engineering - State of the Art and Future Challenges

P-121 Korbinian Herrmann, Bernd Bruegge (Hrsg.) Software Engineering 2008 Fachtagung des GI-Fachbereichs Softwaretechnik

P-122 Walid Maalej, Bernd Bruegge (Hrsg.) Software Engineering 2008 - Workshopband Fachtagung des GI-Fachbereichs Softwaretechnik

P-123 Michael H. Breitner, Martin Breunig, Elgar Fleisch, Ley Pousttchi, Klaus Turowski (Hrsg.)
Mobile und Ubiquitäre Informationssysteme – Technologien, Prozesse, Marktfähigkeit
Proceedings zur 3. Konferenz Mobile und Ubiquitäre Informationssysteme (MMS 2008)

P-124 Wolfgang E. Nagel, Rolf Hoffmann, Andreas Koch (Eds.)
9th Workshop on Parallel Systems and Algorithms (PASA)
Workshop of the GI/ITG Speciel Interest Groups PARS and PARVA

P-125 Rolf A.E. Müller, Hans-H. Sundermeier, Ludwig Theuvsen, Stephanie Schütze, Marlies Morgenstern (Hrsg.)
Unternehmens-IT: Führungsinstrument oder Verwaltungsbürde
Referate der 28. GIL Jahrestagung

P-126 Rainer Gimnich, Uwe Kaiser, Jochen Quante, Andreas Winter (Hrsg.)
10th Workshop Software Reengineering (WSR 2008)

P-127 Thomas Kühne, Wolfgang Reisig, Friedrich Steimann (Hrsg.)
Modellierung 2008

P-128 Ammar Alkassar, Jörg Siekmann (Hrsg.)
Sicherheit 2008
Sicherheit, Schutz und Zuverlässigkeit
Beiträge der 4. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik e.V. (GI)
2.-4. April 2008
Saarbrücken, Germany

P-129 Wolfgang Hesse, Andreas Oberweis (Eds.)
Sigsand-Europe 2008
Proceedings of the Third AIS SIGSAND European Symposium on Analysis, Design, Use and Societal Impact of Information Systems

P-130 Paul Müller, Bernhard Neumair, Gabi Dreo Rodosek (Hrsg.)
1. DFN-Forum Kommunikations-technologien Beiträge der Fachtagung

P-131 Robert Krimmer, Rüdiger Grimm (Eds.)
3rd International Conference on Electronic Voting 2008
Co-organized by Council of Europe, Gesellschaft für Informatik and E-Voting.CC

P-132 Silke Seehusen, Ulrike Lucke, Stefan Fischer (Hrsg.)
DeLFI 2008:
Die 6. e-Learning Fachtagung Informatik

P-133 Heinz-Gerd Hegering, Axel Lehmann, Hans Jürgen Ohlbach, Christian Scheideler (Hrsg.)
INFORMATIK 2008
Beherrschbare Systeme – dank Informatik
Band 1

P-134 Heinz-Gerd Hegering, Axel Lehmann, Hans Jürgen Ohlbach, Christian Scheideler (Hrsg.)
INFORMATIK 2008
Beherrschbare Systeme – dank Informatik
Band 2

P-135 Torsten Brinda, Michael Fothe, Peter Hubwieser, Kirsten Schlüter (Hrsg.)
Didaktik der Informatik – Aktuelle Forschungsergebnisse

P-136 Andreas Beyer, Michael Schroeder (Eds.)
German Conference on Bioinformatics GCB 2008

P-137 Arslan Brömme, Christoph Busch, Detlef Hühnlein (Eds.)
BIOSIG 2008: Biometrics and Electronic Signatures

P-138 Barbara Dinter, Robert Winter, Peter Chamoni, Norbert Gronau, Klaus Turowski (Hrsg.)
Synergien durch Integration und Informationslogistik
Proceedings zur DW2008

P-139 Georg Herzwurm, Martin Mikusz (Hrsg.)
Industrialisierung des Software-Managements
Fachtagung des GI-Fachausschusses Management der Anwendungsentwick-lung und -wartung im Fachbereich Wirtschaftsinformatik

P-140 Oliver Göbel, Sandra Frings, Detlef Günther, Jens Nedon, Dirk Schadt (Eds.)
IMF 2008 - IT Incident Management & IT Forensics

P-141 Peter Loos, Markus Nüttgens, Klaus Turowski, Dirk Werth (Hrsg.)
Modellierung betrieblicher Informations-systeme (MobIS 2008)
Modellierung zwischen SOA und Compliance Management

P-142 R. Bill, P. Korduan, L. Theuvsen, M. Morgenstern (Hrsg.)
Anforderungen an die Agrarinformatik durch Globalisierung und Klimaveränderung

P-143 Peter Liggesmeyer, Gregor Engels, Jürgen Münch, Jörg Dörr, Norman Riegel (Hrsg.)
Software Engineering 2009
Fachtagung des GI-Fachbereichs Softwaretechnik

P-144 Johann-Christoph Freytag, Thomas Ruf, Wolfgang Lehner, Gottfried Vossen (Hrsg.)
Datenbanksysteme in Business, Technologie und Web (BTW)

P-145 Knut Hinkelmann, Holger Wache (Eds.)
WM2009: 5th Conference on Professional Knowledge Management

P-146 Markus Bick, Martin Breunig, Hagen Höpfner (Hrsg.)
Mobile und Ubiquitäre Informationssysteme – Entwicklung, Implementierung und Anwendung
4. Konferenz Mobile und Ubiquitäre Informationssysteme (MMS 2009)

P-147 Witold Abramowicz, Leszek Maciaszek, Ryszard Kowalczyk, Andreas Speck (Eds.)
Business Process, Services Computing and Intelligent Service Management
BPSC 2009 · ISM 2009 · YRW-MBP 2009

The titles can be purchased at:

Köllen Druck + Verlag GmbH
Ernst-Robert-Curtius-Str. 14 · D-53117 Bonn
Fax: +49 (0)228/9898222
E-Mail: druckverlag@koellen.de