Arthur ter Hofstede
Boualem Benatallah
Hye-Young Paik (Eds.)

LNCS 4928

Business Process Management Workshops

BPM 2007 International Workshops
BPI, BPD, CBP, ProHealth, RefMod, semantics4ws
Brisbane, Australia, September 2007, Revised Selected Papers

Springer

Arthur ter Hofstede   Boualem Benatallah
Hye-Young Paik (Eds.)

Business Process Management Workshops

BPM 2007 International Workshops
BPI, BPD, CBP, ProHealth, RefMod, semantics4ws
Brisbane, Australia, September 24, 2007
Revised Selected Papers

Springer

Volume Editors

Arthur ter Hofstede
Business Process Management Group
Queensland University of Technology
Brisbane, Australia
E-mail: a.terhofstede@qut.edu.au

Boualem Benatallah
University of New South Wales
Sydney, Australia
E-mail: boualem@cse.unsw.edu.au

Hye-Young Paik
University of New South Wales
Sydney, Australia
E-mail: hpaik@cse.unsw.edu.au

Library of Congress Control Number: 2008921481

CR Subject Classification (1998): H.3.5, H.4.1, H.5.3, K.4.3, K.4.4, K.6, J.1, J.3

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

ISSN	0302-9743
ISBN-10	3-540-78237-0 Springer Berlin Heidelberg New York
ISBN-13	978-3-540-78237-7 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12230310 06/3180 5 4 3 2 1 0

Preface

These proceedings contain the final versions of papers accepted for the workshops that were held in conjunction with the Fifth International Conference on Business Process Management (BPM 2007) that took place in Brisbane, Australia. Twenty workshop proposals were submitted for this conference of which seven were selected. Ultimately this resulted in six workshops that ran concurrently on September 24 2007. This was the third year running for BPM workshops, a testament to the continued success of the workshop program.

The BPM community's ongoing strong interest in process modelling, design, measurement and analysis were well reflected in the "Business Process Intelligence" and "Business Process Design" workshops. This year's workshops also included two new emerging areas that have gained increased attention: "Collaborative Business Processes"—a topic which explores the challenges in seamless integration of and collaboration between business processes from different organizations, and "Process-Oriented Information Systems in Healthcare"—a topic which recognizes the importance of patient-centered process support in healthcare and looks into the potential benefits and limitations of IT support for healthcare processes. The "Reference Modeling" workshop covered languages for reference modelling, evaluation and adaptation of reference models, and applications of such models. Finally, the "Advances in Semantics for Web Services" workshop considered some of the latest research efforts in the field of Semantic Web services including relevant tools and techniques and real-world applications of such services.

We would like to thank the workshop organizers for their tremendous efforts in the preparation for the workshops, the organization of the reviews, the onsite moderation of the workshops, and the publication process. It would not have been possible to hold such successful workshops without their dedication and commitment.

We extend our thanks also to the authors for their submissions to the workshops, to the Program Committee members and the additional reviewers for their reviews, and last but not least to the invited speakers for contributing to an interesting overall program.

December 2007

Arthur ter Hofstede
Boualem Benatallah

Organization

Workshop Organization Committee

Arthur ter Hofstede, Workshop Co-chair
Queensland University of Technology, Australia

Boualem Benatallah, Workshop Co-chair
University of New South Wales, Australia

Hye-Young Paik, Publication Chair
University of New South Wales, Australia

Business Process Intelligence (BPI)

Malu Castellanos
Hewlett-Packard Laboratories, USA

Jan Mendling
Vienna University of Economics and Business Admin., Austria

Barbara Weber
University of Innsbruck, Austria

Ton Weijters
Technische Universiteit, Eindhoven, The Netherlands

Business Process Design (BPD)

Tom Davenport
Babson College, USA

Selma Limam Mansar
Zayed University, UAE

Hajo Reijers
Eindhoven University of Technology, The Netherlands

Collaborative Business Processes (CBP)

Chengfei Liu
Swinburne University of Technology, Australia

Qing Li
City University of Hong Kong, China

Yanchun Zhang
Victoria University, Australia

Marta Indulska
University of Queensland, Australia

Xiaohui Zhao
Swinburne University of Technology, Australia

Process-Oriented Systems in Healthcare (ProHealth)

Manfred Reichert
University of Twente, The Netherlands

Richard Lenz
University of Marburg, Germany

Mor Peleg
University of Haifa, Israel

Reference Modeling

Jörg Becker
European Research Center for Information Systems, Germany

Patrick Delfmann
European Research Center for Information Systems, Germany

Advances in Semantics for Web Services (semantics4ws)

Steven Battle
Hewlett-Packard Labs, UK

John Domingue
The Open University, UK

David Martin
Artificial Intelligence Center, SRI International, USA

Dumitru Roman
University of Innsbruck, Austria

Amit Sheth
Wright State University, USA

Table of Contents

BPD Workshop

BPI Workshop

Introduction to the Third Workshop on Business Process Intelligence (BPI 2007)

Business process intelligence (BPI) is quickly gaining interest and importance in research industry. BPI refers to the application of various measurement and analysis techniques in the area of business process management to provide a better understanding and a more appropriate support of a company's processes at design time and the way they are handled at runtime. The Call for Papers for this workshop attracted 16 international submissions. Each paper was reviewed by at least three members of the Program Committee and the eight best papers were selected for presentation at the workshop. In addition, the workshop included of a keynote and a roundtable. In his keynote talk "DataMining: Practical Challenges in Analyzing Performance" M. Genrich addressed challenges which arise when applying process performance analysis in practices. Genrich pointed out that events logs are often not sufficient for process analysis, and that the business context has to be considered carefully before drawing conclusions from the data.

The papers presented at the workshop provided a mix of novel research ideas, practical applications of BPI as well as new tool support. The paper by M.L. Pérez and C. Møller presents practical experiences of using BPI for churn prediction in one of Denmark's largest trade unions. The work by J.E. Ingvaldsen and J.A. Gulla contributes to process mining in SAP systems. The paper by M.T. Wynn et al. targets short-term predictions through workflow simulations taking the current state of process execution into account. In addition, A. Rozinat et al. propose a framework for evaluating process mining algorithms and for comparing their quality along several quality metrics. In their paper, A.K. Alves de Medeiros et al. suggest the application of clustering techniques to increase the precision of the mined models. A novel technique for mining process models based on first-order logic is presented by S. Goedertier et al. inspired by machine learning. Exception handling is addressed by M. Golani et al., who propose process models to be enriched for semi-automatic generation of exception handlers. Finally, the work by J.A. Rodrigues et al. presents some initial ideas towards autonomic business processes which can be characterized as being self-configuring, self-healing, self-optimizing and self-protecting.

The roundtable on "What Business Process Intelligence Should Provide to Business Process Management," in which A. Kokkonen, J. Moormann, R. Tregear, and M. zur Muehlen participated, showed that business process intelligence can deliver substantial benefits. However, its application in practice raises several challenges. The summary of the BPI workshop discussions is included in these workshop proceedings.

September 2007

Malu Castellanos
Jan Mendling
Barbara Weber
Ton Weijters

Workshop Organization

Malu Castellanos
Intelligent Enterprise Technologies Lab
Hewlett-Packard Laboratories
1501 Page Mill Rd, CA 94304
USA

Jan Mendling
BPM Cluster, Faculty of IT
Queensland University of Technology
126 Margaret Street, Brisbane Qld 4000
Australia

Barbara Weber
Institut für Informatik
Universität Innsbruck
Technikerstraße 21a, 6020 Innsbruck
Austria

Ton Weijters
Department of Technology Management
Technische Universiteit Eindhoven
Paviljoen, Postbus 513, 5600 MB Eindhoven
The Netherlands

Program Committee

Wil van der Aalst, Technical University of Eindhoven, The Netherlands
Boualem Benatallah, University of New South Wales, Australia
Gerardo Canfora, University of Sannio, Italy
Fabio Casati, University of Trento, Italy
Jonathan E. Cook, New Mexico State University, USA
Umesh Dayal, HP Labs, USA
Peter Dadam, University of Ulm, Germany
Marlon Dumas, Queensland University of Technology, Australia
Gianluigi Greco, University of Calabria, Italy
Dimitrios Georgakopoulos, Telcordia Technologies, Austin, USA
Mati Golani, Technion, Israel
Jon Atle Gulla, Norwegian University of Science and Technology, Norway
Joachim Herbst, DaimlerChrysler Research and Technology, Germany
Ramesh Jain, Georgia Tech, USA
Jun-Jang Jeng, IBM Research, USA
Ana Karla de Medeiros, Technical University of Eindhoven, The Netherlands
Sandro Morasca, Università dell'Insubria, Como, Italy
Michael zur Muehlen, Stevens Institute of Technology, USA
Cesare Pautasso, ETH Zurich, Switzerland
Shlomit S. Pinter, IBM Haifa Research Lab, Israel
Manfred Reichert, University of Twente, The Netherlands
Michael Rosemann, Queensland University of Technology, Australia
Domenico Sacca, Università della Calabria, Italy
Pnina Soffer, Haifa University, Israel
Hans Weigand, Infolab, Tilburg University, The Netherlands
Mathias Weske, Hasso Plattner Institute at University of Potsdam, Germany

Challenges for Business Process Intelligence: Discussions at the BPI Workshop 2007

Michael Genrich[1], Alex Kokkonen[2], Jürgen Moormann[3], Michael zur Muehlen[4], Roger Tregear[5], Jan Mendling[6], and Barbara Weber[7]

[1] Fujitsu Consulting Limited
1 Breakfast Creek Road, Newstead QLD 4006, Australia
michael.genrich@au.fujitsu.com
[2] Johnson & Johnson
1-5 Khartoum Road, North Ryde Sydney NSW 2113, Australia
AKOKKONE@JJPAU.JNJ.com
[3] Frankfurt School of Finance & Management
Sonnemannstraße 9-11, 60314 Frankfurt am Main, Germany
j.moormann@frankfurt-school.de
[4] Stevens Institute of Technology
Castle Point on the Hudson, Hoboken, NJ 07030, USA
Michael.zurMuehlen@stevens.edu
[5] Leonardo Consulting
GPO Box 2046, Canberra ACT 2601, Australia
r.tregear@leonardo.com.au
[6] Queensland University of Technology
126 Margaret Street, Brisbane QLD 4000, Australia
j.mendling@qut.edu.au
[7] University of Innsbruck
Technikerstraße 21a, 6020 Innsbruck, Austria
Barbara.Weber@uibk.ac.at

Abstract. This paper summarizes the discussions at the 3rd Workshop on Business Process Intelligence (BPI 07) which was held at the 5th International Conference on Business Process Management (BPM 07) in Brisbane, Australia. We focus in particular on three cases that were referenced in the BPI roundtable and discuss some practical challenges. Finally, we identify future research directions for business process intelligence.

1 Introduction

Business Process Intelligence (BPI) relates to “a set of integrated tools that supports business and IT users in managing process execution quality” [1]. BPI builds on techniques such as data mining and statistical analysis that were developed or inspired by business intelligence techniques such as data mining or statistical analysis, and adapts them to the requirements of business process management. Recent case studies like [2] clearly show that process mining techniques have gained a level of maturity that makes them applicable to real-world business processes, and that they reveal valuable insight into the way how people really work in organizations.

Even though the application of process mining or similar techniques can provide substantial business benefit, few organizations actually use them in practice. The invited talk and the roundtable discussion at the 3rd Workshop on Business Process Intelligence (BPI 07) had the aim of identifying some of the challenges for successfully utilizing BPI techniques in a real-world business setting. This paper provides a summary of these discussions. In particular, Section 2 describes three cases that different workshop participants experienced in their work and research. Finally, Section 3 identifies challenges for BPI projects in practice, and discusses future directions that could advance BPI as a research area.

2 Experiences

This section describes three cases in which organizations used data about past process executions to get better insight into their processes and performance. The cases involve a German bank, a German insurance company, and an Australian utility company.

2.1 DEA Analysis in a German Bank

In the banking industry fierce competition, pressure from regulation authorities, as well as increased customer demands act as a catalysts for current efforts to gain full transparency about process performance. Process performance management in banks is influenced by the complexity of the products and services, multiple inputs and outputs, and missing efficiency standards [3]. Performance in banks is generally understood as a multi-dimensional phenomenon. Despite this understanding, common performance measurement practice has a strong focus on cost, e.g. by analyzing input consumption and cycle times. Simple ratio-based productivity analysis predominates in banking [4]. In contrast to that, we started a research project to analyze single transactions on a multi-input and multi-output basis to discover process performance deficits. The object of analysis is the *Securities Settlement & Clearing Process*. Like most banking operations processes, this process combines automatic processing and selective manual intervention. From a bank's management point of view, the securities process has high significance, due to its high revenue generation potential.

The research project is conducted in co-operation with Commerzbank AG, one of the leading European commercial banks, and it utilizes the bank's operational data. The goal is to provide a better understanding of, and a more appropriate support for, bank business processes and the way they are handled at runtime. We introduce a DEA-based (Data Envelopment Analysis) approach for process diagnosis. DEA is a non parametric, non-stochastic efficiency measurement method [5,6]. The DEA performance evaluation is based on benchmarking *Decison Making Units* against best observed practices. DEA has been applied to banking, but up to now the focus has been on entities such as banks or bank branches [7]. In our project we apply DEA on the business process level in order to reveal patterns of (in-)efficiency based on the transformation of resources (i.e. labor, processing, data) into outcome characteristics (i.e. costs, quality, risk aspects) [8].

While dealing with operational data there were some operational challenges. Firstly, the securities process is supported by various applications each performing specific processing functions (such as data enrichment, confirmation, settlement, clearing, and booking). Secondly, the applications were built with a functional focus and lack process orientation. Thirdly, functional and technical documentation is scarce as applications are managed by long-tenured staff for years. There is no single contact person for all applications along the process available; instead various application managers needed to be contacted individually. Fourthly, each application is using individual references. A unique and overlapping reference is created via meta-referencing that demands mapping of references across applications. Furthermore, it turned out to be very time-consuming to detect the right databases, extract the data from it, and transform it into a ready-to-use format for the research. We had to handle various database formats that are currently in use (DB2, Oracle, Sybase, IMS), and find the right fields containing the relevant data. Finally, there is a vast amount of operational data. Everyday more than 40,000 database entries with over 100 fields each for almost every application along the processing lifecycle are generated.

This research project is in progress and empirical results are expected in the beginning of 2008. A first analysis shows that there is a significant variance in relation to input and output factors across securities transactions across various cases. This indicates that DEA is an appropriate method to measure process performance in banks. Several circumstances work against the application of BPI analysis in this project. We found that there is no clear definition and agreement of business processes across the industry, such as reference processes or a list of industry-wide business processes. Moreover, there is no common understanding of input and output factors for productivity and efficiency analysis. Then, there is limited understanding of the relevant aspects and measures on the business process level. It would be desirable to find an adequate process modeling language that captures all relevant aspects of the process. Finally, an agreement on standards, e.g. on how to count transactions or for benchmarking across companies, would help. Nothing of the above is available in banks today.

2.2 Activity-Based Costing in an German Insurance Company

Obtaining accurate and timely information about operational processes is a core foundation that enables effective decision-making around process structures and resource allocation. Like many others, the case study company, a medium-sized German insurance company, wanted to improve the platform for managerial decision-making by including information about its business processes. The trigger for the BPI project was the realization that the same information request by executives (e.g. "how many car insurance policies were underwritten last quarter?") resulted in different responses, depending on which system was used to obtain the information. The lack of a true source of data had led to a data warehouse project which had already begun to store customer and policy information. But no transactional or process information was part of the warehouse.

The company had an existing platform to provide activity-based costing information. The problem was that the information provided by the existing system was plain wrong. Both data sources, and the way data was aggregated to provide managerial information, were severely flawed. Activity-based costing systems essentially require information about process and activity frequencies, durations, resources, and cost rates. Since no business process management system was in place, the number of process and activity instances were estimated based on the number of transactions that were recorded in the company's mainframe system. But there was no 1:1 mapping between process activities and mainframe transactions, therefore a conversion factor was used that scaled the number of transactions into the number of activities performed. Now that the number of activities was known, the resources used by each activity had to be determined. The cost rate for individual resources was known from the internal accounting system. However, since the mainframe system recorded just one timestamp per transaction, activity processing time could not be determined. To overcome this lack of information, the organization surveyed its employees and asked ”how long does this transaction typically take?” and took the responses as the basis for transaction durations. By multiplying the (assumed) activity duration with the (converted) number of activities the activity-based costing system now determined the resource utilization, which typically was a fraction of the overall work time of the employee. So another conversion factor was used to scale the work time recorded up to match the annual work time of the employees. After these transformations the organization was finally able to determine the cost of performing a single process instance. The decision makers were well aware that the resulting information was based on unsound computations, but since it was the only information available it was still used for decision-making.

As part of a BPI project, the organization was looking to improve the activity-based costing data. In order to do this, several significant changes were required to the technology infrastructure of the organization. As a first step, the existing paper-based process of work distribution had to be replaced by an electronic document management system. In the next step, the business processes were documented in the workflow component of the document management system. Finally, the audit trail information logged in the workflow component could be used as a basis to provide activity-based costing information. This massive change in infrastructure had a substantial positive impact on the overall performance of the organization, but the trigger to go ahead with the project was the desire of senior executives, notably the CIO, to obtain better information for decision-making. Senior management support was essential to maintain the momentum in the project, which turned from a 3 month prototype assessment to a 36 month infrastructure project. The creative use of available information, even though it was known to be inaccurate, illustrates the desire of organizations to improve their decision-making.

2.3 Performance of Outage Management in an Australian Utility Company

Outages in the storm season and their timely repair is one of the major issues in customer satisfaction of Australian utility companies. The study was conducted within a major state government owned corporation, that is responsible for distribution and retailing of electricity. The organization supported approximately 600,000 customers and operates an electricity network of about 100,000 km. The study was focused on the interaction and performance of the call centre, the network operations centre and field crews during storms and other unplanned network outages. The contact centre is the key point of telephone contact for customers with the network operations centres responsible for managing and switching the electricity load within the network. The field crews are mobile teams that perform the actual analysis and repair of distribution power lines. The initial goal of the project was to understand and mitigate the root causes of communication issues between the three groups during unplanned events. There was also a perception that field crews in some geographies were far more efficient at restoring power than others. During an unplanned outage the goal is to restore power safely as quickly as possible whilst keeping customers informed. This is extremely difficult during events such as storms, when communication to the field is difficult, and both the network operations centre and the contact centre experience heavy transaction loads.

Several problems were encountered while trying to achieve the objective. First, there is a high volume of data being generated during outages. Identifying trends in data across multiple outages required significant data analysis. Second, in several systems not all of the data had been properly captured. Data quality deteriorated as the size and significance of outages increased. Data such as customer contact details, qualitative data on outage descriptions and field data relating to qualitative descriptions were typical of this deterioration. Third, the alignment of data between contact centre, network operations centre and field crew feedback was also problematic. Frequently, multiple outage events were experienced over a short time frame with differences in time stamping between systems challenging the ability to isolate and analyze all data related to an outage. Finally, the utility company had been formed from a number of predecessor organizations, inheriting distribution networks that varied significantly by geography. This made comparative analysis of field crew performance problematic.

The initial rounds of analysis confirmed common expectations that as the significance of the outage increased, the internal communication between the three groups became strained. The contact centre provided lower quality feedback from customer calls. The network centre became less responsive to requests for status from the contact centre. The field crews provided less frequent status updates on repairs as they focused on safely restoring power. The study also confirmed significant variation by geography of the total number of "customer lost time minutes" and work effort to repair similar types of outages. This presented a dilemma as these types of measures are well known within the organization. Through a significant number of workshops and feedback sessions, using data mined from field and contact centre systems, it was identified that the measures used for field crew performance were not effective. Using the data analysis methods identified in the first rounds of the study, the analysis was repeated once reporting had been adjusted to how frequently the crews achieved "80% of the customers restored, in 20% of the time". This measure produced a positive gaming focus within a trial group of field crews with some crews improving performance by a factor of 2 to 3 times in "customer lost time minutes". However, the overall work effort to repair increased. Field crews were now being far more effective at isolating outages (e.g. broken pole), routing power around the affected area to restore most households power, repairing the fault, and then removing the temporary routing. Thus, although overall field crew effort increased, far fewer customers experienced extended power loss.

The key learnings for the organization included an understanding that isolating transaction flows and focusing on efficiency or performance may not provide the optimal customer outcome. Effective performance measures may be behavioral in nature and not directly linked to the transaction being analyzed. An understanding of the desired business outcomes is needed to more effectively interpret large volumes of data.

3 Future Research

The discussions at the BPI workshop highlighted several challenges for BPI initiatives in practice. In essence, three major success factors were identified. First, there is a need for a clear strategy and leadership that aligns BPI projects with the overall business objectives. The importance of this aspect is also mentioned in [9]. Second, beyond the availability of powerful tools, it remains critical to understand the business and the factors that can be controlled by management. The appropriateness of behavior-based or outcome-based control [10] depends on the business process. Third, there is a great diversity of technological infrastructure, data availability, and data quality (cf. [11]). A BPI project is more likely to succeed if different data sources can be easily integrated.

The roundtable participants shared the opinion that many companies miss the opportunity to record event data that could be used in BPI analysis. In this regard, workflow technology is not only an enabler for process execution, but also for process evaluation. It is desirable to enhance the state of the art in BPI by analyzing real-world end-to-end business processes. These processes typically span across several different application systems. By addressing this challenge BPI would likely come up with new techniques for data consolidation that could be valuable to increase its uptake in practice.

References

1. Grigori, D., Casati, F., Castellanos, M., Dayal, U., Sayal, M., Shan, M.: Business Process Intelligence. Computers in Industry 53, 321–343 (2004)
2. van der Aalst, W., Reijers, H., Weijters, A., van Dongen, B., Medeiros, A., Song, M., Verbeek, H.: Business process mining: An industrial application. Information Systems 32, 713–732 (2007)
3. Sherman, H., Zhu, J.: Service Productivity Management - Improving Service Performance using Data Envelopment Analysis (DEA) (2006)
4. Rose, P., Hudgins, S.: Bank Management & Financial Services (2004)
5. Epstein, M., Henderson, J.: Data envelopment analysis for managerial control and diagnosis. Decision Sciences 20, 90–119 (1989)
6. Cooper, W., Seiford, L., Zhu, J.: Data Envelopment Analysis: History, Models and Interpretations. In: Cooper, W., Seiford, L., Zhu, J. (eds.) Handbook on Data Envelopment Analysis, pp. 1–39. Kluwer Academic Publishers, Dordrecht (2004)
7. Paradi, J., Vela, S., Yang, Z.: Assessing Bank and Bank Branch Performance: Modeling Considerations and Approaches. In: Cooper, W., Seiford, L., Zhu, J. (eds.) Handbook on Data Envelopment Analysis, pp. 349–400. Kluwer Academic Publishers, Dordrecht (2004)
8. Burger, A.: Process performance analysis with dea - new opportunities for efficiency analysis in banking. In: Proceedings to the 5th International Symposium on DEA and Performance Management, Hyderabad, India, p. 1 (2007)
9. Corea, S., Watters, A.: Challenges in business performance measurement: The case of a corporate IT function. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 16–31. Springer, Heidelberg (2007)
10. Anderson, E., Oliver, R.: Perspectives on behavior-based versus outcome-based salesforce control systems. Journal of Marketing 51, 76–88 (1987)
11. Ingvaldsen, J., Gulla, J.: Preprocessing support for large scale process mining of SAP transactions. In: Proceedings of the 3rd Workshop on Business Process Intelligence (2007)

The Predictive Aspect of Business Process Intelligence: Lessons Learned on Bridging IT and Business

Moisés Lima Pérez[1] and Charles Møller[2]

[1] Stenmoellen 143, 2640, Hedehusene, Denmark
lima_2001@hotmail.com
[2] Aalborg University, Fibigerstraede 16, 9220 Aalborg O, Denmark
charles@production.aau.dk

Abstract. This paper presents the arguments for a research proposal on predicting business events in a Business Process Intelligence (BPI) context. The paper argues that BPI holds a potential for leveraging enterprise benefits by supporting real-time processes. However, based on the experiences from past business intelligence projects the paper argues that it is necessary to establish a new methodology to mine and extract the intelligence on the business level which is different from that, which will improve a business process in an enterprise. In conclusion the paper proposes a new research project aimed at developing the new methodology in an Enterprise Information Systems context.

Keywords: Business Process Intelligence; Data Mining, Enterprise Information System; Customer Relationship Management.

1 Introduction

In order to stay competitive in dynamic environments, companies must continually improve their processes and consequently align their business, people and technologies. Some companies have built their businesses an their ability to collect, analyze and act on data [1]. The ability to accurately predict consumer demand coupled with the capability to rapidly react and readjust to environmental changes and customer demand fluctuations separates the winners from the losers [2].

Agility in a global context is inevitably tied into technology and modern Enterprise Information Systems (EIS) from major vendors such as SAP, Oracle and Microsoft include the concepts and tools needed for creating a flexible infrastructure [3].

This paper suggests that there is a huge potential contribution in using advanced EIS to transform an entire supply chain and create a better alignment between business and IT. The management of business process and thus the concept of Business Process Management (BPM) are central and one of the techniques is process intelligence (BPI).

The importance for BPI to predict events has already been highlighted in previous studies [8]. In this paper we will explore the predictive aspects of BPI. Based on an analysis of a case study we call for a new approach to BPI that addresses the integration of technology and management. We will present and discuss the existing research on BPI in the next chapter in order to identify the gap.

The background for our prediction case is based on a real data mining project, where a trade union institution in Denmark needed to predict churn among its members. This paper proposes a research project aimed at developing a new methodology for BPI. An important argument of this paper is that BPI needs to look beyond system logs in order to effectively find interesting business patterns that can be used to improve a given business.

2 Business Process Intelligence

Business Process Management (BPM) is a mature concept [9] but BPI has yet to be established as a concept. The concept is used by a group of HP researchers to capture a set of tools in the BMPS suite [10, 11], but is there more in the concept?

Casati et al. explicit states: "we assume that it is up to the (business or IT) users to define what quality means to them, and in general which are the characteristics that they want to analyze" [10]. Grigori et al. focus on a set of tools that can help business IT and users manage process execution quality [11].

In their paper they explain the concepts used to process a system's logs, the architecture and semantics used in their data warehouse that stores this information and the analytics and prediction models used in their cockpit [12].

Recently we have also seen the emergence of the Business process mining concept [7]. Business process mining takes information from systems as CRM and ERP and extracts knowledge from them that can then be used to improve a given aspect of a business.

In consistency with previous studies (e.g. [11, 14]), Ingvaldsen & Gulla [15] present also the need to combine data from external sources, such as the department and employee involved in a process with actual process logs to achieve better knowledge discovery results.

List & Machaczek [16] highlight the need to obtain a holistic view of the corporate performance. The case shows the potentials that lie in using traditional methods of data warehousing to process and extract knowledge from process logs.

In general, our work differs from the ones mentioned above in one area: we have gone beyond the use of process logs from a CRM system and used instead demographic data (age, educational background, working sector) as well as traditional transactional data (i.e. the fee paid by trade union members). We highlight as well the need for a very detailed methodology that starts with a business analysis until the discovery of the mining model that answers the business issue in question.

In the next section we will evaluate the experiences from a business process improvement project which took a traditional BI perspective.

3 Case Study: BI in a Danish Trade Union

One of Denmark's largest trade unions has in recent years faced problems with customer loyalty (footnote: due to a confidentiality agreement the name of the customer has been omitted). Their essential problem is that their churn rate (10 %) has been higher than the rate of customer acquisition (2 %). Last year they were interested in learning what data warehousing and data mining could do to explain the reasons for the customer churn.

The trade union in question was already using CRM so they were interested in checking the efficiency of their existing services. A workshop was conducted in order to understand the nature of those that churned. Before choosing any particular algorithm of data mining, it was decided to follow the steps elaborated in the following sections.

3.1 Definition of a Business Issue to Predict: Success Criteria

It was of paramount importance to conduct a business analysis session, where we focussed on understanding the types of services they provided, which segments of the population they worked with, and in their opinion, the reason for the problem they were facing.

We also needed to determine the type of prediction they wanted, and agreed on building a model that identified those customers churned (marked as 0) against those that did not (marked as 1). The success criteria for the prediction model were decided to be at least 60 percent of accuracy for both those that churned and those that did not.

The aim was to improve their CRM services especially over the phone (e.g. improve the legal and educational offers they offered to those that were potential churners).

3.2 Data Analysis and Preparation

The next step was an initial analysis of available data and its preparation. Data analysis was to yield two important results: the quality and accuracy of the data and its relevance to the business aim at hand, namely churn prediction.

Accuracy of the information was of great importance and therefore we needed an experienced person in the trade union that could help us set up the logic that the data was supposed to comply with. These rules were also used to select the data and transform it when needed.

For the exercise we used SQL server 2005 data mining suite. This suite is able to calculate the relevance of the information in relationship to the issue to be predicted. So during this early stage of the data mining exercise we were able to spot those attributes of almost no relevance and exclude them from the exercise.

One major discovery, which is of relevance to BPI, was that none of the information which came from their CRM systems, such as complaints, types of transactions the customers made etc. were of importance to determine the likelihood of whether a customer would churn or not.

We therefore concentrated on data that was stored in their legacy system such the demography of their customers, the number of years they had with the union before they left it, the type of education they had, the fee they paid, the work they performed, etc.

3.3 Selection of Training and Validation Sets

We divided the data into two sets: one set was to be used to train the model and the other one to test the model. It is important that this division is done by some sort of random selection, so that you avoid bias in either the training or the test set.

One of the issues that caused discussion was the percentage of data that should belong to either the churn or the non-churn side in the training set. After several trials, the ideal proportion for the training set was 50/50. This proportion proved to help our models to "discover" the patterns behind each group and effectively predict real life situations.

The test set had to reflect reality so that we were sure to later use it in real life. So we built a test set that contained only 10 percent of churners and 90 percent of loyal members.

3.4 Data Mining Models: Development

Data mining development implies that you need to compare the effectiveness of the models used. SQL Server 2005 data mining suite comes with several algorithms. We found that models based on decision trees were the most effective to predict a customer's likelihood to churn or not.

All our decision trees identified that membership fee was the most relevant factor. This came as quite a surprise to the trade union as they had even worked hard to keep membership fee as low as possible. Customer seniority and work trade came as the next most relevant factors.

Again none of these attributes were used in their CRM processes and therefore we needed to look for churn reasons in their legacy systems instead of their process logs.

3.5 Model Validation and Test

The platform we used, SQL Server 2005, can validate our model against both a so-called perfect model and another one called random model. In data mining language this is called to "assess the model's lift" or its degree of accuracy. A proof of these models accuracy is the fact that it stays close to the perfect model. In our tests we could see that the decision tree model performed very well already with 30 percent of the population when it predicted those customers that churn or not.

The tests were conducted with a completely new set of data. The training set had 90 percent of its records as loyal customers and 10 as churning. The proportion between the two sets reflected the trade union's actual loyal vs. churn rate.

Table 1. Predicted and real churn

Predict	Real	Count
0	0	82791
0	1	2628
1	0	7209
1	1	7372

Predicted churn versus real churn is presented in table 1. The value 1 represents those that were loyal members and the 0 those that churned. The shaded row illustrates where the model had a 92 percent of accuracy for the loyal cases and a 74 percent for the churn-cases. The "Bar of Excellency" decided at the beginning of the workshop was of 65 percent for either case (the total amount of loyal ones were 90000 and 10000 those that churned).

4 Lessons Learned and Discussion

From a Data mining perspective the trade union case is trivial but with the developed model they are able to enhance their customer service processes considerably. In this case study we discovered that the most business relevant information was found in the legacy enterprise systems and not in the process logs.

Most information mining projects fail due to lack of a proper method. One of the key issues in such an exercise is to start with a clear business goal which should be quantifiable. It is also crucial in any data mining methodology to find relevant and cleansed information from which to develop a model.

This implies that BPI should be considered on two levels: 1) on the system or the BPMS level where most of the present research has been focusing; and 2) on the business level where the contextual information and business issues direct the prediction effort. These two approaches are quite different but we are suggesting that they can supplement each other.

Consequently we advocate that BPI research should enhance its perspective from process logs towards a "holistic" approach where process-derived data is merged with general enterprise system information. Pre-processing this enterprise information through a BI strategy will give a better picture of what elements a BPI model should include and substantially reduce the time needed in the processing efforts to identify the best predictive model for business impact.

5 Conclusion

In this article we have argued that BPI is important to a modern global enterprise and we have emphasized prediction as a key characteristic of the business value of BPI. Through the case study we have argued that existing BPI techniques are missing an important business link and that this link can be extracted from existing enterprise systems. Finally we concluded that research needs to extend its perspective towards these business issues.

This leads to the formulation of a new research project on Business Process Intelligence in the context of a global business [17]. One of the research challenges in this project is to transfer the methods and techniques from the systems level BPI towards business level BPI. The long term vision is to automate business improvement activities using BPI.

Initial studies suggest that a global business with a large and complex enterprise systems setup obtains a substantial benefit from this approach.

References

1. Davenport, T.H.: Competing on analytics. Harvard Business Review 84, 98–107 (2006)
2. Rai, A., Ruppel, C., Lewis, M.: Sense and Respond. University Thought Leadership forum. Georgia State University, Rollins College, Atlanta, Georgia, p. 17 (2005)
3. Møller, C.: ERP II: A conceptual framework for next-generation enterprise systems? Journal of Enterprise Information Management 18, 483–497 (2005)
4. Grigori, D., Lorraine, L.I., Casati, F., Dayal, U.: Improving Business Process Quality through Exception Understanding, Prediction, and Prevention. In: Proceedings of VLDB 2001, Rome, Italy (2001)
5. Møller, C., Tan, R., Maack, C.J.: What is Business Process Management? A two stage literature review of an emerging field. In: CONFENIS 2007. The IFIP International Conference on Research and Practical Issues of Enterprise Information Systems, Beijing, China (2007)
6. Casati, F., Dayal, U., Sayal, M., Shan, M.-c.: Business Process Intelligence (2002)
7. Grigori, D., Casati, F., Castellanos, M., Dayal, U., et al.: Business Process Intelligence. Computers in Industry 53, 321–343 (2004)
8. Sayal, M., Casati, F., Dayal, U., Shan, M.-C.: Business Process Cockpit. In: Bressan, S., Chaudhri, A.B., Lee, M.L., Yu, J.X., Lacroix, Z. (eds.) VLDB 2002. LNCS, vol. 2590, Springer, Heidelberg (2003)
9. Muehlen, M.z.: Process-driven Management Information Systems — Combining Data Warehouses and Workflow Technology. In: ICECR 2004. Proceedings of the International Conference on Electronic Commerce Research, California, Dallas, TX, pp. 550–566. IEEE Computer Society Press, Los Alamitos (2001)
10. Ingvaldsen, J.E., Gulla, J.A.: Model-Based Business Process Mining. Information Systems Management 23, 19–31 (2006)
11. List, B., Machaczek, K.: Towards a Corporate Performance Measurement System. In: Handschuh, H., Hasan, M.A. (eds.) SAC 2004. LNCS, vol. 3357, Springer, Heidelberg (2004)
12. Møller, C.: The Conceptual Framework for Business Process Innovation: Towards a Research Program for Global Supply Chain Intelligence. The Icfai Journal of Supply Chain Management (2007) (Forthcoming)

Process Mining Based on Clustering: A Quest for Precision

Ana Karla Alves de Medeiros[1], Antonella Guzzo[2], Gianluigi Greco[2], Wil M.P. van der Aalst[1], A.J.M.M. Weijters[1], Boudewijn F. van Dongen[1], and Domenico Saccà[2]

[1] Eindhoven University of Technology, P.O. Box 513, 5600MB, Eindhoven, The Netherlands
{a.k.medeiros, w.m.p.v.d.aalst, a.j.m.m.weijters, b.f.v.dongen}@tue.nl
[2] University of Calabria, Via Bucci 41C, 187036 Rende (CS), Italy
guzzo@icar.cnr.it, ggreco@mat.unical.it, sacca@unical.it

Abstract. Process mining techniques attempt to extract non-trivial and useful information from event logs recorded by information systems. For example, there are many process mining techniques to automatically discover a process model based on some event log. Most of these algorithms perform well on structured processes with little disturbances. However, in reality it is difficult to determine the scope of a process and typically there are all kinds of disturbances. As a result, process mining techniques produce spaghetti-like models that are difficult to read and that attempt to merge unrelated cases. To address these problems, we use an approach where the event log is clustered iteratively such that each of the resulting clusters corresponds to a coherent set of cases that can be adequately represented by a process model. The approach allows for different clustering and process discovery algorithms. In this paper, we provide a particular clustering algorithm that avoids over-generalization and a process discovery algorithm that is much more robust than the algorithms described in literature [1]. The whole approach has been implemented in ProM.

Keywords: Process Discovery, Process Mining, Workflow Mining, Disjunctive Workflow Schema, ProM Framework.

1 Introduction

The basic idea of [illegible] is to discover, monitor and improve [illegible] processes (i.e., not assumed processes) by extracting knowledge from event logs [1]. Today many of the tasks occurring in processes are either supported or monitored by information systems (e.g., ERP, WFM, CRM, SCM, and PDM systems). However, process mining is not limited to information systems and can also be used to monitor other operational processes or systems (e.g., web services, care flows in hospitals, and complex devices like wafer scanners, complex X-ray machines, high-end copiers, etc.). All of these applications have in

Table 1. Example of an event log (with 300 process instances) for the process of a one-day conference. Each row refers to process instances that follow a similar pattern in terms of tasks being executed. The first row corresponds to 80 process instances that all followed the event sequence indicated.

Identifier	Process instance	Frequency
1	Start, Get Ready, Travel by Car, Conference Starts, Give a Talk, Join Guided Tour, Join Dinner, Go Home, Pay for Parking, Travel by Car, End.	80
2	Start, Get Ready, Travel by Train, Conference Starts, Give a Talk, Join Guided Tour, Join Dinner, Go Home, Travel by Train, End.	68
3	Start, Get Ready, Travel by Car, Conference Starts, Join Guided Tour, Join Dinner, Go Home, Pay for Parking, Travel by Car, End.	81
4	Start, Get Ready, Travel by Train, Conference Starts, Join Guided Tour, Join Dinner, Go Home, Travel by Train, End.	71

common that [illegible] and that [illegible]. Assuming that we are able to log events, a wide range of [illegible] comes into reach. The basic idea of process mining is to learn from observed executions of a process and it can be used to (1) [illegible] new models (e.g., constructing a Petri net that is able to reproduce the observed behavior), (2) check the [illegible] of a model by checking whether the modeled behavior matches the observed behavior, and (3) [illegible] an existing model by projecting information extracted from the logs onto some initial model (e.g., show bottlenecks in a process model by analyzing the event log).

In this paper, we focus on process discovery. Concretely, we want to construct a process model (e.g., a Petri net) based on an event log where for each case (i.e., an instance of the process) a sequence of events (say tasks) is recorded. Table 1 shows an aggregate view of such a log. This log will be used as a running example and describes the events taking place when people attend a conference. It is a toy example, but it is particularly useful when explaining our approach. Note that in the context of ProM we have analyzed many real-life logs (cf. www.processmining.org). However, the corresponding processes are too difficult to describe when explaining a specific process mining technique. Therefore, we resort to using this simple artificial process as a running example. Each process instance corresponds to a sequence of events (i.e., task executions). Process instances having the same sequence are grouped into one row in Table 1. In total there are 300 process instances distributed over 4 possible sequences. Note that all process instances start with task "Start" and end with task "End". The process instances referred to by the first two rows contain task "Give a Talk" while this event is missing in process instances referred to by the last two rows. This suggests that some people give a talk while others do not. Rows 1 and 3 refer to two occurrences of task "Travel by Car" and the other two rows (2 and 4)

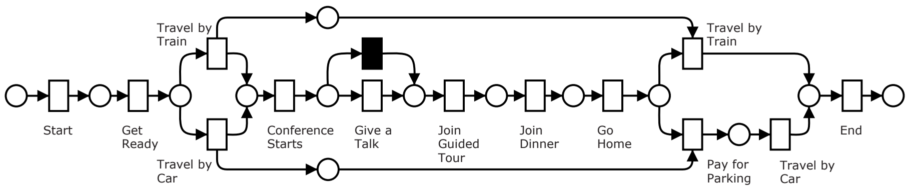


Fig. 1. Example of a mined model for the log in Table 1. This model correctly captures the right level of abstraction for the behavior in the log.

refer to two occurrences of task "Travel by Train". This indicates that the people that arrive by car (train) also return by car (train). People that come by car also execute task "Pay for Parking". Note that Table 1 shows only an aggregate view of the real log. In fact, real logs typically contain much more information, e.g., timestamps, transactional information, information on users, data attributes, etc. However, for the purpose of this paper, we can abstract from this information and focus in the information shown in Table 1.

A possible result of a process discovery mining algorithm for the log in Table 1 is depicted in Figure 1. The process is represented in terms of a Petri net [6], i.e., a bipartite directed graph with two node types: [illegible] and [illegible]. Places (circles) represent states while transitions (rectangles) represent actions (e.g., tasks). A certain state holds in a system if it is [illegible] (i.e., it contains at least one [illegible]). Tokens flow between states by [illegible] (or executing) the transitions. A transition is [illegible] (or may fire) if all of its input places have at least one token. When a transition fires, it removes one token from each of its input places and it adds one tokens to each of its output places. More information about Petri nets can be found in [6]. Figure 1 shows that the process can be started by putting a token in the source place at the left and firing the two leftmost transitions. After this, there is a choice to fire "Travel by Train" or "Travel by Car". Firing one of these two transitions results in the production of two tokens: one to trigger the next step ("Conference Starts") and one to "remember" the choice. Note that after task "Go Home" (later in the process) there is another choice, but this choice is controlled by the earlier choice, e.g., people that come by train return by train. Also note that after task "Conference Starts" there is a choice to give a talk or to bypass this task. The black transition refers to a "silent step", i.e., a task not recorded in the log because it does not correspond to a real activity and has only been added for routing purposes.

The model shown in Figure 1 allows for the execution of all process instances in the original log (Table 1). Moreover, the model seems to be at the right level of abstraction because it does not allow for more behavior than the one in the log (e.g., it explicitly portrays that attendees have used the same means of transportation to go to and come back from the conference) and there does not seem to be a way to simplify it without destroying the fit between the log and model. Note that in Figure 1 some of the tasks are duplicated, there are the transitions labeled "Travel by Train" and "Travel by Car" that occur multiple times. Most mining techniques do not allow for this. If we apply a

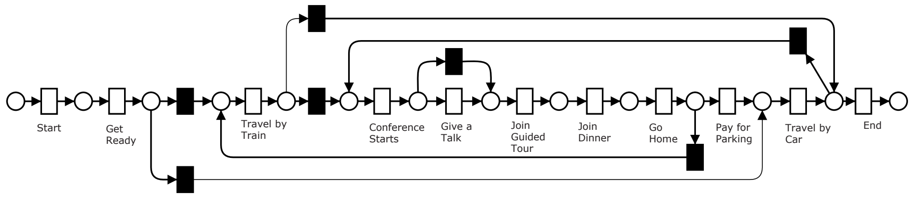


Fig. 2. Example of another mined model for the log in Table 1. This model is more general than necessary.

mining algorithm that does not support duplicate tasks, the resulting model could look like the one in Figure 2. Note that, although this model captures (or can reproduce) the behavior in the log (cf. Table 1), it is more general than necessary because: (i) attendees can use different means of transportation to reach and leave the one-day conference, (ii) attendees can skip the whole conference after traveling by car or by train, and (iii) attendees can return to the conference after going home (cf. task "Go Home"). So, this model is not a precise picture of the traces in the log. For such a small log, it is easy to visually detect the points of over-generalizations in the model in Figure 2. However, when mining bigger logs, there is a need for a technique that can automatically identify these points.

Figures 1 and 2 show that process mining algorithms may produce suitable models but also models that are less appropriate. In particular, the more robust algorithms have a tendency to over-generalize, i.e., construct models that allow for much more behavior than actually observed. A related problem is that event logs typically record information related to different processes. For example, there could be process instances related to the reviewing of papers mixed up with the instances shown in Table 1. Note that the distinction between processes is sometimes not very clear and in a way arbitrarily chosen. In our example, one could argue that there are two processes: one for handling people coming by car and the other one for people coming by train. When processing insurance claims, one could argue that there is one process to handle all kinds of insurance claims. However, one could also have a separate process for each of the different types of insurance policies (e.g., fire, flooding, theft, health, and car insurance). [illegible] This is the core problem addressed by the approach presented in this paper.

To address problems related to over-generalization and mixing up different processes in a single model, we propose to use [illegible]. We do not aim to construct one big model that explains everything. Instead we try to cluster similar cases in such a way that for every cluster it is possible to construct a relatively simple model that fits well without too much over-generalization. The idea for such an approach was already mentioned in [4]. However, in this paper we refine the technique and use a much more powerful process mining technique. Moreover, we report on the implementation of this approach in ProM.

Both the ProM framework and the described plug-ins are publicly available at www.processmining.org.

The remainder of this paper is organized as follows. Section 2 provides an overview of the approach in this paper. Section 3 explains our implementation for this approach and Section 4 describes how to use this implementation to discover over-generalizations in mined models and common patterns in logs. Section 5 presents related work and Section 6 concludes this paper.

2 Approach

As explained in the introduction, the goal of this paper is to allow for the mining of processes with very diverse cases (i.e., not a homogeneous group of process instances) while avoiding over-generalization. Consider for example the application of process mining to care flows in hospitals. If one attempts to construct a single process model for all patients, the model will be very complex because it is difficult to fit the different kinds of treatments into the same model. The different patient groups are ill-defined and share resources and process fragments. For example, what about the patient that was hit by a car after getting a hearth attack? This patient needs both a knee operation and hearth surgery and belongs to a mixture of patient groups. This example shows that it is not easy to define the boundary of a process and that given the heterogeneity of cases it may be impossible to find a clean structure. Moreover, process mining techniques that

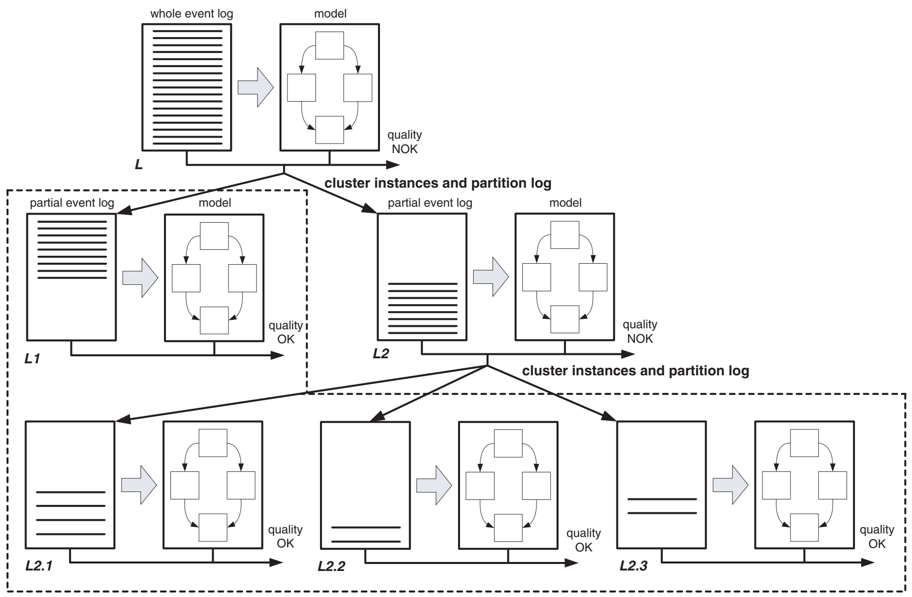


Fig. 3. Overview of the approach: the process instances are iteratively partitioned into clusters until it is possible to discover a "suitable model" for each each cluster

are able to deal with less structured processes have a tendency to over-generalize, i.e., the process instances fit into the model but the model allows for much more behavior than what is actually recorded. To address this we suggest to [illegible]

Figure 3 shows the basic idea behind this approach. First the whole log is considered (denoted by L in Figure 3). Using a discovery algorithm a process model is constructed. By comparing the log and the process model, the quality of the model is measured. If the model has good quality and there is no way to improve it, the approach stops. However, as long as the model is not optimal the log is partitioned into clusters with the hope that it may be easier to construct a better process model for each of the clusters. In Figure 3 log L is split into two logs $L1$ and $L2$, i.e., each process instance of L appears in either $L1$ or $L2$. Then for each cluster, the procedure is repeated. In Figure 3, the quality of the model discovered for cluster $L1$ is good and $L1$ is not partitioned any further. The quality of the model discovered for cluster $L2$ is not OK and improvements are possible by partitioning $L2$ into three clusters: $L2.1$, $L2.2$, and $L2.3$. As Figure 3 shows it is possible to construct suitable process models for each of these three clusters and the approach ends.

Note that the approach results in a hierarchy of clusters each represented by a partial log and a process model. This is quite different from conventional process mining approaches that produce a single model. The leaves of the tree presented in Figure 3 (enclosed by dashed lines) represent clusters that are homogenous enough to construct suitable models.

When applying the approach illustrated by Figure 3, there are basically three decisions that need to be made: [illegible] The approach starts with considering the whole log as a single cluster. Then this cluster is partitioned into smaller clusters which again may be partitioned into even smaller clusters, etc. Note that this always ends because eventually all clusters contain only one process instance and cannot be split anymore. However, it is desirable to have as few clusters as possible, so a good stopping criterion is needed. Stopping too late, may result in a proces model for every process instance. Stopping too early, may result in low quality process models that try to describe a set of cases that is too heterogeneous; [illegible] There are many ways to split a cluster into smaller clusters. An obvious choice is to cluster process instances that have a similar "profile" when it comes to task executions, e.g., split the log into a cluster where "A" occurs and a cluster where "A" does not occur. It is also possible to use more refined features to partition process instances. It may also be possible to use data elements, e.g., in case of hospital data it may be wise to cluster patients based on the diagnosis. Different approaches known from the data mining field can be used here; [illegible] To extract a process model from the process instances in the cluster different process mining algorithms can be used. Some algorithms are very robust but yield models that allow for too much behavior. Other algorithms produce models that are unable to replay the existing log, i.e., there are process instances whose behavior is not allowed according to the model.

The approach presented in this paper is inspired by the process mining algorithm described in [4]. Here the approach is termed Disjunctive Workflow Schema (DWS) and particular choices are made for the three questions listed above. For example, a rather weak algorithm is used for process discovery. The algorithm described in [4] is unable to deal with loops, non-free-choice constructs (e.g., the controlled choice in Figure 1 forcing people to take the same means of transportation home), etc. Moreover, the different parts of the approach are tightly coupled in [4]. In our view, it is essential to allow for different techniques to be plugged into the approach illustrated by Figure 3. Hence, this paper improves this earlier work in several directions: (1) the approach is presented independent of a particular algorithm, (2) several improvements have been made (e.g., replacing the process mining algorithm), and (3) the whole approach is implemented in ProM using an architecture that makes it easy to plug-in new clustering algorithms or process discovery algorithms.

The remainder of this paper presents a particular choice for each of the three questions stated before and describes the new plug-ins implemented in ProM. We use the combination of a very robust process mining algorithm (Heuristics Miner) combined with a clustering approach focusing on over-generalization.

3 Implementation

This section explains how we have answered the questions raised in Section 2 and describes the resulting plug-ins that were implemented in ProM.

3.1 When to Further Partition a Cluster?

A cluster should be further partitioned when its mined model allows for more behavior than what is expressed by the traces in the cluster. So, in our approach, generalizations in the model are captured by [illegible] (or structural patterns) that, while being in principle executable according to the model, have never been registered in the log. If such discrepancies can be identified, then the model is not an accurate representation for the process underlying the log. Hence, we are given some evidence that the model has to be further specialized into a set of different, more specific use cases.

To identify the relevant features, we use an [illegible] [2] like approach. The idea is to incrementally generate sequences of tasks by extending a sequence of length n with another task, and to subsequently check for their frequency in the log[1]. A relevant feature is basically a sequence for which this incremental extension cannot be carried out by guaranteeing that the resulting sequence is as frequent as its two basic constituents. Hence, a relevant feature is a sequence,

[1] Note that log traces are projected while checking for the frequency of these sequences in the log. During the projection, only the tasks that are in a sequence are kept in the log traces.

say $t_1, ..., t_n$, together with a task, say t_{n+1} such that (cf. Figure 4): (i) $t_1, ..., t_n$ is [illegible], i.e., the fraction of projected log traces in which the sequence occurs is greater than a fixed threshold (called [illegible]); (ii) t_n, t_{n+1} is also frequent with respect to the same threshold [illegible]; but,(iii) the whole sequence $t_1, ..., t_n, t_{n+1}$ is [illegible] frequent, i.e., its occurrence is smaller than some threshold [illegible].

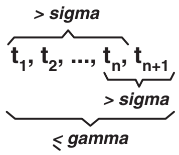


Fig. 4. Feature selection. The subparts ("$t_1...t_n$" and "$t_n t_{n+1}$") of the feature should occur more than *sigma* times in the projected log traces while the whole sequence ("$t_1...t_{n+1}$") should occur at most *gamma* times.

3.2 How to Split a Cluster into Smaller Clusters?

We use the *k*-[illegible] method [5] to split a cluster into sub-clusters. *k*-[illegible] is a clustering algorithm based on Euclidian distance in vectorial spaces. It works by finding central points (or centroids) over which a set of vectors are clustered. Every cluster has a centroid. The parameter k determines the number of centroids (and, therefore, the number of clusters). Consequently, to reuse the *k*-[illegible] clustering method, we have designed an approach for producing a flat representation of the traces by projecting each of them on the relevant features identified as outlined in Section 3.1. In particular, only the most relevant m features are considered. The relevance of each feature is measured on the basis of the frequency of its occurrences in the log (the more frequent, the more relevant). Once features have been selected, each trace in the log is processed and associated with a vector in a m-dimensional vectorial space: for each feature, the corresponding entry of the vector is set to a value that is proportional to the fraction of the feature actually occurring in the trace. In the extreme case where the feature does not characterize the trace, this value is set to 0. When the whole feature matches the trace, this value is set to 1. After the log traces have been projected in the vectorial space, the *k*-[illegible] clustering method takes place.

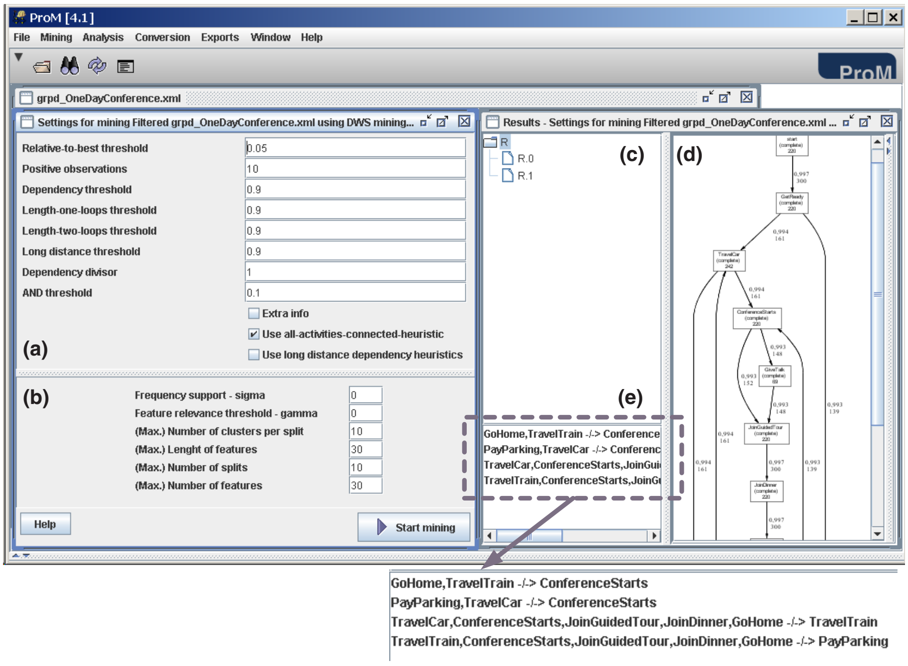
3.3 What Discovery Algorithm to Use?

We have selected the [illegible] (HM) [9] to use as the mining algorithm in our approach. The HM can deal with noise and can be used to express the main behavior (i.e., not all the details and exceptions) registered in an event log. It supports the mining of all common constructs in process models (i.e., sequence, choice, parallelim, loops, invisible tasks and some kinds of non-free-choice), except for duplicate tasks. Therefore, the HM is a more robust algorithm than the mining algorithm originally used in [4]. The HM algorithm has two main steps. In the first step, a [illegible] is built. In the second step, the [illegible] in the dependency graph are set. Due to the lack of space, in this paper we do not elaborate in the relation between these two steps and the threshold values used by the HM. The interested reader is referred to [9].

The next subsection introduces the two plug-ins that were implemented in ProM to support the choices explained so far in this section.

3.4 Implemented Plug-Ins

Two ProM plug-ins have been implemented to provide for the mining of precise models: [illegible] and [illegible]. The [illegible] plug-in implements the full-cycle of the approach in Figure 3. This plug-in starts with an event log and uses the Heuristic Miner (cf. Section 3.3) to mine a model for this log. Afterwards, the plug-in tries to detect relevant features for this model (cf. Section 3.1). If features are found, the plug-in clusters this log based on *k*-[illegible] (cf. Section 3.2). If further sub-clusters can be identified, a model is again automatically mined by the HM for each sub-cluster. This iterative procedure continues until no further clustering is possible. Figure 5 shows a screenshot of applying the DWS Mining plug-in to the log in Table 1. As can be seen, [illegible] has two parts: one for setting the parameters used by the HM (cf. Figure 5(a)) and another for setting the parameters for the feature selection and clustering (cf. Figure 5(b)). The parameters for the HM correspond


ProM [4.1]
File Mining Analysis Conversion Exports Window Help
ProM
grpd_OneDayConference.xml
Settings for mining Filtered grpd_OneDayConference.xml using DWS mining...
Results - Settings for mining Filtered grpd_OneDayConference.xml ...
(a)
(b)
(c)
(d)
(e)
R
R.0
R.1
Help
Start mining
start
GetReady
TravelCar
ConferenceStarts
GiveTalk
JoinGuidedTour
JoinDinner

Relative-to-best threshold	0.05
Positive observations	10
Dependency threshold	0.9
Length-one-loops threshold	0.9
Length-two-loops threshold	0.9
Long distance threshold	0.9
Dependency divisor	1
AND threshold	0.1

☐ Extra info
☑ Use all-activities-connected-heuristic
☐ Use long distance dependency heuristics

Frequency support - sigma	0
Feature relevance threshold - gamma	0
(Max.) Number of clusters per split	10
(Max.) Lenght of features	30
(Max.) Number of splits	10
(Max.) Number of features	30

GoHome,TravelTrain -/-> ConferenceStarts
PayParking,TravelCar -/-> ConferenceStarts
TravelCar,ConferenceStarts,JoinGuidedTour,JoinDinner,GoHome -/-> TravelTrain
TravelTrain,ConferenceStarts,JoinGuidedTour,JoinDinner,GoHome -/-> PayParking

Fig. 5. Screenshot of the *DWS mining* plug-in

to the thresholds mentioned in Section 3.3. The first three parameters in the panel in Figure 5(b) are respectively used to determine the [illegible], [illegible] and k threshold explained in sections 3.1 and 3.2. Note that other three extra parameters are provided ("(Max.) Length of features", "(Max.) Number of splits" and "(Max.) Number of features") which allow for determining upper bounds to the algorithm. The parameter "(Max.) Length of features" sets the length of the sequences up to which the discovery of the frequent features is carried out. In many practical situations, this parameter may be set to 2, meaning that one looks for features of the form t_1, t_2 with a task t_3 such that t_1, t_2, t_3 is not frequent. Larger values for this length may be desirable for models involving many tasks, especially when the paths between the starting task and some final one involve lots of tasks. The parameter "(Max.) Number of splits" is an upper bound on the total number of splits to be performed by the algorithm. The higher its value, the deeper the resulting hierarchy can be. The parameter "(Max.) Number of features" defines the dimension of the feature space over which the clustering algorithm is applied. Note that "(Max.) Number of features" should be greater than k (i.e. parameter "(Max.) Number of clusters per split"). In general, larger values of "(Max.) Number of features" lead to higher quality in the clustering results but it requires more computational time. The [illegible] has three panels: the hierarchy of the found clusters (cf. Figure 5(c)), the model mined for each cluster[2] (cf. Figure 5(d)), and the set of relevant features (cf. Figure 5(e)) used to split each cluster. The subcomponents of each feature are separated by the symbol "-/->". The substrings on the left and right side of this symbol respectively correspond to $t_1 ... t_n$ and t_{n+1} in Figure 5. As can be seen at the bottom of this figure, four features have been found for the settings in this example. These features reflect the generalizations already discussed in Section 1. The results in Figure 5 are discussed in Section 4. The [illegible] plug-in is very similar to the DWS mining one. However, it has the advantage that the approach is decoupled from a specific mining plug-in. The input of the DWS Analysis plug-in is a log [illegible] a model. Its output is a set of partitions (or clusters) for this log. No sub-clusters are provided because the user can again choose which mining plug-ins to use for each of the resulting clusters.

The next section describes how to use the parameters in these two plug-ins to detect (i) over-generalizations in mined models and (ii) frequent patterns in the log.

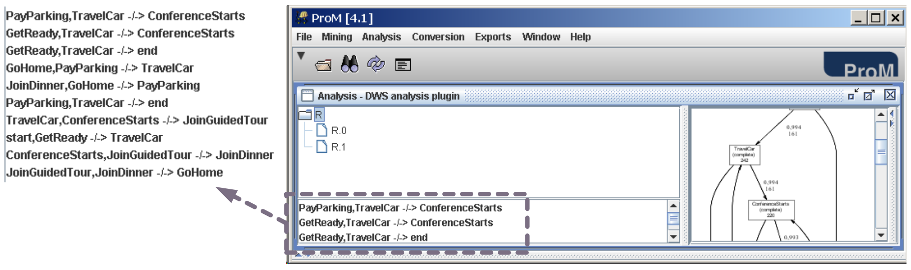
4 Detecting Over-Generalizations and Common Patterns

A (mined) model is over-general when it can generate behavior that cannot be derived from the log. In other words, certain sequences (or features) are

[2] The model mined by the HM for this example is just like the one in Figure 2. However, instead of Petri nets, the HM uses *Heuristics nets* as the notation to represent models. Due to the lack of space and the fact that the two models allow for the same behavior, we will not provide an explanation about Heuristics nets. The interested reader is referred to [9].

possible in the model but do not appear in the log. Using this reasoning, over-generalizations can be detected by the DWS analysis or mining plug-ins whenever we (i) set the [illegible] and [illegible] parameters to 0, (ii) allow for feature sizes that are [illegible] as big as the number of tasks in the model and (iii) set a maximum number of features so that all points of over-generalization are kept in the list of identified features. As an illustration, consider the screenshot in Figure 5. This figure shows the result of applying the DWS mining plug-in to the model in Figure 2 linked to the log in Table 1. Note that the DWS mining plug-in successfully detected [illegible] points of over-generalizations in the model. For instance, the first feature (see bottom-right) states that the task "ConferenceStarts" was never executed after the sequence "GoHome,TravelTrain" has happened, although the sequence "TravelTrain,ConferenceStarts" appears in the log. Actually, the first two features indicate that attendees did not return to the conference after going home and the last two features reflect that attends always used the same means of transportation while reaching of leaving the conference. Note that the algorithm did not capture the over-generalizations for the sequences "GetReady,TravelTrain,End" and "GetReady,TravelCar,End" because features are detected based on their occurrences in [illegible] traces in a log. So, if we project the traces in the log in Table 1 to contain only the tasks in these two sequences, they both will occur in the log and, therefore, are not relevant features. This example shows that the DWS analysis plug-in [illegible] [illegible] [illegible]. This information is especially useful when many parts of a model capture the right level of abstraction and just a few parts do not. In these situations, the DWS analysis or mining plug-ins could guide a designer in modifying the model to make it more precise.

Our approach can also be used to identify common patterns in the log. In these situations, the values for the [illegible] and [illegible] parameters may overlap because the whole feature may also be a common pattern in the log. For instance, if one wants to find out the patterns that happen at least in half of the traces in the log, one can set [illegible] = 0.5 and [illegible] = 1. With these settings, one is


PayParking,TravelCar -/-> ConferenceStarts
GetReady,TravelCar -/-> ConferenceStarts
GetReady,TravelCar -/-> end
GoHome,PayParking -/-> TravelCar
JoinDinner,GoHome -/-> PayParking
PayParking,TravelCar -/-> end
TravelCar,ConferenceStarts -/-> JoinGuidedTour
start,GetReady -/-> TravelCar
ConferenceStarts,JoinGuidedTour -/-> JoinDinner
JoinGuidedTour,JoinDinner -/-> GoHome

PayParking,TravelCar -/-> ConferenceStarts
GetReady,TravelCar -/-> ConferenceStarts
GetReady,TravelCar -/-> end
ProM [4.1]
File Mining Analysis Conversion Exports Window Help
ProM
Analysis - DWS analysis plugin
R
R.0
R.1
TravelCar (complete) 242
0.994 161
0.994 161
ConferenceStarts (complete) 220
0.993

Fig. 6. Screenshot of the result of applying the *DWS analysis* plug-in to detect *common patterns in the log*. The model, log and configuration parameters are the same as in Figure 6, except for the parameters "Frequency support - sigma" and "Frequency relevance threshold - gamma" which were respectively set to 0.5 and 1.

saying that the subparts of the feature (i.e., "$t_1 ... t_n$" and "$t_n t_{n+1}$") happen in more than 50% of the traces in the log and its combination (i.e., "$t_1 ... t_n t_{n+1}$") happens in (i) all traces, (ii) some traces or (iii) none of the traces in the log. As an illustration, let us try to find out the most frequent patterns (above 50%) in the log in Table 1. The results are in Figure 6. As expected, all the patterns identified in the list of features returned by the DWS analysis plug-in occur in process instances 1 and 3 in the log. Together, these process instances correspond to 53.6% of the behavior in the log (cf. Table 1). Based on the ten identified features, the log was partitioned into two clusters: "R.0" contains the process instances 1 and 3, and "R.1" has the process instances 2 and 4.

5 Related Work

This section reviews the techniques that that have been used to detect over-general mined models in the process mining domain. Greco et al. [4] have defined the [illegible] metric, which receives a model and a log as input and calculates the percentage of traces that a model can generate and are not in the log. Since the log is assumed to be exhaustive, this metric only works for [illegible] models. Rozinat et al. [7] have created two notions of [illegible] (a_B), both based on a model and a log. The [illegible] a_B measures the average number of enabled tasks while replaying a log in a model. The problem with this metric is that it does not take into account [illegible] tasks are enabled. The [illegible] a_B calculates the so-called "sometimes" predecessor and successor binary relations for tasks in a log and tasks in a model. The problem here is that the relations are global and binary. So, the approach cannot identify over-generalizations that involve more than two tasks. Alves de Medeiros et al. [8] have defined the [illegible] (B_P) and [illegible] (B_R) metrics, which work by checking how much behavior two models have in common with respect to a given log. Although the metrics quantify the degree of over-generalization in a mined model, they have the drawback that they require a base model (in addition to the mined model and the log). Van Dongen et al. [3] have defined the [illegible] metric, which assesses behavior similarity of two models based on their [illegible]. Like the behavioral precision/recall metrics, the problem with the causal footprint is that a base model is also required. The approach presented in this paper differs from the previously discussed ones because it not only detects that a mined model is over-general, but it also highlights where the over-general points are. Additionally, it only requires a log and a model to run (i.e., no need for a base model).

6 Conclusions and Future Work

This paper has introduced an approach for mining precise models by clustering a log. The approach has been implemented as two ProM plug-ins: the [illegible] and the [illegible]. The DWS analysis plug-in can be used to detect [illegible] in a model or [illegible] in a log. The DWS mining plug-in provides a way to mine a hierarchical tree of process models by using the [illegible], a pre-existing ProM plug-in that is robust to noise and can handle most of the common control-flow constructs in process models. By decoupling the feature selection and clustering steps from a specific mining algorithm, this paper has shown how to broaden the reach of the techniques in [4] such that other process mining algorithms can easily use them. Future work will focus on (i) allowing for the definition of intervals for the thresholds [illegible] and [illegible], and (ii) removing the constraint that features are possible (sub-)paths in a model. This way it would be possible to identify features whose two sub-components are not directly connected.

References

1. van der Aalst, W.M.P., Weijters, A.J.M.M. (eds.): Process Mining. Special Issue of Computers in Industry, vol. 53. Elsevier Science Publishers, Amsterdam (2004)
2. Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules. In: Bocca, J.B., Jarke, M., Zaniolo, C. (eds.) VLDB, pp. 487–499. Morgan Kaufmann, San Francisco (1994)
3. van Dongen, B.F., Mendling, J., van der Aalst, W.M.P.: Structural Patterns for Soundness of Business Process Models. In: EDOC 2006. Proceedings of the 10th IEEE International Enterprise Distributed Object Computing Conference, pp. 116–128. IEEE Computer Society Press, Washington, DC (2006)
4. Greco, G., Guzzo, A., Pontieri, L., Sacca, D.: Discovering expressive process models by clustering log traces. IEEE Transactions on Knowledge and Data Engineering 18(8), 1010–1027 (2006)
5. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a Review. ACM Computing Surveys 31(3), 264–323 (1999)
6. Reisig, W., Rozenberg, G. (eds.): APN 1998. LNCS, vol. 1491. Springer, Heidelberg (1998)
7. Rozinat, A., van der Aalst, W.M.P.: Conformance Testing: Measuring the Fit and Appropriateness of Event Logs and Process Models. In: Bussler, C.J., Haller, A. (eds.) BPM 2005. LNCS, vol. 3812, pp. 163–176. Springer, Heidelberg (2006)
8. van der Aalst, W.M.P., Alves de Medeiros, A.K., Weijters, A.J.M.M.: Process equivalence: Comparing two process models based on observed behavior. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102, pp. 129–144. Springer, Heidelberg (2006)
9. Weijters, A.J.M.M., van der Aalst, W.M.P., Alves de Medeiros, A.K.: Process Mining with HeuristicsMiner Algorithm. BETA Working Paper Series, WP 166, Eindhoven University of Technology, Eindhoven (2006)

Preprocessing Support for Large Scale Process Mining of SAP Transactions

Jon Espen Ingvaldsen and Jon Atle Gulla

Norwegian University of Science and Technology,
Department of Computer and Information Science,
Sem Saelands vei 7-9,
NO-7491 Trondheim, Norway
{jonespi, jag}@idi.ntnu.no

Abstract. Since ERP systems, like SAP, support the backbone operations of companies, their transaction logs provide valuable insight into the companies' business processes. In SAP every transaction is stored and linked to relevant documents, organizational structures and other process-relevant information. However, the complexities and size of SAP logs make it hard to analyze the business processes directly with current process mining tools. This paper describes an ERP log analysis system that allows the users to define at a meta level how events, resources and their inter-relations are stored and transformed for use in process mining. We show how the system is applied to extract and transform related SAP transaction data for the ProM process mining tool.

1 Introduction

SAP is the most widely used ERP system for backbone operations. SAP implementations are configured according to the SAP Reference Model or customized for specific requirements. Even though there may be blue print models defined for how the the systems should support organizational business processes, there are often gaps between how the systems are planned to be used and how the employees actually carry out the operations. To identify these gaps, we need models that reveal how the actual business processes are carried out.

Static blue print models, like the SAP Reference Model, also ignore information about load distributions. This means that a system might be modeled correctly, but for a person reading the model it is not possible to say which parts of the process flows are carried out frequently and which parts are hardly carried out at all. And we cannot know whether the process have been finished within acceptable time limits. To access such information we need to collect historical information about executed process instances.

The functional richness of SAP systems makes the whole SAP Reference Model extensive and complex. Recent research has revealed modeling errors in 5.6% (lower bound) of the Event Process Chains in the SAP Reference Model [1]. As a consequence, the EPC models in the SAP Reference Model are not completely representational for how standard SAP systems are implemented on the system level.

Process mining aims at extracting descriptive models from event logs in Enterprise Resource Planning (ERP) or Workflow Management Systems (WfMS) to reconstruct the underlying business process flows[2]. As process mining models are built upon instance data, we are also capable of enriching the models with key performance indicators, load distribution information and other more detailed analyses of the business flows[3][4].

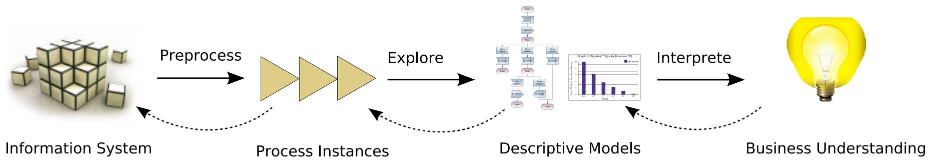


Fig. 1. Process mining project methodology

Figure 1 shows the steps involved in a process mining project. The basis for carrying out a project is a data material that contains event related information fragments. To make use of the raw data material, pre-processing activities are often required before process mining algorithms can be applied. The goal of the pre-processing phase is to extract normalized event logs, a job which involves activities such as data cleansing, feature selection, and merging of distinct data sources.

The output of the pre-processing phase is process or event instances that can be explored through graphs and process- and data mining models. The goal of the exploration phase is to give the user a deeper understanding of his business, which again can be exploited to improve organizational structures and policies. To gain a proper business understanding users typically have to extract several models that describe different perspectives in the process analyses, i.e. control flow, social networks, load distributions, etc.

An important aspect of the process mining projects is that for each phase it might be necessary to return to the previous phase to make improvements or perform additional activities. This makes the nature of process mining projects iterative and interactive.

Some information systems produce event logs that can be fed into process mining algorithms directly with little pre-processing involved. For other systems, the pre-processing is the most time-consuming and work intensive phase.

In this paper, we will describe the Enterprise Visualization Suite (EVS)[1] Model Builder; an application that is designed to support the pre-processing phase of ERP related process mining projects. We will use data from procurement and logistics as examples to show SAP related pre-processing challenges

[1] EVS is a visualization, process- and data mining framework, developed by Businesscape AS (www.businesscape.no).

and show the EVS Model Builder approach to support the work. The output of the EVS Model Builder is identified process instances that are stored as MXML, which is the input format to the open source process mining application ProM[2].

Many SAP tables can be viewed as an event log that individually has some potential for process mining. In this paper, we focus on process mining where process instances are constructed from tracing resource dependencies between executed transactions.

Section 2 describes characteristics of SAP transaction data, and how they create challenges for larger process mining projects. The architecture of the EVS Model Builder and its approach for supporting the pre-processing phase are given in Section 3. Section 4 follows with an discussion of results and alternatives. Related work is described in Section 5, followed by some concluding remarks in Section 6.

2 SAP Transaction Data

SAP contains more than 10000 transaction. Transactions are sub applications of SAP that are accessed by their unique transaction code or through menu hierarchies. It is important to note that transactions do not have a one-to-one mapping to tasks, which means that tasks can be carried out through more than one transaction, which again can incorporate a set of task. A task (also known as function) is an atomic entity describing "what is to be done" in the SAP Reference Model[5].

Transactions carried out in the SAP system store and change data elements in master data and transaction tables. Transaction tables are the largest, since they contain the daily operations data, such as sales orders and invoices. Master data files describe sets of basic business entities such as customers, vendors and users [6]. Most transactions do operations on some resource, typically a document. Documents are most often represented by two tables, one describing header properties and one describing properties for each involved item.

Most dependiencies between SAP documents are stored at the item-level. This makes it possible for one document to depend on a collection of other document resources.

In SAP, each resource operation is logged, also change operations. Most creation and change events on documents are stored in the CDHDR (shown with example data in Table 1) and CDPOS (Change Document Positions) tables.

Master data tables are valuable for accessing textual descriptions of involved business entities and relationships between them. In process mining, events are assigned with a user (originator in MXML). While users in the transaction tables are referred to by their unique and often cryptical username, their full name and department relationship are found in master data sources.

The SAP database also contains ontological information that helps us interpreting the transactions. The two database tables TSTC and TSTCT (shown in Table 2) describe data related to every SAP transtaction. The TEXT attribute

[2] http://sourceforge.net/projects/prom

Table 1. Sample data from the CDHDR (Change Document Header) table

OBJECTID	OBJECTCLAS	CHANGENR	USERNAME	TCODE	UDATE	UTIME
10764301	BANF	33255224	HANSEN	ME51	20030802	144344
00411544	EINKBELEG	33255226	BJARMAN	ME21	20030802	151531
00367081	EINKBELEG	33255227	BJARMAN	ME21	20030803	114030
10445894	BANF	33255243	ANNE	ME51	20030804	090311
00411544	EINKBELEG	33255221	BJARMAN	ME22N	20030804	091740
00411544	EINKBELEG	33255340	TOR	ME22	20030804	123041
00367092	EINKBELEG	33255261	ANNE	ME21N	20030804	130401
04516134	VERKBELEG	33256062	BATMAN	VA02	20030804	135643
74002003	BELEG	33255265	BATMAN	FB02	20030804	150945

Table 2. Sample data from the TSTCT (Transaction Descriptions) table

TCODE	TEXT
MR1M	Post Invoice Document
ME21N	Create Purchase Order
ME57	Assign and Process Requisitions
ME22N	Change Purchase Order
ME52N	Change Purchase Requisition
MIRO	Enter Invoice

in TSTCT gives a full textual description for each transaction. Whenever we see the transaction code ME21N in the logs, we know that this code stands for 'Create Purchase Order'.

A main challenge for doing process mining on the resource-flow between SAP transactions is that there is no defined consistency for how all documents, change events and resource dependencies are stored. Numerous data tables have to be merged, and the data attributes that are interesting for process mining have to be explicitly located in each data table. In the SAP tables there are also missing and faulty data values that must be handled in the pre-processing phase.

To extract process chains that incorporate the creation and changing of purchase requisitions, purchase orders and invoice receipts, and how these events are related to each other, users, departments and transaction descriptions, we have to pre-process information from the EBAN (Purchase Requisitions), EKKO (Purchase Order Header), RBKP (Invoice Receipts), RSEG (Invoice Items), CDHDR (Change Document Header), TSTCT (Textual Transaction Descriptions), and USR03 (User) tables. In larger process mining projects, potentially incorporating elements such as goods movement, goods receipts, sales and delivery orders, shipping documents etc., the pre-processing job would be even more extensive and difficult.

3 EVS ModelBuilder

The EVS Model Builder supports the pre-processing phase of process mining projects by extracting the data elements that are interesting for process mining analysis, constructing process instances and store them as MXML files. As shown in figure 2, the process of constructing process instances consists of three steps:

1. Extraction of business objects and their inter-relations.
2. Extraction of events and their relationships to business objects.
3. Identification of process instances by tracing dependency relationships between events.

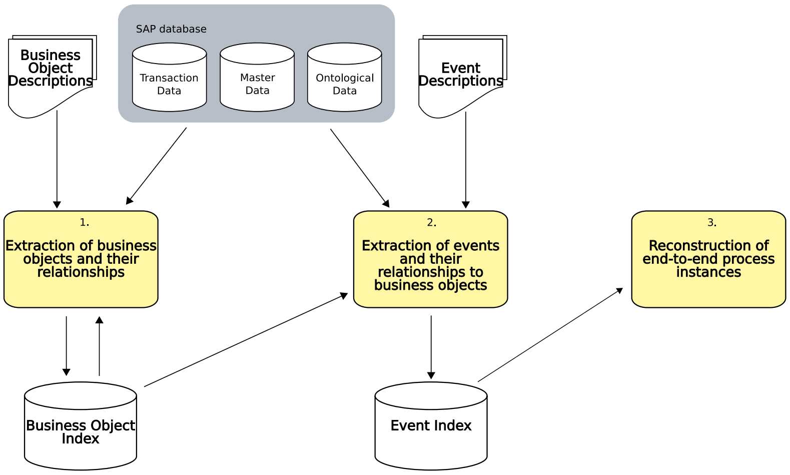

SAP database
Transaction Data
Master Data
Ontological Data
Business Object Descriptions
Event Descriptions
1.
Extraction of business objects and their relationships
2.
Extraction of events and their relationships to business objects
3.
Reconstruction of end-to-end process instances
Business Object Index
Event Index

Fig. 2. The three-step-process for constructing process instances. The data flow is represented with arrows.

Business objects are entities that have a valuable (with respect to out process mining analyses) relationship to the business flow. A business object can be a user, department, transaction, document, and other more project specific entities. Business objects are defined by a unique identifier and a textual description (optional).

Events are happenings at a point of time where a set of business objects co-occur. An example of an event is the alteration of a purchase order. Such an event has a certain timestamp and relates a set of business objects, like a user, a purchase order, and a transaction.

Relationships are labeled such that the process instance construction can treat them differently. To trace the dependencies between the events and to reconstruct process instances, we need to know which business objects an event consumes (inputs) and which it produces (outputs). For an alter purchase order event, a purchase order is consumed, altered, and provided as output. Alteration events consume typically the same resource as they produce. Creation events, on the other hand, uses the consumption resources to produce a new resource. The creation of a purchase order consumes (most often) a purchase requisition, and a purchase order is produced and provided as output.

The EVS Model Builder requires three types of data sources:

1. Transactional data - Information source for constructing event.
2. Master data - The information source for constructing business objects with meaningful names.
3. Ontological data - Information source for interpreting events at the system level.

Together with meta-descriptions, provided by the user, EVS Model Builder knows how to combine these data sources and extract meaningful process intances.

3.1 Defining Element Descriptions

Figure 3 shows a UML class diagram model of how the meta-descriptions are defined. Event and business object descriptions tell us where the information needed to extract their instances exists. Figure 4 shows an meta level description for how business objects, events and their relations can be defined for process mining projects analysing processes related to the creation and chaning of purchase requisitions, purchase orders, and invoice receipts.

In some SAP tables, several types of elements are stored in the same table. CDHDR is one such table, where creation and change events for several document types are stored. Discriminators are used to separate out specific rows of a database table that are of interest to a given element description. A discriminator contains an attribute, an operator (like equals) and a value. For an event description that are only specifying events on purchase requisitions, we can add the discriminator (CDHDR.objectclas = 'BANF').

EVS Model Builder has two ways of describing relationships between event and business object descriptions. Explicit relation descriptions are used to define a connection between two element descriptions, where mapping attributes for both are stated in a single database table. An example of such a relation is between transactions, described in TSTCT, and events in the CDHDR table. Here, we can define an explicit relation description where we map the CDHDR.tcode attribute to the transactions key attribute.

Implicit relations are used to define connection paths that involves several linked tables in the SAP database. An example of such a relation is between purchase orders and creation of invoice receipt events. Here the events are located in the RBKP tables, which have a produce-relation to an invoice receipt (also found in RBKP). These invoice receipts have explicit relations to invoice items found in RSEG, which further points to a set of purchase orders. To create invoice receipt events with a consumption-relation to a set of purchase order, an implicit relation description containing the path of involved element descriptions can be defined.

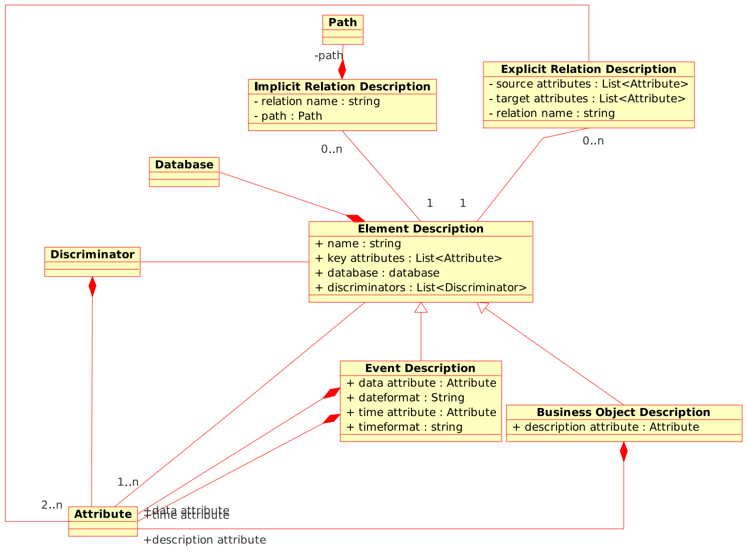

Path
-path
Implicit Relation Description
- relation name : string
- path : Path
Explicit Relation Description
- source attributes : List<Attribute>
- target attributes : List<Attribute>
- relation name : string
0..n
0..n
Database
1
1
Element Description
+ name : string
+ key attributes : List<Attribute>
+ database : database
+ discriminators : List<Discriminator>
Discriminator
Event Description
+ data attribute : Attribute
+ dateformat : String
+ time attribute : Attribute
+ timeformat : string
Business Object Description
+ description attribute : Attribute
1..n
2..n
Attribute
+data attribute
+time attribute
+description attribute

Fig. 3. UML class diagram defining description constructs

3.2 Extracting Business Objects

The first phase in the construction of process instances is to extract all business objects from the set of business objects descriptions that the user has provided. The extraction operation iterates over each business object description and constructs an SQL based on the information they contain.

The SQL statements are executed on the underlying database, and instances from the result sets are stored in a business object index (based on the Lucene Search Engine[3]). This index enables fast lookup of the business objects based on the values of their key and relation mapping attributes. In the index, the business objects refer to each other by id values (loose referencing).

3.3 Extracting Events

The extraction of events creates and executes SQL statements similarly to the extraction of business objects. When the result sets are processed, the date and time values are parsed according to the date and time formats provided in the event descriptions.

To construct relationships to business objects that occur in an event, the business object index is queried. If an event description contains implicit relations to some business objects, these are located by following the relation path and stepwise querying the business object index. All extracted events are stored in a separate event index.

[3] http://lucene.apache.org/

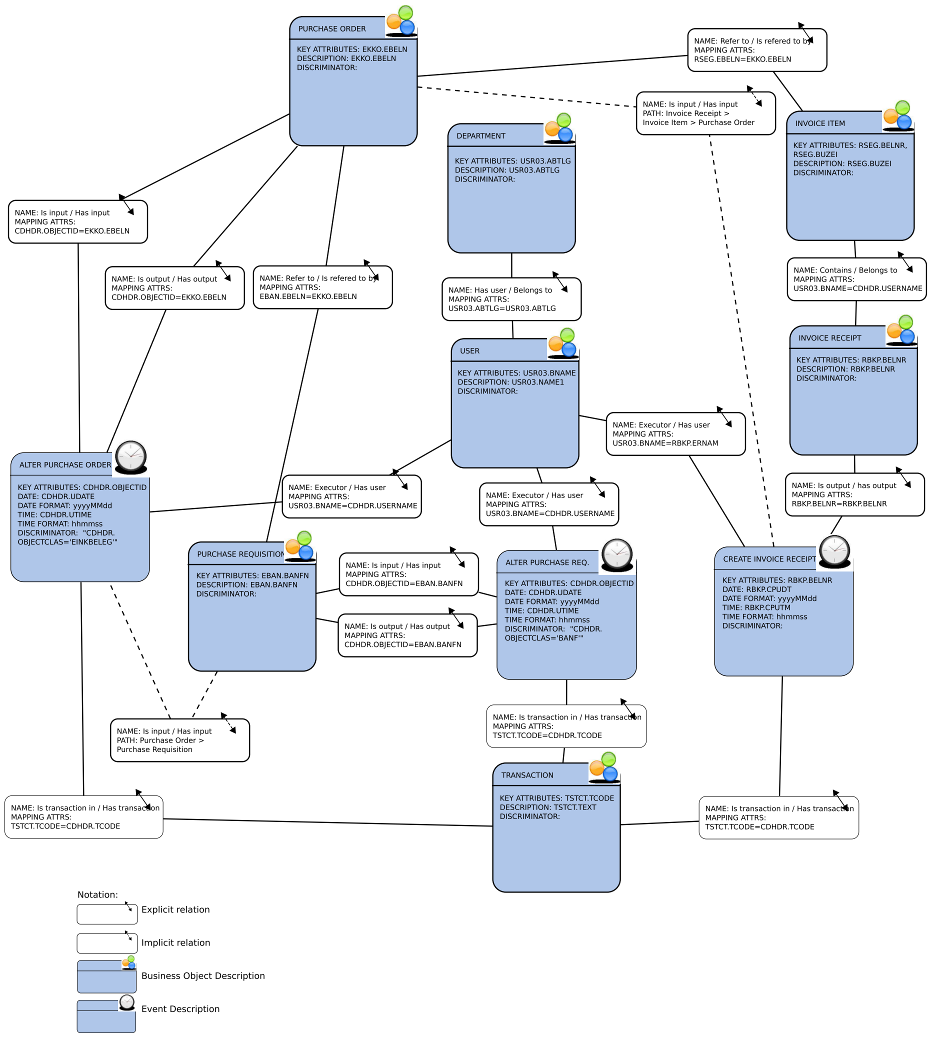

PURCHASE ORDER
KEY ATTRIBUTES: EKKO.EBELN
DESCRIPTION: EKKO.EBELN
DISCRIMINATOR:
NAME: Refer to / Is refered to by
MAPPING ATTRS:
RSEG.EBELN=EKKO.EBELN
NAME: Is input / Has input
PATH: Invoice Receipt > Invoice Item > Purchase Order
INVOICE ITEM
KEY ATTRIBUTES: RSEG.BELNR, RSEG.BUZEI
DESCRIPTION: RSEG.BUZEI
DISCRIMINATOR:
DEPARTMENT
KEY ATTRIBUTES: USR03.ABTLG
DESCRIPTION: USR03.ABTLG
DISCRIMINATOR:
NAME: Is input / Has input
MAPPING ATTRS:
CDHDR.OBJECTID=EKKO.EBELN
NAME: Is output / Has output
MAPPING ATTRS:
CDHDR.OBJECTID=EKKO.EBELN
NAME: Refer to / Is refered to by
MAPPING ATTRS:
EBAN.EBELN=EKKO.EBELN
NAME: Has user / Belongs to
MAPPING ATTRS:
USR03.ABTLG=USR03.ABTLG
NAME: Contains / Belongs to
MAPPING ATTRS:
USR03.BNAME=CDHDR.USERNAME
INVOICE RECEIPT
KEY ATTRIBUTES: RBKP.BELNR
DESCRIPTION: RBKP.BELNR
DISCRIMINATOR:
USER
KEY ATTRIBUTES: USR03.BNAME
DESCRIPTION: USR03.NAME1
DISCRIMINATOR:
NAME: Executor / Has user
MAPPING ATTRS:
USR03.BNAME=RBKP.ERNAM
ALTER PURCHASE ORDER
KEY ATTRIBUTES: CDHDR.OBJECTID
DATE: CDHDR.UDATE
DATE FORMAT: yyyyMMdd
TIME: CDHDR.UTIME
TIME FORMAT: hhmmss
DISCRIMINATOR: "CDHDR.OBJECTCLAS='EINKBELEG'"
NAME: Executor / Has user
MAPPING ATTRS:
USR03.BNAME=CDHDR.USERNAME
NAME: Executor / Has user
MAPPING ATTRS:
USR03.BNAME=CDHDR.USERNAME
NAME: Is output / has output
MAPPING ATTRS:
RBKP.BELNR=RBKP.BELNR
PURCHASE REQUISITION
KEY ATTRIBUTES: EBAN.BANFN
DESCRIPTION: EBAN.BANFN
DISCRIMINATOR:
NAME: Is input / Has input
MAPPING ATTRS:
CDHDR.OBJECTID=EBAN.BANFN
NAME: Is output / Has output
MAPPING ATTRS:
CDHDR.OBJECTID=EBAN.BANFN
ALTER PURCHASE REQ.
KEY ATTRIBUTES: CDHDR.OBJECTID
DATE: CDHDR.UDATE
DATE FORMAT: yyyyMMdd
TIME: CDHDR.UTIME
TIME FORMAT: hhmmss
DISCRIMINATOR: "CDHDR.OBJECTCLAS='BANF'"
CREATE INVOICE RECEIPT
KEY ATTRIBUTES: RBKP.BELNR
DATE: RBKP.CPUDT
DATE FORMAT: yyyyMMdd
TIME: RBKP.CPUTM
TIME FORMAT: hhmmss
DISCRIMINATOR:
NAME: Is input / Has input
PATH: Purchase Order > Purchase Requisition
NAME: Is transaction in / Has transaction
MAPPING ATTRS:
TSTCT.TCODE=CDHDR.TCODE
TRANSACTION
KEY ATTRIBUTES: TSTCT.TCODE
DESCRIPTION: TSTCT.TEXT
DISCRIMINATOR:
NAME: Is transaction in / Has transaction
MAPPING ATTRS:
TSTCT.TCODE=CDHDR.TCODE
NAME: Is transaction in / Has transaction
MAPPING ATTRS:
TSTCT.TCODE=CDHDR.TCODE
Notation:
Explicit relation
Implicit relation
Business Object Description
Event Description

Fig. 4. Example of business object, event and relationship descriptions

As there is no one-to-one mapping between tasks in the SAP Reference Model and transactions, we cannot map executed transactions to the defined business processes. Although this mapping would be preferable, we can still extract meaningful end-to-end process chains of subsequent events. The EVS Model Builder construct process chains by identifying those events that produce and consume the same set of resources.

3.4 MXML Export

The output list of process instances from the EVS Model Builder can be converted to and serialized as MXML. MXML assumes that it is possible to record events such that (i) each event refers to an activity (i.e., a well-defined step in the process), (ii) each event (named AuditTrailEntry) refers to a process instance, (iii) each event can have a performer (the person executing or initiating the activity), and (iv) events have a timestamp and are totally ordered. The structure also incorporates flexibility for other data requirements by having an additional data elemement at each level [4][7].

To store the process instances as MXML, the user must point out which business object type that represents the originator and activity (named Workflow-ModElelement in MXML). For our example in Figure 4, user business objects take the role as originators, while transactions are set as WorkFlowModelElements.

MXML is the event log format for the ProM framework, which is a plug-in based architecture where the kernal offer event log information to its components. Plug-ins within five categories are developed (i) Mining plug-ins (e.g., extraction of Petri nets, social networks, frequency abstraction models, etc.), (ii) Export plug-ins (implementation of "save-as" functionality), (iii) Import plug-ins (e.g., import of instance-EPCs from ARIS PPM), (iv) Analysis plug-ins (implementation of property analysis on som mining results), (v) Conversion plug-ins (implementation of conversion between different data formats, e.g., from EPCs to Petri nets)[4][8].

In ProM the user can explore the data and potentially uncover unknown knowledge about the processes that have been executed. It may be that certain tasks in a process are unreasonably time-consuming compared to others, or that certain operations are under-staffed or over-staffed. More dramatically, the analysis may prove that the organization is not carrying out their business processes according to the policies given. If the uncovered knowledge is used to improve the business processes in an organization, new process mining projects can again be carried out to measure and monitor the new situation.

4 Discussion

The preprocessing support in the EVS Model Builder has been tested in two SAP process mining projects. The projects were carried out at the Norwegian Agricultural and Marketing Cooperative and Nidar (producer and distributor of cholocate and sweets). Both these project target processes related to purchase and logistics, and the analyses targeted vendors behavior specifically. The goals of the projects included:

1. Model discovery - How do the AS-IS models look like?
2. Delivery times - How long delivery time do different vendors have? How do the actual delivery times deviate from planned lead times?
3. Systematic irregulations in deliveries - Do any vendors systematically and over time deliver more or less than what is planned?
4. Order confirmation - To which extent do the vendors send confirmations when orders are placed?

For the model discovery goal, ProM and its modeling plug-ins were used. For the vendor analyses, the constructed process instances enriched with features like planned lead times, and delivery amounts served as the bases for documentation and statistical analyses.

The experiences from the projects is that the preprocessing requires substantial efforts for defining meta descriptions and locating SAP data sources. However, the necessary work of table joins, SQL queries, and date format conversions are automated and hidden at the user level. As a consequence, the time spent on preprocessing and extraction of MXML is drastically reduced. The preprocessing support is especially valuable when longer process chains are mined and the amount of involved database tables is large.As the database structures in implemented SAP systems are mostly consistent, the meta descriptions for preprocessing in one process mining project can be reused and applied in other organizations and process mining projects. That means that if one business object description and its relationships is described for one SAP related process mining project, this information can be reused directly in other SAP process mining projects where the same business object type is involved.

The quality of ontological data in the process mining projects is critical for interpreting events at the system level and constructing meaningful process models. In SAP, information for how to interpret each transaction code is available in the underlying database, but it is not possible to map a transaction directly to tasks and business process hierarchies in the SAP Reference Model. This makes it hard to identify gaps (delta analyses [9]) between the mined models and existing reference models.

Business process models that can be extracted from the MXML describe how transactions, users, departments and other entities depend on each other in the daily operations of a company. In difference from the static reference models, mined process models contain instance data that enable drill-down analyses and the extraction of key performance indicators.

5 Related Work

Also other research activities that target process mining on SAP data have been carried out. In 2004 a master thesis [10] identified potentials for doing process mining on SAP R/3. A two-step methodology was developed, where the first step is to identify the database tables that are needed for the process mining project. They implemented a application, TableFinder, which locates the required tables through business objects[4] in the SAP reference model. The second step in their methodgy is retrieval of the document-flow. In this step they export the data in the tables from step one to an XML format suitable for process mining. The main conclusion of the work is that process mining in SAP R/3 is feasible, but the retrieval of the document-flow is very laborious. The process mining project methodology in figure 1 assumes that the user has good knowledge of the

[4] The EVS Model Builder and the SAP reference model uses the term “business object” to describe different concepts.

underlying data sources. In this way, the methodology by Giessel complements the project methodology in figure 1 with an additional pre-phase for cases where good data knowledge does not exist.

Other reseach projects aim at supporting the conversion of application specific event log data to MXML. The ProM Import Framework[5] allows us to extract process enactment event logs from a set of information systems, including WebSphere, FLOWer, Staffware, PeopleSoft, Eastman, and others. In difference from SAP, these systems offer more complete and defined event log structures.

TeamLog is another tool that supports the pre-processing of event logs from process aware collaboration systems. Like the dependencies between transactions in SAP systems, the processes in such collaboration systems have a highly adhoc structure. The output from TeamLog is XML data that can be analyzed by the EMiT (Enhanced Mining Tool) process mining application [11].

In practical settings, it is more common to use data warehouse solutions to analyze important aspects of ERP-supported business operations. These solutions do not reveal new knowledge patterns, though, as they only reflect the performance indicators already defined by the companies.

6 Conclusions

Doing process mining on SAP data have both positive and challenging aspects. Transactional, master and ontological data that are required for constructing meaningful process models are all available in the underlying SAP database. As the transactions cannot be directly mapped to tasks, we are unfortunately not capable of aggregating the mined transaction flows to the defined processes in the SAP Reference Model.

In this paper, we have addressed pre-processing challenges for process mining projects on SAP transactions. With the aid of pre-processing tools, like the EVS Model Builder, large scale process mining in SAP is feasible. SAP related process mining projects are still a complex task, and to set them up and interprete extracted models correctly we need project members with good knowledge both on the data and business level.

In spite of recent successes in process mining, current tools are still hampered by their reliance on clean and well-structured transaction logs. The logs from ERP-supported industrial business operations are however so complex and large that they do not naturally fit into existing process mining tools. As seen from SAP above, there may also be discrepancies between the application itself and the provided business models, which further complicates the analysis of their logs. As a consequence, there is still very little work on process mining of real large-scale logs, and the technology has so far not shown itself to be very useful in the practical ERP world. Our work on the EVS Model Builder, though, demonstrates that complex SAP transaction logs can be pre-processed and transformed into structures that lend themselves for conventional process mining techniques. It shows not only that mining SAP data per se is valuable, but also that current

[5] http://sourceforge.net/projects/promimport

process mining techniques are relevant in real industrial settings, provided that we have the necessary ontological knowledge of the ERP systems and can deal with the magnitude of data.

References

1. Mendling, J., Moser, M., Neumann, G., Verbeek, H.M.W., van Dongen, B.F., van der Aalst, W.M.P.: Faulty epcs in the sap reference model. In: Business Process Management, pp. 451–457 (2006)
2. van der Aalst, W., Weijters, A.: Process mining: A research agenda (2003)
3. Ingvaldsen, J.E., Gulla, J.A.: Model bases business process mining. In: Information Systems Management, Auerbach Publications, vol. 23 (2006)
4. van der Aalst, W.M.P., Reijers, H.A., Weijters, A.J.M.M., van Dongen, B.F., de Medeiros, A.K.A., Song, M., Verbeek, H.M.W.E.: Business process mining: An industrial application (2007)
5. Keller, G., Teufel, T.: Sap R/3 Process Oriented Implementation. Addison-Wesley Longman Publishing Co. Inc., Boston, MA (1998)
6. Brancroft, N.H., Seip, H., Sprengel, A.: Implementing SAP R/3, 2nd edn. Manning Publications Co., Greenwich, CT (1998)
7. van Dongen, B.F., van der Aalst, W.M.P.: A meta model for process mining data. In: EMOI-INTEROP (2005)
8. van Dongen, B.F., de Medeiros, A.K.A., Verbeek, H.M.W., Weijters, A.J.M.M., van der Aalst, W.M.P.: The prom framework: A new era in process mining tool support. In: Ciardo, G., Darondeau, P. (eds.) ICATPN 2005. LNCS, vol. 3536, Springer, Heidelberg (2005)
9. van der Aalst, W.M.P.: Business alignment: using process mining as a tool for delta analysis and conformance testing. Requir. Eng. 10(3), 198–211 (2005)
10. van Giessel, M.: Process mining in sap r/3: A method for applying process mining to sap r/3. Master Thesis, Eindhoven University of Technology (2004)
11. Dustdar, S., Hoffmann, T., van der Aalst, W.: Mining of ad-hoc business processes with teamlog. Data Knowl. Eng. 55(2), 129–158 (2005)

Process Mining as First-Order Classification Learning on Logs with Negative Events

Stijn Goedertier[1], David Martens[1], Bart Baesens[1,2], Raf Haesen[1,3], and Jan Vanthienen[1]

[1] Department of Decision Sciences & Information Management, Katholieke Universiteit Leuven, Belgium
{myFirstName.myLastName}econ.kuleuven.be
[2] School of Management, University of Southampton, United Kingdom
[3] Vlekho Business School, Belgium

Abstract. Process mining is the automated construction of process models from information system event logs. In this paper we identify three fundamental difficulties related to process mining: the lack of negative information, the presence of history-dependent behavior and the presence of noise. These difficulties can elegantly dealt with when process mining is represented as first-order classification learning on event logs supplemented with negative events. A first set of process discovery experiments indicates the feasibility of this learning technique.

1 Introduction

Event logs of information systems such as ERP, Role Based Access Control, and Workflow Management systems conceal an untapped reservoir of knowledge about the way people conduct every-day business transactions. The vast quantity of available events, however, makes it difficult to analyze event logs using only descriptive statistics. Process mining, in contrast, is the automated construction of process models from event logs [1,2]. Process models that have been discovered through process mining enable organizations to compare the behavior in the event log with the business conduct it would expect from its employees and other stake holders. The latter can be helpful in the context of regulatory compliance or in the context of business process redesign and optimization. Currently, many algorithms have been developed to describe or predict control-flow, data or resource-related aspects of processes. An important but difficult learning task in process mining is the discovery of sequence constraints from event logs, referred to as Process Discovery [3,4]. Other process learning tasks involve, for instance, learning allocation policies [5] and social networks [6].

Process mining faces many difficulties. One difficulty is that it is often limited to the much more difficult setting of unsupervised learning because **negative information** about state transitions that were prevented from taking place is often not available in the event log and consequently cannot guide the search problem. Moreover, much of the behavior displayed in processes is non-local, **history-dependent** behavior. While a history of related events is a potentially strong predictor and is readily available in process logs, the inclusion of such non-local, historic events in the hypothesis space of process mining algorithms poses many difficulties with regard to search space complexity and hypothesis visualization. Another difficulty is that process mining algorithms often overfit the **noise** in event logs.

In this paper these difficulties are addressed by representing process mining as first-order classification learning on event logs supplemented with negative events. We describe a technique to add artificial negative examples to a process log. Additionally, we show how first-order classification learners allow to search for patterns among multiple event rows in the event log and thus allow for detecting history-dependent behavior. The proposed representation is expressive enough to cover many learning tasks in process mining including Process Discovery. The remainder of this article is structured as follows. First an introduction is provided to first-order classification and it is shown how process mining can be represented as a binary classification problem. In section 3 the problem of lacking negative information is discussed and an algorithm is proposed to supplement event logs with artificial negative examples. In section 4 the proposed technique is applied to Process Discovery. Section 5 provides a brief overview of related work.

2 First-Order Classification Learners

Classification learning is learning how to assign an instance to a predefined class or group according to its known characteristics. The result of a classification learning is a model that makes it possible to classify future instances based on a set of specific characteristics in an automated way. Classification techniques are often used for credit scoring [7,8] and medical diagnostic. In process mining classification learning has, for instance, been used for “Decision Mining” [9] and Process Discovery [10].

In this article process mining is represented as a classification problem that models the conditions under which an event can take place (a positive event) or not (a negative event). In this respect, it is useful to think of a **process instance** as a *trajectory* in a *state space* that is span by the domains of the different possible activities, events and business concepts. Declarative classification rules can be used to classify whether a state transition at a give state is allowed (a positive event) or not (a negative event). Each activity in a process instance can undergo a number of distinct state transitions that are recorded as events, for instance:

– `create(AId,BId,PId)`: creates a new activity instance `AId` with business identifiers `BId` in the context of the parent activity `PId`. As a result a `created` event is added to the state of the process instance.
– `assign(AId,AgentId)`: the assignment of activity `AId` to an agent `AgentId` that is recorded as an `assigned` event.

– `addFact(AId,F)`, `removeFact(AId,F)`, `updateFact(AId,F1,F2)`: add, remove or update a business fact `F` in the state space. This is recorded respectively as a `factAdded`, a `factRemoved` or a `factUpdated` event.
– `complete(AId)`: requests the completion of activity `AId`, recorded as an event of the type `completed`.

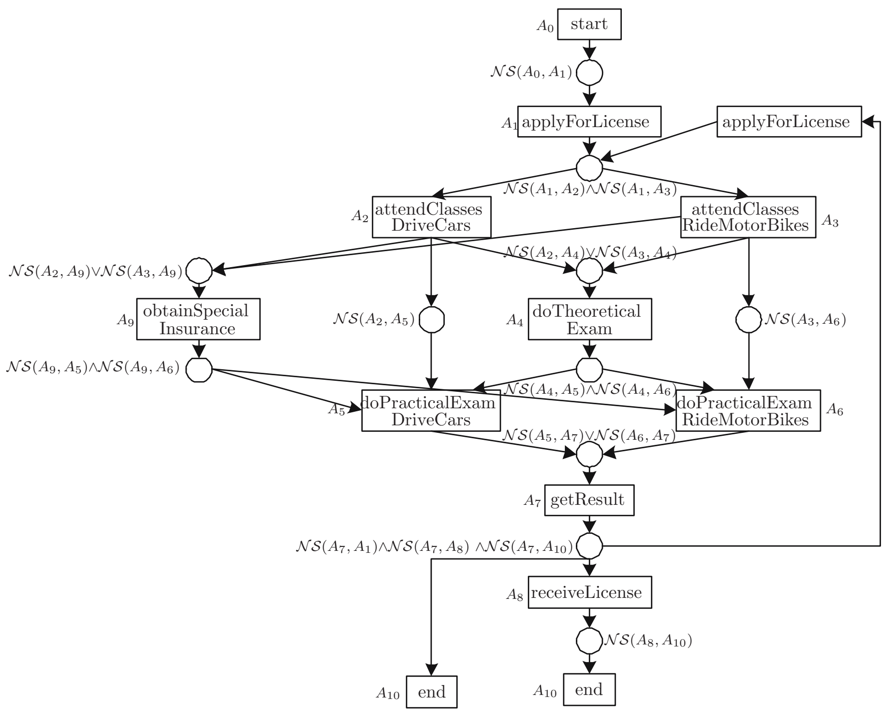
To use first-order learners on an event log, the log has to be represented as a logic program of ground facts. In our experiments, an activity event is represented as an atom `event(AId,AT,BId,ET,AgentId,PL,TS)`, with following arguments:

– `AId` a unique non-business identifier for the activity
– `AT` represents the activity type (e.g. `applyForLicense`)
– `BId` represents a unique business identifier of the activity
– `ET` represents the event type (e.g. `created`, `assigned`, `completed`,...)
– `AgentId` represents the worker that brings about the activity state transition
– `PL` is a list of parameters that pertain to the event
– `TS` is a time stamp

In this article we use an extended version of the "Driver's License" example [4]. This example is a non-trivial Process Discovery problem with non-local non-free choice that has been extended with a parallel task '`obtainSpecialInsurance`' and loop '`applyForLicense`' as displayed in Fig. 1(a). The sample event log below this paragraph represents the activity life cycle of an '`applyForLicence`' activity.

```
event(act92,applyForLicense,driver2,created,worker1,[concept(act92,hasParent,act90)],1).
event(act92,applyForLicense,driver2,factAdded,worker1,[concept(driver2,hasRole,driver)],3).
event(act92,applyForLicense,driver2,assigned,worker1,[concept(act92,assignedTo,driver2)],6).
event(act92,applyForLicense,driver2,factAdded,driver2,[concept(driver2,hasAge,26)],9).
...
event(act92,applyForLicense,driver2,completed,driver2,[],17).
...
event(act99,doTheoreticalExam,driver2,created,worker1,[concept(act96,hasParent,act90)],56).
...
```

Most classification learners are called **propositional or uni-relational classification learners**, because they can only perform classification based on the information within a single row of a dataset. In contrast, **first-order or multi-relational classification learners** can learn classification patterns based on multiple rows within one or more tables of a dataset. For the purpose of discovering history-dependent patterns, this multi-relational property is much desired, as it allows learning based on *global information* in the **event log**. Alternatively the event history of an event log instance could in part be represented as extra propositions (extra columns), for instance including all immediately preceding event information as extra columns in the event log. However, if we want to relate an event to any other previously occurred event within a process instance, it is no longer possible to represent these historic events as extra columns of an event table as the dimensions of the input space would exponentially increase. High dimensional input spaces are typically hard to handle by classical data mining techniques, a problem known as 'the curse of dimensionality' [11].



(a) A Petri net representation

	activity	precondition
A_0	start	*true*
A_1	applyForLicense	$\mathcal{NS}(A_0, A_1) \;\vee$ $(\; count(A_1^{started}) < 3 \;\wedge\; \mathcal{NS}(A_7, A_1)$ $\wedge\; \mathcal{NS}(A_7, A_8) \;\wedge\; \mathcal{NS}(A_7, A_{10})\;)$
A_2	attendClassesDriveCars	$\mathcal{NS}(A_1, A_2) \;\wedge\; \mathcal{NS}(A_1, A_3)$
A_3	attendClassesRideMotorBike	$\mathcal{NS}(A_1, A_2) \;\wedge\; \mathcal{NS}(A_1, A_3)$
A_4	doTheoreticalExam	$\mathcal{NS}(A_2, A_4) \;\vee\; \mathcal{NS}(A_3, A_4)$
A_5	doPracticalExamDriveCars	$\mathcal{NS}(A_4, A_5) \;\wedge\; \mathcal{NS}(A_4, A_6) \;\wedge$ $\mathcal{NS}(A_9, A_5) \;\wedge\; \mathcal{NS}(A_9, A_6) \;\wedge\; \mathcal{NS}(A_2, A_5)$
A_6	doPracticalExamRideMotorBike	$\mathcal{NS}(A_4, A_5) \;\wedge\; \mathcal{NS}(A_4, A_6) \;\wedge$ $\mathcal{NS}(A_9, A_5) \;\wedge\; \mathcal{NS}(A_9, A_6) \;\wedge\; \mathcal{NS}(A_3, A_6)$
A_7	getResult	$\mathcal{NS}(A_5, A_7) \;\vee\; \mathcal{NS}(A_6, A_7)$
A_8	receiveLicense	$\mathcal{NS}(A_7, A_1) \;\wedge\; \mathcal{NS}(A_7, A_8) \;\wedge\; \mathcal{NS}(A_7, A_{10})$
A_9	obtainSpecialInsurance	$\mathcal{NS}(A_2, A_9) \;\vee\; \mathcal{NS}(A_3, A_9)$
A_{10}	end	$\mathcal{NS}(A_8, A_{10}) \;\vee$ $(\; count(A_1^{started}) >= 3 \;\wedge\; \mathcal{NS}(A_7, A_1)$ $\wedge\; \mathcal{NS}(A_7, A_8) \;\wedge\; \mathcal{NS}(A_7, A_{10})\;)$

(b) A representation with activity preconditions

Fig. 1. An extended version of the Diver's License example [4]

The hypotheses to be tested by first-order learners are described in terms of language constructs and constraints. The latter is called the language bias $\mathcal{L}$ of the learning task. The effectiveness by which a multi-relational learner can be applied to a learning task depends in part on the chosen language bias. When searching for a hypothesis, multi-relational learners refine the current hypothesis using the information of the language bias. Too simple refinements result in new hypotheses that have little or no extra explanatory power. Too complex refinements might result in too large a hypothesis space, making search inefficient. Another requirement for the language bias is that expressions in the chosen language can be transformed into graphical models such as Petri nets. Therefore, we use a simple event operator, that, in combination with conjunction (`,`), disjunction (`;`) and negation-as-failure (`not`) provides a reasonably expressive language bias that yields good results in learning non-local classification problems. This operator is called the $\mathcal{NS}$ operator and is defined as follows in Prolog:

```
ns(AT1,AT2,BId,Now) :-
    event(_AId,AT1,BId,completed,_AgentId,_Parameters,Time1),
    Time1 < Now,
    not(eventFromTill(AT2,BId,completed,Time1,Now)).
eventFromTill(AT,BId,ET,From,Till) :-
    event(_AId,AT,BId,ET,_AgentId,_Parameters,Time),
    From < Time, Time < Till.
```

Each transition is characterized as an activity type – event type pair `AT-ET`. The statement `ns(AT1,AT2,BId,Now)`, abbreviated as $\mathcal{NS}(AT_1, AT_2)$, evaluates to true when for a given process instance `BId` at time `Now` an `AT1-completed`-transition has taken place, but that it has not (yet) been followed by an `AT2-completed` state transition. For instance, the expression $\mathcal{NS}(A_0, A_1)$ is true when for a given process instance the activity A_0 has completed at the time of inspection without the activity A_1 being completed. Notice that the $\mathcal{NS}$ operator can be extended to other kinds of state transitions that record, for instance, the assignment (`assigned`), start (`started`) or skipping (`skipped`) of activities. Figure 1(b) shows how the preconditions in the Petri net can be represented as conjunctions and disjunctions of $\mathcal{NS}$ atoms. The conversion from $\mathcal{NS}$ preconditions to Petri nets requires the conditions to refer to local, immediately preceding events as much as possible.

3 Inducing Artificial Negative Events

Without negative information learning can be much harder. For instance, a two-year old will have more difficulties in learning a precise definition of the concept 'balloon' when shown only a balloon than when presented both a ball and a balloon and pointed to their difference. Event logs rarely contain such negative information that allows to identify the distinguishing properties that characterize the underlying process model. Because of the lack of negative information, many learning tasks in process mining are in principle limited to the more difficult setting of unsupervised learning to which classification learners cannot be applied.

To make process mining a supervised learning problem suitable for classification, we propose to include negative information in the event log in the form of negative events. A **negative event** reports that a state transition could not take place. For each positive activity event type one can think of a negative one. For instance, for the event types `created` and `assigned` the event types `createRejected` and `assignRejected` can be conceived. Learning the classification rules that predict whether, given the state of a process instance, a particular state transition can occur, then boils down to learning the classification rule that predicts when either a positive or a negative event occurs. In this way, we have formulated process mining tasks such as Process Discovery and authorization rule learning as classification problems.

Sometimes, process logs naturally contain negative events. An access log, for instance, contains information about the workers that have obtained authorization, and information about the workers who were refused authorization to perform a particular task. In many cases, however, information systems do not reveal their internal functioning in terms of negative events. For instance, when a WfMS creates a number of work items and assigns them to several work trays, it will not expose the work items it did not create or provide information about the work trays to which it could not allocate a work item.

Negative examples can be introduced by replaying the positive events of each process instance event trace t_i and by checking whether a state transition of interest ϵ could occur. At each event $e_{(i,k)} \in t_i$, it is tested for each possible activity state transition of interest ϵ whether there exists up to that point k similar traces $t_j : \forall l, l < k, similar(e_{(i,l)}, e_{(j,l)})$ in the event log in which at that point a state transition $e_{(j,k)}$ has taken place that is similar to ϵ, as denoted by a similarity operator $similar(e_{(j,k)}, \epsilon)$. If such a state transition does not occur in similar traces, this is an indication that the state transition should be prevented from occurring. Consequently, a negative event can be added at this point k in the event trace t_i. On the other hand, if a similar trace is found in which the state transition ϵ does occur, this behavior is present in the event log and no negative event is generated. More formally, this process of adding negative examples can be described as follows:

```
1  For each process instance $t_i$ in the event log
2     For each event $e_{(i,k)}$ in $t_i$
3        For each activity state transition $\epsilon$ of interest
4           if $\nexists\ t_j : \forall l, l < k, similar(e_{(i,l)}, e_{(j,l)}) \wedge similar(\epsilon, e_{(j,k)})$
5           then recordNegativeEvent($t_i$,k,$\epsilon$,$\pi$)
```

To avoid an imbalance in the proportion of negative versus positive events the addition of negative events can be manipulated with a negative event injection probability π. In the above pseudo code it is possible that $i = j$. Evidently, t_i is similar to itself. If at point k in t_i a state transition of interest ϵ occurs, this provides enough evidence not to include ϵ as a negative event in t_i.

The procedure for injecting negative events does not supplement the event log with noisy negative events in the presence of noisy positive ones. Noise represents additional, low-frequent behavior that originates from log errors or the occurrence of exceptions outside the scope of the process mining task. Although compared to noise-free logs, less negative events are added because of the additional noisy behavior, our negative event induction technique does not lead to the addition of noisy negative events.

Supplementing event logs with artificial negative events, adds the completeness assumption [3] that all possible trajectories in the process model to be learned have corresponding process instances in the event log. Formulated differently, adding artificial negative examples to an event log on which later on classification is performed, forces a classification learner to conclude that trajectories that do not occur in the original event log, should not occur in the induced process model. This is a much desired property, as it is the intention of process mining to induce a process model that only portrays the behavior in the event log. For instance, when mining a control-flow model for the purpose of delta-analysis, the induced process model should preferably cover all the presented process instances, but no more than the presented ones. Without this completeness assumption as inductive bias, it is not possible to learn from positive examples only as any model in which the preconditions impose no constraints would cover all process instances in the event log. The completeness assumption also results in requiring a large number of process instances. This is particularly the case when the underlying process model contains a lot of concurrent (parallel) activities. A possible solution is to limit the number of possible activity events in the log, for instance, by only considering activity `completed` events. Another solution is to to leave out or regroup a number of concurrent tasks in the event log. In order to capture the behavior in the event log, this behavior must be present in the event log. The requirement for sufficient data is similar to the data requirements in any other process mining technique.

In the above procedure, similarity is a relative notion that depends on the learning task at hand. Three factors play a role in determining whether two process instances t_i and t_j are similar: the contents of the event traces t_i, the definition of the event similarity operator $similar(e_{(i,l)}, e_{(j,l)})$ and the depth k. It is important to notice that the event traces t_i should only include those events that are relevant to the learning task at hand. For instance, when learning sequence constraints among activities, the events involving scheduling, assignment and data manipulation are likely to be left out of consideration with regard to the induction of negative events. Likewise, the event similarity operator $similar(e_{(i,l)}, e_{(j,l)})$ is task-dependent. For instance, in the context of Process Discovery it might be acceptable to say that the events

```
event(act92,applyForLicense,driver1,completed,driver2,[],17)
event(act403,applyForLicense,driver3,completed,driver3,[],17)
```

are similar. However, in the context of authorization rule learning, this might not be the case, as in the first event the activity is completed by a different agent (`driver2`) than indicated by the business id (`driver1`) of the case. The depth k is another factor in determining whether two process instances are similar. Arguably, the above procedure, requiring two process instances to have exactly the same trace up to point k, imposes too strong of a similarity requirement. In some cases, it might be sufficient to have particular chunks (part of a loop) of the traces in common. This can be solved by a careful preprocessing of the event log or by an adaptation of the procedure. Both adaptations are likely to be problem dependent.

4 Process Discovery as Learning Preconditions

Process Discovery involves the discovery of the process control flow from event logs [3,4] and has been the main focus of process mining. Several algorithms have been proposed, such as the α [3] algorithm and a genetic algorithm [4]. In this section, we make use of Tilde [12], a first-order decision tree learner available in the ACE-ilProlog data mining system [13]. Tilde is a first-order generalization of the well-known C4.5 algorithm for decision tree induction [14]. Like C4.5, **Tilde** [12,13] obtains classification rules by recursively partitioning the dataset according to logical conditions that can be represented as nodes in a tree. This top-down induction of logical decision trees (Tilde) is driven by refining the node criteria according to the provided language bias $\mathcal{L}$. Unlike C4.5, Tilde is capable of inducing first-order logical decision trees (FOLDT). A FOLDT is a tree that holds logical formula containing variables instead of propositions. Below this paragraph, a decision tree is depicted that represents the learned precondition of the activity `receiveLicense`. To learn such preconditions, the language bias $\mathcal{L}$ of Tilde was restricted to the `ns(AT1,AT2,BId,Now)` event operator $\mathcal{NS}$ and aggregate operator `count(AT,BId,Now)` that counts the number of occurrences of an activity type within a specific process instance. Tilde's C4.5 gain ratio was used as a heuristic for selecting the best branching criterion. In addition, Tilde's C4.5 post pruning method was used with a standard confidence level of 0.25.

```
canStartReceiveLicense(BId,Time,-C)
ns(getResult,applyForLicense,BId,Time) ?
+--yes: ns(getResult,receiveLicense,BId,Time) ?
|       +--yes: ns(getResult,end,BId,Time) ?
|       |        +--yes: [started]
|       |        +--no:  [startRejected]
|       +--no:  [startRejected]
+--no:  [startRejected]
```

The artificial event log was generated with 450 process instances from the process model in Fig. 1(a) with a maximum of three allowed loops. As is common, learning was performed on a training set, whereas the reported performance was done on the test set (out-of-sample performance), as to provide an objective measure for the predictive performance on new, unseen examples. The **test log** is created as follows. The entire event log, consisting of about 7300 activity `completed` events is first supplemented with about 7000 negative `completeRejected` events by applying the above described procedure with a negative event injection probability π of 100%. After this procedure the first 350 process instances (the first 350 drivers) are removed from event log to retain a test set of 100 process instances.

To correctly evaluate the proposed learning technique, it is important that the negative events in the test set accurately indicate the state transitions that are not present in the event log. For this reason, the negative events in the test log are created with information from the entire event log. Should the negative event injection procedure be applied on the 100 selected process instances only, it is possible that additional, erroneous negative events are injected because it is possible that some behavior is not present in the test set. To avoid that the injected negative events in the test set become dependent on the sampling policy used, the negative events in the test set are generated using all information in the event log. The same procedure cannot be applied to come up with the training log.

The **training log** is composed of the first 350 process instances. The log consisting of some 5300 `completed` events was supplemented with some 4400 negative `completeRejected` events on the basis of training log events only. To test the performance of first-order activity precondition learning under noise, the training set has been modified with different types of noise. After adding noise, the noisy training sets were supplemented with negative events also with a negative event injection probability π of 10%. Alves de Medeiros et al. describe six noise types [4, p. 41]: *missing head*, *missing body*, *missing tail*, *swap tasks*, *remove task*, and *mix all*. For reasons of brevity we report performance results with *swap tasks*, identified as being the most difficult [4], and *mix all*, which a combination of all other noise types. The used noise levels of 10% and 30% are higher than the 5% and 10% levels reported in [4].

In Process Discovery it is important that the discovered preconditions allow almost every event trace in the log (completeness) but preferable no more event traces that do not occur in the log (preciseness) [4]. Rather than using accuracy as a performance measure, we therefore propose two performance measures that are more suitable to the problem domain of Process Discovery:

- **true positive rate** TP or **completeness**: the frequency of correctly classified positive events in the test set. This probability can be estimated as follows: $TP = E^{+}_{positive}/E^{total}_{positive}$, where $E^{+}_{positive}$ is the amount of correctly classified positive events and $E^{total}_{positive}$ is the total amount of positive events.
- **true negative rate** TN or **preciseness**: the frequency of correctly classified negative events in the test set. This probability can be estimated as follows: $TN = E^{-}_{negative}/E^{total}_{negative}$, where $E^{-}_{negative}$ is the amount of correctly classified negative events and $E^{total}_{negative}$ is the total amount of negative events.

Notice that the true negative rate gives an accurate idea of the preciseness of the learned precondition as negative events are precisely representatives for traces that are not in the sample log. In Table 1 we report these evaluation measures for each precondition learned under different noise circumstances.

Under zero noise conditions, one can observe perfect completeness and preciseness each activity precondition in Table 1. However, rather than favoring local preconditions the decision tree induction algorithm often favors preconditions with immediate discriminating power. For the moment, this non-preference for local conditions complicates the construction of a graphical model from the learned preconditions. Under conditions of noise, it is observed with regard to the completeness criterion that every induced precondition portrays a perfect recall

Table 1. Out-of-sample performance of the learned preconditions. Both completeness TP and preciseness TN is given as in the following pattern: TP;TN.

Activity Type		**Noise Type**				
		no noise	*10% mix all*	*10% swap task*	*30% mix all*	*30% swap task*
A_0	start	1.00;1.00	1.00;1.00	1.00;1.00	1.00;1.00	1.00;1.00
A_1	applyForLicense	1.00;1.00	1.00;0.91	1.00;0.91	1.00;0.91	1.00;0.91
A_2	attendClassesDriveCars	1.00;1.00	1.00;1.00	1.00;1.00	1.00;1.00	1.00;0.83
A_3	attendClassesMotorBikes	1.00;1.00	1.00;1.00	1.00;1.00	1.00;1.00	1.00;0.83
A_4	doTheoreticalExam	1.00;1.00	1.00;1.00	1.00;1.00	1.00;0.91	1.00;0.82
A_5	doPracticalExamDriveCars	1.00;1.00	1.00;1.00	1.00;1.00	1.00;0.92	1.00;1.00
A_6	doPracticalExamMotorBikes	1.00;1.00	1.00;1.00	1.00;1.00	1.00;0.92	1.00;1.00
A_7	getResult	1.00;1.00	1.00;1.00	1.00;0.83	1.00;0.92	1.00;0.83
A_8	receiveLicense	1.00;1.00	1.00;1.00	1.00;0.91	1.00;1.00	1.00;0.82
A_9	obtainSepcialInsurance	1.00;1.00	1.00;0.91	1.00;1.00	1.00;0.72	1.00;0.69
A_{10}	end	1.00;1.00	1.00;1.00	1.00;1.00	1.00;1.00	1.00;0.82

of the positive events. With respect to the preciseness criterion it is observed that the preconditions relax, allowing negative events to take place and thus scoring lower on the preciseness criterion. For example, under 30% swap tasks noise, the induced activity precondition for the parallel task `obtainSpecialInsurance` deteriorates to 0.69, indicating that 31% of the identified negative events are not classified correctly. The reason is that the extra behavior that is introduced by the added noise has also in part been included in the preconditions. 30% is nonetheless a high noise level, and it can be seen that under the 10% noise level (the highest noise level reported in [4]) the learned preconditions still have an almost perfect recall of the negative events. This robustness to noise can be attributed to the robustness of Tilde's C4.5 tree induction algorithm.

The eleven preconditions could always be learned in under half an hour. In general, first-order classification problems potentially have an extremely large search space. However, we have tried to limit the hypothesis space by limiting the language bias $\mathcal{L}$ to the two aforementioned language constructs. The greedy search strategy performed by Tilde's C4.5 top down induction of decision trees also contributes to this computational efficiency result.

5 Related Work

Process mining can be seen as an application of the machine learning of grammars [15,16]. Gold has shown that important classes of recursively enumerable languages cannot be learnt from positive examples only [15]. Instead, a complete presentation of both positive and negative examples is required for grammar learning to distinguish from an infinite number of grammars that fit the positive examples. In grammar learning, the hypothesis space is often expressed as production rules, automata or regular expressions. In this paper, however, a different hypothesis space is used. Moreover, the possibility of noise is taken into account.

Several authors have represented process mining as classification learning. For instance, Maruster et al. [10] were among the first to investigate the use of rule-induction for Process Discovery. The authors use propositional rule induction techniques on a table of direct metrics for each process task in relation to the other process tasks, which is generated in a pre-processing step. This transformation is needed to deal with the absence of negative examples and to use the uni-relational classification learner RIPPER [17]. In contrast, the multi-relational nature of first-order classification learners allows to directly perform classification on the event log and is capable of dealing with non-local dependencies. Rozinat et al. [9] discuss the use of uni-relational classification for the purpose of "decision mining". In decision mining so-called decision points are semi-automatically identified in process logs, and the classification problem consists of determining which case data properties lead to taking certain paths in the processes.

Ferreira and Ferreira apply a combination of ILP learning and partial-order planning techniques to process mining [18]. Rather than generating artificial negative events, negative examples are collected from the users who indicate whether a proposed execution plan is feasible or not. By iteratively combining planning and learning, a process model is discovered that is represented in terms of the case data preconditions and effects of its activities. In addition to this new process mining technique, the contribution of this work is in the truly integrated BPM life cycle of process generation, execution, re-planning and learning. Alves de Medeiros et al. [4] point out the difficulties that process mining algorithms have when only taking into account local information. The authors have implemented a genetic algorithm for Process Discovery. No negative examples are introduced, but this problem is circumvented by the incorporation of both a completeness and preciseness measure in the fitness function that drives the genetic algorithm towards suitable models.

6 Conclusion

In this paper we have demonstrated the feasibility of process mining as a first-order classification learning on event logs supplemented with artificial negative events. A first set of Process Discovery experiments has shown promising results on a non-trivial learning problem with loop, parallelism and non-local non-free choice constructs. In the experiment without noise, a model was discovered with perfect completeness and preciseness, indicating the suitability of the proposed language bias for Process Discovery. Additional experiments have indicated the technique to be robust to noise. With this paper we certainly do not claim to have solved all the important problems, but we think to have pointed out the potential of applying the techniques of first-order learners and negative event induction to process mining in general.

References

1. van der Aalst, W.M.P., van Dongen, B.F., Herbst, J., Maruster, L., Schimm, G., Weijters, A.J.M.M.: Workflow mining: A survey of issues and approaches. Data & Knowledge Engineering 47(2), 237–267 (2003)
2. van der Aalst, W., Reijers, H., Weijters, A., van Dongen, B., de Medeiros, A.A., Song, M., Verbeek, H.: Business process mining: An industrial application. Information Systems 32(5), 713–732 (2007)
3. van der Aalst, W., Weijters, A., Maruster, L.: Workflow mining: Discovering process models from event logs. IEEE Transactions on Knowledge and Data Engineering 16(9), 1128–1142 (2004)
4. de Medeiros, A.A., Weijters, A.J., Aalst, W.M.: Genetic process mining: An experimental evaluation. Data Mining and Knowledge Discovery 14(2), 245–304 (2007)
5. Ly, L.T., Rinderle, S., Dadam, P., Reichert, M.: Mining staff assignment rules from event-based data. In: Bussler, C., Haller, A. (eds.) BPM 2005. LNCS, vol. 3812, pp. 177–190. Springer, Heidelberg (2006)
6. van der Aalst, W., Reijers, H., Song, M.: Discovering social networks from event logs. Computer Supported Cooperative Work 14(6), 549–593 (2005)
7. Baesens, B., Van Gestel, T., Viaene, S., Stepanova, M., Suykens, J., Vanthienen, J.: Benchmarking state-of-the-art classification algorithms for credit scoring. Journal of the Operational Research Society 54(6), 627–635 (2003)
8. Martens, D., Baesens, B., Gestel, T.V., Vanthienen, J.: Comprehensible credit scoring models using rule extraction from support vector machines. European Journal of Operational Research 183(3), 1466–1476 (2007)
9. Rozinat, A., van der Aalst, W.M.P.: Decision mining in ProM. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102, pp. 420–425. Springer, Heidelberg (2006)
10. Maruster, L., Weijters, A.J.M.M., van der Aalst, W.M.P., van den Bosch, A.: A rule-based approach for process discovery: Dealing with noise and imbalance in process logs. Data Mining and Knowledge Discovery 13(1), 67–87 (2006)
11. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Addison-Wesley, Reading (2005)
12. Blockeel, H., De Raedt, L.: Top-down induction of first-order logical decision trees. Artificial Intelligence 101(1-2), 285–297 (1998)
13. Blockeel, H., Dehaspe, L., Demoen, B., Janssens, G., Ramon, J., Vandecasteele, H.: Improving the efficiency of inductive logic programming through the use of query packs. Journal of Artificial Intelligence Research 16, 135–166 (2002)
14. Quinlan, J.R.: C4.5: programs for machine learning. Morgan Kaufmann Publishers Inc., San Francisco, CA (1993)
15. Gold, E.M.: Language identification in the limit. Information and Control 10(5), 447–474 (1967)
16. Angluin, D.: Inductive inference of formal languages from positive data. Information and Control 45(2), 117–135 (1980)
17. Cohen, W.: Fast effective rule induction. In: Prieditis, A., Russell, S. (eds.) Proceedings of the 12th International Conference on Machine Learning, Tahoe City, CA, pp. 115–123. Morgan Kaufmann, San Francisco (1995)
18. Ferreira, H., Ferreira, D.: An integrated life cycle for workflow management based on learning and planning. International Journal of Cooperative Information Systems 15(4), 485–505 (2006)

Modeling Alternatives in Exception Executions

Mati Golani[1], Avigdor Gal[2], and Eran Toch[2]

[1] Ort Braude College, Israel
matig@braude.ac.il
[2] Technion - Israel Institute of Technology
avigal@ie.technion.ac.il, erant@techunix.technion.ac.il

Abstract. To date, the ability of a business process designer to produce a solid, well-validated workflow models is limited, especially since all necessary scenarios that need to be covered by the workflow are hard to predict. Workflow management systems (WfMSs), serving as the main vehicle of business process execution, should recognize those limits, and increase its support to designers in this task. One aspect of such assistance is in exception handlers generation. In this paper we propose a model language enrichment for expressing workflow semantics, in the context of alternative solutions, within the process model. Thus, enabling the designer to state which possible alternatives and their applicability to changing execution paths states. Using this enrichment, an inference algorithm can efficiently find an adequate alternative. The model language is used as a basis for a design tool and an execution environment, which semi-automatically generates exception handlers, resulting, due to a reduced search space, in a smaller set of exceptions for the designer/user to choose from.

1 Introduction

The ability of a business process designer to produce solid, well-validated workflow models is limited as it is difficult to predict all various scenarios, needed for the process model. Workflow management systems (WfMSs), serving as the main vehicle of business process execution, should recognize these limits and become more case oriented, assisting the designers in this task. One aspect of such assistance is in exception handlers generation. To illustrate this point, we next present an example, involving Web services. First, we observe that there is no existing mechanism that can solve the designer's effort in modeling alternative paths. Web services, for example, merely provide syntactic information regarding their input, output and processing logic, through standards such as WSDL [18]. Usually, such descriptions fail to convey constraints and restrictions. Web services choreography does not refer to alternative paths affinity. This means that we cannot tell whether one path can be executed as a result of exception, while some Web service was invoked on an alternative one .Modeling using Web services, therefore, is likely to make the validation of workflow models more difficult [8], and more exceptions at run-time are to be expected. Efficient exception handling is a fundamental component of WfMSs and is critical to their successful implementation in real-world scenarios [2,10] and has great impact on system performance.

Designing efficient exception handlers – specifications of exception handling processes – is not a simple task. By their very nature, exceptions are rare events that do not enjoy the advantages of common processes, which are easily programmed with much expert information injected into them. Thus, exception handlers may well be ill-designed, affecting both the correctness and the efficiency of the process. Avoiding their design altogether is also not a valid option. During runtime, process operators observe a narrow perspective of the process, and given an exception, will not have sufficient information to effectively manage it. In those cases in which the operator mitigates the exception, the solution may be neither optimal nor effective.

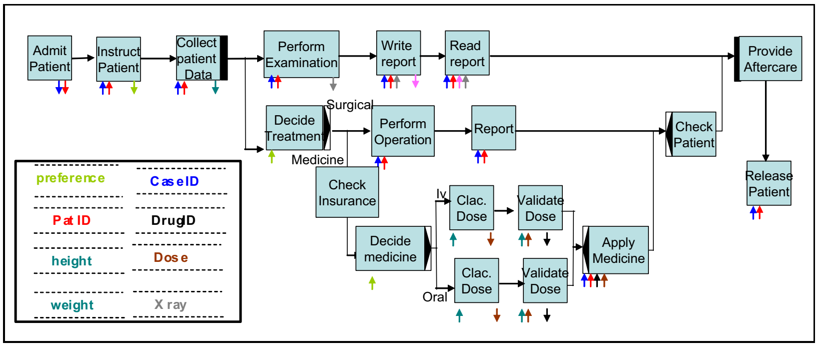


Fig. 1. A medical process case study

Consider Figure 1 that presents a scenario of a medical care within a hospital. In this process there are two points of decision making (namely *"Decide treatment"*, and *"Decide medicine"*). In first one, the doctor decide whether a surgery is required, or a medicine treatment is sufficient. The second activity refers to the type of medicine treatment (oral vs. iv). The conservative approach semantics means, that the various options are mutual exclusive. Suppose that the designer is introduced with some more knowledge from the domain experts (i.e. doctors):

– If an operation begun or completed, in case of some exception (reduction in the patient stability), the path of the medicine treatment can be taken.
– After a medicine was taken, performing a surgery is forbidden. If a decision to take a medicine was taken, there is still an option of performing an operation on emergency cases unless the medicine was actually applied.
– Until the medicine is applied, one can switch - in case of short inventory-from oral to iv and vice versa.

This knowledge is, needless to say, not intuitive, but rather quite complicated to model using the conservative constructs. However it looks like a classical case for alternative path exception handler (i.e. rollback and execute another direction from a visited Xor split activity). Therefore, there is a demand for an extended model that considers such alternatives, of which some are state based. Such a model can assist the designer in generating more accurate exception handlers, and a whole more accurate process model.

In this paper we propose a model enrichment for expressing alternative execution paths, using real world semantics. By augmenting the process model with additional meta-data, the designer is capable of stating applicable alternative paths, with respect to changing world states and conditions. Once the process model is defined, a tool may use the additional information in order to assist both the designer during design-time, or the user during runtime, with relevant exception handlers. This extension provides a more complete reflection of reality. When referring to auto-generated exception handlers, it reduces the number of illogical solutions, and generates a smaller set of exceptions for the designer/user to choose from. Our objective is to minimize the necessary user intervention when handling an exception in the business process execution. Therefore, we suggest methods to use the world knowledge and process model to reduce the decision space of the user.

The rest of the paper is organized as follows. In Section 2, we provide the preliminary constructs on which we base our model. Then, Section 3 presents the extension to the model, to address the relations and interactions between alternatives. Section 4 introduces the various semantic options between alternatives in a Xor split activity, such as the *conservative mutual Xor*, *asymmetric alternative path execution*, and *conditional alternative path execution*. We also discuss the quality metrics in terms of a minimal change. In Section 5 we give an illustrated example. Section 6 reviews related work on this topic, and finally we conclude in Section 7.

2 Preliminaries

There are various ways to describe a workflow process model. Workflows define a business process in terms of *activities* (also called actions or tasks). Activities, together with temporal constraints on execution ordering define a business process [17]. A workflow model can be graphically described as a workflow net [1] for example, or ADEPT WSM net [16]. A WSM net $G(V,E)$ ($V = (V_a \cup V_d)$; $E = (E_c \cup E_d \cup E_s)$), combined of activity and data nodes, connected by control, data, and synchronization edges. Synchronizing edge are quite useful, allowing activities on parallel threads to be synchronized, even though they are not connected with control edges.

Given a workflow graph $G(V,E)$ we define for an activity $a \in V_a$, *the Nearest Xor split point of a* (denoted $NXSP(a)$) as the nearest Xor activity that provides an *alternative path* execution for a (path of execution that avoids a_i) [10]. In what follows, and following the WFMC definition (in interface 1) of *full-blocked workflows,* we assume the use of well structured processes.

2.1 Exception Handling

An *exception handler* is a workflow $X(V_X, E_X)$, executed in response to an occurrence of an exception for which it was defined. Given a workflow $G(V, E)$ and an exception handler $X(V_X, E_X)$, we define an operator *Apply* such that by applying $X(V_X, E_X)$ to $G(V, E)$ one receives a revised workflow model

$$G'(V', E') = Apply(G(V, E), X(V_X, E_X), v_s, V_e),$$

where $\{v_s\} \cup V_e \subseteq V$. v_s specifies the failing node in G and V_e is a set of nodes in G from which the normal operation of G will resume [15]. An exception handler can be schematically partitioned into two sections, namely *rollback* and *forward stepping*. A *rollback* section executes compensating activities and a *forward stepping* section activates and reactivates activities. For each exception handler we can define three reference activities. We denote by a_{sr} a *start activity*, the failing activity. a_{sp} is a stop activity, the activity where the control is returned to the original process. Finally, a target activity (a_{tr}) is the activity where the rollback section ends.

There are three types of activities an exception handler can use. The first type can activate activities in the workflow (for the first time), or reactivate them. The second type invokes *compensation* activities, also known as *undo* activities and *semantic rollback* activities [11,9]. A compensating activity needs to be predefined, is associated with a single or combined set of workflow activities, and is typically used for reversing the impact of activities that were already performed for a given instance. Lastly, an exception handler can use activities that are not defined in the workflow altogether.

In [15] two types of exception handlers were presented. A *repeat activation exception handler*. Such an exception handler attempts to repeat the activation of a subgraph of a workflow model by first applying compensating activities to the part that was already activated, followed by reactivation of activities. The second type is denoted an *alternative path exception handler* which was first introduced in [10]. Alternative path exception handlers combine the use of compensating activities, reactivation of activities and first-time activity activation. For an alternative path exception handler a_{sp} is always a *Xor split point* which is a direct or predecessor *NXSP* for the referred activity [10].

Another distinction is between *actual* and *logical* execution in an exception handler. An actual activation of an activity a involves the invocation of a routine associated with a or performing a new work item in an item list of some role in the organization. A logical activation of a requires only recording its activation in the WfMS without actually activating it. $C(a)$ is set to 0 whenever a requires only a logical activation. If an actual activation is involved, then $C(a)$ is assigned with its full cost. The cost of exception handler X is the sum of costs of all activities in X.

3 Modeling Alternative Flows

In [10] it was stated that the user may be invoked with too many and infeasible alternatives, while a relaxation can somehow address this issue. A process model reflects real world context. Exception, by its nature, is context related. In the following section, we extend and study further the model semantic context in order to minimize the user invocation, and infer better and faster exception handling solutions. This is done by a classification of the process model context. Each Xor split activity is augmented with additional meta-data, enabling the modeling of the following classification.

- *Global class (Stateless)* - can be evaluated by inspecting the model regardless a specific instance data.
- *Instance based class* - requires instance data for exception feasibility evaluation
  - *Direction related* - dependent on the actual path taken from the Xor split activity
  - *State related* - dependent on the non/activation of a specific activity in the path taken from the Xor split activity.

We continue by using a predicate model as a formal extension meta-data to the process model in order to enrich the process language. Each decision edge e_{ij} has the following attached structure: $Dec = \langle \Phi_{ij}, I_{ij}, c_{ij} \rangle$.

- $I = \{\iota_1, \iota_2, \ldots\}$ - a set of invariants, one for each $e_{ik} : k \neq j$ in a CNF structure for evaluating ι_{ik} as a condition for executing e_{ij} as an alternative path.
- $\Phi_{ij} = p_1 \wedge p_2 \wedge \ldots$ - a logical formula in CNF structure representing the routing criteria decision.
- $c : I \rightarrow price$ - a pricing function.

W - a CNF structured formula $p_1^w \wedge p_2^w \wedge \ldots$that represents the world's state. $\forall \{parameter, operator, value\} \in p_i^w : parameter \in V_d$. each valid state W is evaluated to *true*. W used to determine the solution space for which a formula composed from Φ and I, is evaluated for available solutions, to be discussed in Section 4.

4 Alternative Related Knowledge Modeling

Given a workflow process G that includes some Or split activity a_x, the semantic reasoning of the process domain may provide us with different meaning of relating to this activity a_x. The conservative approach claims that the two Xor activity branches are mutually exclusive. That means that if the process instance took one branch, it cannot take the other. Consider the example in Figure 2, and the Xor split activity 3. The actual branching direction is taken according to the value of c. We also assume that this value is not system dependent but rather a user input (e.g. a controllable choice [4]). An example for such a case may be a medical patient registration that includes different pregnancy tests, while activity 3 checks the pregnancy status of the patient. In this case there is no real alternative meaning for the two directions. Either the patient is pregnant or not.

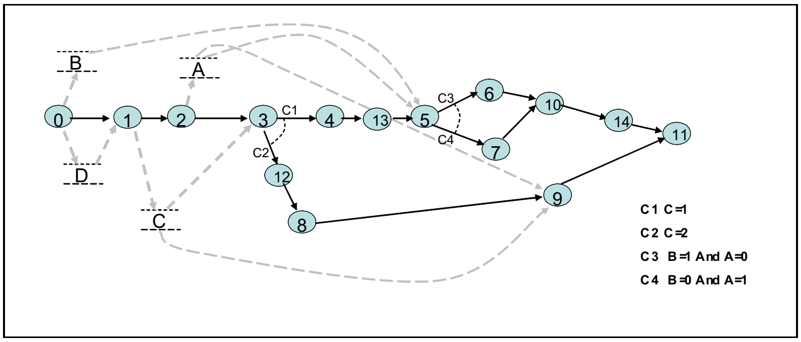


Fig. 2. Mutual exclusive split analysis example

In that case, $\exists i \in I_{3-4} : i == \Phi_{3-12} = c2$. The compound formula is $c1 \wedge c2 = (c = 1) \wedge (c = 2)$ which is evaluated to $false$. Thus, for a given a world state W, and an execution that was carried out using e_{i-j}, there is no solution $W^{desired}$ that satisfies the execution via e_{i-k}. Formally put:

$$W \Rightarrow \nexists W^{desired} \models (\iota_{i-j} \wedge \Phi_{i-k}) \tag{1}$$

$$W \Rightarrow \nexists W^{desired} \models (\iota_{i-k} \wedge \Phi_{i-j})$$

The $\wedge$ operator is used in order to produce a formula that takes the values in the original path (left hand side proposition i.e. ι_{i-j}) but ensures its validity on the alternative one (right hand side proposition).

A more forgiving option is providing an execution over the alternative path [10]. Here, using the *alternative path exception handler*, though the process has been executed on one path in can be compensated and continue over another path. in this case the ι referring to the other path is an empty set. Referring again to our example, that means, that while executing activity 12 the process may in some cases rollback to activity 3 and continue its execution in the path going through activity 4. An example for such a case is an ordering process, were the Xor split differ between gold customer and regular customer. In case of some malfunction in the gold customer path, the order can be always executed as a regular customer (given the inheritance nature with some degree of compensation).

Formally put

$$W \Rightarrow \exists W^{desired} \models (\iota_{i-k} \wedge \Phi_{i-j}) = F$$

And in our example there is a solution $W^{desired}$ which satisfies F.

$$\left[\iota_{3-12} : (\iota_{3-12} \in I_{3-4})\right] = \varnothing \Rightarrow \iota_{3-12} \wedge \Phi_{3-4} = \Phi_{3-4} = \{(c = 2)\} = F \tag{2}$$

$$\left\{W^{desired} = (c = 2)\right\} \models F$$

4.1 Asymmetric Alternative Path Execution

We would like to draw the readers' attention that the above formula does not state just the routing conditions (as appear in Figure 2), but also the set of propositions that can hold along the left side propositions set. Practically this means, that a process with original value of $c = 2$ may be executed on e_{3-4} since modification of c in this direction is allowed.

On the other hand, in our scenario, the opposite direction is not valid (i.e. a regular customer treated as a gold one). We can enforce it by

$$(\iota_{3-4} \in I_{3-12}) \wedge (\Phi_{3-12}) \overset{allways}{=} false$$

Formally put

$$\begin{array}{c} (\iota_{i-j} \in I_{i-k}) \wedge \Phi_{i-k} \overset{always}{=} false \\ (\iota_{i-k} \in I_{i-j}) \wedge \Phi_{i-j} = true \end{array} \tag{3}$$

The usage of Equation 3 is presented in Algorithm 1.

Algorithm 1. Semantic inference execution algorithm

```
1:  Input: Graph $G'$, $Inst(G)$ -Instantiation, $a_j$ -alternative path candidate activity.
2:  Output: $approval$ - a boolean value map that represents the approval for each
    alternative.
3:  Process:
4:  //execute over the possible alternative paths from $a_j$
5:  set $e_{j,k}$ to be the activated edge on the original process
6:  for each $e_{j,l}$ where $e_{j,l}$.history=false do
7:     if $\iota_{e_{j,k}} \wedge \Phi_{j,l}$ then
8:        set $approval(e_{j,l})$=true
9:     else
          set $approval(e_{j,l})$=false
10:    end if
11: end for
12: return $approval$
```

4.2 Conditional-Alternative Path Execution

Consider a case were a certain path may rolled back unless some specific activity is already executed or completed. Taking for example a chemical analysis execution process, which involves providing a sample (a_1), declaring the type of analysis (a_2 type Xor split directing to a_3 with condition c or to activity a_{10} with condition $\bar{c}$), preparing the analysis kit (a_3), and performing the analysis on the sample (a_4). Suppose that the given activities describes a specific test (e.g. Gas chromatography) that involves some modifications on the sample in activity a_4 (for preparations to the test). In a case of a desired rollback for executing another analysis, if activity a_4 is already executed, then the sample was modified. Thus, the alternative path could not be taken.

On the contrary case, where the decision is taken while activity a_3 is active, the rollback is valid and may be taken. This can be modeled as a state predicate within the propositions. This predicate is evaluated during runtime. In order to satisfy this constraint the following clause in $\iota_{e_{2-3}}$ should hold

$$c \implies (a4.activated \ \vee \ a4.completed \ \vee \ a4.aborted)$$
$$a4.activated \implies a4.active$$
$$a4.completed \implies a4.active$$
$$a4.aborted \implies a4.active$$

For simplicity we classify every activity that was activated during the execution as *active*, which brings us to a shorter form:

$$c \implies a4.active$$

Using first order logic equivalence rules

$$c \vee \neg(a4.active)$$

Since we apply the conjunction of ι and Φ we get in resultl

$$\bar{c} \wedge (c \vee \neg(a4.active)) = \overbrace{(\bar{c} \wedge c)}^{AlwaysFalse} \vee (\bar{c} \wedge \neg(a4.active))$$

Obviously, the first section is always evaluated to false. Therefore, the second section should be always evaluated to true in order to permit a solution $W^{desired}$ which executes along the alternative path. This means that the activation of activity a_4 is not allowed

For a given set $\iota_{e_{i-j}}$,

$$\forall\ (\neg a_k.active) \in \iota_{e_{i-j}} : a_k \ was\ not\ activated \implies approval(e_{ij}) = true$$

For simplicity each clause p in I is fragmented into two fragments a model fragment f_m, and a runtime (instance) fragment f_i, where $f = f_m \wedge f_i$. in the above example $f_i = (\neg a4.active)$

$$\exists W^{desired} \models f_m(p_x \in \iota_{_{i-j}}) \wedge \Phi_{_{i-k}} \tag{4}$$

Equation 4 is a mandatory condition but not a satisfactory one for using the $i-j$ direction as an exception handler. The satisfactory condition contains in addition

$$W^{desired} \models f_i(p_x \in \iota_{_{i-j}}) \tag{5}$$

4.3 Model Extension Usage

Figure 3 presents a general overview referring to the interaction of the designer and the user during the various stages of the model generation and extension.

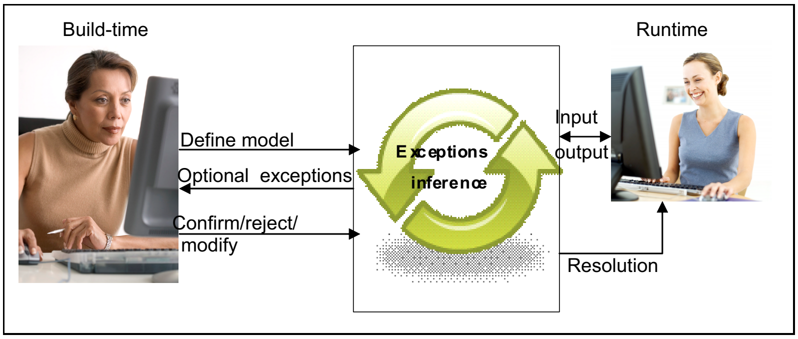


Fig. 3. User/Designer high level interaction

It starts by the conservative model definition (i.e. $G(V, E)$ and Φ). Then, the designer continue and define the extended data (i.e. I_{jk}). Once the model is completed, exceptions can be suggested to the designer. Each exception can be either confirmed or rejected. This exception is available later on for the users during runtime.

Figure 4 takes a closer look on the activities taken by the designer. Once the designer finishes defining the initial process model, he refers to previously defined constructs (e.g. activities and conditions) while creating the invariants for this model. During this stage, the designer may became aware of wrong assumptions taken previously, resulting in model refinement. Once the model is refined, exception handlers can be inferred. A valid exception handler is one that for a given world state W can provide a modified world $W^{desired}$ which satisfies a formula F that includes the relevant I and Φ segments.

4.4 Discussion

Once an alternative path has been established, the procedure proposed in [10] involves identifying the data items that need to be modified to allow the workflow instance to use the alternative path (dubbed, heretoafter, *the change set*). For each such data item, we need to trace back the activity in which it was modified and request to modify the data item value to the new value. It would be desirable to find the minimal change set for two main reasons. First, our aim is to avoid, to the extent possible, performing redundant extra work. Therefore, changing a single data item is preferred over changing two data items to turn the condition on the outgoing edge of the Xor node of the alternative path to be true. Second, if we are able to compute the minimal *change set*, we can use this information to optimize our efforts to recover from an exception. This way, we can compare the "cost" of various alternative paths, rank them in an increasing order of their cost, and try them one by one.

Although finding the minimal change set is desirable, it is not an easy task. In fact, finding the minimal change set is an NP-complete problem [3]. This can be shown by performing a reduction from the minimal set cover problem. Therefore,

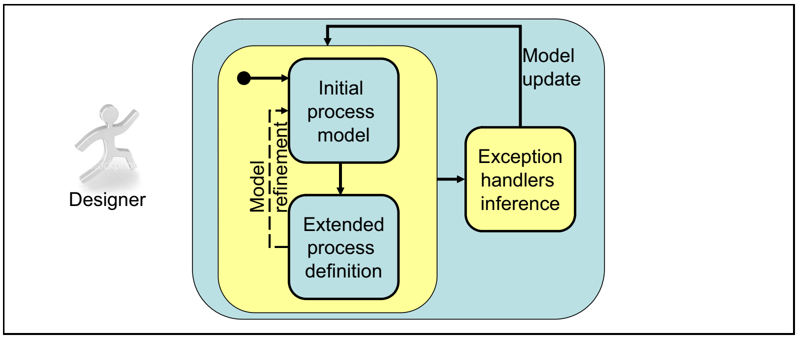


Fig. 4. Activity diagram of the suggested solution

in our future work we shall attempt to identify good heuristics that take into account the semantics of workflows to improve the solution performance.

5 Illustrating Example

Returning to Figure 1, we next show how to model the extended requirements presented in Section 1. For simplicity we tag some of the activities with numbers: 1=decide_treatment; 2=check_ins; 3=decide_medicine; 4=IV_calcDosg; 5=Oral_calcDosg; 6=apply_med; 7=perform_op. In our example: $\Phi_{17} = \{t = surgical\}, \Phi_{12} = \{t = medicine\}$ $\Phi_{34} = \{m = IV\}, \Phi_{35} = \{m = Oral\}$.

Now, let us refer to the given requirements.

– If an operation begun or completed, in case of some exception (reduction in the patient stability), the path of the medicine treatment can be taken.

$I_{12} = \varnothing$

– After a medicine was taken, performing a surgery is forbidden. If a decision to take a medicine was taken, there is still an option of performing an operation on emergency cases unless the medicine was actually applied.

$I_{17} = \{(t = medicine) \vee \neg(6.active)\}$

– Until the medicine is applied, one can switch - in case of short inventory- from oral to iv and vice versa.

$I_{24} = \varnothing, I_{25} = \varnothing$. There is no requirement to refer to the "apply medicine" activity state, since it is the closing join activity of split activity 3. Thus, once this activity is activated, activity 3 is not an option for alternative path. Recall that we check the feasibility of $I \wedge \Phi$ on candidate XSP activities, and activity 3 is not such one.

6 Related Work

As mentioned in the introduction, there are several approaches for dealing with exception modeling, while not addressing the semantic alternative availability of other paths. A compensation based rollback as described in [7], using design time specification [5,13,6]. A second approach is dynamic exception handling generated on runtime [12]. Other works used a dynamic interaction with the user for exception handling inference [10]. In [14], the authors addressed a relevant field of automatic service composition for providing an alternative execution while specifying some shortcomings, and refer to the requirement of *alternative control flow*, and *uncertainty in the initial state and service effect*.

7 Summary and Outlook

In this paper we proposed a model enrichment for expressing real world semantics, in the context of alternative solutions, within the process model. By injecting extra meta-data into the process model, the designer is capable of stating those semantics. Once the process model is defined, a case tool may use this extended data in order to assist both the designer during design-time, or the user during runtime, with relevant exception handlers.

This extension provides a more complete reflection of reality. When referring to auto-generated exception handlers, it reduces the number of illogical solutions, and generates a smaller set of exceptions for the designer/user to choose from.

Currently we implement a prototype as a proof of concept, and intend to evaluate it on real world scenarios. Future work may include integration with a real workflow engine. Present plans are to focus on costs, and run solvers (such as SAT) in order to rate available alternatives.

References

1. van der Aalst, W.M.P.: Petri-net-based Workflow Management Software. In: Sheth, A. (ed.) Proceedings of the NFS Workshop on Workflow and Process Automation in Information Systems, Athens, Georgia, pp. 114–118 (1996)
2. Agostini, A., De Michelis, G.: Improving flexibility of workflow management systems. In: van der Aalst, W., Oberweis, J. (eds.) BPM. Models, Techniques, and Empirical Studies, pp. 218–234. Springer, Heidelberg (2000)
3. Cook, S.A.: The complexity of theorem-proving procedures. In: STOC 1971. Proceedings of the third annual ACM symposium on Theory of computing, New York, NY, USA, pp. 151–158 (1971)
4. Dehnert, J., Zimmermann, A.: Making workflow models sound using petri net controller synthesis. In: Meersman, R., Tari, Z. (eds.) OTM 2004. LNCS, vol. 3290, pp. 139–154. Springer, Heidelberg (2004)
5. Du, W., Davis, J., Shan, M.-C.: Flexible specification of workflow compensation scopes. In: GROUP, pp. 309–316. ACM, New York (1997)
6. Eder, J., Liebhart, W.: Workflow recovery. In: CoopIS, pp. 124–134 (1996)

7. Eder, J., Liebhart, W.: Contributions to exception handling in workflow management. In: Burkes, O., Eder, J., Salza, S. (eds.) Proceedings of the Sixth International Conference on Extending Database Technology, Valencia, Spain, March 1998, pp. 3–10 (1998)
8. Gaaloul, W., Bhiri, S., Godart, C.: Discovering workflow transactional behavior from event-based log. In: On the Move to Meaningful Internet Systems 2004: CoopIS, DOA, and ODBASE, pp. 3–18. Springer, Heidelberg (2004)
9. Garcia-Molina, H., Salem, K.: Sagas. In: Proceedings of the ACM SIGMOD Conference on Management of Data, pp. 249–259 (May 1987)
10. Golani, M., Gal, A.: Flexible business process management using forward stepping and alternative paths. In: van der Aalst, W.M.P., Benatallah, B., Casati, F., Curbera, F. (eds.) BPM 2005. LNCS, vol. 3649, pp. 6–8. Springer, Heidelberg (2005)
11. Hagen, C., Alonso, G.: Exception handling in workflow management systems. IEEE Trans. Software Eng. 26(10), 943–958 (2000)
12. Hwang, G.-H., Lee, Y.-C., Wu, B.-Y.: A new language to support flexible failure recovery for workflow management systems. In: Favela, J., Decouchant, D. (eds.) CRIWG 2003. LNCS, vol. 2806, pp. 135–150. Springer, Heidelberg (2003)
13. Kamath, M., Ramamritham, K.: Failure handling and coordinated execution of concurrent workflows. In: ICDE, pp. 334–341. IEEE Computer Society Press, Los Alamitos (1998)
14. Meyer, H., Weske, M.: Automated service composition using heuristic search. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102, pp. 81–96. Springer, Heidelberg (2006)
15. Golani, M., Gal, A.: Optimizing exception handling in workflows using process restructuring. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102, pp. 407–413. Springer, Heidelberg (2006)
16. Reichert, M., Dadam, P.: Adept$_{flex}$-supporting dynamic changes of workflows without losing control. Journal of Intelligent Information Systems (JIIS) 10(2), 93–129 (1998)
17. Worflow management coalition. the workflow reference model (wfmc-tc-1003) (1995), http://www.wfmc.org
18. Specification: Web Services Description Language (WSDL) version 2.0, http://www.w3.org/TR/wsdl

Business Process Simulation for Operational Decision Support

Moe Thandar Wynn[1], Marlon Dumas[1], Colin J. Fidge[1],
Arthur H.M. ter Hofstede[1], and Wil M.P. van der Aalst[1,2]

[1] Faculty of Information Technology, Queensland University of Technology,
GPO Box 2434, Brisbane QLD 4001, Australia
{m.wynn,m.dumas,c.fidge,a.terhofstede}@qut.edu.au
[2] Dept. of Mathematics and Computer Science, Eindhoven University of Technology,
PO Box 513, NL-5600 MB Eindhoven, The Netherlands
w.m.p.v.d.aalst@tue.nl

Abstract. Contemporary business process simulation environments are geared towards design-time analysis, rather than operational decision support over already deployed and running processes. In particular, simulation experiments in existing process simulation environments start from an empty execution state. We investigate the requirements for a process simulation environment that allows simulation experiments to start from an intermediate execution state. We propose an architecture addressing these requirements and demonstrate it through a case study conducted using the YAWL workflow engine and CPN simulation tools.

1 Introduction

Business process simulation enables the analysis of business process models with respect to performance metrics such as throughput time, cost or resource utilization. A number of business process modelling tools support simulation to varying degrees [7]. However, this tool support is largely geared towards *a priori*, i.e., design time, comparison of candidate business process models. Accordingly, they assume that simulation experiments are run from an empty initial state, for a very large number of cases, to give analysts insight into the average, long-term benefits of process improvement options.

This contrasts markedly with the requirements of *operational decision support*, where the goal is to evaluate short-term options for adjusting an already deployed business process in response to contextual changes or unforeseen circumstances. In this situation, the current system state and recent event history cannot be ignored, and the emphasis is on understanding the short-term implications of making a change to the system.

Another shortcoming of contemporary process simulation tools with respect to operational decision support is the inability to set different completion horizons for simulation experiments. The focus of traditional simulation experiments is to identify average long-term behaviour, over a wide variety of contextual scenarios. By contrast, operational decision making introduces the need to make short-term decisions, based on the current state and specific recent history. To do this we need the ability to limit the simulation's forward-looking 'horizon', to enable rapid evaluation of the consequences of various decisions. A typical example is the need to determine if redeploying resources will eliminate a temporary backlog of unprocessed jobs within a given time frame. Simulation horizons of interest include absolute times (e.g., 30 June 9pm), time durations (e.g., 5 hours from now), the number of jobs completed (e.g., 200th case), and the number of resources consumed (e.g., when 80% of employees are busy).

In this paper we define the requirements for an operational process simulation environment which addresses these issues, and describe a suitable toolset architecture. To demonstrate the feasibility of the concept, we also describe the outcomes of a proof-of-concept case study performed using existing, off-the-shelf tools, the YAWL workflow engine and the CPN simulation tools.

2 Previous and Related Work

Business process simulation involves developing an accurate simulation model which reflects the behaviour of a process, including the data and resource perspectives, and then performing simulation experiments to better understand the effects of running that process [13]. In general, a business process simulation model consists of three components: basic model building blocks (e.g., entities, resources, activities, and connectors); activity modelling constructs (e.g., branch, assemble, batch, gate, split and join); and advanced modelling functions (e.g., attributes, expressions, resource schedules, interruptions, user defined distributions) [13]. Business process simulation is regarded as an invaluable tool for process modelling due to its ability to perform quantitative modelling (e.g., cost-benefit analysis and feasibility of alternative designs) as well as stochastic modelling (e.g., external factors and sensitivity analysis) [4]. Simulation has been used for the analysis and design of systems in different application areas [13], a "decision support tool" for business process reengineering [6] and for improving orchestration of supply chain business processes [12].

Simulation functionality is provided by many business process modelling tools based on notations such as EPCs or BPMN. These tools offer user interfaces to specify basic simulation parameters such as arrival rate, task execution time, cost, and resource availability. They allow users to run simulation experiments and to extract statistical results such as average cycle time and total cost. Process simulation can also be performed using a more general class of simulation techniques known as discrete event simulation [13].

Even though simulation is well-known for its ability to assist in long-term planning and strategic decision making, it has not to date been considered a mainstream technique for operational decision making due to the difficulty of obtaining real-time data in the timely manner to set up the simulation experiments [8]. Nevertheless, a number for recent developments point out how aspects of the problem can be handled, and form the basis of our approach.

A novel use of discrete event simulation, close to our own aims, is short-interval scheduling of a shop floor control system where the ability of the simulation to "look ahead" at the expected performance of the system in the near future, given its current status, is used to provide real-time responses to dynamic status changes [3,11]. We aim to generalise this specific capability to arbitrary business models. More significantly, Reijers et al. [8], introduced the concept of 'short-term simulation'. They went on to experiment with short-term simulations from a 'current' system state to analyse the transient behaviour of the system, rather than its steady-state behaviour [9]. A similar resource-oriented approach is provided by the proprietary Staffware prediction engine[1]. Our goal is to design such a 'short-term' analysis architecture in the context of widely-used, off-the-shelf workflow tools, and without the specific focus on resourcing.

To do this, we have experimented with a combination of the YAWL workflow engine [1] and the CPN Tools simulator [2]. A number of previous such experiments have informed our work. For instance, Gottschalk et al. [5] used a YAWL subset to generate CPN models, and Verbeek et al. [14], integrated the ExSpect simulator with Protos 7.0 to provide modelling and simulation facilities in one tool. Also, Rozinat et al. [10] showed how event logs produced by CPN models can be 'mined' to discover the operational characteristics of the model. Our aim is to combine both these notions, i.e., creating simulation models from workflow processes and feeding back simulation results to calibrate the model, but with a particular emphasis on incorporating observed behaviours from the 'real', operational system into the predictive simulation.

3 Requirements for Operational Process Simulation

In this section we use a simple example to motivate the requirements for operational decision support. Consider the credit card application process expressed as a workflow model in Figure 1. The process starts when an applicant submits an application. When an application is received, a credit clerk checks whether the application is complete. If the application is found to be incomplete, the clerk requests additional information and waits until the information is received before proceeding. For a complete application, the clerk performs further checks to validate the applicant's income and credit history. The validated application is then passed on to a manager to make the decision. The manager decides either to accept or reject an application. For an accepted application, a credit card is produced and delivered to the applicant. For a rejected application, the applicant is given a timeframe to request a review of the decision. If a review request is not received, the process ends.

A typical question for the credit application process might be "How long will it take to process a credit card application?" Using conventional tools for business process simulation, it is possible to answer this question with an *average* duration, assuming some 'typical' knowledge regarding the available resources and expected execution times for the involved activities. However, if the business

[1] `http://www.tibco.com/`

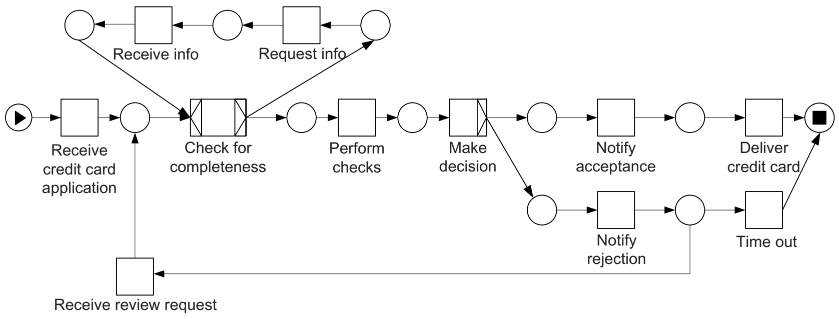


Fig. 1. Workflow model of a credit card application process

process is already operational, and it is supported by a workflow management system, the same question can be asked for observed, specific states of execution. For instance, we could ask ourselves, “how long will it take to complete processing a particular application, provided that all documentation is complete and the application is now ready for a manager to make the decision?” Most importantly, this can be done using the actual state of the system’s resources, such as the number of clerks already occupied with other applications.

While performing ‘short-term’ system predictions, we need to define when a simulation experiment should stop, i.e., the completion horizon. This can be defined as a bound on various aspects of simulation, such as end times and durations, as well as the number of case completions and at various resource utilisation rates. For the credit card application example, some interesting completion horizons include: 12 or 24 hours duration from now; the time at which the delay for decision making is over 3 days; the point at which 1000 applications have been processed, etc. Operational decision makers seeking to adjust the credit card process following spikes in demand, or delays caused by unexpected events, would benefit from being able to perform simulations with different horizons.

Consider for example the case where the company runs a highly successful promotion campaign and receives unexpectedly large numbers of credit applications. As a result, the company now has a backlog of applications (e.g., 100 applications) waiting to be processed. In this case, the average time (e.g., five days) to process a credit card application cannot be guaranteed with the current number of staff members (3 clerks and 1 manager). At this stage, it is desirable to obtain more realistic input data to determine the cycle time by taking into consideration the current number of applications in the queue, and other observable properties of the ‘live’ system. Understanding this can lead to a more effective resource planning for the manager. Given the current state of system, the following questions might be of interest to a manager:

1. What is the cycle time to process an application at this current load?
2. Is it possible for all applications in the queue to be processed after a certain duration (e.g., in 12 hours)?

3. What are the consequences of adding five additional clerks and two managers to assist in processing?

None of these questions can be answered with precision using the 'average' results produced by a conventional simulation from an empty state. Overall, therefore, the requirements for an operational process simulation toolkit are:

1. The ability to start a simulation from a non-empty state, using data obtained from the operational system's actual behaviour.
2. The ability to specify (multiple) breakpoints in a simulation experiment based on different criteria such as the number of cases completed, the time horizon, or based on conditions encountered in the simulated environment (e.g., queue or resource utilization dropping below preset levels).
3. The ability to automatically extract and process historical execution data, and in particular recent data, in order to calibrate the simulation model.

4 Architecture for Operational Process Simulation

In this section, we first propose the generic architecture for simulation and then discuss the various process components to realise this architecture.

4.1 Generic Architecture

Figure 2 shows a data flow diagram of the proposed architecture as a tool chain to support operational process simulation. The process modelling and analytics phase of the tool chain is concerned with developing a stateful simulation model while the process simulation phase focuses on running various simulation experiments and providing simulation reports as well as detailed simulation logs for use as input into (re)design of the simulation model. The diagram shows a "step-by-step" translation of a simulation template: first by enriching the template with historical data to derive the various simulation parameters and second by including the starting state to develop a stateful simulation model. The resulting stateful simulation model is then used to run various simulation experiments. The external input from observed "real-world" logs plays a crucial role in this architecture and it is envisioned that a number of extraction functions will be used to derive the historical data and the starting state from these logs. The architecture also supports the use (and conversion) of simulation logs to derive historical data and a starting state.

The main data objects (depicted as hexagons) and activities (depicted as rectangles) comprising the architecture in Fig. 2 are as follows.

Simulation template. A simulation template includes the representation of control, data, and resource requirements of a business process (process definition) as well as necessary setup information for simulation experiments. To run simulation experiments, a simulation template defines various input and output parameters, breakpoints for completion horizons and derivation functions. At

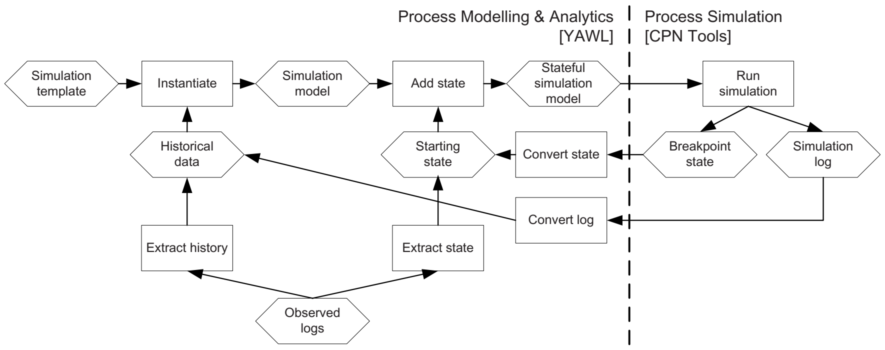


Fig. 2. Architecture of the operational process simulation toolkit

a minimum, a simulation template needs to specify the following setup information: arrival rates of cases, a resource calender, simulation parameters (Key Performace Indicators), completion horizons (breakpoints) and simulation report requirements (monitors). Furthermore, various parameters in the template are also enriched with information on how to generate the data used in the simulation (estimated or derived). In our proposed architecture such parameters can be specified either by entering estimations or by specifying various derivation functions over the observed and simulated log files. For example, the case arrival rate parameter is typically specified over a Poisson distribution. Similarly, the average execution time of a task is specified using mean and standard deviation. Abstract data types for case arrival rates and execution times in a simulation template can be specified as follows:

$$\mathit{ArrivalRate} : (\mathit{ArrivalRateFunction} \cup \mathit{HistoricalData} \rightarrow \mathit{ArrivalRateFunction})$$
$$\mathit{ExecutionTime} : \mathit{Task} \rightarrow (\mathit{TimeFunction} \cup \mathit{HistoricalData} \rightarrow \mathit{TimeFunction})$$

Instantiation. This activity takes a simulation template with derivation functions and historical data from the logs to generate a simulation model. It is essential that log data contains relevant information that can be used for a given derived parameter. Obviously, the requirements for logs could vary depending on a given parameter and the derivation function used. The logs data can be based either on observations from a running process engine or from prior simulation logs.

Historical data. Historical data to instantiate a simulation template could be extracted from the execution logs of a process engine or from previous simulation runs. For instance, if an average execution time of a task is to be derived using log data to calculate the mean and standard deviation, the log should contain information about when all instances of a given task are executed and completed. If a derivation function is also based on resources (i.e., the time it takes to execute a task by a manager), then the log should contain information about resource utilisation in addition to time. Conversions and adjustments might be necessary if log data is incompatible with the requirements in the template. These conversions take place during the *Extract history* activity for observed logs and the *Convert log* activity for simulation logs. The necessary abstract data type for a log to derive case arrival rate and execution times is as follows:

$$\begin{aligned}&\mathit{Cases} : \mathit{Case} \times \mathit{CreationTime} \times \mathit{CancellationTime} \times \mathit{CompletionTime}\\&\mathit{CompletedTasks} : \mathit{Case} \times \mathit{Task} \times \mathit{StartTime} \times \mathit{EndTime} \times \mathit{Resource}\end{aligned}$$

Observed logs and Simulation logs. While the observed logs represent the data and metrics from executing process engines, the simulation logs provide the information from prior simulation runs. Both types of historical data are useful in determining appropriate values for simulation parameters.

Simulation model. After ensuring that all derivable simulation parameters have been instantiated with historical data, a simulation model is generated. It is now possible to use this simulation model to run simulations from the initial state.

Adding a state. This activity takes a simulation model and a given state to set the starting state of a simulation experiment. If a simulation experiment is to be started from scratch (an empty state), minimal transformation is required to include resource scenario for a simulation experiment. On the other hand, if a simulation experiment is to be started from a given state, current state information is added to obtain a stateful simulation model. In cases where some of the tasks are already running for a certain amount of time in the simulation's starting state, we propose to use a truncated probability distribution so that the duration randomly assigned to an active task during the simulation is always greater than the amount of time for which the task has already been running.

Starting state. The state information can be derived from historical logs and also from prior simulation runs. At a minimum, the logs used to derive state should contain information on active cases, resource availability and active and enabled tasks information. Inconsistencies are possible between a given model and the data obtained from the logs and conversions might be necessary. The following abstract datatypes for logs capture the minimum information requirements to generate an initial simulation state:

$$\begin{aligned}&\mathit{ActiveCases} : \mathit{Case} \times \mathit{CreationTime}\\&\mathit{ActiveTasks} : \mathit{Case} \times \mathit{Task} \times \mathit{StartTime} \times \mathit{Resource}\\&\mathit{EnabledTasks} : \mathit{Case} \times \mathit{Task}\\&\mathit{ResourceAvailability} : \mathit{Resource} \times \mathit{Role}\\&\mathit{LogTime} : \mathit{Time}\end{aligned}$$

Importantly, this allows us to inject observed characteristics of the system into the simulation. For instance, let's assume a simulation model may specify the (initial) availability of three staff members, whereas the observed logs show there are actually five staff members currently assigned to this process.

Breakpoint state. Capturing the full state of the simulation model at the end of a simulation experiment provides an opportunity to use the breakpoint state as the initial state for another simulation, thus facilitating the conduct of simulation experiments with different breakpoints.

Stateful simulation model. A stateful model is obtained by enriching a simulation model with starting state information for simulation runs. More than one stateful simulation model can be developed where each one represents the state at a certain point in time. The *LogTime* parameter from the state is used to set the starting time of the simulation experiments.

Running the simulation. Simulation experiments can now be started using a stateful model and stopped at various completion horizons. In addition to the generation of simulation reports for analysis, the architecture makes provision for the generation of both breakpoint states and simulation logs. This data can then be used as input for later simulation runs after necessary conversions.

4.2 A Practical Instantiation of the Architecture

It is possible to realise the proposed simulation architecture in a number of ways using a suitable process editor, a process engine (with logging functionality) and a simulation tool that is flexible enough to support our requirements. For our research, we are using the YAWL workflow environment for both modelling and analytics components and the simulation capabilities within CPN Tools for process simulation, as shown by the partition in Fig. 2.

The YAWL workflow environment was chosen because of its formal foundation in Petri nets, its expressiveness in providing support for workflow patterns, its easy-to-use graphical editor that has the ability to generate executable process models, and its extensive logging function for process execution. There are also mappings available between various business process modelling notations (EPC, BPMN, BPEL) and Petri nets. Furthermore, the YAWL workflow language is supported by an open-source implementation[2]. The YAWL editor is an ideal candidate for the process modelling component in the architecture as the user can specify control, data and resource requirements of a business process using a natural graphical notation and then export the process definition as an XML file ready for execution in the engine. Various verification functionalities are also available in the editor to ensure the correctness of the process model before execution. The YAWL editor can be easily extended to capture various setup parameters for a simulation template. The logging module in the YAWL engine can be used to record the statistics of various cases (such as the start and end times, the resource, whether the task is cancelled or completed, etc). These logs provide sufficient information to generate stateful simulation models.

The current YAWL implementation does not provide simulation functionality. However, it is rather straightforward to transform YAWL models into Coloured Petri Nets (CPNs) [2], modulo some restrictions, and to exploit the simulation capabilities of CPN Tools[3]. Coloured Petri nets can be used to model Petri nets with time constraints and hierarchy. In contrast to contemporary BPM simulation tools, CPN Tools can be customised to support our requirements, i.e., the ability to start simulation experiments from a given state and the specification of

[2] `http://www.sourgeforge.net/yawl`

[3] `http://www.daimi.au.dk/CPnets/`

different completion horizons using breakpoints. It is possible to incorporate the log data by specifying derivation rules using ML functions. Rather than modelling the business process directly in CPN Tools (i.e., modelling the process as one or more Coloured Petri nets), our arrangement avoids the need for detailed knowledge of Petri nets, which would make CPN Tools unsuitable for business process designers. For these reasons, we combine use of YAWL for modelling and execution of business processes with CPN Tools for BPM simulation, to leverage their strengths in their respective areas.

5 Proof of Concept

To validate the simulation architecture, we developed a stateful simulation model for the credit card application process from Section 3 and performed simulation experiments using the CPN Tools. The credit card application workflow process from Fig. 1 was (manually) translated into a corresponding hierarchical and timed colored Petri net as depicted in Fig. 3.

The translations are similar to the Gottschalk et al.'s approach [5] where a YAWL condition is mapped to a CPN place and a YAWL task to a CPN substitution transition (see the *Make decision* task Fig. 4). As the credit card process does not contain tasks with complex split and join behaviours, we elected to map most YAWL tasks to CPN transitions directly for simplicity. The *Environment* subpage controls the arrival of cases. Tokens in the *resource* place indicates the number of employees available to execute the tasks (e.g., clerks and managers). Execution times for tasks were specified using ML functions and a time delay was attached to the task (e.g.,$a@+getExecTime(mean = 7200, stdD = 3600)$). In Fig. 4, the XOR-split behaviour of the task is modelled as two transitions that share the same input place (*busy*). The start transition for the *Make decision* task has another input from the resource place with a guard to that only a manager can carry out this task ($[\#role(e) = mgr]$).

To determine the feasibility of the proposed architecture, we tested out the resource planning scenario as discussed in section 3. For testing purposes, the stateful simulation model is populated with the following data.

– a case arrival rate of 1 application per hour following a Poisson distribution;
– 100 active cases as the starting state where 80 credit card applications are in the *ready* place, 15 in the *complete check* place, 3 in the *receive* place, 1 in the *accept* place and 1 in the *busy* place of the *Make decision* task;
– breakpoint monitors with 12-hour and 24-hour time horizons and marking size monitors to observe the two places of interest, the *ready* place and the *complete check* place;
– execution times (sample mean and standard deviation) for each task; and
– two resource availability scenarios, namely 5 clerks and 1 manager, and 10 clerks and 3 managers.

Multiple simulation runs are carried out using the starting state with 100 active cases, two different time horizons (12 hours and 24 hours) and with two

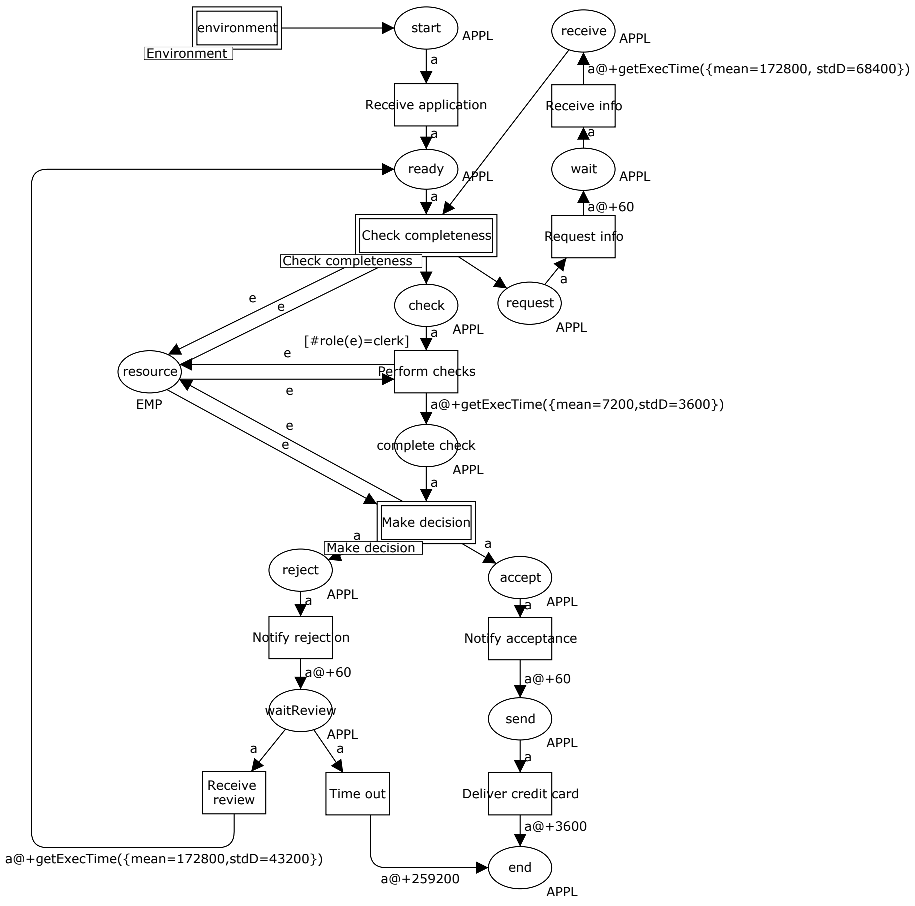


Fig. 3. CPN model of credit card application process

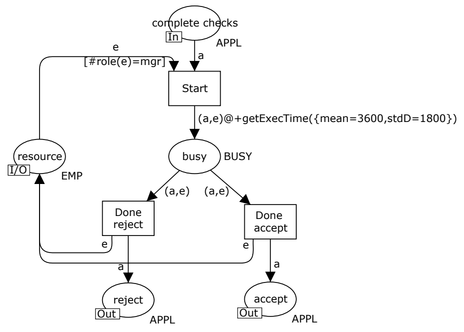


Fig. 4. CPN subpage for making a credit card decision

different resource scenarios. The results (see Table 1) show that when processing the applications with 5 clerks and 1 manager, on average (with 95% probability) between 48 to 55 applications are still in the queue in the *ready* (R) place and between 17 to 28 in the *complete check* (C) after 12 hours and it becomes 13-19 (R) and 14-22 (C) after 2 24 hours. On the other hand, the queue is reduced to 12-20(R) and 3-9 (C) after 12 hours, and 10-17(R) and 0-2(C) when 10 clerks and 3 managers are available. This scenario illustrates the possibilities opened by operational process simulation, in terms of being able to perform "what-if" analysis based on the current situation and to different completion horizons.

Table 1. Summary of simulation results

Resource availability	Duration (12 hours)	Duration (24 hours)
5 clerks and 1 manager	48-55 (R) and 13-19 (C)	17-28 (R) and 14-22 (C)
10 clerks and 3 managers	12-20 (R) and 3-9 (C)	10-17(R) and 0-2 (C)

In this proof-of-concept demonstration we were specifically interested in validating the feasibility of inserting a non-empty starting state and multiple completion horizons into simulation experiments. For instance, we assumed an arrival rate of 1 application per hour instead of deriving the actual arrival rate for these observations. Nevertheless, we have confirmed that the YAWL engine logs contain sufficient data to instantiate the simulation template and to add state. The same goes for the resource availability statistics and the current state. In the next stage, we plan to implement the necessary extraction and conversion functions to derive simulation parameters and starting states automatically from logs.

6 Conclusion and Future Work

To produce accurate short-term predictions a workflow simulation environment must start its analysis in a state that incorporates the actual, observed properties of the operational system, including its recent history. In this paper we have demonstrated the feasibility of building such a simulation environment using off-the-shelf workflow modelling and system simulation tools. A general design for such a system was defined in terms of its essential capabilities and a (manual) feasibility study was conducted using the YAWL and CPN Tools toolkits. Currently we are implementing the various 'gluing' components needed to automate the transformation of 'mined' log values to produce simulation inputs.

Acknowledgement. This research was funded by the Australian Research Council grant DP0773012.

References

1. van der Aalst, W.M.P., ter Hofstede, A.H.M.: YAWL: Yet Another Workflow Language. Information Systems 30(4), 245–275 (2005)
2. Beaudouin-Lafon, M., Mackay, W., Andersen, P., Janecek, P., Jensen, M., Lassen, H., Lund, K., Mortensen, K., Munck, S., Ratzer, A., Ravn, K., Christensen, S., Jensen, K.: A Tool for Editing and Simulating Coloured Petri Nets. In: Margaria, T., Yi, W. (eds.) ETAPS 2001 and TACAS 2001. LNCS, vol. 2031, pp. 574–577. Springer, Heidelberg (2001)
3. Drake, G., Smith, J., Peters, B.: Simulation as a planning and scheduling tool for flexible manufacturing systems. In: Proceedings of the 27th conference on Winter simulation, pp. 805–812 (1995)
4. Giaglis, G., Paul, R., Doukidis, G.: Simulation for intra- and inter-organisational business process modelling. In: Proceedings of the 28th conference on Winter simulation, pp. 1297–1308 (1996)
5. Gottschalk, F., van der Aalst, W.M.P., Jansen-Vullers, M.H., Verbeek, H.M.V.: Protos2CPN: Using Colored Petri Nets for Configuring and Testing Business Processes. In: Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools, Aarhus, Denmark, (October 2006), Published online at: http://www.daimi.au.dk/CPnets/workshop06/
6. Hlupic, V., Robinson, S.: Business process modelling and analysis using discrete-event simulation. In: Proceedings of conference on Winter simulation, vol. 2, pp. 1363–1369 (1998)
7. Jansen-Vullers, M., Netjes, M.: Business process simulation – a tool survey. In: Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools, Aarhus, Denmark (October 2006), Published online at: http://www.daimi.au.dk/CPnets/workshop06/
8. Reijers, H.A., van der Aalst, W.M.P.: Short-term simulation: bridging the gap between operational control and strategic decision making. In: Proceedings of the IASTED Conference on Modeling and Simulation, Philadelphia, USA, pp. 417–421 (1999)
9. Reijers, H.A.: Design and Control of Workflow Processes. Springer, New York (2003)
10. Rozinat, A., Mans, R., van der Aalst, W.M.P.: Mining CPN Models: Discovering Process Models with Data from Event Logs. In: Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools, Aarhus, Denmark (October 2006), Published online at: http://www.daimi.au.dk/CPnets/workshop06/
11. Smith, J., Wysk, R., Sturrock, D., Ramaswamy, S., Smith, G., Joshi, S.: Discrete event simulation for shop floor control. In: Proceedings of conference on Winter simulation, Lake Buena Vista, FL, vol. 2, pp. 962–969 (1994)
12. Tewoldeberhan, T.W., Verbraeck, A., Msanjila, S.: Simulating process orchestrations in business networks: a case using BPEL4WS. In: Proceedings of the 7th international conference on Electronic commerce, pp. 471–477. ACM Press, New York (2005)
13. Tumay, K.: Business process simulation. In: Proceedings of the 28th conference on Winter simulation, pp. 93–98 (1996)
14. Verbeek, H.M.V., Hattem, M.: Simulation made accessible. In: Ciardo, G., Darondeau, P. (eds.) ICATPN 2005. LNCS, vol. 3536, pp. 465–474. Springer, Heidelberg (2005)

Autonomic Business Processes Scalable Architecture
Position Paper

José A. Rodrigues Nt.[1], Pedro C.L. Monteiro Jr.[1], Jonice de O. Sampaio[1], Jano M. de Souza[1,2], and Geraldo Zimbrão[1]

[1] COPPE/UFRJ, Graduate School of Engineering
[2] DCC-IM Dept. of Computer Science, Institute of Mathematics
Federal University of Rio de Janeiro, Brazil
PO Box 68.511 – Zip code: 21945-970 - Rio de Janeiro - Brazil
{rneto, calisto, jonice, jano, zimbrao}@cos.ufrj.br

Abstract. Organizations have to face new challenges, hold new opportunities, conquer and maintain important customers and always find a better strategic position. The principles used in Autonomic Computing can be adapted to help them survive in dynamic business scenarios. Thus, organizations should count on processes that can be able to self-manage and self-adapt to better answer market and organization's changes, as well as new challenges – Autonomic Business Processes. This work proposes a multi-agent rule-based scalable architecture to provide business processes with autonomic properties, reducing the need for human intervention, and improving overall organization's response time.

Keywords: Autonomic Computing, Business Process, Workflow.

1 Introduction

The world is confronted with the new knowledge-driven economy, where processes' dynamics are high and their impact on management unavoidable. New approaches to support management ought to be developed. Autonomic Computing (AC), which appeared due to the increasing complexity of managing current computational solutions, is a good model. The principles used in AC can be adapted to help firms survive in dynamic business scenario. Organizations should count on processes that can be able to self-manage and self-adapt to new challenges and market changes – Autonomic Business Processes. Therefore, we believe the marriage of business management needs with AC principles opens new opportunities.

This work proposes a multi-agent rule-based architecture to provide business processes with autonomic properties. The rule-based approach, supported with layered blackboards, considers expert knowledge and treats business processes under the Complex Adaptive Systems paradigm [1], in the sense that the many variables involved in processes executions and their relationships demand systems capable of presenting emergent behaviors. It promotes flexibility by adaption, as classified in [2]. Additionally, the multi-level architecture provides a flexible solution that can be scaled up to work with higher level abstractions, closer to or at an organization's strategic level, through the composition of lower level autonomic processes.

2 Related Works

Most work found on autonomic business process, or autonomic workflow, focus on predictable workflows, in the sense that a baseline execution path can be defined and all alternates paths mapped; flexible but a priori defined workflow instances; or software or system oriented workflows, like in grid applications.

While the work of Savarimuthu, Purvis and Fleurke [3] deals with business process execution in a multi-agent workflow system, it neither describes how agents actually handles flexibility, nor it worries with autonomic properties.

Work done on medical workflows [4] [5] concentrate on treating exceptions and related mechanisms. They have no provision for self-optimization and support self-configuration and self-protection in minor scale. This is also the case of AgentWork [6] that concentrates on failure prediction and reaction.

Mangan and Sadiq [7] do not focus on treating healing and protection issues and do not dedicate enough attention to real-time monitoring and reaction. Nevertheless, their analysis of processes definition and handling approaches helps on understanding the need for a non-deterministic component in our solution.

A dynamic workflow for grid environment is described in [8]. This solution just works with workflows in grids, for job execution, not business processes.

In [9], the authors propose a continuous and optimized computing environment. While it considers business objectives as the driving force to process optimization, it directs the optimization efforts towards IT assets utilization.

FEEDBACKFLOW is an adaptive workflow generator for system's management [10].

Another related work is the view of a multi-agent workflow enactment as an Adaptive Workflow [11]. It touches some important aspects closely related to our solution, as the use of multi-agent systems for coordination and the use of containers to preserve the workflow state of execution.

Web services oriented workflows are also subject of other studies. Autonomic Web Services (AWP) are web processes that support AC properties [12]. In AWP, the processes are configured according to business policies. Failures are quickly responded and the workflow can be reconfigured due to environmental changes. The work of Pautasso, Heinis and Alonso [13] about web services composition, only evaluates policies for composition configuration. On the same grounds, Pankatrius and Stucky [14] establish a formal foundation for workflow composition, instrumental to provide reconfiguration capabilities to workflow applications.

In [15] the system has a component to determine if the current configuration is optimal and, if appropriate, proposes an alternate execution plan.

Our work evolves from several aspects presented on these previously mentioned works and relies on some of their mechanisms for proper implementation, e.g. the formalisms proposed in [14] and [16].

3 Concepts and Related Technologies

3.1 Autonomic Computing

The concept originates from the human autonomic nervous system, which is responsible for managing functions that humans do automatically, i.e., without reasoning and giving instructions. The AC paradigm aims at mimicking the nervous system, providing computer systems with self-management capabilities, reducing human intervention.

The 4 basics aspects of AC are [17]:

- Self-configuring: refers to installation and activation of the system in an automated way;
- Self-healing: the ability to discover, diagnose and correct problems;
- Self-optimizing: resource monitoring and optimization; and
- Self-protecting: identification, detection and protection against threats.

3.2 Agent

An agent is anything that has sensors to perceive the environment and act on it [18]. Agents can interact with other agents forming multi-agent systems.

Agents present, at least, the following properties [19]: reactive to the environment, autonomous, goal-driven and continuous execution.

Agents approach provides techniques to decompose the control intelligence of flow execution and to encapsulate distributed resources [20].

3.3 Blackboard

Blackboard is a repository style architecture where loosely coupled entities share a common knowledge space [21]. In [22], the blackboard system is divided in 3 components: the blackboard, a global data structure that usually holds system's state information; knowledge sources; and a control component, driven by the blackboard state indication.

3.4 Peer-to-Peer

System implementation can benefit from the use of P2P technologies. COPPEER [23], a multi-agent framework, can support the proposed architecture, adapting its shared repositories, to behave as needed blackboards.

4 Autonomic Business Process

4.1 Attribute, Fact, Condition, Action, Rule and Priority

Attribute is a process's characteristic which is of interest. *Value* is the attribute's measurement. A *Fact* is an observed attribute with a specific value.

The architecture supports the creation of any kind of resources, as time (expected task duration), human (available executors) or cost (process estimated cost).

An *Attribute* can be associated to a value through relational operators (=, <>, >, <, >= and <=) forming an atomic *condition*. A *composed condition* is the association of atomic conditions through the use of the Boolean operators.

Action is an intervention on a process, that can be direct, e.g. allocation of others resources to the process, or indirect, i.e. notification of an occurrence to another handling instance, e.g., the assertion of new facts to a blackboard or a message delivered to another system or user.

A *Rule* is constructed using conditions and actions – ***if*** *condition* ***then*** *actions*. For each rule a priority is defined, thus, allowing for the proper processing order.

We haven't explicitly used ECA [16] because the working characteristics of monitoring agents/expert systems lead us to think that a fact treatment is more appropriate, since the event that can affect workflow execution is usually a composition of many facts. At the same time, considering that comparing to events, facts may be seen as lower level constructs, we believe this approach is more suitable to model expert's knowledge.

4.2 System Architecture

The proposed architecture is shown in figure 1. Each defined activity of the target process has associated *Monitor agents*, a *Local Blackboard* and *Actuator agents*. Such schema over all the process' activities defines the lower level of the architecture, i.e., the level closer to the primitive activities.

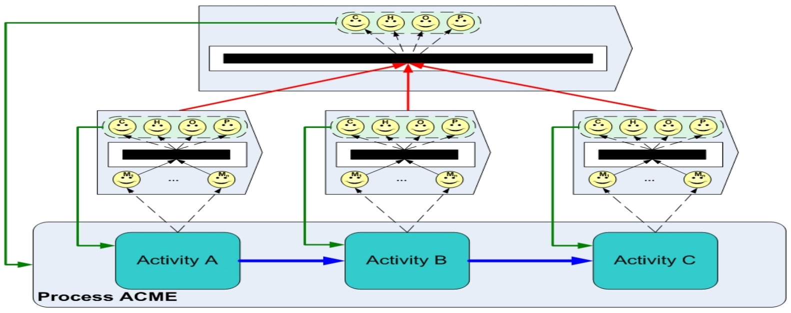


Fig. 1. System Architecture

Monitor agents (letter M inside them in figure 1) have the function of sensing the activity and writing facts, onto the *Local Blackboard*, that receives facts related to all resources defined for the activity.

Actuator agents are responsible for fact interpretation and reactions. They contain rules defined by the user and work like any expert system, mimicking processes' specialists. They observe the *Blackboard* looking for facts and when a rule is triggered, they execute the prescribed action. Each *Blackboard* works with four *Actuator agents* and each one is responsible for one autonomic computing dimension – CHOP, shown above *local Blackboards* in figure 1.

The next level of the architecture has a *global Blackboard* that serves a group of activities, e.g., the whole process. This *global Blackboard* receives facts from lower level *Actuators*. The *Actuators* at the lower level work as monitors for the upper level.

We can now briefly describe how the architecture works. The user defines the process, i.e. the workflow, and for each activity defined, specifies attributes, facts, condition and rules that shall be applied. When the process is started, the *Monitors* start sensing the activities, registering facts in the *local Blackboard*. The *Actuator*s read the *Blackboard* triggering the matched rules, i.e., executing the prescribed actions. Usually, actions include the assertion of a fact onto the next upper level *Blackboard*, promoting coordination, i.e., allowing for the implementation of the autonomic behaviors for the process as a whole. The same behavior explained for the lower level is then manifested at the upper level.

5 Conclusions and Future Works

We believe the proposed architecture is simple, yet powerful. The approach combining agents and rule-based systems allows for its use with modern software technology, e.g. SOA, brings the systems closer to the organization, since it is also based on business expert's knowledge, and innovates business management technology.

It is important to note that although presented here in two levels only, the proposed architecture can easily be scaled up to work with many levels. We believe, and we are also researching, the assembly of such architecture in multiple levels. In the highest level though, in a process-oriented organization, we expect to deliver what we call the Autonomic Balanced Scorecard.

References

1. Tan, J., Wen, H., Awad, N.: Health Care and Services Delivery Systems as Complex Adaptive Systems. Communications of the ACM 48(5) (2005)
2. Heinl, P.: Exceptions during workflow execution. In: Schek, H.-J., Saltor, F., Ramos, I., Alonso, G. (eds.) Proceedings of the 6th International Conference on Extending Database Technology, Valencia, Spain (March 1998)
3. Savarimuthu, B.T., Purvis, M., Fleurke, M.: Monitoring and controlling of a multi-agent based workflow system. In: Hogan, J., Montague, P., Purvis, M., Steketee, C. (eds.) Proceedings of the 2nd Workshop on Australasian Information Security, Data Mining and Web intelligence, and Software Internationalisation, vol. 32. ACM International Conference Proceeding Series, vol. 54, pp. 127–132. ACS, Darlinghurst (2004)
4. Han, M., Thiery, T., Song, X.: Managing exceptions in the medical workflow systems. In: ICSE 2006. Proceeding of the 28th International Conference on Software Engineering, Shanghai, China, ACM Press, New York (2006)
5. Mourão, H., Antunes, P.: Supporting effective unexpected exceptions handling in workflow management systems. In: SAC 2007. Proceedings of the 2007 ACM Symposium on Applied Computing, Seoul, Korea, pp. 1242–1249. ACM Press, New York
6. Mueller, R.: Event-Oriented Dynamic Adaptation of Workflows: Model, Architecture and Implementation. PhD thesis, University of Leipzig (2002)

7. Mangan, P., Sadiq, S.: On building workflow models for flexible processes. In: Proceedings of the 13th Australasian Database Conference, vol. 5, Melbourne, Australia. ACM International Conference Proceeding Series, vol. 18, pp. 103–109. ACS, Darlinghurst (2002)
8. Nichols, J., Dermikan, H., Goul, M.: Autonomic Workflow Execution in the Grid. In: IEEE Transactions on Systems, Man, and Cybernetics (2006)
9. Aiber, S., Gilat, D., Landau, A., Rainkov, N., Sela, A., Wasserkrug, S.: Autonomic self-optimization according to business objectives. In: Proceedings of the International Conference on Autonomic Computing (2004)
10. Andrzejak, A., Herman, U., Sahai, A.: FEEDBACKFLOW - An Adaptive Workflow Generator for System Management. In: International Conference on Autonomic Computing (2005)
11. Buhler, P.A., Vidal, J.M., Verhagen, H.: Adaptive Workflow = Web Services + Agents. In: Proceedings. of the International Conference on Web Services (2003)
12. Verma, K., Sheth, A.P.: Autonomic Web Processes. In: Proceedings of the 3rd International Conference on Service Oriented Computing (2005)
13. Pautasso, C., Heinis, T., Alonso, G.: Autonomic execution of Web Service Compositions. In: ICWS 2005. Proceedings of the IEEE International Conference on Web Services, pp. 435–442. IEEE Press, Los Alamitos (2005)
14. Pankratius, V., Stucky, W.: A Formal Foundation for Workflow Composition, Workflow View Definition, and Workflow Normalization based on Petri Nets. In: Hartmann, S., Stumptner, M. (eds.) APCCM 2005. Proc. 2nd Asia-Pacific Conference on Conceptual Modelling, CRPIT, 43, Newcastle, Australia, ACS, pp. 79–88 (2005)
15. Heinis, T., Pautasso, C., Alonso, G.: Design and Evaluation of an Autonomic Workflow Engine. In: Proceedings of the 2nd International Conference on Autonomic Computing (2005)
16. Casati, F., Ceri, S., Paraboschi, S., Pozzi, G.: Specification and implementation of exceptions in workflow management systems. ACM Trans. Database Syst. 24(3), 405–451 (1999)
17. Murch, R.: Autonomic Computing. Prentice-Hall, Englewood Cliffs (2004)
18. Russel, S., Norvig, P.: Artificial Intelligence: A Modern Approach. Prentice-Hall, Englewood Cliffs (1995)
19. Oshima, M., Lange, D.B.: Programming and Deploying Java Mobile Agents with Aglets. Second Printing. Addison Wesley, Boston (1998)
20. Zhao, Z., Belloum, A., Sloot, P.M.A., Hertzberger, L.O.: Agent Technology and Generic Workflow Management in an e-Science Environment. In: Zhuge, H., Fox, G.C. (eds.) GCC 2005. LNCS, vol. 3795, pp. 480–485. Springer, Heidelberg (2005)
21. Shaw, M., Garlan, D.: Software Architecture – Perspectives on an Emerging Discipline. Prentice-Hall, Englewood Cliffs (1996)
22. Corkill, D.D.: Blackboard Systems. AI Expert 6(9), 40–47 (1991)
23. Miranda, M., Xexeo, G., Souza, J.: Building Tools for Emergent Design with COPPEER. In: CSCWD 2006. Proceeedings of the 10th International Conference of CSCW in Design (2006)
24. http://www.uml.org

The Need for a Process Mining Evaluation Framework in Research and Practice

Position Paper

A. Rozinat, A.K. Alves de Medeiros, C.W. Günther, A.J.M.M. Weijters, and W.M.P. van der Aalst

Eindhoven University of Technology
P.O. Box 513, NL-5600 MB, Eindhoven, The Netherlands
{a.rozinat,a.k.medeiros,c.w.gunther,a.j.m.m.weijters, w.m.p.v.d.aalst}@tue.nl

Abstract. Although there has been much progress in developing process mining algorithms in recent years, no effort has been put in developing a common means of assessing the quality of the models discovered by these algorithms. In this paper, we motivate the need for such an evaluation mechanism, and outline elements of an evaluation framework that is intended to enable (a) process mining researchers to compare the performance of their algorithms, and (b) end users to evaluate the validity of their process mining results.

1 Introduction

Process mining has proven to be a valuable approach that provides new and objective insights into the way business processes are actually conducted within organizations. Taking a set of real executions (the so-called "event log") as the starting point, these techniques attempt to extract non-trivial and useful process information from various perspectives, such as control flow, data flow, organizational structures, and performance characteristics. A common *mining XML* (MXML) log format was defined in [3] to enable researchers and practitioners to share their logs in a standardized way. However, while process mining has reached a certain level of maturity and has been used in a variety of real-life case studies (see [1] for an example), *a common framework to evaluate process mining results is still lacking*. We believe that there is the need for a concrete framework that enables (a) process mining researchers to compare the performance of their algorithms, and (b) end users to evaluate the validity of their process mining results. This paper is a first step into this direction.

The driving element in the process mining domain is some operational process, for example a business process such as an insurance claim handling procedure in an insurance company, or the booking process of a travel agency. Nowadays, many business processes are supported by information systems that help coordinating the steps that need to be performed in the course of the process. Workflow systems, for example, assign work items to employees according to their roles and the status of the process. Typically, these systems record *events* related to the activities that are performed, e.g., in audit trails or transaction logs [3].[1] These event logs form the input for process mining algorithms.

In this paper we focus on providing a means of comparison for algorithms that discover the *control-flow perspective* of a process (which we simply refer to as process discovery algorithms from now on). In particular, we focus on *validation techniques* for these process discovery algorithms. We argue that this evaluation can take place in different dimensions, and identify ingredients that are needed for an evaluation framework. Note that in an extended version of this paper [11] we describe two different validation approaches: one based on existing validation metrics, and another based on the so-called k-fold cross validation technique known from the machine learning domain. We applied both approaches to the running example. Furthermore, in [11] we also present an extensible `Control Flow Benchmark` plug-in to directly support the evaluation and comparison of different mining results in the context of the ProM framework[2].

The remainder of this paper is organized as follows. Section 2 motivates the need for an evaluation framework. Then, Section 3 outlines first steps towards such a common framework. Finally, Section 4 concludes the paper.

2 Process Discovery: Which Model Is the "Best"?

The goal of a process discovery algorithm is to construct a process model which reflects the behavior that has been observed in the event log. Different process modeling languages[3] can be used to capture the causal relationships of the steps, or activities, in the process. The idea of applying process mining in the context of workflow management was first introduced in [5]. Over the last decade many process mining approaches have been proposed [6,9]. While all these approaches aim at the discovery of a "good" process model, often targeting particular challenges (e.g., the mining of loops, or duplicate tasks), they have their limitations and many different event logs and quality measurements are used. Hence, no standard measure is available.

To illustrate the dilemma, we consider the simple example log in Figure 2(a), which contains only five different traces. We applied six different process mining algorithms that are available in ProM and obtained six different process models (for every plug-in, we used the default settings in ProM 4.1). Figure 1 depicts the mining results for the `Alpha` miner [4], the `Heuristic` miner [12], the `Alpha++`

[1] It is important to note that information systems that do not enforce users to follow a particular process often still provide detailed event logs, e.g., hospital information systems, ERP systems etc.

[2] ProM offers a wide range of tools related to process mining and process analysis. Both documentation and software (including the source code) can be downloaded from *http://www.processmining.org*.

[3] In the remainder of this paper we will use Petri nets, motivated by their formal semantics. Note that in our tool ProM there exist translations from process modeling languages such as EPC, YAWL, and BPEL to Petri nets and vice-versa.

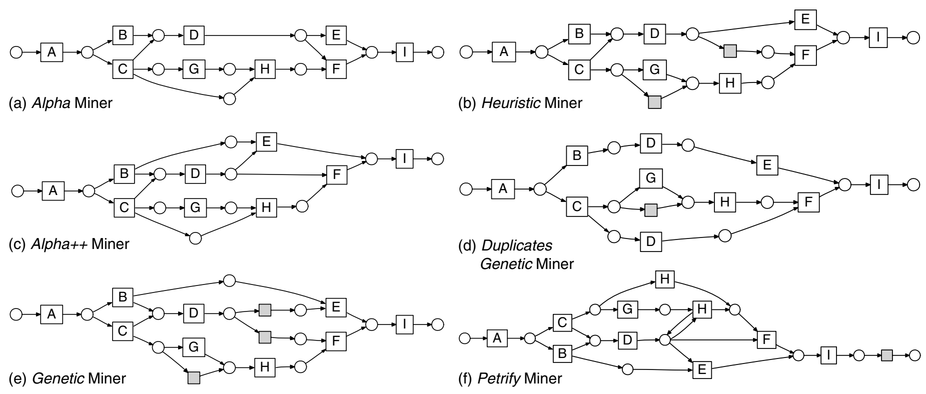


Fig. 1. Process models that were discovered by different process discovery algorithms based on the same log

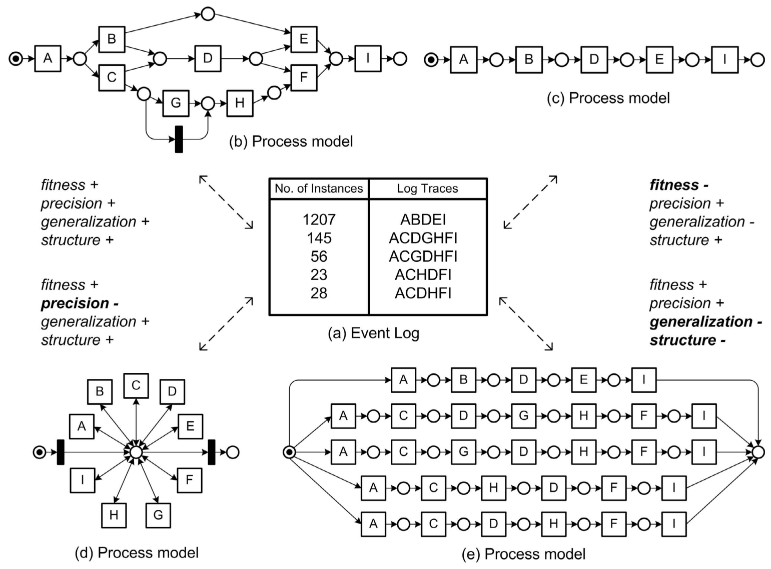
miner [13], the `Duplicates Genetic` miner and the `Genetics` miner [8], and the `Petrify` miner [2]. The models seem similar, but are all different[4]. Are they equivalent? If not, which one is the "best"?

These questions are interesting both for researchers and end users: (a) Researchers typically attempt to let their process discovery algorithms construct process models that completely and precisely reflect the observed behavior in a structurally suitable way. It would be useful to have common data sets containing logs with different characteristics, which can be used within the scientific community to systematically compare the performance of various algorithms in different, controlled environments. (b) Users of process discovery techniques, on the other hand, need to know how well the discovered model describes reality, how many cases are actually covered by the generated process description etc. For example, if in an organization process mining is to be used as a knowledge discovery tool in the context of a Business Process Intelligence (BPI) framework, it must be possible to estimate the "accuracy" of a discovered model, i.e., the "confidence" with which it reflects the underlying process. Furthermore, end users need to be able to compare the results obtained from different process discovery algorithms.

3 Towards a Common Evaluation Framework

In an experimental setting, we usually know the original model that was used to generate an event log. For example, the log in Figure 2(a) was created from the simulation of the process model depicted in Figure 2(b). Knowing this, one could leverage *process equivalence* notions to evaluate the discovered model with respect

[4] Note that throughout this paper the invisible (i.e., unlabeled) tasks need to be interpreted using the so-called "lazy semantics", i.e., they are only fired if they enable a succeeding, visible task [8].



No. of Instances	Log Traces
1207	ABDEI
145	ACDGHFI
56	ACGDHFI
23	ACHDFI
28	ACDHFI

(b) Process model
(c) Process model
(a) Event Log
(d) Process model
(e) Process model
fitness + precision + generalization + structure +
fitness - precision + generalization - structure +
fitness + precision - generalization + structure +
fitness + precision + generalization - structure -

Fig. 2. The evaluation of a process model can take place in different dimensions

to the original model. But in many practical situations no original model is available. However, if we assume that the behavior observed in the log is what really happened (and somehow representative for the operational process at hand), it is possible to compare the discovered model to the event log that was used as input for the discovery algorithm. This essentially results in a *conformance analysis* problem [10,7]. In either case quality criteria need to be determined.

Evaluation Dimensions. Figure 2 depicts an event log (a) and four different process models (b-e). While Figure 2(b) depicts a "good" model for the event log in Figure 2(a), the remaining three models show undesirable, extreme models that might also be returned by a process mining algorithm. They illustrate that the evaluation of an event log and a process model can take place in different, orthogonal dimensions.

Fitness. The first dimension is fitness, which indicates how much of the observed behavior is captured by (i.e., "fits") the process model. For example, the model in Figure 2(c) is only able to reproduce the sequence *ABDEI*, but not the other sequences in the log. Therefore, its fitness is poor.

Precision. The second dimension addresses overly general models. For example, the model in Figure 2(d) allows for the execution of activities $A - I$ in any order (i.e., also the sequences in the log). Therefore, the fitness is good, but the precision is poor. Note that the model in Figure 2(b) is also considered to be a precise model, although it additionally allows for the trace *ACGHDFI* (which is not in the log). Because the number of possible sequences generated by a process model may grow exponentially, it is not likely that all the possible behavior has been observed in a log. Therefore, process mining techniques strive for weakening the notion of *completeness* (i.e., the amount of information a log needs to contain to be able to rediscover the underlying process [4]). For example, they want to detect parallel tasks without the need to observe every possible interleaving between them.

Generalization. The third dimension addresses overly precise models. For example, the model in Figure 2(e) only allows for *exactly* the five sequences from the log. In contrast to the model in Figure 2(b) no generalization was performed. Determining the right level of generalization remains a challenge, especially when dealing with logs that contain *noise* (i.e., distorted data). Similarly, in the context of more unstructured and/or flexible processes, it is essential to further abstract from less important behavior (i.e., restriction rather than generalization). In general, abstraction can lead to the omission of connections between activities, which could mean lower precision or lower fitness (e.g., only capturing the most frequent paths). Furthermore, steps in the process could be left out completely. Therefore, abstraction must be seen as a different evaluation dimension, which needs to be balanced against precision and fitness.

Structure. The last dimension is the structure of a process model, which is determined by the vocabulary of the modeling language (e.g., routing nodes with AND and XOR semantics). Often there are several syntactic ways to express the same behavior, and there may be "preferred" and "less suitable" representations. For example, the fitness and precision of the model in Figure 2(e) are good, but it contains many duplicate tasks, which makes it difficult to read. Clearly, this evaluation dimension highly depends on the process modeling formalism, and is difficult to assess in an objective way as it relates to human modeling capabilities.

Evaluation Framework. To systematically compare process mining algorithms, it would be useful to have common data sets, which can be used and extended by different researchers to "benchmark" their algorithms on a per-dataset basis. For instance, in the machine learning community there are well know data sets (e.g., the UCI Machine Learning Repository, CMU NN-Bench Collection, Proben1, StatLog, ELENA-data, etc.) that can be used for testing and comparing different techniques. Such a process mining repository could be seen as an element in a possible evaluation framework, and should also provide information about the process or log characteristics as these may pose special challenges. Furthermore, the results of an evaluation could be stored for later reference.

At the same time it is necessary to be able to influence both the process and log characteristics. For example, one might want to generate an event log containing noise (i.e., distorting the logged information), or a certain timing behavior (some activities taking more time than others), from a given model. For log generation, simulation tools such as CPN Tools can be used. Another example for log generation is the generation of "forbidden" scenarios as a complement to the actual execution log.

Clearly, many different approaches for evaluation and comparison of the discovered process models are possible. As a first step, in [11] we have looked at existing evaluation techniques both in the process mining and data mining domain.

4 Conclusion

Adequate validation techniques in the process mining domain are needed to evaluate and compare discovered process models both in research and practice. Many obstacles such as bridging the gap between different modeling languages, defining good validation criteria and metrics for the quality of a process model etc. remain, and should be subject to further research. Moreover, a comprehensive set of benchmark examples is needed.

References

1. van der Aalst, W.M.P., Reijers, H.A., Weijters, A.J.M.M., van Dongen, B.F., Alves de Medeiros, A.K., Song, M., Verbeek, H.M.W.: Business Process Mining: An Industrial Application. Information Systems 32(5), 713–732 (2007)
2. van der Aalst, W.M.P., Rubin, V., van Dongen, B.F., Kindler, E., Günther, C.W.: Process Mining: A Two-Step Approach using Transition Systems and Regions. BPM Center Report BPM-06-30, BPMcenter.org (2006)
3. van der Aalst, W.M.P., van Dongen, B.F., Herbst, J., Maruster, L., Schimm, G., Weijters, A.J.M.M.: Workflow Mining: A Survey of Issues and Approaches. Data and Knowledge Engineering 47(2), 237–267 (2003)
4. van der Aalst, W.M.P., Weijters, A.J.M.M., Maruster, L.: Workflow Mining: Discovering Process Models from Event Logs. IEEE Transactions on Knowledge and Data Engineering 16(9), 1128–1142 (2004)
5. Agrawal, R., Gunopulos, D., Leymann, F.: Mining Process Models from Workflow Logs. In: Sixth International Conference on Extending Database Technology, pp. 469–483 (1998)
6. Cook, J.E., Wolf, A.L.: Discovering Models of Software Processes from Event-Based Data. ACM Transactions on Software Engineering and Methodology 7(3), 215–249 (1998)
7. Cook, J.E., Wolf, A.L.: Software Process Validation: Quantitatively Measuring the Correspondence of a Process to a Model. ACM Transactions on Software Engineering and Methodology 8(2), 147–176 (1999)
8. Alves de Medeiros, A.K.: Genetic Process Mining. PhD thesis, Eindhoven University of Technology, Eindhoven (2006)
9. Herbst, J.: A Machine Learning Approach to Workflow Management. In: Lopez de Mantaras, R., Plaza, E. (eds.) ECML 2000. LNCS (LNAI), vol. 1810, pp. 183–194. Springer, Berlin (2000)
10. Rozinat, A., van der Aalst, W.M.P.: Conformance Checking of Processes Based on Monitoring Real Behavior. Information Systems 33(1), 64–95 (2008)
11. Rozinat, A., Alves de Medeiros, A.K., Günther, C.W., Weijters, A.J.M.M., van der Aalst, W.M.P.: Towards an Evaluation Framework for Process Mining Algorithms. BPM Center Report BPM-07-06, BPMcenter.org (2007)
12. Weijters, A.J.M.M., van der Aalst, W.M.P.: Rediscovering Workflow Models from Event-Based Data using Little Thumb. Integrated Computer-Aided Engineering 10(2), 151–162 (2003)
13. Wen, L., Wang, J., Sun, J.G.: Detecting Implicit Dependencies Between Tasks from Event Logs. In: Zhou, X., Li, J., Shen, H.T., Kitsuregawa, M., Zhang, Y. (eds.) APWeb 2006. LNCS, vol. 3841, pp. 591–603. Springer, Heidelberg (2006)

BPD Workshop

Introduction to the Third Workshop on Business Process Design

If one considers business processes that are in use today, be it by banks, insurance companies, governmental agencies, hosptitals, etc., it is sometimes difficult to imagine that they were once consciously designed. Their structures can be extremely complicated, responsibilities for their parts may have been widely distributed, and no particular participant really has an overview of the entire process anymore. Such business processes typically evolved organically over dozens of years, mostly with a focus on mending the operational problems that occurred on a day-to-day basis. But this will no longer do.

Organizations are under increasing pressure both to deliver the performance and to comply with the regulations that stakeholders impose on them. This means that they will need to consider their processes, evaluate them, and – if this is required – design them for the better.

The aim of the third workshop on Business Process Design was to continue the discussions that took place within the setting of previous workshops editions in Nancy and Vienna, to further develop the field of knowledge that is concerned with the disciplined, well-understood and appropriately evaluated design of business processes.

The Call for Papers for this workshop attracted 19 high-quality international submissions. Within a rigorous process, in which each paper was reviewed by at least three experts, we selected seven papers for inclusion in this workshop. The workshop took place in conjunction with the 5th International Conference on Business Process Management, in Brisbane, where authors for all invited papers presented on their work. In addition, Stefan Jablonski from the University of Bayreuth gave a key-note talk on the current developments in the field. The modified versions of the accepted papers are part of this volume.

We are very grateful to the efforts of all authors related to writing, revising and presenting their papers. Finally, we appreciate the indispensable support of the members of the Program Committee, who provided excellent feedback and valuable directions for the authors to improve their work.

November 2007

Tom Davenport
Selma Mansar
Hajo Reijers

Co-chairs

Tom Davenport
School of Executive Education
Babson College at Wellesley
Babson Park, MA 02457-0310
USA

Selma Limam Mansar
College of Business Sciences
Zayed University
P.O. BOX 19282, Dubai
UAE

Hajo Reijers
Department of Technology Management
Eindhoven University of Technology
5600 MB, Pav.D14, Eindhoven
The Netherlands

Workshop Program Committee

Challenges Observed in the Definition of Reference Business Processes

Liming Zhu[1,2], Leon J. Osterweil[3], Mark Staples[1,2], and Udo Kannengiesser[1,2]

[1] Empirical Software Engineering, NICTA, Australian Technology Park, NSW, Australia
[2] School of Computer Science and Engineering, University of New South Wales
[3] Laboratory for Advanced Software Engineering Research (LASER), University of Massachusetts at Amherst, 140 Governors Drive, Amherst, MA 01003
{Liming.Zhu, Mark.Staples, Udo.Kannengiesser}@nicta.com.au, ljo@cs.umass.edu

Abstract. In many modern enterprises, explicit business process definitions facilitate the pursuit of business goals in such ways as best practice reuse, process analysis, process efficiency improvement, and automation. Most real-world business processes are large and complex. Successfully capturing, analysing, and automating these processes requires process definition languages that capture a variety of process aspects with a wealth of details. Most current process modeling languages, such as Business Process Modeling Notation (BPMN), focus on structural control flows among activities while providing inadequate support for other process definition needs. In this paper, we first illustrate these inadequacies through our experiences with a collection of real-world reference business processes from the Australian lending industry. We observe that the most significant inadequacies include lack of resource management, exception handling, process variation, and data flow integration. These identified shortcomings led us to consider the Little-JIL language as a vehicle for defining business processes. Little-JIL addresses the afore-mentioned inadequacies with a number of innovative features. Our investigation concludes that these innovative features are effective in addressing a number of key reference business process definition needs.

1 Introduction

As is the case with many other enterprises in modern society, business looks to processes to facilitate its pursuit of key goals such as greater efficiency, faster response to customers, more effective utilization of resources, and improvements in quality of work [5, 6, 10]. Processes can facilitate the pursuit of these goals because they can prescribe which entities are to perform which tasks at what times. As such, careful examination of process descriptions can help performers to know their jobs better and interact with other performers more effectively. Managers can use process descriptions to identify bottlenecks, inefficiencies, process defects, and faulty coordination. Addressing these problems by modifying processes can then lead to reaching the goals that business seeks. Well defined business processes also lay the foundation for potential automation and e-Business exchanges by way of IT systems across organizations.

While this approach to systematic process improvement seems straightforward, it has proven to be harder than expected to achieve in actual practice [4]. Indeed, there are substantial challenges to be overcome before reaching the enumerated goals [8, 13, 14]. We note, in particular, that real processes generally turn out to be large and complex. Even "easy" processes reveal an unexpected wealth of details when examined carefully. In addition, these details are not simply elaborations of higher level steps into lower level steps. The details required to adequately capture a real-world process are very diverse in multiple dimensions [11]. Thus, a central challenge is to define such large and complex processes in sufficient detail, and in all dimensions. We observe that some of the principal obstacles to doing so are:

Resource Management. The foci of most process definitions are activities and control flows [15, 17]. The rich language features addressing control flow in most process definition languages reflect this. In reality, resources including people, systems, and other physical entities prove to be equally important, if not more important, than activity centric process elements. The use of resources, the ability to specify resource capabilities, and sophisticated policies for resource allocation are some of the important semantic issues in real-world process definition.

Exception Handling. The name Exception can suggest that an exception is a rare, even less important, after-thought in a process definition. In any real-world complex system, however, we observe that exceptions often dominate business scenarios. Handling of exceptions is not merely an extra-step or minor diversion but has major impact on control flow, data flow and resource management [9]. Complete process definitions must encompass articulate specification of exception management. In particular, we note that contexts must be considered in determining exception handling strategy.

Process Variation. Processes always vary, and for different reasons. We want to empower people to act in the most effective way, rather than bind them by prematurely limiting their flexibility in process execution [21]. We need agility and the ability to adapt existing processes to fast-changing business environments. In theory, it is possible to capture such variants in particular versions of a process or through existing "OR"-based control flows. However, we observe that the sheer numbers of possible variations, and the way process models are used, render static representation of variation impractical. Addressing this problem seems to require large-scale process reuse through both planned variation points and property-observing mechanisms.

Reconciliation of Control and Data Flow. While data flows and data-driven information systems are pervasive and vital parts of any business process, process definitions largely treat data items as second-class entities. The definition of the activity aspects of processes should not be isolated from other process aspects such as data [22]. In reality, precise reconciliation of control flows and data flows has become a recurring demand. Such reconciliation improves process understanding, enables process analysis, and guides unambiguous process implementation and deployment.

In addressing all of these issues, we note that it is essential that these complex process details be described carefully and rigorously. Use of a rigorous description language allows for the assignment of precise meanings to process details. This serves as a basis for resolving possible disputes among humans in the validation of the process definition. A rigorously defined process description also serves as a reliable basis for the analysis and implementation of the process, which supports the key business goals of optimization, defect removal, and speed.

Addressing the foregoing issues to achieve the effective exploitation of processes requires the use of a process notation that is detailed, precise, and rigorously defined. In view of the need for such a language to address a particularly broad and challenging spectrum of semantic issues, this is quite a challenge.

In the work that we present here, we describe a specific business process that we have been addressing for quite some time. In seeking benefits such as superior coordination, improved efficiency, automation, and the removal of defects, we have attempted to describe this process using a generally accepted standard process definition language – Business Process Modelling Notation (BPMN) [12] from the Object Management Group (OMG). We describe some of the major challenges that our problem domain has presented, and then indicate ways in which we found BPMN to be unequal to meeting these challenges acceptably well. This list of unmet process description challenges has led us to consider some process definition research languages that seemed to offer the superior process description facilities required by the demands of our problem domain. We describe our experiences in using one of them, Little-JIL, in our attempts to address these challenges. Our experiences with Little-JIL suggest to us the need to incorporate some of this language's stronger semantic features into process description languages if they are to indeed be effective in supporting the pursuit of the goals that business has set for the use of process.

2 Motivating Case

NICTA (National ICT Australia) has been working with a leading Australian e-Business industry standardization body – Lending Industry XML Initiative (LIXI) [1] – that serves the lending industry in Australia. The LIXI e-business standards are composed of XML-based business data models (associated with message exchange patterns), business process reference models, reference architecture and implementations. Reference models are models that are developed explicitly with the goal of being reused for different, but similar purposes while accommodating different stakeholder perspectives. Business process reference models in LIXI are intended to serve multiple purposes, including:

- Capturing standard and best practices in the lending industry at different levels of abstraction, with reference to IT systems and people within the eco-system.
- Describing how business objects in the data standards are exchanged among process elements of the eco-system and transformed within process elements.
- Being the high-level inputs to model-driven transformations resulting in implementation models, such as web service coordination models and executable models expressed in Business Process Execution Language (BPEL).

- Evaluating standards compliance for individual implementations of the standard processes.
- Performing process analysis for achieving business goals of optimization, defect removal, and speed across the whole lending industry.

In this work, we use the property valuation part of the larger LIXI business process as a motivating example. During a loan application process, a lending organization needs to determine the market value of loan-related property. This process is called "valuation" and is usually conducted by a valuation firm. The process is typically represented by a workflow description that starts with an initial valuation request. The request contains information such as the address of the property, the type of valuation (desktop or curb-side) and urgency. The request progresses through a number of processing stages within the valuation firm until it is completed and a response is sent back to the requestor. The initiation of a valuation request and the downstream activities after a valuation also interface with other process elements in the LIXI business processes. Most activities are long-lasting business transactions which may span several days. During that time, backchannel messages regarding the progress of the valuation are exchanged upon requests or status changes. Cancelling requests, amending requests and fee negotiations can happen at any time. They may result in variant execution paths and exceptions affecting both the current sub-process and the whole gamut of LIXI processes and involved organizations. Many activities involve human-aided interactions, depending on the automation sophistication of participants and nature of the activities. Intermediaries who act as proxies of lending institutions and valuation firms may be involved. Some intermediaries are legal entities that act on behalf of multiple valuation firms or lending institutions. Other intermediaries are simply technical mediators who provide almost transparent infrastructure for transaction mediation. Fee negotiation is allowed at any time during the request.

A major goal of LIXI business processes is to support highly interoperable and efficient e-Business transactions within the lending industry. Accordingly, it is expected that all business process definitions will eventually be mapped to software implementations through Web services or other technologies. Thus, we have produced reference implementations using BPEL and Web services. In doing so we have identified the challenges enumerated in Section 1 of this paper. Specifically:

Resource Management. The LIXI eco-system does not have clear people/system boundaries. The scale of the companies involved varies widely. Some companies have sophisticated systems that can automate most tasks while others still rely on fax and manual processing. Messages and activities in the e-Business standards can map to different systems or people depending on resource capabilities, availability, mobility and rules of allocation of specific process actors. For this reason, our process definitions need: 1) a more flexible abstraction for defining the process actors and the ways in which they are to participate in a defined process, 2) an ability to express management of allocation policies, and 3) a way to specify when and where process performers are to be allowed the flexibility of autonomous behavior.

Exception Handling. The LIXI business processes are large, complex and long lasting. A home loan process lasts weeks if not months during its application stage and requires servicing and refinancing spanning decades. The long lasting nature demands process definition capabilities for being precise about timing constraints and compensational policies that may vary depending upon contexts and scopes.

In addition, exception handling across organizations and policy changes is complicated. Thus we need exception handlers that can be dynamically determined, and that consider execution context rather than being pre-defined. Scoping is essential in such scenarios. Also, it is equally important to have the ability to specify precisely and flexibly what happens after the handling of an exception has been completed.

Process Variation. The following LIXI business process characteristics demand high flexibility:

- In a system as large as LIXI, inherently conflicting requirements exist. Most parties in LIXI want complexity to reside in others' parts of the overall system, want information to be shared, but do not want to share their own information. Technical solution companies provide and favor intermediary gateways and custom-built applications, while smaller players typically want commoditized applications and to remove intermediaries. LIXI reference business process definitions can not impose a single structured static process.
- As a business-to-business process standard, LIXI business processes govern cross-boundary transactions and leave private business processes behind organizational boundaries as their competitive advantage. However, there is a subtle balance between promoting interoperability through prescriptive standards and allowing innovation through minimal prescription.
- LIXI business processes require multiple-levels of abstraction and serve different purposes, from high level communication to process analysis and implementation

We need a number of features in process definition to address the variation issue. Among them are:

- Process variations that can be expressed either through explicit enumeration or, preferably, through a process property, pre/post condition driven approach.
- A mixture of both proactive and reactive control mechanisms for variant process coordination.
- The use of resources and data as means of constraining and managing process execution, in order to supplement more conventional control flow capabilities.
- Rigorous analysability of the process definition so that we can determine whether a particular process variation is standard-compliant.
- A mechanism for hierarchical abstraction decomposition that allows a single model to be used in different contexts

Reconciliation of Control and Data Flow. In addition to business process models, LIXI also has data model standards that exist in the form of a controlled vocabulary and XML schemas. There is a high demand within LIXI among both managers and developers to cross-reference control flows with data flows rigorously. We need to be able to express the data and artifacts used by an activity. Data flows must not be confined to the control flow and data access scope. A more flexible channel mechanism associable with control flow is needed to allow maximum data flow flexibility without compromising activity views.

3 Limitations of Traditional Workflow Languages

As LIXI is a standardization body, we initially decided to promote further standards use by adopting the international standard notation BPMN for our process definitions.

Using BPMN models did serve the purpose of standards-based communication, which leverages existing expertise in use of the standard and in tooling support for graphic modeling. However, we found it did not meet our other business process needs, and did not address the process definition challenges previously enumerated.

Resource Management. Using roles to express process participants in BPMN models is limited to mapping activities to role-based swim lanes [20]. Thus, there is no satisfactory way to express resource allocation mechanisms other than by direct role-based allocation. Overall, facilities for specifying policies for the allocation of resources are very limited. As discussed earlier, we need capability-based and other types of rule-based resource allocation mechanisms that can be associated with an autonomous process actor.

Exception Handling. Scoping and dynamic handler determination is not supported in BPMN. The ability to specify what to do after an exception is handled is also limited to structural control flows and some forms of resource re-allocation (still solely through the use of concrete role-based capabilities [16]). More sophisticated exception handling mechanisms are needed [7].

Process Variation. Expressiveness of process variability is limited to static structural enumeration of possibilities. Parameterized activities are achieved through implicit data sharing. There is no direct way to express overall process properties or pre/post/invariants for individual activities. This limits the specification of sufficiently broad variation, and also hinders the possibility of systematic process analysis. Levels of abstraction are of paramount importance in LIXI business processes. Abstraction specification capabilities must enable specification of references to internal private business processes and detailed process implementations as well as communications among different stakeholders. Every activity must be decomposable. BPMN offers abstraction only through sub-workflows, which do not offer adequate abstraction and decomposition semantics. Thus, for example, sub-workflows do not incorporate parameter binding semantics needed to implement context-sensitive reuse. Such limitations can be addressed by using more configuration-based approach [3].

Reconciliation of Control and Data Flow. Annotating fine-grained data elements in a BPMN process model is largely limited to associating information artefacts to control flow edges [18]. Data sharing across arbitrary activities not connected by direct control flow is limited to the use of data access scopes. BPMN models also lack precise definitions, thus raising ambiguities in their translation into implementation models such as BPEL. Strict rules defining the semantics of BPMN are needed to support its deterministic translation.

The issue is not that BPMN is not expressive enough relative to other workflow languages. A systematic comparison shows that traditional block-based workflow languages, such as UML activity diagrams, BPEL, and XML Process Definition Language (XPDL), are more or less similar to BPMN. Indeed, because of the non-executable and informal nature of BPMN, it is often considered to be more expressive than these other notations. But, as the foregoing has indicated, BPMN still lacks key expressive capabilities that are needed in order to specify LIXI processes. Another notable effort is YAWL (Yet Another Workflow Language) [2] and the latest newYAWL [19] that cover most of the workflow patterns and have better support for resource modeling and exception handling. However, they are still token-passing based discrete workflow languages which do not address some process definition challenges, such as modeling continuous behavior, property-based variation flexibility and more flexible parameter passing between activities.

4 Little-JIL

In order to address the business process definition needs we have identified, we sought more innovative process definition languages from the research community.

Little-JIL is a language originally developed for defining the processes by which software is developed and maintained. Wise [23] provides full technical details of the language. A Little-JIL process is defined by specifying three components: an artifact collection, a resource repository, and a coordination specification. Each addresses a different area of concern. The artifact collection contains the various items, initial, intermediate, and final, that are the focus of the activities carried out by the process. The resource repository specifies the agents and other capabilities that are available to support performance of the activities. The coordination specification ties these together by specifying precisely which agents, aided by which supplementary capabilities, will perform which activities upon which artifacts at which exact time(s). Because of its central role in specifying all of this, the coordination diagram is generally the central focus of a Little-JIL process definition.

A Little-JIL coordination diagram is essentially a hierarchical decomposition of steps, where a step represents a task to be done by an assigned agent. Each step has a name and a set of badges to represent control flow among its sub-steps, its interface (a specification of its input/output artifacts and the resources it requires), the exceptions it handles, etc. A step with no sub-steps is called a leaf step and represents an activity to be performed by an agent, without any guidance from the process.

We have found that a number of features of Little-JIL are particularly useful in addressing our needs, especially as compared to other languages. Among the key features of Little-JIL that distinguish it from most process languages are 1) its use of abstraction to support scalability and clarity, 2) its use of scoping to make the use of step parameterization clear, 3) its facilities for specifying both artifact and data flow in a single notation, 4) its extensive capabilities for defining how to handle exceptional conditions, and 5) the clarity with which iteration can be specified and controlled. These capabilities allow Little-JIL to address the previously enumerated process definition needs relatively more successfully. To make this clearer, we now show how Little-JIL can be used to define part of the valuation process.

The purpose of this process, as noted above, is to show how managers, valuers, and clients collaborate to arrive at a valuation of a property. As will be seen, this entails some iteration, and must take into account the participation of various agents, as well as the possibility of handling contingencies of various kinds. Figure 1 defines the high level of this process, which we refer to as **Check Property Value**. We define this as a hierarchical decomposition into three substeps, **Assign Valuer, Perform Inspection,** and **Propose Valuation Response**. The right arrow in the **Check Property Value** step bar indicates that these three substeps are to be executed in sequence. Little-JIL also supports specifying that substeps can be executed in parallel as well, although this capability is not shown in this example. Annotations on the edges between the parent step and its children define the flow of process artifacts. Thus, in particular, note that the process begins with the flow of an artifact, **Prop ID** from **Check Property Value** to **Assign Valuer** (the downward arrow indicates that this artifact is passed from parent to child). Execution of the next substep, **Perform Inspection,** also receives **Prop ID** as an input, and in addition produces **Inspection Data** as an output (note the upward pointing arrow next to this artifact). It is important to note that the specification of artifact flows in Little-JIL is done both by annotating edges as shown, and also by specifying the input/output behavior of a step by attaching to the step's external interface icon (the round circle atop each step) an enumeration of the artifacts that are taken as input artifacts, and those that are produced as output artifacts. Note that a Little-JIL step can be thought of as a procedure, and thus these annotations function as arguments to the procedure.

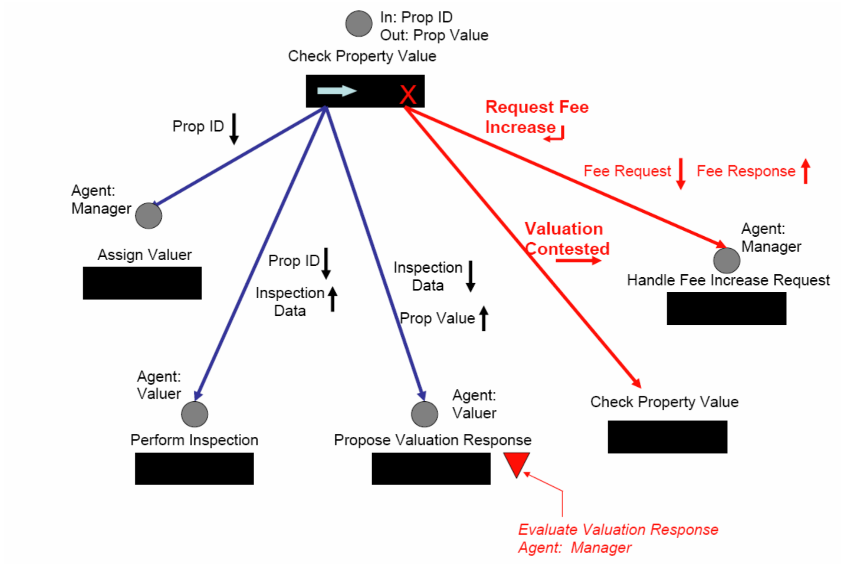


Fig. 1. Valuation Process Modeled in Little-JIL

Figure 1 also shows that each step is annotated with a specification of the type of agent that is required in order to perform the step. Thus, note that the agent for **Assign Valuer** is of type **Manager**, while the agent for **Perform Inspection** is of type **Valuer**. In all cases, these specifications indicate the type, rather than the specific instance, that is required to perform the step. The type specification is passed at runtime by the Little-JIL interpreter to a resource respository that uses the specification to select the specific agent that will be assigned to perform the step.

Figure 1 also shows that each step incorporates the ability to handle exceptions. Thus, note that **Check Property Value** also has two substeps, connected to the parent by red edges, emanating from the red X on the right of the step bar. Each such edge is annotated (using bold face type) by the type of the exception that triggers execution of the step on its end. Thus, for example, note that a **Valuation Contested** exception can be raised, in this case by executing the postcondition (indicated by a red downward pointing arrowhead) of the **Propose Valuation Response** step, and getting a negative outcome (ie. that that valuation has been rejected). Figure 1 has attached to it an informal comment (indicated by the use of italics) noting that this postcondition is in fact an entire step, **Evaluate Valuation Response**, and that it is to be executed by an agent of type Manager. The response to this outcome is the execution of the step at the end of the exception handling step bearing the **Valuation Contested** label. Note that this step is a recursive invocation of the **Check Property Value** step. In this case, the step is reinvoked in the context of the rejection of the valuation, thereby enabling the agent assigned to perform the reevaluation to understand that this is being done in the context of this rejection. Finally note that there is a right arrow under the **Valuation Contested** label, which indicates that once the exception has been handled, control continues as though the parent, **Check Property Value**, has completed.

This availability of context may be seen more clearly by examining the handling of the **Request Fee Increase** exception, which is done by another exception handler substep of the **Check Property Value** step. In this case, the exception is raised by the execution of one of the substeps of **Check Property Value** (rather than by a step postcondition), and arises when the agent decides that the fee offered for the valuation is inadequate. This exception is handled by another step, and in this case we see that its invocation has defined arguments, namely **Fee Request** (as an input) and **Fee Response** (as an output). Finally note that the specification of the type of the exception also includes an angled arrow that points to the left, which indicates that, once the exception has been handled, control returns to the location where the exception was raised. This is as needed, indicating that the outcome of the request was a response to the request for a fee increase, at which time execution must resume.

Note also that **Handle Fee Increase Request** is a step that is defined in further detail by the diagram shown in Figure 2, illustrating the use of hierarchical decomposition, but also indicating that the elaboration is indeed an invocation of a procedure, taking the indicated artifacts as its arguments. Thus, note that the annotations on the circular external interface icon show that the arguments are assigned as the values of the parameters, **Request** (an input), and **Fee Response** (an output). This step is decomposed into two substeps, and one exception handler substep. The first substep is **Manager Discussion**, and the second substep is **Client Consideration**. Each is labeled with an appropriate type of agent and the appropriate artifact flow specifications. Note that each has a postcondition that specifies what is to be done in case the agent does not go along with the request. Each postcondition throws the **Request Denied** exception, which is handled as the invocation of the procedure that is represented by the appropriately labeled exception handler. In this case, there is no step required to handle the exception, only the binding of the value of the **Fee** artifact to the **Fee Response** artifact (this is indicated by the documentation shown in italics), and by the right arrow at the end of the exception edge, which indicates that after this binding, control continues as though the parent step, **Handle Fee Increase Request**, has concluded, at which time control is returned to the part of the process depicted in Figure 1. Finally note that if neither substep raises an exception, then the second substep, **Client Consideration**, terminates by binding the value of the **Request** argument artifact as the value of the **Fee Response** artifact, properly reflecting that the request has been approved.

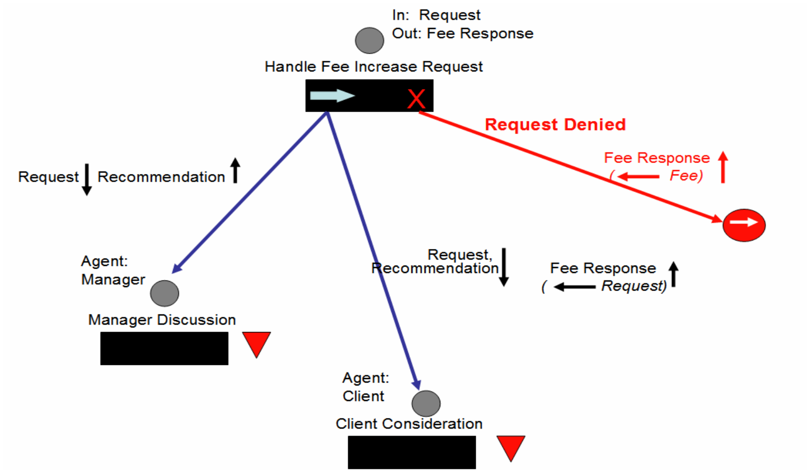


Fig. 2. Exception Handling

In this short example, we see that Little-JIL is capable of clearly and tersely capturing very complex control flow variation, and meshing it successfully with artifact flow. The handling of the request for a fee increase indicates this nicely, showing how a variety of decisions will affect both data and control flow. It also shows how powerful exception handling helps a great deal. In this case we show how the exception can arise in different places, but is channeled to the same exception handler, which itself has further exception handling capability. Despite this structure, it is clear how execution is guided back to the right location before proceeding.

Little-JIL has previously been used to define processes in such areas as healthcare provision, labor-management negotiation, ecological data processing, and engineering. In all of these domains we have found that end-users are willing and able to read the Little-JIL definitions rather successfully. We do not advocate that end-users write the Little-JIL definitions, but it is reasonable to expect that they can read them to benefit from some of the advantages.

5 Discussion and Conclusions

Our example has shown the value of late binding of agents to steps, making it clear that a specified capability is what is needed by a particular step, but leaving the identification of the agent for late binding. Thus we see that Little-JIL's **Resource Management** capabilities meet some of our important needs. The example also indicates the complex **Exception Management** needs of LIXI processes, and likewise demonstrates that Little-JIL's powerful exception management capabilities are needed here. The example only begins to show the value of incorporating abstraction into a process language, indicating the value of thinking of a step as a procedure. In this example we see the value of parameter passing and the use of scoping. All of these features add greatly to the ability of Little-JIL to represent important forms of **Process Variation** that are lacking in most other process languages that we have investigated. These same features also seem to make the **Reconciliation of Process and Data Flow** views of our processes clearer. Thus we see how Little-JIL seems effective in addressing the four areas of need identified early in this paper.

Indeed, our experience suggests that there is further value to be derived from Little-JIL language features. We note in particular that the use of abstraction in Little-JIL fosters reuse, clarity, and terseness of expression of complex process content. We have also found that the ability to specify concurrency is extremely important, as are other Little-JIL language semantic capabilities that we do not elaborate upon here due to the lack of space in this paper.

We will continue to use BPMN models in some aspects of the process definition, mainly because of its status as an international standard and because of its abundant tooling support in graphical modeling. On the other hand, in order to meet all the needs that a process definition has to support, and to address unique challenges in the highly dynamic and variable nature of process definitions in domains such as LIXI (i.e. industry-wide highly adaptable reference processes), a more sophisticated process definition language must be used. Initial experience with Little-JIL has largely met all of these needs. Further work will be done to fully explore the advanced features of Little-JIL. In addition, we note that preliminary work indicates that Little-JIL's strong semantic basis renders processes defined in the language amenable to analysis by powerful finite state verification tools. We will also explore leveraging the use of these analysis tools by applying them to LIXI process definitions.

Acknowledgements

NICTA is funded by the Australian Government's Department of Communications, Information Technology, and the Arts and the Australian Research Council through Backing Australia's Ability and the ICT Research Centre of Excellence programs.

This research was also supported by the US National Science Foundation under Award Nos. CCR-0204321 and CCR-0205575. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. National Science Foundation, or the U.S. Government.

References

[1] Lending Industry XML Initiative (LIXI), http://www.lixi.org.au

[2] van der Aalst, W.M.P., ter Hofstede, A.H.M.: YAWL: Yet Another Workflow Language. Information Systems 30(4), 245–275 (2005)

[3] Becker, J., Delfmann, P., Dreiling, A., Knackstedt, R., Kuropka, D.: Configurative Process Modeling - Outlining an Approach to Increased Business Process Model Usability. In: Information Resources Management Association Conference (2004)

[4] Desai, N., Mallya, A.K., Chopra, A.K., Singh, M.P.: Interaction protocols as design abstractions for business processes. IEEE Transaction on Software Engineering 31(12), 1015–1027 (2005)

[5] Eriksson, H.-E., Penker, M.: Business modeling with UML: Business Patterns at Work. John Wiley & Sons, New York (2000)

[6] Georgakopoulos, D., Hornick, M.F., Sheth, A.P.: An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure. Distributed and Parallel Database (3), 119–153 (1995)

[7] Grigori, D., Casati, F., Dayal, U., Shan, M.-C.: Improving Business Process Quality through Exception Understanding, Prediction, and Prevention. In: 27th International Conference on Very Large Data Bases (VLDB) (2001)

[8] Gruhn, V., Laue, R.: What business process modelers can learn from programmers. Science of Computer Programming 65(1), 4–13 (2007)

[9] Klein, M., Dellarocas, C.: A Knowledge-Based Approach to Handling Exceptions in Workflow Systems. Computer Supported Cooperative Work (CSCW) 9, 339–412 (2000)

[10] Laguna, M., Marklund, J.: Business process modeling, simulation, and design. Pearson/Prentice Hall, Upper Saddle River, NJ (2004)

[11] Muehlen, Z., Rosemann, M.M.: Multiparadigm process management. In: The Fifth Workshop on Business Process Modeling, Development, and Support (BPMDS) (2004)

[12] OMG, Business Process Modeling Notation Specification (version 1.0 Final Adopted Version) (2006)

[13] Osterweil, L.: Software Processes Are Software, Too, Revisited. In: 19th International Conference on Software Engineering (ICSE), Boston, MA, pp. 540–558 (1997)

[14] Osterweil, L., Sondheimer, N.K., Clarke, L.A., Katsh, E., Rainey, D.: Using Process Definitions to Facilitate the Specifications of Requirements. In: Department of Computer Science, University of Massachusetts, Amherst, MA (2006)

[15] Rolland, C.: A comprehensive View of Process Engineering. In: Pernici, B., Thanos, C. (eds.) CAiSE 1998. LNCS, vol. 1413, Springer, Heidelberg (1998)

[16] Russell, N., van der Aalst, W.M.P., ter Hofstede, A.H.M.: Exception Handling Patterns in Process-Aware Information Systems, BPM Center Report BPM-06-04 (2006)

[17] Russell, N., ter Hofstede, A.H.M., van der Aalst, W.M.P., Mulyar, N.: Workflow Control-Flow Patterns: A Revised View, BPMcenter.org BPM Center Report BPM-06-22 (2006)

[18] Russell, N., ter Hofstede, A.H.M., Edmond, D., van der Aalst, W.M.P.: Workflow Data Patterns, Queensland University of Technology FIT-TR-2004-01 (2004)

[19] Russell, N., ter Hofstede, A.H.M., Edmond, D., van der Aalst, W.M.P.: newYAWL: Achieving Comprehensive Patterns Support in Workflow for the Control-Flow, Data and Resource Perspectives, BPMcenter.org (2007)

[20] Russell, N., ter Hofstede, A.H.M., Edmond, D., van der Aalst, W.M.P.: Workflow Resource Patterns, Eindhoven University of Technology BETA Working Paper Series, WP. 127 (2004)

[21] Simidchieva, B.I., Clarke, L.A., Osterweil, L.J.: Representing Process Variation with a Process Family. In: International Conference on Software Process (ICSP) (2007)
[22] Ward, P.T.: The transformation schema: An extension of the data flow diagram to represent control and timing. IEEE Transaction on Software Engineering 12(2), 198–210 (2005)
[23] Wise, A.: Little-JIL 1.5 Language Report, Department of Computer Science, University of Massachusetts, Amherst, MA. (2006)

Trade-Offs in the Performance of Workflows – Quantifying the Impact of Best Practices

M.H. Jansen-Vullers, P.A.M. Kleingeld, M.W.N.C. Loosschilder, M. Netjes, and H.A. Reijers

Department of Technology Management, Eindhoven University of Technology
P.O. Box 513, NL-5600 MB, Eindhoven, The Netherlands
{m.h.jansen-vullers, p.a.m.kleingeld, m.netjes, h.a.reijers}@tue.nl

Abstract. Business process redesign is one of the most powerful ways to boost business performance and to improve customer satisfaction [14]. A possible approach to business process redesign is using redesign best practices. A previous study identified a set of 29 different redesign best practices [18]. However, little is known about the exact impact of these redesign best practices on workflow performance.

This study proposes an approach that can be used to quantify the impact of a business process redesign project on all dimensions of workflow performance. The approach consists of a large set of performance measures and a simulation toolkit. It supports the quantification of the impact of the implementation of redesign best practices, in order to determine what best practice or combination of best practices leads to the most favorable effect in a specific business process.

The approach is developed based on a quantification project for the parallel best practice [8] and is validated with two other quantification projects, namely for the knockout and triage best practices.

Keywords: Business Process Redesign, Business Process Simulation, Best Practices, Performance Measurement.

1 Introduction

The domain of business process redesign can roughly be divided into two different approaches: the revolutionary and the evolutionary approach. In the revolutionary approach, a redesign starts from a clean sheet. In the evolutionary approach, the existing business process is taken as a starting point. An example of this approach is the application of redesign best practices. Reijers provided an overview of all best practices currently encountered in literature [18]. Further, a rough qualitative estimation of the expected impact was given [19]. However, quantitative research is necessary to determine a more concrete impact of one or more redesign best practices on the performance of a workflow.

Although not much is known about the impact of redesign best practices on the performance of a workflow, some papers have been found that are based on a quantitative study. These studies include several best practices: knockout best practice [1], extra resources best practice [6], specialist-generalist best practice [6,17], flexible assignment best practice [17] and task composition, triage and case types best practice [20].

The main shortcoming of the above mentioned literature is that none of the authors, with the exception of [1] provided guidelines for the redesign of workflows: what best practice should be applied in what situation, process, or setting? Other deficiencies are the lack of a general approach to quantify the impact of best practices, the limited number of different dimensions of performance, and the limited number of aspects per measured dimension. Further, none of the authors, with the exception of [17], quantified the impact of the simultaneous implementation of more than one best practice.

In our research, we aimed to quantify redesign best practices on as many dimensions as possible. This paper provides an overview of possible performance dimensions and related performance measures. These performance measures have been applied in a simulation study to quantify the impact of a redesign best practice, i.e. the parallel best practice. In the parallel best practice one considers whether tasks may be executed in parallel.

The setup of the paper is as follows. In Section s:perf the dimensions of performance are summarized. In Section s:plan the quantification approach is introduced, including the setup of the simulations, the approach when comparing different variants, and the statistical analysis. We carried out three simulation projects; one to develop the approach and two to validate it. The results of these simulations (i.e. the impact on the identified performance measures) for the best practices involved are shown in Section s:results. The paper concludes with a discussion of the results.

2 Performance Measurement

This study focused on the quantification of the impact of a redesign best practice on the performance of a business process. Subject of study was the business process that is being redesigned, in contrast to, for example, the performance of individual employees or entire organizations.

In the last twenty years a variety of performance measurement systems has been developed. We assessed the literature on this subject to see what dimensions of performance the authors discerned and which are suitable for measuring business process performance. The following six systems have been considered: Performance pyramid [5], Performance measurement matrix [10], Results/determinants matrix [4], Balanced scorecard [9], Devil's quadrangle [3] and Performance prism [2]. The assessment resulted in five dimensions of performance: time, cost, external quality, internal quality, and flexibility. These dimensions are all present in the devil's quadrangle. Furthermore, the other performance measurement systems do not provide additional relevant dimensions. An extensive overview and validation of the dimensions, the relevant measures per dimension and their operationalization can be found in [8]. Here, we suffice with a brief overview.

The Time Dimension. Time has been described as both a source of competitive advantage and a fundamental measure of performance. Based on the information on time measurements found in the literature, we derived a set of performance measures for the time dimension, specifically for workflows, consisting of lead time and throughput time.

Lead time is the time it takes to handle an entire case. Throughput time is the time between the moment a task is completed and the moment the next task is completed. Throughput time is composed of: service time, queue time, wait time, move time, and setup time.

The Cost Dimension. The cost dimension is closely related to the other dimensions. For example, long lead times can result in a more costly process, low quality can lead to expensive rework, and low flexibility can also result in a more costly process execution. Focusing on the direct costs of running a process, we discerned running costs (for labor, machinery, training), inventory costs, transport costs, administrative costs, and resource utilization costs.

The External Quality Dimension. The quality of a workflow can be judged from at least two angles. External quality is defined from the customer's side, i.e., the person or organization that initiates the workflow and will receive the output. Internal quality is defined from the worker's side.

External quality can be measured as client satisfaction with either the product (output) or the process. Satisfaction with the product is the degree to which the customer feels that the product is according to specification or feels satisfaction with the delivered product. The satisfaction of a customer with the process relates to the way a workflow is executed [18]. Literature has been found on both the quality of a product and the quality of a process. Quality of the output takes into account product performance, conformance and serviceability, whereas quality of the process considers information availability and bureaucratic language simplification. These measures were included in our study.

The Internal Quality Dimension. Internal quality can be seen as the quality of a workflow from an operator's perspective. In this context, internal quality involves the working conditions. Task design characteristics and social factors are very important. High internal quality can result in high motivation, high job satisfaction, high psychological well-being, high external quality, and low absenteeism.

The Flexibility Dimension. Flexibility is the least noted criterion to measure the effect of a redesign effort. Flexibility can be defined as "the ability to react to changes". It appears that flexibility can be identified for individual resources, for individual tasks, and for the workflow (process) as a whole. Five types of flexibility can be distinguished. Mix flexibility is the ability to process different kinds of cases (per resource, task, or workflow). Labor flexibility reflects the ability to perform different tasks (per resource or per workflow). On the workflow level we further distinguished routing flexibility (the ability to process a case by using multiple routes, i.e. the number of different sequences in the workflow), volume flexibility (the ability to handle changing volumes of input) and process modification flexibility (the ability to modify the process, e.g., the number of sub flows in the workflow, complexity, number of outsourced tasks, etc.)

Operationalization. Operationalization of the time, cost, and flexibility dimensions is quite straightforward. Measuring internal and external quality in a workflow model is less straightforward than measuring time or costs because many different factors influence and determine quality. For example, with respect to internal quality differences among people moderate how they react to the complexity and challenge of their work [7]. To settle this, we decided to list (measurable) aspects of those dimensions and consider them proxies: a change in one or more of the aspects will have some impact on the quality dimension. However, the exact extent of impact cannot be determined in a simulation model.

3 Quantification Approach

Based on the quantification project performed for the parallel best practice, a generalized quantification approach was developed. This approach starts with a redesign quantification plan, based on [12] and [15]. The plan consists of 8 steps, of which steps 1 to 4 are mainly general steps in a simulation study: (1) project definition, (2) definition and building of a model of the original situation, (3) validation of this model, and (4) definition and building of a model of the redesigned situation. Step 5 (design of the experiments), step 6 (execution of the simulation runs), and step 7 (analysis of the output) are more specific for this kind of quantification projects. Finally, in step 8, conclusions are drawn.

3.1 The Redesign Best Practices Quantification Plan

1. Project definition. The main objective of a quantification project is the collection of evidence to reject or support a proposition. In this case the impact of the implementation of a certain redesign best practice was quantified. Literature can be used to set the objectives. The work of Reijers [18] can be used as a literature guide.
2. Definition and building of a model of the original situation. We created a high-level Petri net model of the original situation in CPN Tools, which could be used as a starting point for the simulations jensenboek97. The model can be used directly or changed where necessary in order to measure the impact of a certain best practice. The model is very flexible and easy to adapt and also includes monitors for the specified operational performance measures.
3. Validation of the model. Our basic model was validated through a comparison of the results of the simulation with the analytical outcomes of mathematical queuing models [15]. The mathematical model is a network of queues, i.e. a Jackson network [11].

   With the formulas of Kulkarni [11] a number of performance measures could be calculated: utilization of the resources, expected number of cases in the queue, expected queuing time, and expected time of a case in the system. After simulation of the CPN model, the results were collected and analyzed, and the 95% confidence intervals were calculated.

4. Definition and building a model of the redesigned situation. Based on the model of the original situation, a redesign was created. Again, the work of Reijers [18] could be used as a literature guide to acquire detailed insight. The CPN model of the original situation can be adapted to benefit from the structure and monitors already available.
5. Design of the experiments. This step consists of five sub steps that should be followed before the actual simulation runs can be executed. These sub steps are a very important part of the project, because the correct setup of the simulations is essential for the success of the simulation project. The first two sub steps concern the selection of introducible variations. The parameters of the simulations are calculated in the remaining sub steps.

   Choice of variations
   Variations are introduced in the simulation models of the original and redesigned situation, to test the impact of a specific best practice under different settings. Variations in arrival rates, resource classes, number of resources, service times, and resource skills are examples of introducible variations. The types and degrees of variation should be chosen in such a way that eventually conclusions can be drawn about the impact of the implementation of the best practice in different situations.

   Specification of model variants
   Model variants specify what combinations of variations are used. An example of a model variant is a model with a high arrival rate, low service times, and two resource classes. The number of variations and model variants determines the number of simulation runs.

   Calculation of the warm-up period
   The warm-up period is the amount of time a model needs to come to steady state. In this study the time series method was used to calculate this. This was done based on a pilot run of 20 replications and the calculation of the WIP costs (Work In Progress) in relation to the model time [15]. This resulted in a warm-up length of 4800 minutes (=2 simulation weeks).

   Determination of run length
   CPN Tools resets the model after every replication. We assumed that the seed of the random generator in CPN Tools produced independent number streams and that the results thus were independent. We used a run length of 10 working weeks. As the warm-up length was 4800 minutes, there were 19200 minutes remaining for data collection.

   Calculation of the number of replications
   Due to the very nature of random numbers, it is imprudent to draw conclusions from a model based on the results generated by a single model run [15]. We adopted the approach proposed in [12] to calculate the number of

replications based on a pre-specified precision of the collected data. As a result, 21 replications were used in this study.

6. Execution of the simulation runs. In this step all original and redesigned models are created and simulated and the results are recorded and stored. The simulations are set up according to the parameters (calculated in the previous step) and all performance measures (specified in step 1) are measured. One should bear in mind that simulation of the models of all model variants in CPN Tools requires a lot of time and computer power.
7. Analysis of the output. Before the actual analysis of the output data can be done, the comparisons between the different model variants are determined. It is decided what model variants need to be compared in order to comply with the objectives. For example: two model variants with equal resource setups and service times but different arrival rates can be compared, if one of the sub-objectives is to determine what the impact of a certain best practice is on systems with different arrival rates. The selected comparisons form the basis of the analysis of the output data.

   When comparing results of simulated real systems, equality of variance cannot be assumed. Therefore a separate-variance-t-test such as the Welch test is recommended as it is more reliable and conservative [12]. Thus, the hypothesis H_0 was tested against H_1 for every performance measure by means of the Welch approach, in order to see what performance measures change significantly in the redesigned model.

   When comparing more than two alternatives and calculating several confidence interval statements simultaneously, the individual confidence levels of the separate comparisons have to be adjusted upwards, in order to reduce the number of Type 1 errors (rejecting the null hypothesis when it is true). For this purpose, the Bonferroni equality can be used [12,16].

   Then the confidence intervals for all differences between the original model and the redesigned model (the Welch confidence intervals with the Bonferroni corrected values) are calculated and this is repeated for all setups and all variants. When the confidence intervals of two or more setups overlap, it can be concluded that the difference between these setups is not significant. Conclusions can be drawn both within and between different model variants.
8. Conclusions. Finally, conclusions are drawn based on the analysis and the sub-conclusions of the model variants. Furthermore, a reflection on the quantification is made by comparing the quantitative results and conclusions of the simulation project with the qualitative results of the research of Reijers and Limam Mansar [14,19] and possibly with earlier quantification efforts found in the literature.

3.2 Validation of the Quantification Approach

The quantification approach consists of three elements: (1) the set of performance measures, (2) the quantification plan, and (3) auxiliary files to support the execution of the quantification plan. The basis of the approach is the redesign best practices quantification plan, which should be followed step by step in the simulation process. The auxiliary files (several MS Excel sheets, CPN Tools simulation models and user guides) were created for use in combination with the quantification plan. The files and models are created to increase the consistency of the project, to increase the usability and to save time when quantifying redesign best practices. This holds true for the design of the model, but especially for the monitors in the model that automatically measure all operationalized performance measures. Together, these tools, the performance measures, and the redesign best practices quantification plan form the quantification approach.

The approach has been developed with the simulations of the parallel best practice and validated with the quantification of the knockout and the triage best practices. The setups and results of these quantification processes can be found in [13]. The validation showed that the developed approach is suitable for the quantification of other best practices. The iterative nature of steps 5, 6 and 7 is stressed, as is the difficulty of measuring internal and external quality. Some of the results of the simulation projects are reported in the next section.

4 Results of Quantification Projects

The quantification approach was developed based on a simulation project for the parallel best practice and validated based on simulation studies for the knockout and triage best practices. Due to space limitations, only the main results for the parallel and knockout best practices are reported here. Each project included about 150 simulations, i.e. 150 * 21 replications. In this section the main results of the first two studies are presented. In each subsection, the best practice is described shortly, followed by a number of observations of when the best practice could be applied. This mainly depends on the intensity of the arrival of cases, the assignment of resource classes to particular tasks, and service times of tasks.

4.1 Quantification of the Parallel Best Practice

The parallel best practice runs as follows: consider whether tasks may be executed in parallel. The obvious effect of applying this best practice is that the throughput time may be reduced considerably. The applicability of this best practice in workflow redesign is large. When analyzing existing workflows in organizations we noted that tasks were mostly ordered sequentially without the existence of hard logical restrictions prescribing such an order. A possible disadvantage of introducing more parallelism in workflows with checks is an increase in costs or decrease in flexibility.

The original model we used for this study consisted of a process with six tasks, named A to F, in a sequence. From this model we created two redesign models: one model with two tasks, B and C, in parallel, and one with three tasks, B, C and D, in parallel. Further, we came up with several variations to test under which conditions a process would benefit from the application of the parallel best practice. We will elaborate on one of the variations in more detail and then present the results for other variations.

We assumed it would make a difference whether the parallel tasks would be performed by the same resource class or by different resource classes, and this became one of the variations we investigated. Table 1 shows the output data resulting from the simulation of this model variant for the model in which tasks B and C are in parallel. The variant consisted of four resource setups (ABC-DEF, AD-BC-EF, AC-BD-EF and ACE-BDF). In this context ABC-DEF, for instance, means there were two resource classes, the resources in the first class were able to execute tasks A, B and C, while the resources in the second class executed tasks D, E and F. Tasks B and C were put in parallel, so for this setup these tasks shared their resources. Except for the resource classes, settings were the same for each setup. Table 1 shows the lower bounds (LB) and the upper bounds (UB) of the confidence intervals of the relative differences between the original model and the four redesigns for eight performance measures. From these confidence intervals it can be seen that the implementation of the best practice in this example decreased the lead time and the WIP costs. All other measures had insignificant differences with the original situation, as their intervals included 0. This means that these measures were not affected by the implementation of the parallel best practice.

Table 1. Output data of resource class variations

	ABC-DEF		AD-BC-EF		AC-BD-EF		ACE-BDF	
	LB	UB	LB	UB	LB	UB	LB	UB
LeadTime	-7,4274	-6,3066	-6,2194	-5,1947	-5,6806	-4,5908	-2,4849	-1,3248
QueueTime	-0,0382	0,0731	-0,0204	0,0716	-0,0286	0,0523	-0,0205	0,0595
Utilisation1	-0,4782	0,4864	-0,4522	0,4637	-0,4765	0,4830	-0,4481	0,4604
Utilisation2	-0,5167	0,5283	-0,5163	0,5265	-0,4984	0,5100	-0,4962	0,5123
Utilisation3	–	–	-0,5052	0,5126	-0,5010	0,5100	–	–
WIP costs	-6.3598	-5.2654	-4.9655	-3.9877	-4.6589	-3.5116	-2.1593	-1.0374
LabFlexWF	-3,6751	9,9113	-4,4070	6,5565	-2,8213	10,0001	-3,6216	8,8427
VolumeFlex	-11,8040	12,3658	-8,7345	18,9280	-17,3928	3,0930	-7,0646	9,7734

Another comparison that can be made with the output data from different resource class variations is between the various redesigns. It allows for the selection of the best redesign alternative. Figure 1 graphically depicts the confidence intervals for two measures: lead time and volume flexibility. From these graphs it can be seen that the decrease in lead time of ABC-DEF was significantly higher compared to the other setups. The difference in lead time between setup AD-BC-EF and AC-BD-EF was nonsignificant, because the confidence intervals of both setups overlapped. The decrease in lead time of ACE-BDF was significantly lower than the decrease of the other setups. From the graph of volume flexibility it can be concluded that this measure was not affected by the redesign effort. An automated MS Excel sheet was created to generate this output.

Next to the resource classes we also varied the arrival rate and the service times. The variations in arrival rate showed that the observed positive impact on performance only held for processes with a low arrival rate. The positive result became smaller or even nonsignificant when the arrival rate increased. With a low

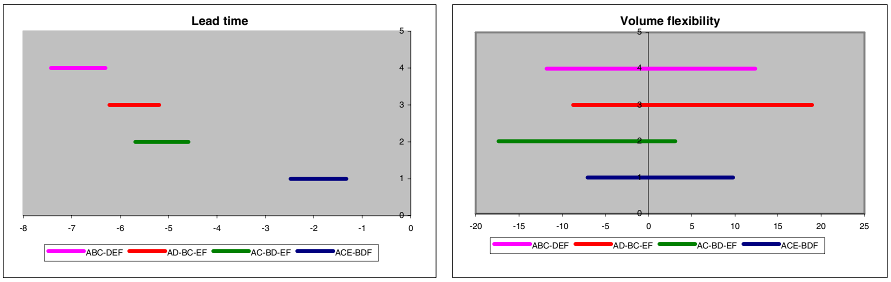


Fig. 1. Confidence intervals for the lead time and volume flexibility

arrival rate the positive impact of the parallel best practice was higher for tasks with equal parallel service times than for tasks with completely different parallel service times. In both situations, implementation of the parallel best practice led to a decrease in lead time and WIP costs and therefore appears to be advisable. However, the differences in impact between the two service time variants decreased or even became nonsignificant when the arrival rate increased. Concluding, we advise to implement the parallel best practice when the arrival rate is low. Further, the improvement will benefit from involved tasks sharing resources and having equal service times. Implementation of the parallel best practice changed the number of parallel tasks, which is a proxy of external quality and process modification flexibility. An increase in the number of parallel tasks led to a more complex workflow, which can result in slightly lower external quality and lower process modification flexibility. The other proxies of external quality and the remaining measures of the flexibility dimension remain unchanged with the implementation of the parallel best practice. Putting tasks in parallel does not change any of the proxies of the internal quality dimension. It is expected that the parallel best practice does not affect the internal quality of a workflow.

4.2 Quantification of the Knockout Best Practice

A typical part of a workflow is the checking of various conditions that must be satisfied to deliver a positive end result. Any condition that is not met may lead to a termination of that part of the workflow, the knockout. The knockout best practice comprises three possible redesigns:

- Swapping tasks rule. If there is freedom in choosing the order in which the various conditions are checked, the condition that has the most favorable ratio of expected knockout probability versus the expected effort to check the condition should be pursued.
- Combining tasks rule. If two tasks are executed by the same resource class, the combination of two tasks into one larger task is considered. As a result, this task can be executed by one resource without interruption.
- Parallel tasks rule. Putting tasks in parallel reduces the total flow time. The flow time in minimized by putting as much tasks in parallel as possible.

However, if one of the parallel tasks returns NOK, the result of the other task is not relevant anymore.

Swapping tasks rule. Applying the swapping tasks rule to processes with knock-out tasks results in lower, more balanced utilizations and lower WIP costs, both leading to a less costly process execution. In addition, also labor flexibility and volume flexibility increase, which positively influences the performance of the workflow as well. In most processes, implementation of the swapping tasks rule results in a decrease in lead time. However, when the arrival rate is too low to cause queues, or the utilizations of the resource classes are too unbalanced for the rule to balance them, implementation of the swapping tasks rule does not result in a reduction of lead time. External quality, internal quality, process modification flexibility, or any of the other measures are not affected by the swapping tasks rule.

Combining tasks rule. Implementation of the combining tasks rule leads to a considerable decrease in lead time. In some settings it also has a positive impact on the utilizations, the WIP costs, labor flexibility and volume flexibility. The combination of two or more KO tasks into one task can lead to too large tasks, which reduces the external quality and the process modification flexibility. The number of task and the scope of a task are proxies for internal quality. The number of executed tasks for one case per resource is reduced by the combining tasks rule. This would indicate lower internal quality. However, these tasks will have a larger scope, which would indicate higher internal quality. Overall, internal quality is expected to remain approximately the same.

Parallel tasks rule. Putting sequential KO tasks in parallel leads to a decrease in lead time and to lower WIP costs. The highest positive impact can be expected when the following conditions are satisfied: (1) The service times of the parallel tasks are of the same order of magnitude, (2) the parallel reject probabilities are small, (3) the arrival rates are low, and (4) none of the resource classes are overloaded as a result of putting tasks in parallel. The positive impact of the parallel tasks rule decreases and some measures are even negatively affected when one or more of the conditions are not satisfied.

The increase in number of parallel tasks is a proxy of lower external quality and lower process modification flexibility, because the complexity of the workflow increases. Internal quality increases, because the number of executed tasks per resource increases, which is a proxy for internal quality.

5 Discussion

The quantification of the impact of a business process redesign project has been standardized into an approach that considers all dimensions of performance of a workflow and can be used for the quantification of redesign best practices. The results indicate which impact on performance is to be expected in which situations and settings. Were the application of the approach applied to all best practices identified by Reijers [18], a clear picture would emerge on what best practice should be implemented to improve one or more performance dimensions.

Quantification of the three best practices in this research project resulted in some unexpected, counterintuitive outcomes, which are different from the qualitative evaluation results of [19]. This may be due to differences in the level of detail of these studies. The qualitative results of [19] were based on expectations and rules of thumb. The predicted impacts were mostly averages, which were based on one measure supplemented with some possible extreme impacts. In contrast, the impacts in this study are the result of employing a complete set of measures for all dimensions, using a simulation model. More measures have been used per dimension, and a more precise impact has been provided. The impacts of the best practices have also been quantified in models with different settings, to obtain a good view of the impact of implementation in different situations.

From a comparison of Van der Aalst's study on knockout processes [1] and this study, it can be concluded that most of the findings of [1] are supported by the results of this research project. The results of this study also identify situations in which some best practices do not hold true or in which the conditions for the application of the best practice are different. In addition, more aspects of performance have been included, which can be seen as an extension of [1].

To obtain a complete view on the impacts of the total set of redesign best practices identified by Reijers [18], the exact impact of the remaining best practices and combinations of best practices should be executed in a future research project. This would support the identification of the correct choice when selecting a redesign best practice to improve a specific performance dimension. Further, the approach should be applied to a real life redesign project to test its applicability to real life data. In this test, the results of individual best practices should be used to determine what redesign best practice could provide the most favorable results. With respect to generalizability, an interesting research topic would therefore be the relationship between the complexity of a business process and the applicability of the presented approach.

A weakness of the approach is that it cannot quantify the impact of a business process redesign effort on the external and internal quality of a workflow. Other methods that can be used to quantify the impact on these dimensions should be found. The use of surveys among customers of the process is proposed as an alternative method for the quantification of the impact on the external quality dimension. The quality of the output and the process, perceived by different customers, can be measured and analyzed. The same method is proposed for the quantification of the impact on the internal quality dimension. For this purpose, a survey among employees can be used. Whether these methods are suitable for the quantification of the impact on both quality dimensions should also be investigated in a subsequent research project.

References

1. van der Aalst, W.M.P.: Reengineering Knock-out Processes. Decision Support Systems 30(4), 451–468 (2001)
2. Adams, C., Neely, A.: Prism Reform. Financial Management 5, 28–31 (2002)

3. Brand, N., van der Kolk, H.: Workflow Analysis and Design. Kluwer Bedrijfswetenschappen (in Dutch) (1995)
4. Brignall, T.J., Fitzgerald, L., Johnston, R., Silvestro, R.: Performance Measurement in Service Businesses. Management Accounting 69(10), 34–36 (1991)
5. Cross, K.F., Lynch, R.L.: The "SMART" Way to Define and Sustain Success. National Productivity Review 8(1), 23–33 (1988/1989)
6. Goverde, R.H.J.J.M., Reijers, H.A.: Resource Management: a Clear-Headed Approach to Ensure Efficiency. Workflow Magazine 4(6), 26–28 (1998)
7. Hackman, J.R., Oldham, G.R.: Motivation through the design of work: Test of a theory. Organizational Behavior and Human Performance 16(2), 250–279 (1976)
8. Jansen-Vullers, M.H., Kleingeld, P.A.M., Loosschilder, M.W.N.C., Netjes, M.: Sequential or in Parallel – Measuring the impact of a Business Process Redesign Best Practice (submitted for publication, 2007)
9. Kaplan, R.S., Norton, D.P.: The Balanced Scorecard: Measures that Drive Performance. Harvard Business Review 70, 71–79 (1992)
10. Keegan, D.P., Eiler, R.G., Jones, C.R.: Are your Performance Measures obsolete? Management Accounting 70(12), 45–50 (1989)
11. Kulkarni, V.G.: Modeling, Analysis, Design, and Control of Stochastic Systems. Springer, New York (1999)
12. Law, A.M., Kelton, W.D.: Simulation Modeling and Analysis, McGraw-Hill Book Co., Singapore (2000)
13. Loosschilder, M.W.N.C., Jansen-Vullers, M.H.: Quantification of the implementation of the parallel, knockout and triage heuristic, BETA Working papers 203, 204 and 205, Eindhoven University of Technology, The Netherlands (2007)
14. Limam Mansar, S., Reijers, H.A.: Best Practices in Business Process Redesign: Validation of a Redesign Framework. Computers in Industry 56, 457–471 (2005)
15. Mehta, A.: Smart modeling: Basic methodology and advanced tools. In: Joines, J.A., Barton, R.R., Kang, K., Fishwick, A. (eds.) Proceedings of the winter simulation conference, pp. 241–245 (2000)
16. Miller Jr., R.G.: Simultaneous Statistical Inference. Springer, New York (1981)
17. Netjes, M., van der Aalst, W.M.P., Reijers, H.A.: Analysis of Resource Constrained Processes with Colored Petri Nets. In: Jensen, K. (ed.) Proc. of the 6th Workshop on Practical Use of Coloured Petri Nets and the CPN Tools (2005)
18. Reijers, H.: Design and Control of Workflow Processes: Business Process Management for the Service Industry. Springer, Berlin (2003)
19. Reijers, H.A., Limam Mansar, S.: Best Practices in Business Process Redesign: an Overview and Qualitative Evaluation of Successful Redesign Heuristics. Omega, The International Journal of Management Science 33, 283–306 (2005)
20. Zapf, M., Heinzl, A.: Evaluation of Generic Process Design Patterns: An Experimental Study. In: van der Aalst, W.M.P., Desel, J., Oberweis, A. (eds.) Business Process Management. LNCS, vol. 1806, pp. 83–95. Springer, Berlin (2000)

Compliance Aware Business Process Design

Ruopeng Lu, Shazia Sadiq, and Guido Governatori

School of Information Technology and Electrical Engineering,
The University of Queensland, Brisbane, Australia
{ruopeng, shazia, guido}@itee.uq.edu.au

Abstract. Historically, business process design has been driven by business objectives, specifically process improvement. However this cannot come at the price of control objectives which stem from various legislative, standard and business partnership sources. Ensuring the compliance to regulations and industrial standards is an increasingly important issue in the design of business processes. In this paper, we advocate that control objectives should be addressed at an early stage, i.e., design time, so as to minimize the problems of runtime compliance checking and consequent violations and penalties. To this aim, we propose supporting mechanisms for business process designers. This paper specifically presents a support method which allows the process designer to quantitatively measure the compliance degree of a given process model against a set of control objectives. This will allow process designers to comparatively assess the compliance degree of their design as well as be better informed on the cost of non-compliance.

Keywords: Business Process Design, Process Compliance Control, Business Process Modeling.

1 Background and Motivation

Compliance essentially means ensuring that business processes, operations and practice are in accordance with a prescribed and/or agreed set of norms. Compliance is increasingly gaining importance as well as raising the pressure for organizations in practically all industry sectors. Although this is not a new issue, but recent events, particularly high profile corporate scandals, as well as new regulations such as the Sarbanes-Oxley act have raised a new set of challenges for businesses.

Compliance is predominantly viewed as a burden, although there are indications that businesses have started to see the regulations as an opportunity to improve their business processes and operations. Industry reports [7] indicate that up to 80% of companies said they expected to reap business benefits from improving their compliance regimens.

Currently there are two main approaches towards achieving compliance. First is *retrospective reporting*, wherein traditional audits are conducted for “after-the-fact” detection, often through manual checks by expensive consultants. A second and more recent approach is to provide some level of automation through *automated detection*. The bulk of existing software solutions for compliance follows this approach. The proposed solutions hook into variety of enterprise system components (*e.g.* SAP HR, LDAP Directory, Groupware etc.) and generate audit reports against hard-coded checks performed on the requisite system. These solutions often specialize in certain class of checks, for example the widely supported checks that relate to Segregation of Duty violations in role management systems. A major issue with the two discussed approaches is the lack of sustainability. Even with automated detection facility, the hard coded check repositories can quickly grow out of control making it extremely difficult to evolve and maintain them for changing legislatures and compliance requirements. The complexity of the situation is exasperated by the presence of dynamically changing collaborative processes shared with business partners. The diversity, scale and complexity of compliance requirements warrant a highly systematic and well-grounded approach.

We believe that a sustainable approach for achieving compliance should fundamentally have a preventative focus, thus achieving *compliance by design*. Incorporating compliance issues within business process design methodology can assist process designers in tackling this complex issue using known successful strategies. However, at the same time, there is evidence that dealing with compliance may be a rather distinct activity from business process management within organizational structures.

This paper presents a particular method to study the relationship between compliance requirements modeled as controls, and process requirements modeled as business process models. Specifically we will present a quantitative measure of compliance for a given process model against a set of control objectives. The associated methods will allow process designers to comparatively assess the compliance degree of their design as well as be better informed on the cost of non-compliance.

Related work can be found in the research of [1, 2, 3, 10]. Space does not allow further elaboration of these works, but a distinctive feature of our work is that most related works present solutions for runtime monitoring, where as we focus on design time support.

The remaining paper is structured as follows. Section 2 presents the underlying methodology for *compliance aware* business process design. In section 3, we present the technique to quantitatively measure the degree of compliance during business process design. We conclude this paper in section 4.

2 Compliance by Design Methodology

Regulations and other compliance directives are complex, vague and require interpretation. Business will typically deal with a number of regulations/standards at one time. Often in legalese, these mandates need to be translated by experts. Tackling this issue warrants a systematic methodology [9].

Firstly, there is a need to provide a structured means of managing the various (expert) interpretations within regional, industry sector and organizational contexts. As a first step, a facility for *control directory management* (e.g. SAP GRC Repository) needs to be realized by repositories of control objectives (and associated parameters) against the major regulations.

Interpretation of regulations from legal /financial experts comes in the form of textual descriptions (see the examples in Section 2.1). *Establishing an agreement* on terms and usage between these descriptions and the business processes and constituent activities/transactions is a difficult but essential aspect of the overall methodology. However, it is evident that several controls may be applicable on a given business task, and one control may impact on multiple tasks as well.

A fundamental question in this regard is the appropriate formalism to undertake the task of *representing controls objectives* in a precise and unambiguous manner. Our observation is that a compliance requirement (or its translation into a control objective and subsequently internal controls) can be reduced to the identification of what obligations an enterprise has to fulfill to be deemed as compliant.

The motivation to model control objectives is multifaceted: Firstly, a generic requirements modeling framework for compliance by design will provide a substantial improvement over current after-the-fact detection approaches. Secondly, it will allow for an analysis of compliance rules thus providing the ability to discover hidden dependencies, and view in holistic context, while maintaining a comprehensible working space. Thirdly, a precise and unambiguous (formal) specification will facilitate the systematic enrichment of business processes with control objectives.

Subsequent to the modeling of control objectives, there is a need to provide the ability to *enhance enterprise models* (business processes) with compliance requirements. This may constitute visualization schemes [9], which facilitates a better understanding of the interaction between the two specifications for both stakeholders (process owners as well as compliance officers).

However, the visualization is only a first step. The new checks introduced within the process model, can in turn be used to analyse the model for measures such as *compliance degree* that can provide a quantification of the effort required to achieve a compliant process model. Eventually, process models may need to be modified to include the compliance requirements.

In this paper, we are focused on this last aspect, that is to assist process designers in creating compliant business processes. The presence of the previous phases of the methodology is assumed. As such, the goal of this so-called *compliance aware business process design* is to design the process while keeping track of relevant control objectives and ensuring that high risk controls are not ignored or violated.

In the rest of the paper, we first discuss the approach to model the controls objectives and present an appropriate language for their representation, followed by a simple formalization for the business process model. We then introduce the technique to map the controls objectives and the process model into a canonical form, such that the degree of compliance in the process model can be compared with regard to the controls objectives. The subsequent discussion is based on a sample procurement process (*cf.* Figure 1).

The procurement process may be subject to a number of control objectives from various restrictions such as regulations, industrial standards and partner obligations etc. The control objectives will typically have a corresponding risk statement, and a translation to an internal control indicating effective implementation of the control objective.

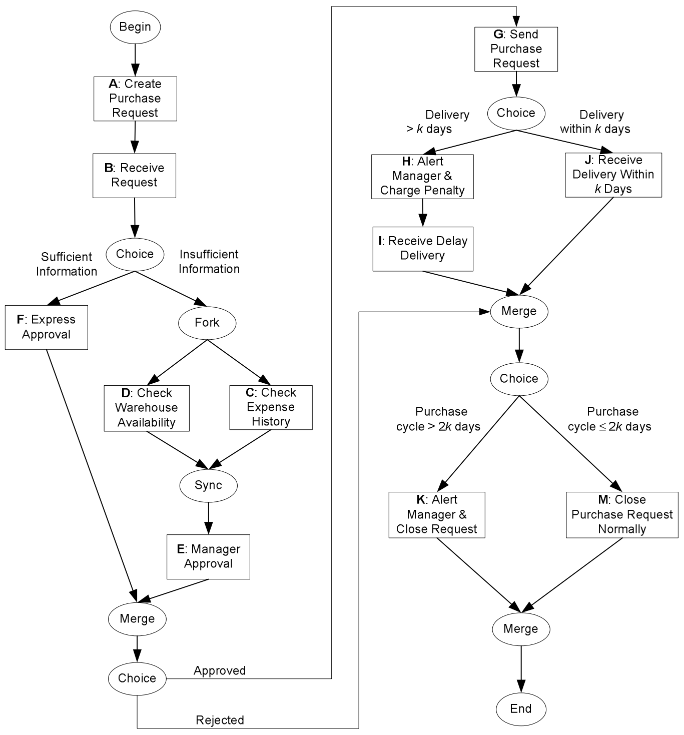


Fig. 1. Example procurement process

Table 1. Control objectives of the procurement process

Control Objective	Risk	Internal Control
Process efficiency	Process delays due to repeated or additional activities.	Purchase request with necessary information should be fast-tracked without management level approval.
Ensure adequate supply of materials	Production delays due to lack of resources/ materials	Supplier can be charged a penalty if goods not received within *k* days of receipt of goods shipment notice.
Timely and efficient procurement process	Production delays due to lack of resources/ materials	Purchase requests not closed (declined or converted to Purchase Orders) within 2*k* days should raise an alert to purchasing manager.

Table 1 provides examples of such control objectives for the procurement process. Typically, these internal controls cover multiple aspects of business process, including:

– Model structure, *e.g.*, task execution restrictions (every purchase order must be initially checked before passing to the Manager for approval).
– Data integrity, *e.g.*, every Purchase Order must contain a valid purchase request number.

- Resource allocation, *e.g.*, segregation of duty constraint (the creation and approval of purchase order must not be by the same officer).
- Temporal restrictions, *e.g.*, deadline (all purchase requests must be closed within $2k$ days).

2.1 Modeling Control Objectives

Although our work is primarily targeted at achieving compliance by design by adopting a preventative approach facilitated by business process models, the work on formal modeling of control objectives has taken into account the violations and resultant reparation policies that may surface at runtime. The objective is to be able to examine how compliant the (possible) runtime behaviors of a process model is with regard to the control objectives. We consider the behaviors of a process model to be reflected by actual *execution sequences* (of tasks in the process). The focus is then on the measurement for how "close" between the behaviors of the process model, and the compliance controls. To allow for the comparison, the formal representations of compliance controls, and the model behaviors (execution sequences) are given.

The compliance controls can be represented in a formal language, such as Formal Contract Language (FCL) [4, 5]. FCL is a combination of an efficient non-monotonic formalism (defeasible logic) and a deontic logic of violations. We illustrate how to use this formalism to represent and reason about "normative" specifications relative to a business process. For detailed presentation of the rationale and formalism of FCL, we refer to [4, 5].

Definition 1 (FCL Rule). A rule in FCL is an expression of the form

$$r: A_1,..., A_n \Rightarrow B$$

where r is the name of the rule (unique for each rule), $A_1,..., A_n$ are the premises (propositions in the logic), and B is the conclusion of the rule (also a proposition of the logic).

The propositions of the logic are built from a finite set of atomic propositions, and the following operators: $\neg$ (for negation), O (for obligation), P (for permission), and $\otimes$ (for violation/reparation). The formation rules are as follows:

- every atomic proposition is a proposition;
- if p is an atomic proposition, then $\neg p$, is a proposition;
- if p is a proposition then Op is an obligation proposition and Pp is a permission proposition; obligation propositions and permission propositions are deontic propositions
- if p_1, ..., p_n are obligation propositions and q is a deontic proposition, then $p_1\otimes ... \otimes p_n\otimes q$ is a reparation chain.

A simple proposition corresponds to a factual statement. A reparation chain, for example $B_1\otimes B_2$ captures obligations and normative positions arising in response to violations of obligation. Thus the expression above means that it is obliged to perform

B_2, in case B_1 is not fulfilled (i.e., the obligation is violated) then the “secondary” obligation B_2 has to be fulfilled. The control objectives shown in Table 1 can be expressed in the following FCL rules:

Purchase request should be supplied with sufficient background information in order to streamline the approval process.

r_1: *CreatePurchaseRequest*, *ReceiveRequest* ⇒ *ExpressApproval* ⊗(*CheckWareHouseAvailability*;*CheckExpenseHistory*;*ManagerApproval*)

Supplier can be charged a penalty if goods not received within k days of receipt of goods shipment notice, while manager should be alerted.

r_2: *SendPurchaseRequest* ⇒ *ReceiveeDeliveryWithinkDays* ⊗(*ChargePenalty&AlertManager*;*ReceiveDelayDelivery*)

If purchase order is not closed within 2k days the manager should be alerted.

r_3: *ReceiveDeliveryWithinkDays* ⇒ *ClosePurchaseRequestWithin2kDays* ⊗(*AlertManager&CloseRequest*)

r_4: *ReceiveDelayDelivery* ⇒ *ClosePurchaseRequestWithin2kDays* ⊗(*AlertManager&CloseRequest*)

For the ease of discussion, we use the letters associated with each task on Fig. 1 to denote the tasks in the process model. r_1 - r_4 can thus be denoted by:

r_1: $A, B \Rightarrow F \otimes (C;D;E)$; r_2: $G \Rightarrow J \otimes (H;I)$; r_3: $I \Rightarrow M \otimes K$; r_4: $J \Rightarrow M \otimes K$

2.2 Business Process Model

We provide a formal definition for a simple business process model. Through which the runtime behaviors of the process as reflected by execution sequences can be defined.

Definition 2 (Process Model). A process model W is a pair (N, E), which is defined through a directed graph consisting a finite set of nodes N, and a finite set of flow relations (edges) $E \subseteq N \times N$. Nodes are classified into tasks T and coordinators C, where $N = C \cup T$, and $C \cap T = \emptyset$. T is the set of tasks in W, and C contains coordinators of the type {*Begin*, *End*, *Fork*, *Synchronizer, Choice, Merge*}, which have typical workflow semantics. A sub-process model is a special type of W, which is a fragment of a process model in which {*Begin*, *End*} is excluded from its coordinator nodes.

Given a process model W and a task $T_i \in T$, *Trigger*(W, Ti) denotes the set of tasks that can be triggered by task T_i in W as the result of execution. E.g., *Trigger*(W, A) = {B} (*cf.* Fig. 1). For tasks followed by a *Fork* (*AND-SPLIT*) or a *Choice* (*XOR-SPLIT*) coordinator, we consider all subsequent tasks after the coordinator can be triggered. E.g., *Trigger*(W, B) = {C, D, F}, *Trigger*(W, G) = {H, J}. *Disable*(W, T_i) denotes the set of tasks disabled as the consequence of executing Ti, which is defined to realize the semantics of the *Choice* coordinator. For example, *Disable*(W, H) $=\{J\}$, which means either H or J is executed but not both. *Initial*(W) is a function returning the first task node in W.

An *execution sequence* of a process models referred to as the trace of execution in a process model, which reflects a possible order of task executions at runtime. Typically, a process model with parallel branches (*Fork*) or alternative branches (*Choice*) contains more than one possible execution sequences.

For example, for tasks A, B, C, D, E, and F in W (*cf.* Fig. 1), there are three possible execution sequences $<A, B, F>$, $<A, B, C, D, E>$ and $<A, B, D, C, E>$, since F and C, D, E are in alternative branches, and C, D in parallel branches.

We follow the general sequence definition to define an execution sequence: A finite sequence $s = \{s_1, s_2, \ldots, s_n\}$ is a function with the domain $\{1, 2, \ldots, n\}$, for some positive integer n. The i-th element of s is denoted by s_i.

Definition 3 (Execution Sequence). An execution sequence s^W of a process model W is a finite sequence of tasks $T' \subseteq T$ in W, which is defined by the sequence $<T_1, T_2, \ldots, T_n>$, $n \geq 1$. An execution sequence ss^W is a *subsequence* of s^W if every element in ss^W is an element of s^W, and the elements in ss^W occur in the same order as in s^W.

2.3 Measurement of Compliance

It is desirable to transform the control objectives given in FCL into a form such that it is comparable to business process design. We establish the connection between FCL and business process model through execution sequences and the so called state of idealness [6]. Through the states of idealness we can determine whether a process model is compliant with the control objective (i.e., how well the process model supports such “ideal” states in execution).

Intuitively an *ideal* situation is a situation where execution sequences do not violate FCL expressions, and thus the execution sequences are fully compliant with the control rule. A *sub-optimal* situation is a situation where there are some violations, but these are repaired. Accordingly, processes resulting in *sub-optimal* situations are still compliant to a control rule even if they provide sub-optimal performance of the control objective. A situation is *non-ideal* (*non-compliant*) if it violates a control objective and the violations are not repaired.

There are two possible reasons for a process not to comply with a control rule: 1) the process executes some tasks which are prohibited by the control rule (or equivalently, it executes the opposite of obligatory tasks); 2) the process fails to execute some tasks required by the control rule. For example consider the rule

$$r: A \Rightarrow B \otimes C$$

which means that, if A occurred then it must be followed by B, or in alternative, in case B does no occur, it must be followed by C. An *ideal* state for r is the situation (a possible execution sequence) $s_1 = <A, B>$. A *sub-optimal* situation can be $s_2 = <A, C>$ where the first obligation B is not fulfilled. Note that we also consider $s_3 = <A, B, C>$ a *sub-optimal* situation since it is not required to perform C when B is already in place. The *non-ideal* situation is $s_4 = <A>$.

Definition 4 (Idealness of execution sequence). Let S^W be the set of all possible execution sequences of a process model W, r: $A_1, \ldots, A_m \Rightarrow B_1 \otimes \ldots \otimes B_n$ be a control objective in FCL.

- A sequence $s \in S^W$ is an *ideal* execution sequence to r *iff* sequence $<A_1, \ldots, A_m, B_1>$ is a subsequence of s.
- A sequence $s \in S^W$ is a *sub-optimal* execution sequence to r *iff* $\exists B_i$, $1 < i \leq n$ such that $<A_1, \ldots, A_m, B_i>$ is a subsequence of s.
- A sequence $s \in S^W$ is a *non-ideal* execution sequence to r *iff* sequence $<A_1, \ldots, A_m>$ is a subsequence of s and s is neither *ideal* nor *sub-optimal.*

Given a control rule r, we denote the set of *ideal* and *sub-optimal* execution sequences as S^r_{ideal} and $S^r_{sub\text{-}optimal}$ respectively. Table 2 shows such for control rules $r_1 - r_4$. Note that for compliance checking purpose, *sub-optimal* execution sequences only contain the *consequences* of the control rule, *i.e.*, right hand side of r. Because the *antecedent*, *i.e.*, left hand side of r is irrelevant in *sub-optimal* states.

The above definition for *non-ideal* covers the second type of non-compliant situation where the process fails to execute some required tasks. We argue that the first situation where the process executes prohibited task(s) can be checked by simple sequence (string) matching technique (for execution sequences between control rule and process model) and hence not discussed further. In the next section, we discuss the technique to check for compliance degree for *ideal* and *sub-optimal* execution sequences.

Table 2. State of idealness of control rules $r_1 - r_4$

Control Rule	S^r_{ideal}	$S^r_{sub\text{-}optimal}$
r_1: $A, B \Rightarrow F \otimes (C;D;E)$	$<A, B, F>$	$<C, D, E>$, $<F, C, D, E>$, $<C, D, E, F>$
r_2: $G \Rightarrow J \otimes (H;I)$	$<G, J>$	$<H, I>$, $<J, H, I>$, $<H, I, J>$
r_3: $I \Rightarrow M \otimes K$	$<I, M>$	$<K>$, $<M, K>$, $<K, M>$
r_4: $J \Rightarrow M \otimes K$	$<J, M>$	$<K>$, $<M, K>$, $<K, M>$

3 Compliance Degree

We now have all the machinery to define the measure for compliance between a process model and a given control rule. We propose to use the notion of compliance degree as a quantitative measurement. The notion further utilizes the concept of *support*: Given a set of execution sequences S and a process model W, the *support* of W based on a sequence $s \in S$ is given by the proportion of tasks in s that can be executed in W. The range of support is a real number between 0 and 1, where 0 indicates no support (s is not executable in W at all) and 1 complete match (the entire sequence s can be executed in W, i.e., it is possible to derive an execution sequence s^w from W such that $s = s^w$). The *support* of W based on S is the weighted sum of *support* from all sequences in S, which is also between 0 and 1.

In order to calculate the *ideal* and *sub-optimal compliance degree*, we need to first extract the set of *ideal* and *sub-optimal* execution sequences for each control rule r, and calculate the degree of support for these sequences in the process model. The rationale of this technique is to measure how well a given process model W represents the *ideal* and *sub-optimal* situations in control rule *r* by calculating the support for *W* against the set of *ideal* and *sub-optimal* execution sequences representing *r*. We refer to the support for *ideal* and *sub-optimal* sequences as *ideal* and *sub-optimal compliance degree* respectively. The first measurement indicates whether the *ideal* situation (the exact sequence) can be fully or partially supported in *W* (*ideal compliance degree* = 1, or between [0, 1]) respectively). Similarly, the latter measurement indicates whether *W* allows *sub-optimal* situation(s) and by what degree.

We first extract a sub-process from the process model which contains only the relevant tasks as in the set of *ideal* and *sub-optimal* execution sequences of *r*. To achieve this we use a technique called `SELECTIVE_REDUCE` [8]. For example, the procurement process model *W* (*cf.* Fig. 1) is reduced into W_1, W_2, W_3 and W_4 (Fig. 2) against control rule r_1, r_2, r_3 and r_4 respectively.

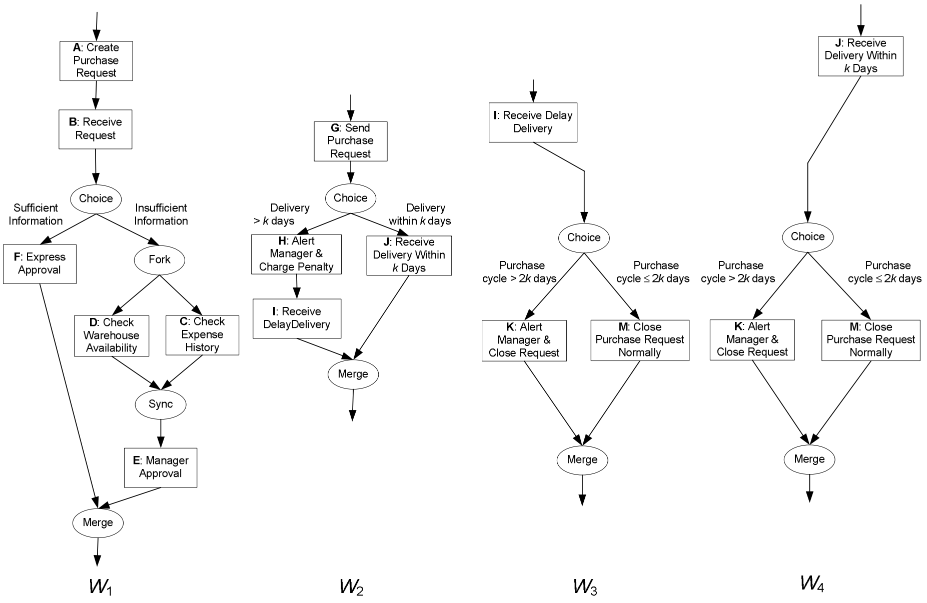


Fig. 2. Sub-processes of the procurement process

We then calculate the compliance degree through the algorithm given in Fig. 3. The algorithm takes as inputs a process model *W*, a set of sequences *S*, and the control rule *r*, produces the compliance degree *comp*. Functions *Trigger*, *Disable* and *Initial* given in Definition 2 are utilized. An additional function *SubInitial*(*W*, *r*) returns the set of task node(s) which are immediate after the last antecedent task in *r*. For example, *SubInitial*(W_2, r_2) = {*H*, *J*}, where *G* is the last task in the antecedent of r_2. Function *Sub-optimal*(S^W) returns TRUE if is S^W the set of *sub-optimal* sequences.

For each sequence *s* in *S*, *Tr* is initially given the first task in *W* in step 4. For each task T_i in a sequence *s* (in this case, $T_i = s_i$ where s_i is the *i*-th element in *s*), *Tr* is the current set of triggered tasks as the result of executing task T_i in *W*. Step 8 checks whether the triggered tasks in *Tr* includes T_i. Step 11 calculates the proportion of tasks in *W* triggered by tasks in *s*. After all different sequences in *S* have been accounted for, the final compliance degree is scaled according to the total number of sequences in *S* and returned (step 12). The algorithm complexity is bound by the number of tasks in the sequence and the number of different sequences in *S*.

For example, to compute the *ideal compliance degree* of *W* with regard to r_1: $A, B \Rightarrow F \otimes(C;D;E)$, we input W_1, the sub-process of *W* relevant to r_1 (*cf.* Fig. 2), and S^{r1}_{ideal}, the set of *ideal* execution sequences of r_1, where $S^{r1}_{ideal} = \{<A, B, F>\}$. Since there is only one sequence in S^{r1}_{ideal}, the *ideal compliance degree* is (1+ 1+1)/3 = 1 (step 11), because <*A*, *B*, *F*> is an exact execution sequence executable in W_1.

```
Procedure. COMPLIANCE_DEGREE
Input W, S, r
Output degree
1. degree, count, comp ← 0
2. For each different sequence s in $S^W$
3.     If Sub-optimal($S^W$)                    // for sub-optimal compliance degree
4.         Tr ← SubInitial(W, r)
5      Else                                     // for ideal compliance degree
6.         Tr ← Initial(W)
7.     For each task in s denoted by $T_i$, $i$ ← 1, …, |s|
8.         If $T_i \in Tr$
9.             count = count + 1
10.        $Tr \leftarrow (Tr - \{T_i\} - Disable(W, T_i)) \cup Trigger(W, T_i)$
11.    $comp \leftarrow comp + \frac{count}{|s|}$
12. Return $degree \leftarrow \frac{comp}{|S|}$
```

Fig. 3. An algorithm to compute compliance degree

The *sub-optimal compliance degree* of *W* with regard to r_1 can also be computed. We again input W_1 and $S^{r1}_{sub\text{-}optimal}$, the set of *sub-optimal* execution sequences of r_1, where $S^{r1}_{sub\text{-}optimal} = \{<C, D, E>, <F, C, D, E>, <C, D, E, F>\}$. For each sequence *s* in $S^{r1}_{sub\text{-}optimal}$, we display in Table 3 the intermediate result of *degree*, which is the support of W_1 received from *s*. Sequence <*C*, *D*, *E*> has *degree* of 1 since it is an exact sequence executable in W_1. Sequence <*F*, *C*, *D*, *E* > has *degree* of 0.25 because after triggering *F* in W_1, C, D, and E will be disabled ((1+0+0+0)/4 = 0.25 in step 9). Similarly, sequence <*C*, *D*, *E*, *F*> has *degree* of 0.75 since after triggering *C*, *D*, and *E* in W_1, *F* will not be triggered ((1+1+1+0)/4 = 0.75). The *sub-optimal compliance degree* is 0.67, which is the average of the three *degrees*.

Suppose there is a process *W'* containing a subgraph (subprocess) W'_1 relevant to r_1, where tasks D and E are not included (*cf.* Fig. 4). In this case the there is no ideal situation in *W* since the *ideal compliance degree* is (1+1+0)/3 = 0.67 ≠ 1. The *sub-optimal compliance degree* is also reduced to 0.5.

We use the *ideal compliance degree* to evaluate how well the process model supports a given control rule. *degree* = 1 indicates all ideal situation(s) of the control objective are represented in the process model *W*, (i.e., it is possible to find out the

Table 3. Intermediate result for applying COMPLIANCE_DEGREE to $S^{r1}_{sub\text{-}optimal}$ and W_1

$S^{r1}_{sub\text{-}optimal}$	*degree*
$< C, D, E >$	1
$< F, C, D, E >$	0.25
$< C, D, E, F >$	0.75
sub-optimal compliance degree	**0.67**

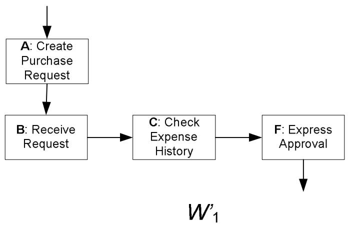


Fig. 4. Sub-process relevant to r_1 of an alternative procurement process

exact ideal execution sequence(s) in the relevant sub-graph of *W*, hence the process is an *ideal design* for the control rule *r*). While 0 indicates none of the ideal situation(s) is represented in *W*, from which we can immediately conclude that *W* is *non-compliant* with *r*. If none of the task in any sequence of *ideal* or *sub-optimal* execution sequences S^{r}_{ideal} is presented in the process model *W*, then one can only derive an empty sub-graph from *W* which contains the relevant tasks in S^{r}_{ideal}, Thus the algorithm returns 0 in this case, which is corresponding to a *non-compliant* situation. Lastly, having a number between 0 and 1 indicates *W* represents part of some *ideal* situation (i.e., it is not possible to find out exact but partial ideal execution sequence(s) in the relevant sub-graph of *W*).

In addition, from the *sub-optimal compliance degree* we can find out whether the process model may contain some *sub-optimal* situations. There can be many interpretations for *sub-optimal compliance degree*. Here we consider it as an auxiliary measurement to examine the expressiveness of the process model, in terms of expressing both *ideal* and *sub-optimal* executions. For example, in the case when two arbitrary process models W_α and W_β are both *ideal* to a control rule, but W_α has a higher *sub-optimal compliance degree* of W_β, then W_α is a better design.

Table 4 lists the *ideal* and *sub-optimal* compliance degree for control rules r_1 - r_4 respectively. The overall compliance degree is the sum of the compliance degree of each control rule. Note that we can also take a weighted approach for calculating the *sub-optimal compliance degree*. For each control rule *r*, a weight can be assigned to

Table 4. Compliance measurement for process model *W*

Control Rules	*Ideal Compliance Degree*	Risk (Weight)	*Sub-optimal Compliance Degree*
r_1	1	10%	0.67
r_2	1	50%	0.67
r_3	1	20%	1
r_4	1	20%	1
TOTAL	**1**	100%	**0.80**

reflect the relative importance of compliance with respect to *r*. Weights are assumed to be determined by experts defining internal controls, as an indication of the risk (or cost) of non-compliance. The overall *sub-optimal compliance degree* for *W* undertakes such approach. The results show that *W* is compliant with all ideal situations according to control rules r_1 - r_4, and *W* supports *sub-optimal* situations to a large extend.

4 Conclusion and Future Work

This paper presents an overall methodology for compliance by design, and specifically proposes a method to measure the degree of compliance between control objectives and business process models during process design. The proposed method based on the notion of compliance degree will assist process designers in undertaking compliance aware design so that an appropriate balance between the two, often conflicting, objectives can be achieved.

The approach presented so far is focused on assessing compliance of a process model through execution sequences. However, control objectives may also refer to other aspects of the process such as resource allocations, or data flow. Consideration of these aspects is part of our future work through which we hope to extend the proposed notion of compliance degree.

References

1. zur Muehlen, M., Ho, D.T.: Risk Management in the BPM Lifecycle. In: Bussler, C.J., Haller, A. (eds.) BPM 2005. LNCS, vol. 3812, pp. 454–466. Springer, Heidelberg (2006)
2. Christopher, G., Müller, S., Pfitzmann, B.: From Regulatory Policies to Event Monitoring Rules: Towards Model-Driven Compliance Automation. IBM Research Report RZ 3662, IBM Zurich Research Laboratory (2006)
3. Goedertier, S., Vanthienen, J.: Designing Compliant Business Processes with Obligations and Permission. In: Eder, J., Dustdar, S. (eds.) BPM Workshops 2006. LNCS, vol. 4103, pp. 5–14. Springer, Heidelberg (2006)
4. Governatori, G.: Representing Business Contracts in RuleML. International Journal of Cooperative Information Systems 14(2–3), 181–216 (2005)
5. Governatori, G., Milosevic, Z.: A Formal Analysis of a Business Contract Language. International Journal of Cooperative Information Systems 15(4), 659–685 (2006)
6. Governatori, G., Milosevic, Z., Sadiq, S.: Compliance checking between business processes and business contracts. In: Proceedings of the 10th IEEE Conference on Enterprise Distributed Object Computing (2006)
7. Hagerty, J.: SOX Spending for 2006. AMR Research, Boston USA (2007)
8. Lu, R., Sadiq, S.: Managing Process Variants as an Information Resource. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102, Springer, Heidelberg (2006)
9. Sadiq, S., Governatori, G., Naimiri, K.: Modeling Control Objectives for Business Process Compliance. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, Springer, Heidelberg (2007)
10. Zdravkovic, J., Kabilan, V.: Enabling Business Process Interoperability Using Contract Workflow Models. In: Meersman, R., Tari, Z. (eds.) OTM 2005. LNCS, vol. 3760, pp. 77–93. Springer, Heidelberg (2005)

Transforming Object-Oriented Models to Process-Oriented Models

Guy Redding[1], Marlon Dumas[1], Arthur H.M. ter Hofstede[1], and Adrian Iordachescu[2]

[1] Queensland University of Technology, Brisbane, Australia
{g.redding, m.dumas, a.terhofstede}@qut.edu.au
[2] Shared Web Services Pty Ltd, Sydney, Australia
adrian@sws.com.au

Abstract. Object-oriented modelling is an established approach to document the information systems. In an object model, a system is captured in terms of object types and associations, state machines, collaboration diagrams, etc. Process modeling on the other hand, provides a different approach whereby behaviour is captured in terms of activities, flow dependencies, resources, etc. These two approaches have their relative advantages. In object models, behaviour is split across object types, whereas in process models, behaviour is captured along chains of logically related tasks. Also, object models and process models lend themselves to different styles of implementation. There is an opportunity to leverage the relative advantages of object models and process models by creating integrated meta-models and transformations so that modellers can switch between these views. In this paper we define a transformation from a meta-model for object behavior modeling to a meta-model for process modeling. The transformation relies on the identification of causal relations in the object model. These relations are encoded in a heuristics net from which a process model is derived.

Keywords: Process model, object model, model transformation.

1 Introduction

Object modelling and process modelling are two established approaches to describe information systems [1]. Each of these approaches adopts a different perspective and has its own way of thinking. Modelling an information system in terms of objects leads to the definition of object types, associations, intra-object behaviour and inter-object interactions, which are captured using notations such as UML class, state and collaboration diagrams [2]. Object models group related data and behaviour into classes, thus promoting modularisation and encapsulation. Purported advantages of this approach include reuse and maintainability.

Meanwhile, process models are structured in terms of activities (which may be decomposed into sub-processes), events, control and data-flow dependencies, and associations between activities and resources. BPMN [3], UML activity diagrams, BPEL [4] and YAWL [5] are examples of notations that capture the behaviour of a system in a process-oriented manner at various levels of details. Process models provide a holistic view on the activities and resources required to achieve a goal. Accordingly, they lend themselves to analysis through simulation and other quantitative analysis techniques, and they have proven instrumental in enabling communication between business and IT stakeholders [6].

Moreover, object modelling and process modelling typically lead to different implementation styles. Whereas object modelling lends itself to implementation in an object-oriented programming environment, process models naturally lead to workflow applications or other types of process-aware information systems.

There is an opportunity to reconcile object-oriented and process-oriented approaches to information systems engineering in order to benefit from their relative strengths. Each of these modelling approaches adopts a different perspective. Information captured in one approach may be missing in the other approach. For example, a class in an object model may contain references to activities (e.g. in a sequence diagram), but objects are predominately state-centric and do not explicitly define activities or the control flow relations between them. Likewise, a process model contains implicit references to states (cf. the event-driven and the deferred choice in BPMN and YAWL respectively) or to classes representing resources, but a process model is predominately activity-centric.

Figure 1 shows the typical phases and deliverables involved in the object-oriented development approach (OODA) and in the process-oriented development approach (PODA). In this paper, we investigate how to bridge the deliverables produced by the design phases of these two approaches. Specifically, we present a transformation from detailed object behaviour models to process models. The transformation relies on the identification of causal relations in the object model. These relations are encoded in a causal matrix (also called heuristics net) from which we derive a process model represented in YAWL.

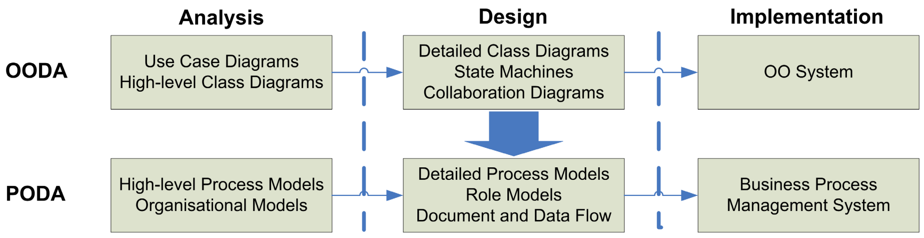


Fig. 1. Transforming an Object-oriented to a Process-oriented Approach

The paper is organised as follows. Section 2 introduces a motivating example. Section 3 defines a meta-model for object behaviour modelling. Section 4 introduces an algorithm to transform an object behaviour model into a process model. Section 5 discusses related work and Section 6 concludes the paper.

2 Example

In this section we introduce an example of a process that we have used as a test scenario. The example deals with a process for inspection and maintenance of heavy equipment such as open mine excavators and shipping container cranes. Such equipment is subjected to inspections at regular intervals when a number of issues requiring maintenance may be raised with the equipment. Depending on the severity of an issue and the criticality of the equipment some issues will be determined to be resolved with more urgency than others. The application domain is presented as a high-level class diagram in Figure 2. Each class may have one or more state machines as discussed later.

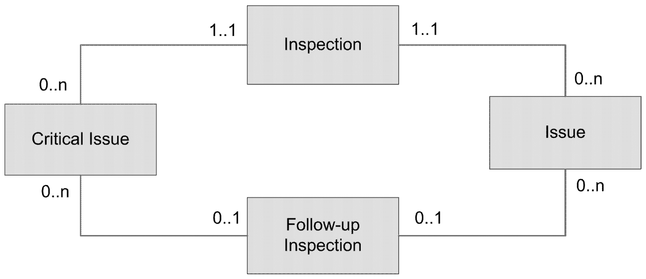


Fig. 2. Asset Maintenance Process – High-level class diagram

An *inspection* for a particular piece of equipment may uncover zero or more *issues* to be resolved. These are added to the set of existing *issues* that can be derived for that piece of equipment so that the main *inspection* can only be completed after all *issues* have been resolved and completed. A *critical issue* may also be detected during an *inspection* or *follow-up inspection*. These *critical issues* are identified separately due to the elevated need to have them resolved. An *issue* may raise a number of *follow-up inspections*, which in turn may lead to new (critical) *issues* being raised, and so on. The multiplicity between classes such as 1:1 and 0:n specifies the number of instances that a class interacts with at runtime. It should be noted that the reason that an *inspection* is modelled as a separate entity to a *follow-up inspection* is that in this scenario these classes have different state machine lifecycles, as do an *issue* and *critical issue*.

3 Object Behaviour Meta-model

In order to present our transformation approach, we first need to agree on meta-models for representing object behaviour models and process models. To represent process models, we use the YAWL language as discussed in the next section. Meanwhile, to represent object models we adopt a meta-model inspired by FlowConnect [7], a system that supports the development of software applications based directly

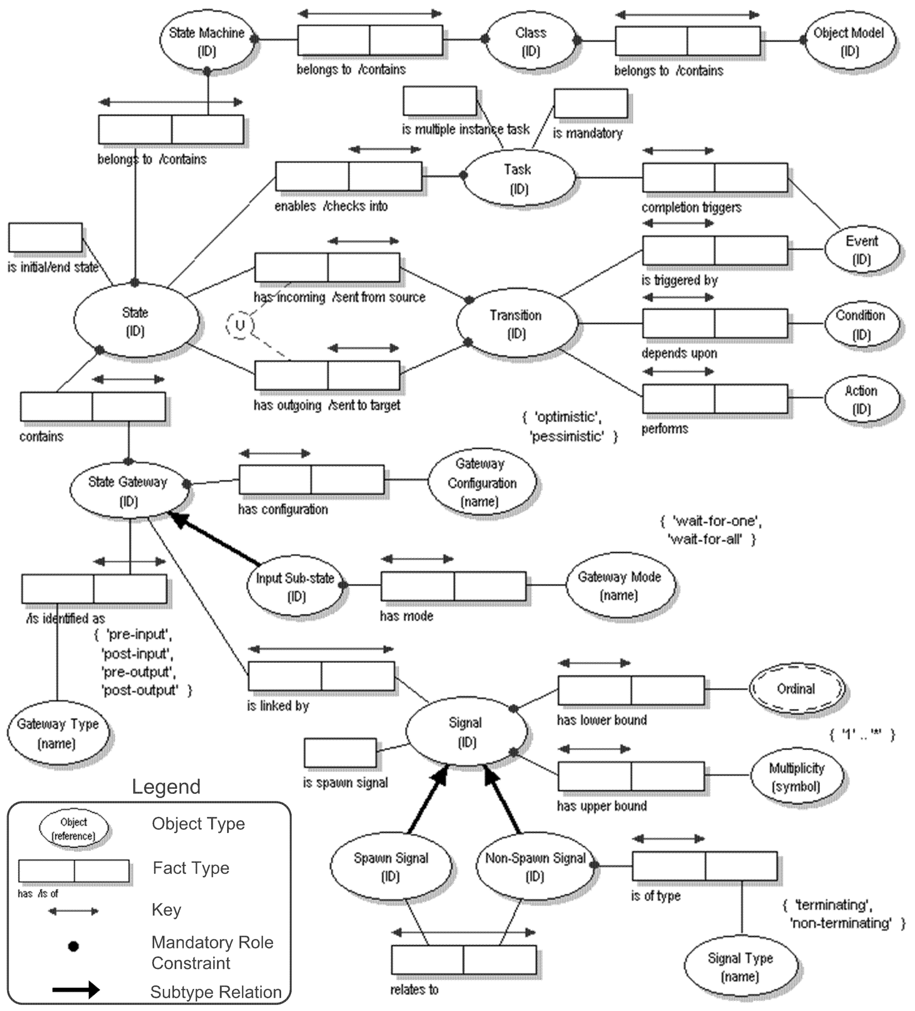


Fig. 3. Object Behaviour Meta-Model

on executable object behaviour models. The meta-model is presented as an Object Role Model (ORM) [8] in Figure 3.

FlowConnect is an attractive source meta-model for our proposal for two reasons. Firstly, FlowConnect seamlessly integrates concepts from UML state diagrams with concepts from UML sequence diagrams, allowing us to capture both intra-object and inter-object behavior in the same model. Secondly, the FlowConnect-based meta-model is a representative of other object-oriented meta-models (e.g. Proclets [9], Merode [10], OCoN [11]), thus it is possible to adapt the results presented here to other meta-models.

At the highest level an **object model** is a container for all classes in an object-oriented model. We define a class as a "cluster" of related states that share some common context. An example of shared context are states that belong to an *inspection*. Grouping these states together allows control flow associations to be made between them and an *inspection* begins to take shape. The object model contains one or more **classes** that contain one or more **state machines**. A state machine contains one or more **states**. A state models a moment in a process lifecycle where the context of a process can be distinguished and named, e.g. *Distribute Report*, *Load Data* or *Test Structural Strength*.

A **transition** connects the output of a state (the source state) to the input of another state (the target state) in the same state machine. Transitions may have an optional Event-Condition-Action (ECA) rule. The occurrence of an **event** will cause the transition labeled by that event to be performed. A transition can have a **condition** associated with it that must be satisfied before the transition can occur. When a transition is performed it may also execute an **action**. The details of an ECA rule language are not specified since this is out of scope. Transitions are indicated on a link between two states by an arrow with an open arrowhead.

Each state contains three sub-states; a **pre-gateway**, **main processing sub-state** and **post-gateway**. The main processing sub-state is where work is completed in a state and it contains zero or more atomic **tasks**. The pre- and post-gateways are the entry and exit points of the main processing sub-state respectively. A pre-gateway is entered when the state it belongs to has been entered by an incoming transition. A gateway may send signals to other state gateways and receive (wait for) signals from other state gateways using an input sub-state to receive incoming signals and an output sub-state to send outgoing signals. The order in which these signals are sent or received depends upon the gateway configuration, i.e. a *pessimistic* gateway will wait to receive all signals it expects before sending any whereas an *optimistic* gateway sends signals before waiting to receive any. An input sub-state also has a mode to specify whether it should wait for the first signal (*wait-for-one*) or all signals (*wait-for-all*) before control flow will be released.

A **signal** establishes a one-way connection between state gateways that belong to two different state machines. In contrast to a transition, a signal does not have an ECA rule. There are three types of signal that are distinguished by the arrowhead on a link between two gateways: **spawn** (i.e. creates a new state machine) indicated by a double-filled arrowhead, **finish** (i.e. terminates a state machine) indicated by a double-empty arrowhead and **message** (i.e. non-terminating) indicated by a single solid arrowhead. A signal has a lower and upper bound, which are the minimum and maximum number of times it can be sent, i.e. a spawn signal with a lower bound of 1 and upper bound of 5 can create between 1 and 5 objects.

Some signals occur in response to, or following another signal. For example, a non-terminating (message) signal *Sig02* can only occur following a spawn signal *Sig01*. Accordingly, the meta-model includes an association between signals which form a **relationship**.

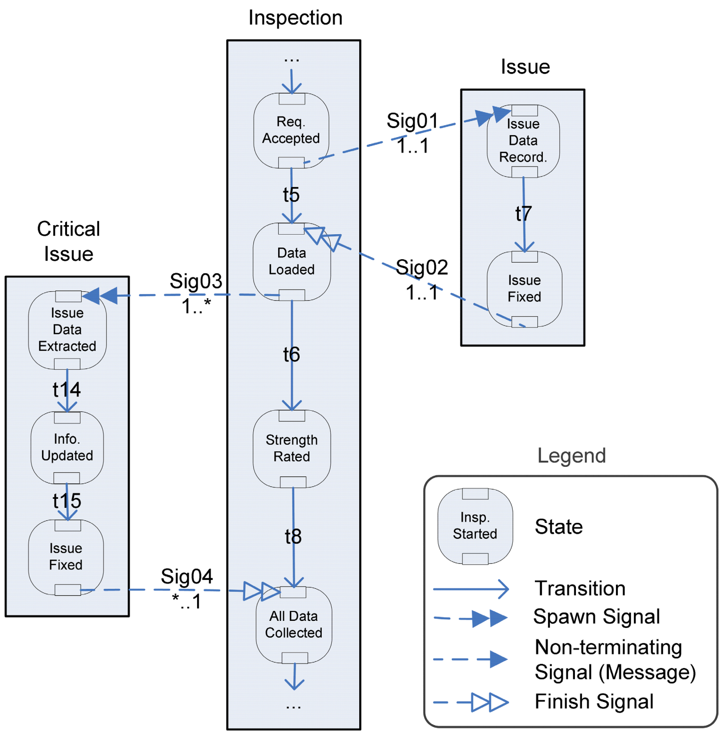


Fig. 4. Example of an Object Behaviour Model

As an example, a fragment of a state machine corresponding to the Inspection class, as well as two related state machines corresponding to the Critical Issue and the Issue classes are shown in Figure 4.

4 From Object Behaviour Models to Process Models

In this section we introduce a proposal to map an object behaviour model to a process model. We use the asset maintenance process as an illustration of this proposal. The essence of the proposal is to analyse the object behaviour model in order to extract a set of elementary causal dependencies between events and signals. These elementary causal dependencies are represented as a *causal matrix*, also known as a *heuristics net* [12]. The idea of using a heuristics net comes from the ProM framework [13], where heuristics nets are used as an intermediate representation to construct a Petri net from an event log.

A heuristics net is composed of a set of transitions, which we call "tasks" to put them in the context of this paper. Each task has an *input* and an *output*. The input of a task T represents the different ways in which task T can be started. Concretely, the input of a task is a set. If this set is empty, it means that the task can be started even if no other task has been completed (i.e. this is the initial task in the process model). If the input of a task is not empty, it contains one of several *disjunctions*. Each of these disjunctions should be read as an "Or" of several tasks. For example, a disjunct { F,B,E } means that either task F or task B or task E have completed. The different disjunctions in an input are implicitly linked through an "And", meaning that each disjunct must be satisfied before the target task can be started. For example, the input of task D is { {F,B,E}, {E,C}, {G} }. For task D to be executed, either F or B or E must be completed, and either E or C must be completed, and G must be completed.

Symmetrically, the output of a task determines which other tasks can be executed after a given task completes. An empty output denotes a final task in the process. Meanwhile, a non-empty output must be read as a set of disjuncts. For example, a disjunction of the form { A,B,C } means that either A or B or C can be executed. An output can contain multiple disjunctions. The output of task A is { {F,B,E}, {E,C}, {G} }, which means that after A completes, either F or B or E will be executed, and either E or C will be executed, and G will be executed. Readers familiar with Petri nets will recognise that a heuristics net is a Petri net of a particular form. The transformation procedure depicted in Figure 5 consists of these steps:

I - Generate a heuristics net from an object model/state machine diagrams.
II - Generate a Petri net from a heuristics net.
III - Transform the Petri net into a YAWL process model.

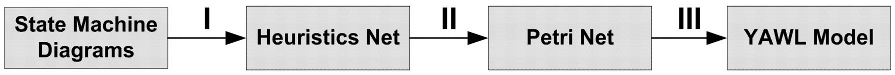


Fig. 5. An Overview of the Transformation Procedure

Below, we present an algorithm that automates **Step I**. For each state in an object model, this algorithm generates two tasks corresponding to the pre- and post-gateway. In other words, each pre- or post-gateway in the object model will lead to one task in the generated heuristics net. Because the main-processing sub-state has no more than one input (from the pre-gateway) and one output (to the post-gateway) it is not represented explicitly in the heuristics net.

Algorithm 1 takes as input an object model and produces the corresponding heuristics net. The algorithm iterates over each state gateway in order to generate an input set (preTask) and output set (postTask). Because there is a one-to-one mapping between state gateways and tasks, the algorithm treats them interchangeably, meaning that it uses the identifiers of gateways in the source object model as identifiers of tasks in the generated heuristics net.

The following auxiliary functions are used in Algorithm 1:

– *states* : ObjectModel → Set of State, is the set of states in an object model.
– *pre, post* : State → Gateway, yields the pre or post gateway of a state.

- *inputTransitions, outputTransitions* : State → Set of Transition, yields the set of input/output transitions.
- *source, target* : Transition → State, yields a transition's source/target.
- *inputSignals, outputSignals* : Gateway → Set of Signal, yields a gateway's input/output signals.
- *mode* : Gateway → GatewayMode, yields a gateway's mode.
- *explode* : Set of Signal → Set of Set of Signal. explode($\{e_1, e_2, \dots, e_n\}$) = $\{\{e_1\}, \{e_2\}, \dots, \{e_n\}\}$.

Algorithm 1. Generation of a Heuristics Net

```
**Input:** om : ObjectModel
**Output:** preTask, postTask : Task → Set of Set of Task
predecessors, successors : Set of Gateway
**foreach** $s \in$ *states(om)* **do**
    predecessors := { post(source(t)) | t ∈ inputTransitions(s) };
    successors := { pre(target(t)) | t ∈ outputTransitions(s) };
    preInputSignals := { source(g) | g ∈ inputSignals(pre(s)) };
    preOutputSignals := { source(g) | g ∈ inputSignals(post(s)) };
    postInputSignals := { target(g) | g ∈ outputSignals(pre(s)) };
    postOutputSignals := { target(g) | g ∈ outputSignals(post(s)) };
    **if** *mode(pre(s)) = wait-for-one* **then**
        preTask(pre(s)) := { predecessors, preInputSignals };
    **else**
        preTask(pre(s)) := { predecessors } ∪ explode(preInputSignals);
    postTask(pre(s)) = { { post(s) }, postInputSignals(s) };
    **if** *mode(post(s)) = wait-for-one* **then**
        preTask(post(s)) := { { pre(s) }, preOutputSignals(s) };
    **else**
        preTask(post(s)) := { { pre(s) } } ∪ explode(preOutputSignals(s));
    postTask(post(s)) := { successors } ∪ explode(postOutputSignals(s));
**end**
```

To analyse the inbound and outbound causal dependencies of a gateway, we conceptually decompose each gateway into two parts: the input and the output. The input corresponds to the signals the gateway has to wait for, while the output corresponds to the signals it has to send out. Figure 6 depicts the decomposition of the pre- and post-gateways of a state into an input and output part.

The input and output sets are generated as follows. The pre-gateway input set is the union of the source of each incoming transition with the source of each incoming signal, depending on the gateway mode (i.e. *wait-for-one* or *wait-for-all*). If the gateway mode is *wait-for-one* then the preTask set is the set of input transition sources (predecessors) and the set of pre-gateway input signal sources. However if the gateway mode is *wait-for-all* then the preTask set is the union of the set of input signal sources converted to a set of set of signals by the *explode* function with the set of predecessors. The postTask set consists of the post-gateway and the targets of all outgoing signals sent from the pre-gateway. The procedure for constructing the input and output sets for a post-gateway is

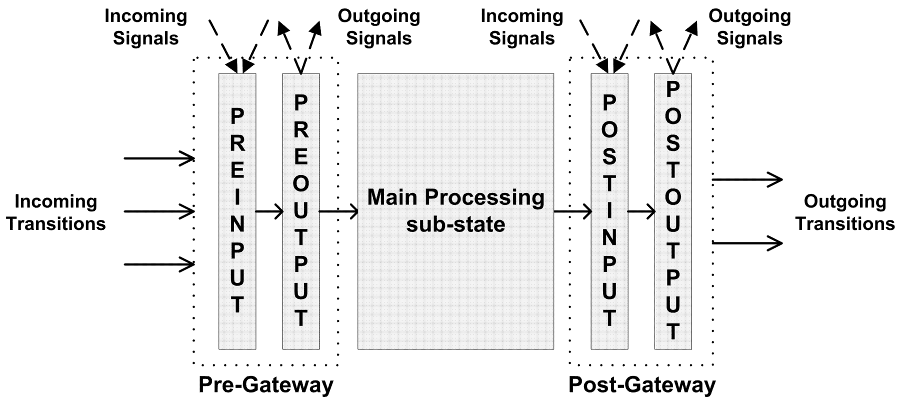


Fig. 6. Pre- and Post-Gateways of a State

symmetric to the corresponding procedure for a pre-gateway. After completion of the algorithm a heuristics net is constructed by inserting the input and output set as individual rows in the net.

A heuristics net is a 'flat' representation of a process since it does not capture sub-processes. When converting a heuristics net to a YAWL net, it is desirable to incorporate sub-processes. This can be achieved by identifying "sub-process delimiters" in the object behaviour model. These delimiters are the points where an instance of a state machine is created, and the point(s) where a state machine returns a terminating signal to its parent. In the absence of message signals between the parent and child state machines, the region between a spawn signal and the finish signal in the resulting YAWL net will be a single-entry-single-exit (SESE) region.

A SESE region in the resulting YAWL net whose entry point corresponds to a spawn signal with a multiplicity of one, and whose exit point correspond to a matching finish signal is converted into a YAWL composite task (i.e. the YAWL construct for capturing sub-processes). Similarly, a SESE region whose entry point corresponds to a spawn signal with a multiplicity greater than one is converted into a YAWL multiple instance composite task (i.e. a sub-process that is executed multiple times concurrently). This procedure of identifying sub-process delimiters in an object model and restoring the delimiters back in the resulting YAWL net allows us to obtain more modular YAWL nets.

In **Step II** the heuristics net is passed to the Heuristics Net conversion tool in ProM to obtain a Petri net. **Step III** is performed using a workflow net conversion plugin in the ProM framework to combine the sub-process delimiters with the derived Petri net to create an 'unflattened' YAWL process model where properties such as task multiplicity, state machine multiplicity and sub-process definitions are restored in the YAWL model as shown in Figure 7. This step is merely a syntactic transformation that aims to exploit the constructs in YAWL because any Petri net can be seen as a YAWL process model. The complete

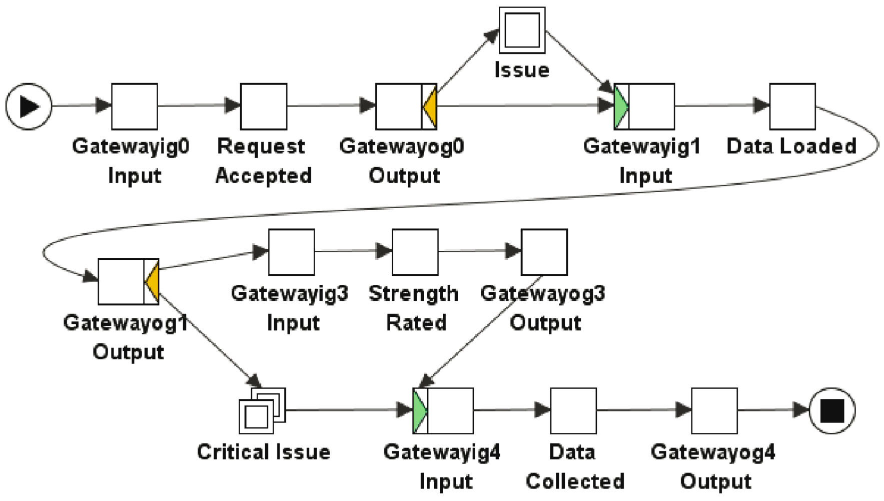


Fig. 7. Resulting YAWL Process Model

model is a process-oriented view of the control flow between the input and output gateways for every state in an object model.

Implementation
The proposal has been implemented in Java on top of the Eclipse platform.[1] The tool includes a graphical editor for object behaviour models and a module that support the transformation of object behaviour models to YAWL nets. The tool implementation relies on libraries from the ProM framework to perform the transformation from heuristic nets to Petri nets and from Petri nets to flat YAWL models. Subsequently, these YAWL models are unflattened as explained above.

The modelling tool and the model transformation technique have been tested using the asset maintenance example. Future plans for this tool includes adding support for data flow modelling and resource modelling and enhancing the model transformation technique to cater for these modelling perspectives.

5 Related Work

Object-oriented (OO) design methodologies that use the UML to design and develop Information Systems have been proposed such as OCoN [11]. These proposals link UML diagrams to phases of the process development lifecycle to produce a schema as output at the conclusion of the lifecycle. Our proposed approach would extend these design methodologies and allow process analysts and designers to produce a completely process-oriented view of an OO model. FlowConnect [7] and Proclets [9] are examples of an OO system and notation.

[1] `http://www.eclipse.org/platform/`

Reijers et al. proposed a methodology for Product-Based Workflow Design (PBWD) that presented an analytical clean-sheet approach for process design specified by the bill-of-material for products that are affected by the process [14]. Since PBWD focuses on the bill-of-material, we consider this is an OO modelling approach. In PBWD there is no notion of object life-cycles, which is an area covered by artefact-centric process modelling [15]. The artefact-centric approach unifies data and process in an "Operational Specification" but does not consider converting these specifications to other modelling representations.

An architecture for mapping between OO and activity-oriented process modelling approaches has been proposed by Snoeck et al. [10]. This architecture maps OO development to process modelling. Object associations and business rules are captured using object-relationship diagrams and an object-event table models the behaviour of domain objects, which are similar to our mapping artefacts.

6 Conclusions and Future Work

Object technology is a mainstream approach to implementing Information Systems. Mainstream object-oriented analysis and design practices (e.g. those based on UML) are based on concepts of objects whose structure is captured as classes and whose behavior and interactions are captured as state machines, sequence diagrams and similar notations. On the other hand, recent trends have seen an uptake of approaches to Information Systems engineering that treat processes as a central concept throughout the development lifecycle.

The co-existence of these two approaches may lead to situations where a project starts with a model corresponding to one approach and needs to switch to a model corresponding to the other approach. In this paper, we have proposed an approach to help bridge these differences in terms of the control flow logic and discussed how the conversion technique has been implemented.

Future work will continue on the topic of transforming object-oriented models to process-oriented models and vice-versa. There is a need to cover not only the control-flow aspects as outlined in this paper, but also data flow and resource allocation. We also note that a similar problem arises in the opposite direction, i.e. moving from a process-oriented to object-oriented design for the purpose of implementing process-oriented design models using object-oriented technology. Therefore another future challenge will be a proposal of a reverse transformation from process-oriented to object-oriented models.

Acknowledgement. This work is supported by an Australian Research Council Linkage grant (LP0562363) co-funded by Shared Web Services Pty Ltd.

References

1. Kueng, P., Bichler, P., Kawalek, P., Schrefl, M.: How to compose an object-oriented business process model? In: Proceedings of IFIP TC8, WG8.1/8.2 working conference on method engineering, pp. 94–110. Chapman and Hall, London, UK (1996)

2. Booch, G., Rumbaugh, J., Jacobson, I.: The Unified Modeling Language User Guide. Addison-Wesley Professional, Reading (1998)
3. Object Management Group: Business Process Modelling Notation, Ver 1.0 (2006), http://www.bpmn.org
4. Andrews, T., Curbera, F., Dholakia, H., Goland, Y., Klein, J., Leymann, F., Liu, K., Roller, D., Smith, D., Thatte, S., Trickovic, I., Weerawarana, S.: Business Process Execution Language for Web Services, Version 1.1 (2003), http://dev2dev.bea.com/webservices/BPEL4WS.html
5. van der Aalst, W.M.P., ter Hofstede, A.H.M.: YAWL: Yet Another Workflow Language. Information Systems 30, 245–275 (2005)
6. Becker, J., Kugeler, M., Rosemann, M.: Process Management. Springer, Heidelberg (2003)
7. Shared Web Services Pty. Ltd.: FlowConnect Model (August 2003)
8. Halpin, T.: Information modeling and relational databases: from conceptual analysis to logical design. Morgan Kaufmann Publishers Inc., San Francisco (2001)
9. van der Aalst, W.M.P., Barthelmess, P., Ellis, C.A., Wainer, J.: Proclets: A Framework for Lightweight Interacting Workflow Processes. International Journal of Cooperative Information Systems 10, 443–481 (2001)
10. Snoeck, M., Poelmans, S., Dedene, G.: An architecture for bridging OO and business process modelling. In: 33rd International Conference on Technology of Object-Oriented Languages (TOOLS), Mont-Saint-Michel, France, pp. 132–143 (2000)
11. Wirtz, G., Weske, M., Giese, H.: The OCoN Approach to Workflow Modeling in Object-Oriented Systems. Information Systems Frontiers 3, 357–376 (2001)
12. van der Aalst, W.M.P., de Medeiros, A.K.A., Weijters, A.J.M.M.: Genetic Process Mining. In: Ciardo, G., Darondeau, P. (eds.) ICATPN 2005. LNCS, vol. 3536, pp. 48–69. Springer, Heidelberg (2005)
13. van Dongen, B.F., de Medeiros, A.K.A., Verbeek, H.M.W., Weijters, A.J.M.M., van der Aalst, W.M.P.: The ProM Framework: A New Era in Process Mining Tool Support. In: 26th International Conference on Applications and Theory of Petri Nets (ICATPN), Miami, USA, pp. 444–454 (2005)
14. Reijers, H.A., Limam, S., van der Aalst, W.M.P.: Product-Based Workflow Design. Journal of Management Information Systems 20, 229–262 (2003)
15. Nigam, A., Caswell, N.S.: Business artifacts: An approach to operational specification. IBM Systems Journal 42, 428–445 (2003)

Perspective Oriented Business Process Visualization

Stefan Jablonski and Manuel Goetz

Chair for Databases and Information Systems, University of Bayreuth
Universitätsstraße 30, 95447 Bayreuth, Germany

Abstract. Visualizing business process models in various ways supports modelers in creation and users in understanding. Therefore we present a flexible and extensible meta model based approach to enhance business process models by multiple visualizations. These visualizations are geared to the so called perspectives of a business process model and emphasize different aspects of a business process model.

1 Introduction

Most process modeling systems offer some kind of graphical notation. There are also standards like Business Process Modeling Notation (BPMN) [13] that even define the appearance of such a notation. Although these systems and standards have different notations, they all aim at the visualization of business process models in a manner that modelers and users can better grasp them to improve modeling and decrease misunderstandings. For instance, the OWL-S Editor plugin [4] for Protégé is good for modeling web services and their calling dependencies. However, the visualization is not flexible enough to change between different views on modeled web services. This would, among other things, foster the understanding of complex web service scenarios. As another example, the YAWL system itself is flexible and extensible [1], but the YAWL designer [18] concentrates on visualizing an advanced control flow model and its decomposition. New visualizations, for instance a Rule Editor, can be added, but this just adds information to a given visualization and does not introduce a different new visualization. Improvise [2] allows attachment of arbitrary and even multimedia data to common nodes in order to increase comprehensibility of a business process model. In [17] business process models are visualized three-dimensional to achieve this goal. While these approaches extend a single visualization, ARIS offers a number of different views [6] on a business process model. Among other things, there is one view to show the complete business process and there are other views that are refinements of special perspectives of a business process; for instance, one view shows the underlying organizational structure of a business process model.

We principally favor the provision of different views on a business process model. However, in conventional process modeling tools these visualizations are implemented firmly. We foster in this paper an approach that provides visualizations flexibly according to the so called perspectives of a business process model. That concretely means that

- we offer a set of different visualizations and
- these visualizations can be flexibly applied to different perspectives of a business process model.

This concept – for instance – facilitates to represent a business process model in a bubble-and-arc notation according to [16] which is depicted in Figure 6. However, it is also possible to represent the same business process model in a swimlane presentation (cf. Figure 11). Potentially, the organizational perspective (Section 0) of the same business process model is shown exclusively; but this latter model can also be extended by information about the operational or other perspectives (Figure 11). Moreover, it is possible to switch between the various presentations flexibly.

We are convinced that multiple views onto a business process model are important to communicate to the modeler and/or user since business process models are complex artifacts. Being able to analyze them in multiple ways does very often facilitate their better understanding. We experienced this in multiple projects where through the variable presentation of a business process model they could be improved and could be better adjusted to the real application scenarios.

The business process model visualization concept contributed in this paper is based on the so called perspective oriented process modeling which accordingly will be presented in Section 0. Section 0 then discusses the visualization concept in detail.

2 Perspective Oriented Process Modeling

We pursue the approach of perspective oriented process modeling POPM ([8], [9]). The idea behind POPM is that a modeling construct consists of several orthogonal building blocks, which we call perspectives. The composition of these blocks then defines a process modeling construct, e.g. a process step. We identified five main perspectives for a basic process modeling language:

- The Functional perspective identifies a process step and defines its purpose. Also the composition of a process is determined by this perspective.
- The Data (flow) perspective defines data used in a process and the flow of data between process steps. Furthermore, this perspective also relates process data to (external) data models.
- The Operational perspective specifies which operation (service) is invoked in order to execute a process step. It relates processes to services stemming from (external) service libraries. Execution of a process step is done after selecting a task from a worklist. The module rendering and executing the service is specified in the Operational Perspective.
- The Organizational perspective defines agents (for instance users, roles) who are eligible and/or responsible to perform a process step. It relates agents to (external) organizational models/charts. These agent definitions are interpreted at execution runtime to assign tasks to people.
- The Control flow or Behavioral perspective is used to define causal dependencies between modeling elements (e.g. step "B" may only be executed after step "A"). Often these dependencies are called control flow.

It is crucial that this list of perspectives is neither complete nor fixed. Further perspectives can be added easily. In order to be able to support this extensibility the POPM is based on a layered meta model [7]; the layering of this meta model is borrowed from the Meta Object Facility (MOF) meta model [14] (refer to Figure 1, left column). It is out of the scope of this paper to define this meta model, but it has to be mentioned that perspectives are defined on the meta model level M2. On level M1, concrete occurrences of these perspectives are defined, i.e. business process types are specified. The usages of these business process types are then defined on level M0. So in principle, the description of a business process model is done on this level M0; here sub-processes, agents to execute a concrete process, applications to use within a process and data to be passed between processes are specified.

3 Visualization of Business Process Models

3.1 Using a Meta Model for Visualization

We exploit the meta model structure of the POPM approach to business process modeling by organizing business process model visualizations in a similar manner as the POPM approach is organized.

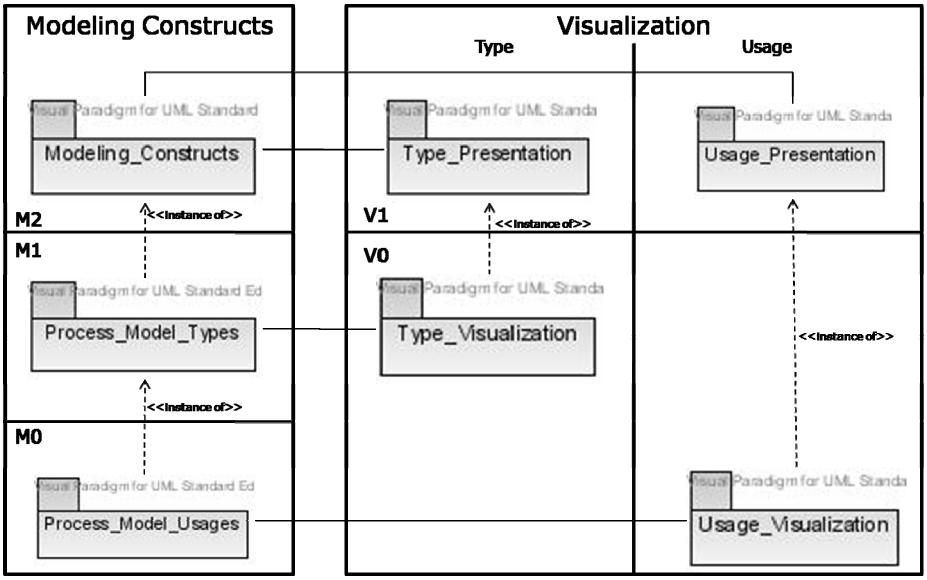


Fig. 1. Relationship between meta model for business process models and its visualization

In Figure 1 the principle relationship between the meta model for business process models and its visualization is depicted. The visualization meta model distinguishes between visualizations of process types and process usages. Besides it is slightly differently structured than the meta model for business process models which is justified according to [5]. On meta level V1 pools for graphical modeling elements are contributed (*Type_Presentation, Usage_Presentation*) which can be taken to form different types of diagrams. For instance, one diagram for swimlanes (Figure 11) and another diagram for bubbles-and-arcs notation (Figure 6) is defined. We principally distinguish pools for process type and process usage presentations (where a process usage is an instance of a process type), although in this paper we just concentrate on usage presentation to comply with paper length instructions. The two pools for graphical modeling elements are associated with the modeling constructs (*Modelling_Constructs*) of the meta model for business process models.

On meta level V0 concrete visualizations for process types (*Type_Visualization*) and process usages (*Usage_Visualization*) are instantiated. These two packages are connected to the corresponding process type definitions and process usage definition of the meta model for business process models.

Up to now the visualization meta model describes how each process type and process usage, respectively, is presented in – possibly – different diagram types. For instance, the organizational perspective is alternatively presented as glue-box (cf. Figure 6) and as swimlane (cf. Figure 11). So, the static part of the presentation of a business process model is given. In order to layout such a process description a layout algorithm has to be provided that determines how the diverse elements of a process model are displayed. In [3] several algorithms to define such a layout are given. Since we provide presentation information for each layout explicitly in the visualization meta model, the number of layout algorithms grows in a linear manner with the number of presentations that are provided for a business process model. In this paper we do not investigate the application of layout algorithms in more detail but concentrate on the provision of presentation information in the visualization meta model.

3.2 Detailing Presentation for Process Instances on Meta Level V1

In this subsection the meta model for business process visualization will be analyzed, i.e. the right column of Figure 1 will be refined. Firstly, we detail the structure of the package *Usage_Presentation* (Figure 2). This package is divided into three layers. On the upper layer (Basic Shape Layer) all available graphical modeling elements are described (Figure 3); it is also described how these modeling elements can be connected. For instance, circles, rectangles and arcs are defined; moreover, it is described that arcs could connect rectangles, but it is forbidden that arcs are connected to further arcs.

On the middle layer (Diagram Type Layer) of the package *Usage_Presentation* different types of diagrams are defined by restricting the modeling features of the package *Basic_shapes*. Here we define – for the time being – two different diagram types: "normal" graph structures are presented by the package *Presentation_graphStructured*; block structured diagrams can be derived from the package *Presentation_blockStructured*. Among other things, block structured diagrams allow nesting of modeling constructs (here especially: rectangles), while this nesting is forbidden for graph structures. Both packages are specializations of the package *Basic_shapes* what supports the reuse of visualization constructs.

The lower layer (Presentation Layer) of the package *Usage_Presentation* defines presentations which are instantiated afterwards for the visualization of concrete business process models. These presentations are specializations of the two packages of the Diagram Type Layer. The classes of the packages of the Presentation Layer are associated with concrete classes of the business process meta model in order to define how actual business processes are visualized.

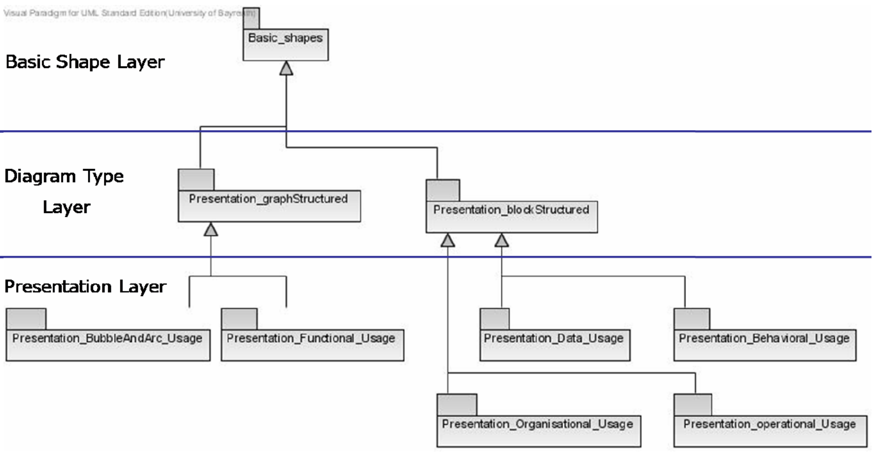

Visual Paradigm for UML Standard Edition(University of Bayreuth)
Basic Shape Layer
Basic_shapes
Diagram Type Layer
Presentation_graphStructured
Presentation_blockStructured
Presentation Layer
Presentation_BubbleAndArc_Usage
Presentation_Functional_Usage
Presentation_Data_Usage
Presentation_Behavioral_Usage
Presentation_Organisational_Usage
Presentation_operational_Usage

Fig. 2. Structure of *Usage_Presentation*

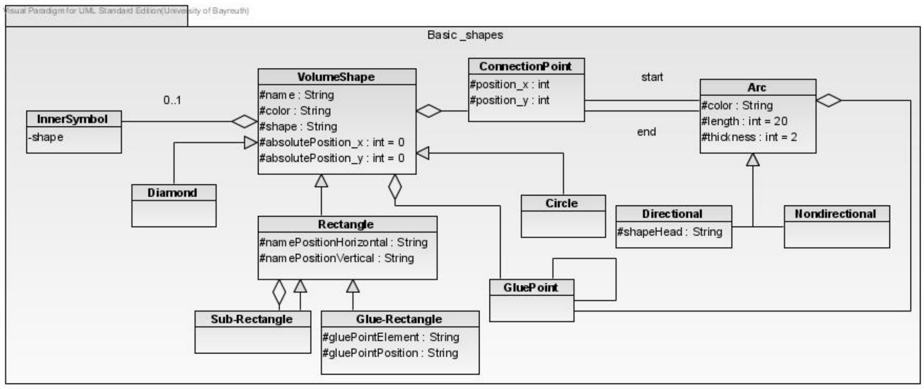


Fig. 3. Basic shapes

In the following we detail some of the packages of Figure 2 in order to demonstrate how flexible visualizations for business process models can be enacted.

3.3 Presentations of the Diagram Type Presentation_graphStructured

Graph structured diagrams (Figure 4) describe business processes through rectangles, circles and diamonds (and further elements) that are connected through arcs. Thus the classes *Rectangle*, *Diamond* and *Circle* of the package *Basic _shapes* are specialized and *ConnectionPoints* are assigned to those shapes in order to provide connecting points for arcs that connect those shapes in a visualization. These specializations together with the class *Glue_RectangleRectangle* are the main classes of the package *Presentation_graphStructured* while the other classes of that package mainly describe what shapes may be connected. The class *Main_Rectangle* can also have glue-boxes, i.e. rectangles that are glued to a *Main_Rectangle*. For that reason

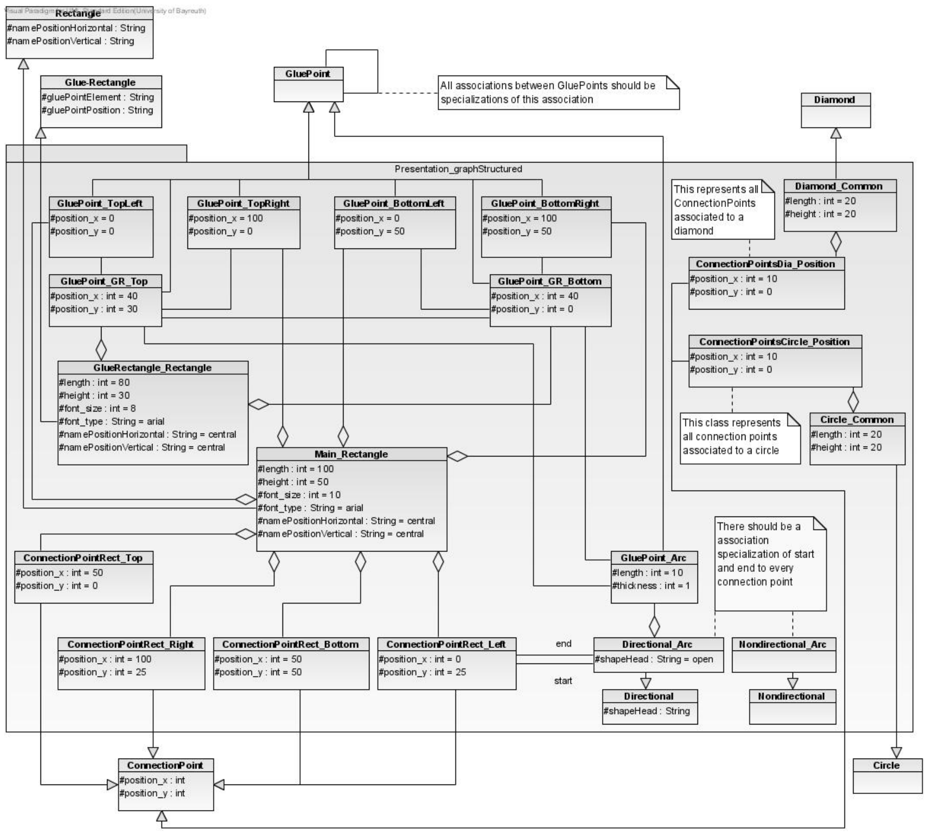

Rectangle
#namePositionHorizontal : String
#namePositionVertical : String
Glue-Rectangle
#gluePointElement : String
#gluePointPosition : String
GluePoint
All associations between GluePoints should be specializations of this association
Diamond
Presentation_graphStructured
GluePoint_TopLeft
GluePoint_TopRight
GluePoint_BottomLeft
GluePoint_BottomRight
This represents all ConnectionPoints associated to a diamond
Diamond_Common
GluePoint_GR_Top
GluePoint_GR_Bottom
ConnectionPointsDia_Position
ConnectionPointsCircle_Position
GlueRectangle_Rectangle
Main_Rectangle
This class represents all connection points associated to a circle
Circle_Common
There should be a association specialization of start and end to every connection point
ConnectionPointRect_Top
GluePoint_Arc
ConnectionPointRect_Right
ConnectionPointRect_Bottom
ConnectionPointRect_Left
end
start
Directional_Arc
Nondirectional_Arc
Directional
Nondirectional
ConnectionPoint
Circle

Fig. 4. Definition of graph structured diagram layout

Glue_RectangleRectangle and *Main_Rectangle* have associations to glue points. Glue-boxes can also be glued to a directional arc in order to provide additional information concerning that arc, e.g. the data that “flows” on an arc.

As all associations defined in the *Presentation_graphStructured* are specializations of associations of the package *Basic_Shapes*, restrictions concerning the connectivity of elements are established. A glue-box, for instance, is not allowed to be connected to another element with an arc.

A derived presentation of a graph structured diagram and hence a package of the *Presentation Layer* is the *Presentation_BubbleAndArc* (Figure 5). As allowed connections are already defined in the corresponding diagram type no further refinement is needed. The graphical elements are derived from the main classes in the *Presentation_graphStructured* and associated with modeling constructs. Additional information concerning the graphical representation on this level is the initial setting of attributes. A process step (*Process_BubbleAndArc_Usage*), for instance, is derived from the class *Main_Rectangle* and hence is allowed to be connected with arcs and to have glue-boxes; the additional graphical information in package *Presentation_BubbleAndArc* is the initial setting of the color which is yellow.

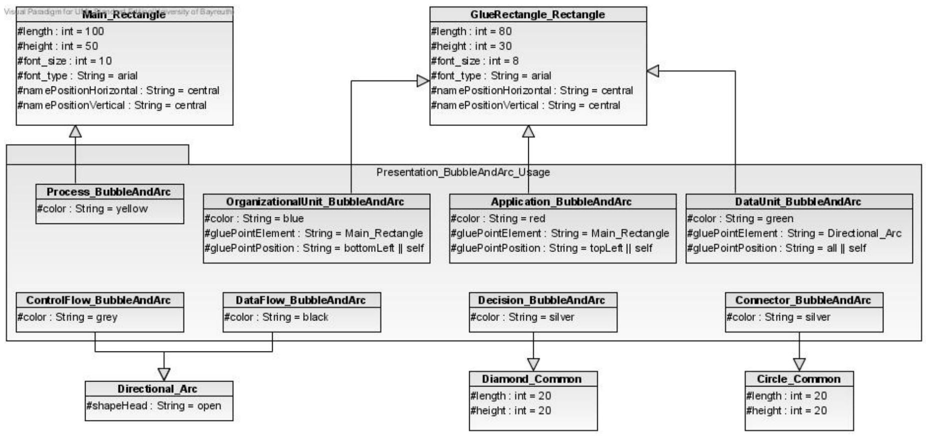


Fig. 5. Derived presentation of bubble-and-arc visualization

A business process model associated with an instance of the *Presentation_BubbleAndArc_Usage* is depicted in Figure 6 and Figure 7 whereas the process depicted in Figure 7 is a sub process of process step *Pre-clinical Preparation Phase* of Figure 6. Process steps bear organizations and applications as glue-boxes and are connected with directional arcs. These arcs can also have glue-boxes, which is meaningful in case of a data flow for displaying the corresponding data. This example shows an extract of a process model created in a clinical project where a hip replacement has been modeled. Figure 6 visualizes the top level process which consists of the main phases of a hip replacement, whereas Figure 7 details one of its main phase, namely *Pre-clinical Preparation Phase* (decomposition).

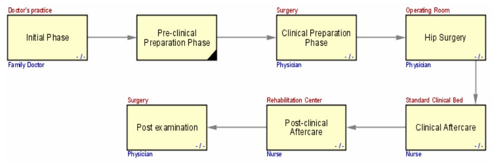


Fig. 6. Hip Prostheses in bubble-and-arc visualization

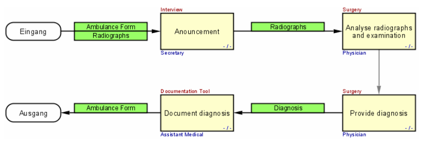


Fig. 7. Pre-clinical Preparation Phase

The instantiated classes of *Presentation_BubbleAndArc_Usage* contain the status of the single graphical elements while a business process model is visualized. For instance, Figure 8 shows some objects needed when displaying Figure 7.

Another presentation derived from the package *Presentation_graphStructured* is the *Presentation_Functional_Usage*. The main purpose of this presentation is to display the hierarchy of processes and sub-processes which is usually done by displaying a process tree. Although the two presentations are different with respect to its visualization, they are graphically derived from the same diagram type, namely *Presentation_graphStructured*; for instance, flows displayed as arcs when using the bubble-and-arc visualization are visualized as glue-boxes for readability reasons when

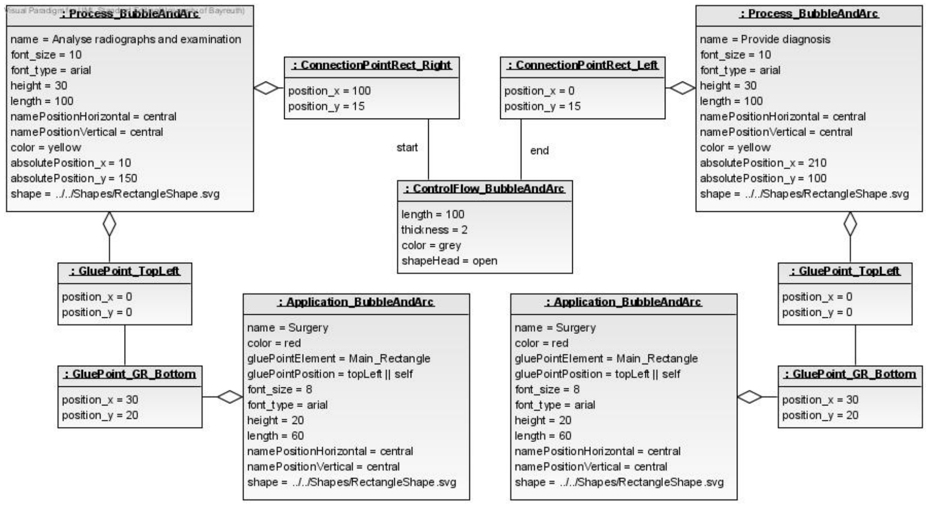


Fig. 8. Resulting objects for bubble-and-arc visualization (Figure 7)

using the functional presentation. From a business process modeling point of view, these two visualizations highlight completely different aspects of a business process.

As an example, the process model of Figure 6 and Figure 7 (bubble-and-arc visualization) is depicted in Figure 9 being visualized according to *Presentation_Functional_Usage*. This visualization was helpful while defining the process when interviewing physicians: the composition structure of the "Hip Prosthesis" could be grasped at a glance and misunderstandings could be corrected in a very early project phase.

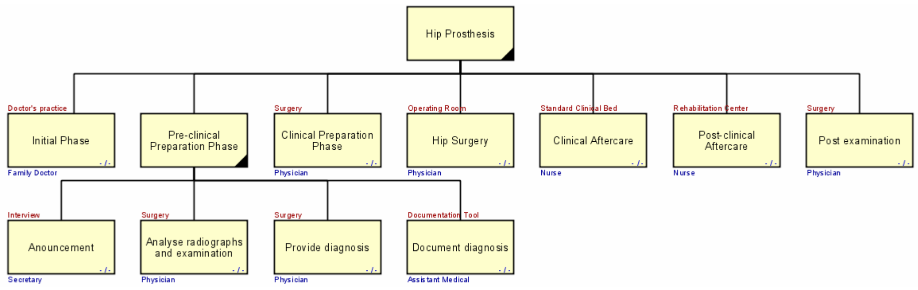


Fig. 9. Process "Hip Prosthesis" in functional visualization

3.4 Presentations of Diagram Type Presentation_blockStructured

Although the graph structured diagram type is commonly used and supports some very useful visualizations, it cannot express all visualization needed. For instance, one restriction of the graph structured presentation is that nesting is forbidden. Therefore another diagram type called block structured diagram is defined in package

Presentation_blockStructured (Figure 10). Its main purpose is the creation of a graphical notation containing swimlanes to display dependencies between business process modeling constructs this way. Swimlanes are – from a graphical viewpoint – nested rectangles.

Main classes of this presentation are *Main_Rectangle* and *Level1_Sub-Rectangle*, which are derived from *Rectangle* respectively *Sub-Rectangle* of package *Basic_shapes* (Figure 3). *Sub-Rectangles* are nested into *Main_Rectangles* in this presentation as *Main_Rectangles* represent the swimlanes. As opposed to *Main_Rectangles*, glue-boxes are allowed to be glued to *Sub-Rectangles*. Glue-Boxes are both sub-rectangles (as they are in a *Main_Rectangle*) and glued boxes (as they are glued to a *Level1_Sub-Rectangle*). Therefore the class *Level2_GlueSub-Rectangle* is derived from both corresponding classes in the package *Basic_shapes*.

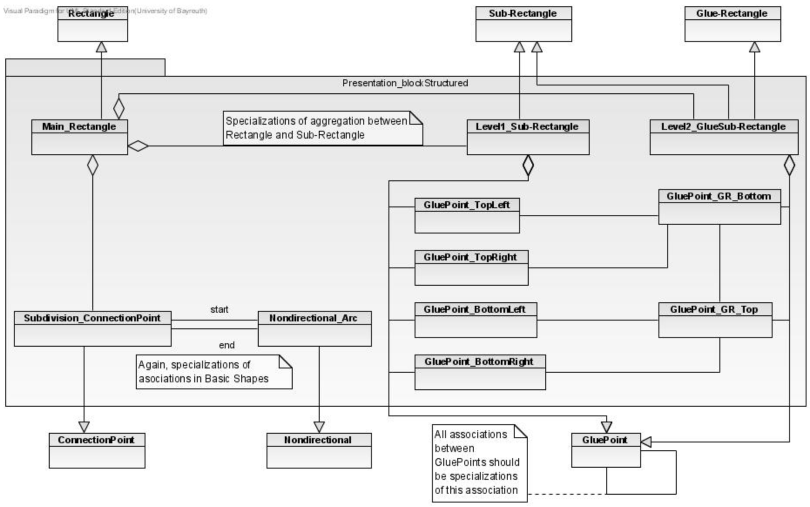


Fig. 10. Definition of block structured diagram layout

All of the remaining perspectives (Figure 2) imply 1:n-dependencies with respect to their visualization purposes:

- One organization is responsible for several processes
- One application is used for execution of several process
- One data usage can be produced and consumed by several processes
- One workflow pattern [19] can appear several times in a process model

An appropriate visualization for 1:n-dependencies is a swimlane notation, as with nesting the n elements into one swimlane the relationship is obvious. Since a swimlane notation is appropriate for displaying 1:n-dependencies, it is used for displaying the remaining perspectives.

In the following, a visualization emphasizing the organizational perspective will be discussed. Again a package for a presentation is defined which is derived from *Presentation_blockStructured* (Figure 10). This presentation focuses on organizations and the processes organizations are responsible for. Therefore the graphical construct for an organization in the organizational presentation is specialized from *Main_Rectangle* which represents the swimlane. The graphical construct for a process is derived from *Level1_Sub-Rectangle* and hence the only element that can directly be nested in a swimlane. Remaining classes are derived from *Level2_Sub-Rectangle* and can be glued to a process to provide additional information concerning the processes.

The example process of Figure 6 and Figure 7 visualized in an organizational visualization (Figure 11) shows the difference in layout and visualization structure compared to the diagrams instantiated and derived from the graph structured diagram type. For instance, there are no arcs defined connecting processes. In our clinical project, the organizational visualization has been used mainly for administration purposes like surgery planning. Furthermore treatment costs could be calculated more precisely as the different organizations participating in a treatment can be derived easily from this visualization. Combining these process-to-organization associations with average salaries and execution times for process steps, good calculations of the hip surgery could be derived.

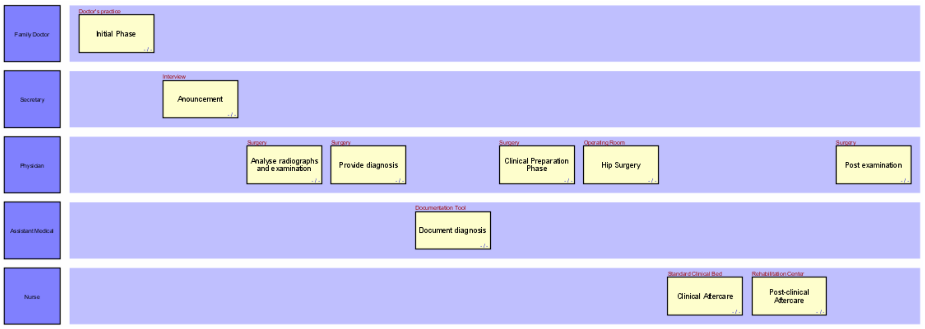


Fig. 11. Organizational visualization

3.5 Evaluation and Use Cases

The approach proposed offers multiple advantages. As visualization and business process modeling is defined on a meta level, it is extremely extensible and customizable in different ways. Single visualization elements, diagram types or presentations can be added, removed or modified depending on the application. This is a major advantage, especially when domain specific modeling is pursued. Here, adding new modeling constructs is a most convenient method to support application specific modeling [10]. We pursue the idea of domain specific modeling since it increases acceptance of process models by users decisively [11]. One conceptually minor customization, but important adjustment from an users' point of view, is to select appropriate graphical elements for modeling: for example, it might be very important to describe processes by circles instead of rectangles since the users are accustomed to this.

Another advantage of our approach is its arbitrary customizability to process modeling methods. For instance, BPMN provides a standard visualization mechanism for business process models. Adopting this visualization mechanism can be done in a straightforward manner in our approach: the package *Basic_shapes* already contains all required modeling elements. Consequently defining a diagram type and a presentation is the only effort to support BPMN process modeling in our approach.

Another advantage of our approach is the ability to include more than one business process model into a visualization. For instance, if more than one business process model share the same organizational elements (i.e. the same agents are responsible to perform the steps of the corresponding business processes), it is possible to depict these multiple business processes within one visualization. Ambition is – among other presentations – to integrate these different business processes into a swimlane presentation. The great advantage of this is that profound analysis of inter-process dependencies is made possible.

4 Conclusion

The main contribution of this paper is to introduce a flexible visualization mechanism. The flexibility of the visualization is achieved by defining presentations on a meta level and associating their elements to modeling constructs. Experiences with a first prototypic implementation have confirmed that perspective oriented visualization improves the design of business process models and decreases misunderstandings between modelers and users. Furthermore the outlook of business process models can be adapted to customers' needs.

The implementation of our approach is heavily making use of sophisticated modeling concepts like powertypes [11] to implement the desired flexibility and handle the mentioned problems concerning the meta levels. We did not discuss implementation details here but focused on the approach in general and especially on its effect on acceptance of process models for users.

As the presented approach is that generic, it also can be used for integration purposes. For instance, several business process models can be displayed in one diagram, which is meaningful for some visualizations in order to analyze a group of business process models with respect to special aspects. Furthermore, transformation of business process models between different notations is supported whereupon we assume that these notations have been defined as presentations on level V1 before.

Finally, we conclude that creating and integrating a visualization mechanism on a meta level is a adequate way to meet many modelers' and customers' special needs with maintainable effort as requirements necessary for that purpose like flexibility and extensibility are implemented on a conceptual level. Therefore reuse of graphical elements and diagram types is feasible and implementation effort is reduced a lot.

References

1. Adams, M., ter Hofstede, A.H.M., Edmond, D., van der Aalst, W.: Implementing dynamic flexibility using worklets. BPMCenter Report, vol. BPM-06-06. BPMCenter.org (2006)
2. Barghouti, N., Koutsofios, E., Cohen, E.: Improvise: Interactive Multimedia Process Visualization Environment. In: Proceedings of the Fifth European Software Engineering Conference (1995)

3. Chok, S., Mareriott, K., Paton, T.: Constraint-based Diagram Beautification. In: Proceedings of the IEEE Symposium on Visual Languages (1999)
4. Elenius, D., Denker, G., Martin, D., Gilham, F., Khouri, J., Sadaati, S., Senanayake, R.: The OWL-S Editor – A Development Tool for Semantic Web Services. In: ESWC. The Semantic Web: Research and Applications, Proceedings of the Second European Semantic Web Conference, Crete, Greece, pp. 78–92 (2005)
5. Gonzalez-Perez, C., Henderson-Sellers, B.: A powertype-based metamodelling framework. Software & System Modeling 5(1), 72–90 (2006)
6. IDS Scheer: "ARIS Platform", (retrieved 2007-06-05), http://www.ids-scheer.de/sixcms/media.php/2152/PR0207-D-BR_final.pdf
7. Jablonski, S., Volz, B., Dornstauder, S.: A Meta Modeling Framework for Domain Specific Process Management and its Implementation. Technical Report, University of Bayreuth, Department of Applied Informatics, (submitted for publication, 2007)
8. Jablonski, S.: Functional and behavioural aspects of process modelling in workflow management systems. In: Chroust, G., Benczur, A. (eds.) Proceedings of CON 1994, Workflow Management: Challenges, Paradigms and Products, Linz, Austria, R. Oldenbourg München, pp. 113–133 (1994)
9. Jablonski, S., Bussler, C.: Workflow Management: Modeling Concepts, Architecture and Implementation. Thomson International Computer Press (1996)
10. Jablonski, S., Faerber, M., Schneider, T.: A Comprehensive Modeling Language for Clinical Proceses. In: ECEH 2007. 2nd European Conference on eHealth, Oldenburg, Germany (2007)
11. Jablonski, S., Lay, R., Meiler, C., Faerber, M., Volz, B., Dornstauder, S., Götz, M., Müller, S.: Integrated Process and Data Management for Healthcare Applications. International Journal of Healthcare Information Systems and Informatics (accepted for publication)
12. Lenz, R., Kuhn, K.: Towards a continuous evolution and adaption of information systems in healthcare. International Journal of Medical Informatics 73, 75–89 (2004)
13. Object Management Group: Business Process Modeling Notation Specification (retrieved 2007-06-01), http://www.bpmn.org/Documents/OMG%20Final%20Adopted%20BPMN%201-0%20Spec%2006-02-01.pdf
14. Object Management Group: Meta Object Facility Core Specification version 2.0 (2006)
15. Prodato Integration Technology: Handbook i>PM Integrated Process Manager. Software documentation, Erlangen (2005)
16. Rinderle, S., Bobrik, R., Reichert, M., Bauer, T.: Business Process Visualization – Use Cases, Challenges, Solutions. In: Proceedings of the 8th International Conference on Enterprise Information Systems, pp. 204–211 (2006)
17. Schönhaage, B., Ballegooij, A., Eliens, A.: 3D Gadgets for Business Process Visualization. In: Proceedings of the fifth symposium on Virtual reality modeling language, pp. 131–138 (2000)
18. van der Aalst, W., Aldred, L., Dumas, M., ter Hofstede, A.H.M.: Design and Implementation of the YAWL System. In: Persson, A., Stirna, J. (eds.) CAiSE 2004. LNCS, vol. 3084, Springer, Heidelberg (2004)
19. van der Aalst, W., ter Hofstede, A.H.M., Kiepuszewski, B., Barros, A.: Workflow Patterns. Distributed and Parallel Databases 14(1), 5–51 (2003)

A Practical Experience in Designing Business Processes to Improve Collaboration

Andréa Magalhães Magdaleno[1], Claudia Cappelli[1], Fernanda Baiao[1,2], Flavia Santoro[1,2], and Renata Mendes de Araujo[1,2]

[1] NP2Tec – Research and Practice Group in Information Technology – UNIRIO
[2] Graduate Program in Informatics – Department of Applied Informatics – UNIRIO
Av. Pasteur, 458, Urca, Rio de Janeiro –RJ, Brazil, 22290-240
{andrea.magalhaes, claudia.cappelli, fernanda.baiao, flavia.santoro, renata.araujo}@uniriotec.br

Abstract. Organizations have been relying upon collaboration for knowledge sharing and productivity improvement in order to achieve cost reduction or revenue improvement. However, organizations still can not assure collaboration is properly conducted in daily work. This work presents an approach to stimulate collaboration between professionals involved in a real scenario of a petroleum company in Brazil. The project is an initiative towards improving decision-making during one of the business processes of the company, and establishing inter-professionals collaboration through information sharing. Our approach combines the use of a BPM methodology with the CollabMM collaboration maturity model and its corresponding method. The result was a to-be business process model with specific activities to plan, conduct and evaluate collaboration between professionals involved in the business process.

1 Introduction

Complex organizations very frequently rely on their professional personal skills and expertise in order to achieve their planned goals. In these organizations, business processes are highly based on tacit knowledge. For that reason, they increasingly face the challenge of making information easily accessible and shared.

Group work turned out to be an important business strategy, and is frequently used as an instrument to overcome these problems [1] [2] [3] [4]. However, in current business processes practices, collaboration is usually left implicit. Despite recognizing collaboration can bring advantages, many organizations still do not know how to encourage it [5] [6].

Organizations can take advantage of understanding and modeling their processes if they are able to explicitly embed into these processes collaboration aspects. Magdaleno et al. [7] argue that collaboration can be systematically enhanced in organizations by explicitly considering it during process modeling. The authors proposed a collaboration maturity model (CollabMM) and a corresponding method which organizes collaboration practices to be introduced in business processes.

We worked in real case from an oil organization, where a project have been implemented concerning issues such as: how to establish collaborative processes through the use of collaborative room technology, and how to make information about business processes available and shared among participants during discussion sessions. We applied the collaboration maturity model CollabMM and its method in order to design the organizational processes for collaboration, helping the organization to enhance the use of collaborative technology as well as to improve information sharing.

The goal of this paper is to describe the approach we have followed to combine the CollabMM method and a typical business process modeling methodology in this real case scenario. Section 2 describes the project context within the organization. Section 3 explains details of CollabMM method, presents the approach we have conducted and its set of artifacts, and analyzes its results. Section 4 concludes the paper and presents future work.

2 Real Case Scenario: The Northeast GEDIG Pilot Projects in PETROBRAS

PETROBRAS is the largest and most important oil company in Brazil. It is responsible for the majority of derivatives of petroleum explored and produced in this country. Since Brazil has a huge geographical area, PETROBRAS has a great number of production fields, both onshore and offshore. Production fields are grouped into Business Units. Professionals working in each field frequently travel long distances to interact with each other, in order to discuss and solve technical problems, change experiences and define best practices. Although there are collaboration tools available in the company, it does not fully benefit from sharing knowledge, experiences and solutions.

Benchmark actions conducted within PETROBRAS concluded that oil production could be enhanced through the use of technological artifacts, including physical information sensors (such as pressure and temperature), information systems and collaborative workplaces.

Therefore, the company launched a corporative program called GEDIG (Integrated Digital Management) aiming at improving oil recovery, optimizing oil production, and reducing company costs through the digital integrated management of oil and gas fields. Specific goals of GEDIG activities are: train people, apply information, automation, simulation and modeling technologies.

In order to reach its goals, the project addressed people, process and technology issues, and focused on applying real-time process monitoring, revising workflow processes and promoting integration among people through technology. For example, one of its technologies is the use of collaborative rooms where people could virtually meet to make decisions.

Business Units receive production information from the fields through Operational Control Room Centers. Management centers should also be equipped with control rooms (GEDIG rooms). People from both sites should communicate and share real time information, as well as use collaborative applications to support discussion and decision-making. Figure 1 shows the configuration of the GEDIG project.

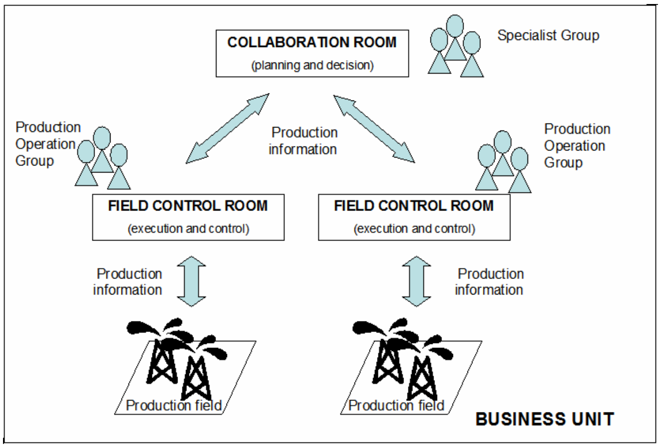
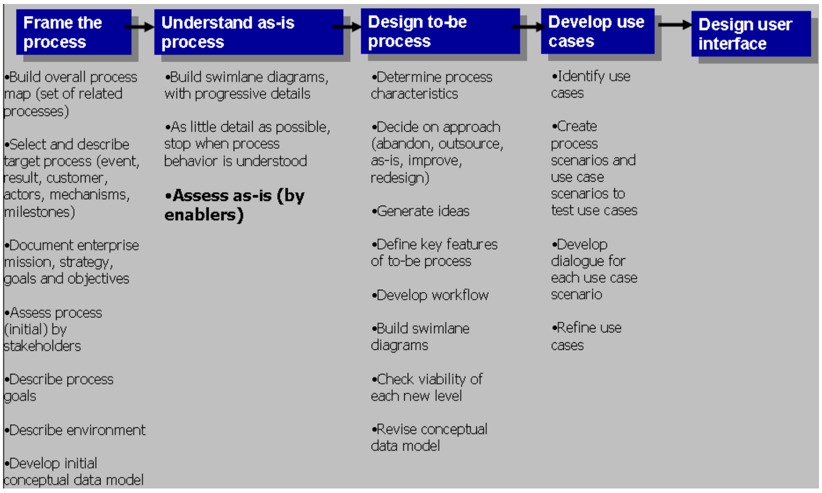


Fig. 1. GEDIG's Configuration Proposal

To encourage the use of the collaborative rooms, PETROBRAS conducted a business process modeling (BPM) strategy that followed the classical (AS-IS, TO-BE) methodology [8] in eight initial processes. In this methodology, current "as-is" processes are modeled and assessed looking for improvement opportunities (figure 2). The as-is assessment is performed considering six enablers for each process: workflow; information technology; motivation and measurement; policies and rules; human resources and facilities [8]. In the present work, we propose the creation of a seventh enabler, which is collaboration. The improvement opportunities found for the process are then incorporated to constitute the "to-be" process.



Frame the process →	Understand as-is process →	Design to-be process →	Develop use cases →	Design user interface
•Build overall process map (set of related processes)	•Build swimlane diagrams, with progressive details	•Determine process characteristics	•Identify use cases	
•Select and describe target process (event, result, customer, actors, mechanisms, milestones)	•As little detail as possible, stop when process behavior is understood	•Decide on approach (abandon, outsource, as-is, improve, redesign)	•Create process scenarios and use case scenarios to test use cases	
•Document enterprise mission, strategy, goals and objectives	**•Assess as-is (by enablers)**	•Generate ideas	•Develop dialogue for each use case scenario	
•Assess process (initial) by stakeholders		•Define key features of to-be process	•Refine use cases	
•Describe process goals		•Develop workflow		
•Describe environment		•Build swimlane diagrams		
•Develop initial conceptual data model		•Check viability of each new level		
		•Revise conceptual data model		

Fig. 2. BPM Methodology from [8]

In this paper, we illustrate how we have designed collaboration in one business process (figure 3). In this AS-IS process, participants deal with business information and must analyze and interpret them to be able to start actions to solve problems whenever they emerge. The participants are an Assistant, a Technician and a Manager. The Assistant works at the field and is responsible to collect and summarize information. The Technician is able to examine business indicators coming from measured data, and write a report about discrepancies. The Manager receives those reports and defines strategic actions to be done to correct problems.

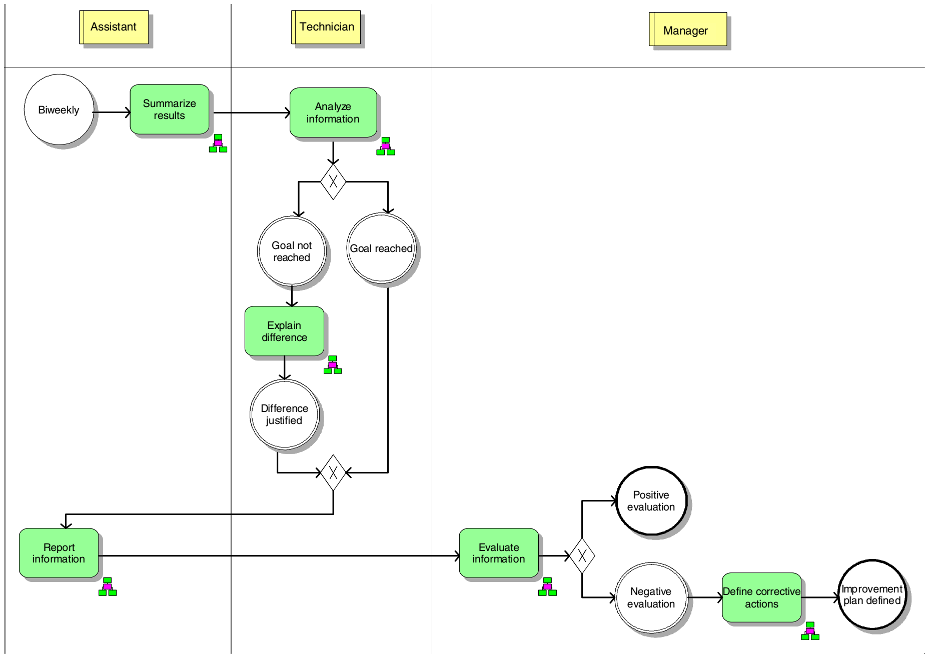


Fig. 3. The AS-IS business process

3 Designing Collaboration

In order to design collaboration in the process, we used our previously proposed CollabMM method [7] as a basis. This section reviews the main concepts of the CollabMM model and its corresponding method, details our approach and analyzes the results we have obtained.

3.1 A Maturity Model for Collaboration

In Magdaleno et al [7], the authors claim that organizations can benefit from thinking about, modeling and defining their business processes while explicitly taking collaboration aspects into account. The work proposes a collaboration maturity model for business processes – CollabMM, which incorporates a set of practices that enhance collaboration in business processes.

CollabMM presents an analogy to other maturity models [9] [10] [11] [12] [13] that exist for different domains. It was empirically defined based on well-known group supporting aspects: communication, coordination, awareness and memory [14][15][16][17]. CollabMM is a staged model that comprises four maturity levels for a process: Ad-hoc, Planned, Aware and Reflexive, as shown in figure 4. Each specific level comprises a group of related activities which can be executed together, aiming at improving the process collaborative capability.

Level 1 – Ad-Hoc: In this level, collaboration is not explicitly represented in business processes. Collaboration may happen, but it is still dependent on individual initiative and skills, and its success depends on the relationship and/or affinity among people.
Level 2 – Planned: Business processes in organizations start to be modified aiming at including basic collaboration activities. It includes planning for collaboration (formalizing groups, roles and responsibilities), and defining the appropriate communication channels among group members. Coordination is another strong aspect at this level.
Level 3 – Aware: In this level the process includes activities for monitoring and controlling how collaboration occurs.
Level 4 – Reflexive: In the reflexive level processes are designed to provide ways for self-understanding, identifying the relevance of the results that had been produced and sharing this knowledge inside the organization.

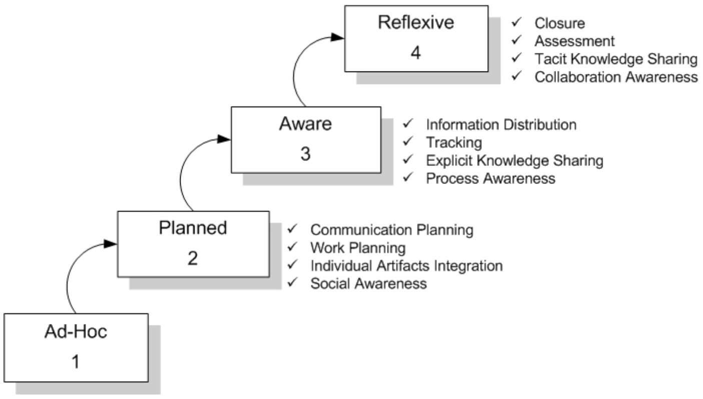

Reflexive
4
✓ Closure
✓ Assessment
✓ Tacit Knowledge Sharing
✓ Collaboration Awareness
Aware
3
✓ Information Distribution
✓ Tracking
✓ Explicit Knowledge Sharing
✓ Process Awareness
Planned
2
✓ Communication Planning
✓ Work Planning
✓ Individual Artifacts Integration
✓ Social Awareness
Ad-Hoc
1

Fig. 4. Overview of the CollabMM [7]

As a maturity model, CollabMM acts as a framework by organizing collaborative practices, without being committed to explain how to implement them. Therefore, in [7] the authors also presented a detailed method that helps the application of the CollabMM. The main objective of the proposed method is to explicit and design collaboration in business process modeling, according to CollabMM.

3.2 An Approach for Designing Collaboration in the PETROBRAS Business Process

Our approach for explicitly representing collaboration in one of the business processes of the northeast GEDIG Project combined the use of a BPM methodology [8] with the CollabMM model and its corresponding method [7]. The BPM methodology contributed with the definition of the main activities (model the AS-IS process, evaluate the AS-IS process, model the TO-BE process), while CollabMM helped in guiding the execution of these activities so as to design for collaboration.

After the AS-IS process model was built, we conducted three activities:

(a) Determine the process AS-IS collaboration maturity level
(b) Define the desired TO-BE collaboration maturity level
(c) Design TO-BE process according to (b).

Determine the process AS-IS collaboration maturity level

We proposed a set of questions (Figure 5) for evaluating which collaboration practices were implemented in the organization, and which ones are yet to be implemented. The questions should be answered by the process modeling team, thus acting as a guide for team members in assessing the collaboration enabler. The questions were derived from our analysis of the practices proposed in CollabMM [7].

Level 2 – Planned

- Q1. Is there a communication plan among process actors?
- Q2. Is each process actor aware of other actors involved in the process?
- Q3. Do process actors collaborate during artifacts integration for generating the final product of a group work?
- Q4. Is there a team work plan?

Level 3 – Aware

- Q5. Is necessary information adequately available for all process actors?
- Q6. Do process actors interact to discuss important issues in the process?
- Q7. Do process actors understand the process definition in which they are involved?
- Q8. Is there a repository accessible by all the actors involved in the process?
- Q9. Are the artifacts generated during process execution stored in the repository and shared among process actors?
- Q10. Are there mechanisms for keeping track of the work that is being done, according to what was planned?

Level 4 – Reflexive

- Q11. Do actors interact to analyze success and challenges, as well as share and discuss lessons learned and ideas for future improvements collected during process execution?
- Q12. Do process actors understand how people collaborate during the process execution?
- Q13. Is there a channel where the group can share informal knowledge - ideas, facts, questions, opinions, debates, discussions and decisions?
- Q14. Are there mechanisms to evaluate the contribution of each actor to the group results?

Fig. 5. The set of proposed questions for evaluating an organization collaboration maturity level according to CollabMM

The answers to these questions helped the organization in determining the current level of collaboration for the process (referred to as the "AS-IS collaboration level"). In the northeast GEDIG project we obtained the answers for these questions from the modeling team manager, who led the AS-IS process modeling and therefore knew the process in detail. Table 1 presents the answers obtained for all questions. According to the answers presented in Table 1, it is clear that there were some practices of each maturity level not yet implemented.

Table 1. Answers for the business process

Planned Level		Aware Level		Reflexive Level	
Question	**Answer**	**Question**	**Answer**	**Question**	**Answer**
Q1	No	Q5	No	Q11	No
Q2	Yes	Q6	No	Q12	No
Q3	Yes	Q7	Yes	Q13	Yes
Q4	No	Q8	Yes	Q14	Yes
		Q9	Yes		
		Q10	Yes		

Define the desired TO-BE collaboration level

The process manager defined the Reflexive Level (4) as the desired collaboration maturity level for the process, in order to provide ways for self-understanding and knowledge sharing inside the organization. Since CollabMM is a continuous approach for achieving higher collaboration maturity levels, to achieve the Reflexive Level the organization must address all the practices suggested by the levels bellow: Planned Level (3) and Aware Level (2).

Design TO-BE process according to the TO-BE collaboration maturity level

In order to implement the planned collaboration level (2), the method suggests the elaboration of the role-based model (Figure 6) representing the interactions between process roles. This model makes information flow among process actors explicit, thus increasing comprehension of process participants.

This model was based on the organization model, which describes the roles that are responsible for each activity execution. The interaction among process roles is represented by a graph where each node corresponds to a role on process swimlanes. Each arrow that links two activities corresponds to an edge between two graph nodes.

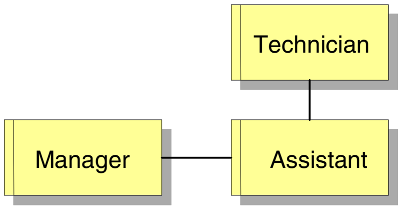


Fig. 6. The role based model of the business process

The analysis of this model evidenced the absence of some important actors that are involved in the process execution, such as the Supervisor, the Specialist and the Engineer. The Supervisor acts in the operational level tracking the daily problems and conducting a global analysis of the operational results obtained during process execution. When the Supervisor is not able to solve the problem, he sends specific problems to the Technician to solve. In some complex problems, the Technician can ask for help by the Specialist or Engineer located in other city.

Following the method, we added two activities in the process (elaboration of communication and coordination plans) (figure 7). These activities are the responsibility of the manager. This role was chosen because it has an intrinsically potential for coordination above other ones, since it occupies a higher position in the organizational structure.

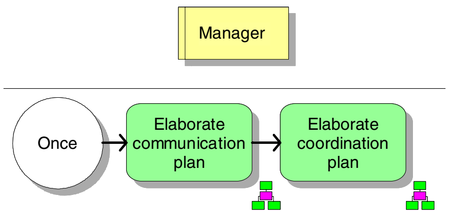


Fig. 7. Adding new activities in the business process

Moreover, the "Define corrective actions" activity was detailed in a sub-process, in order to explicitly define all the necessary steps for the collaborative construction of the "Action Plan" artifact (figure 8).

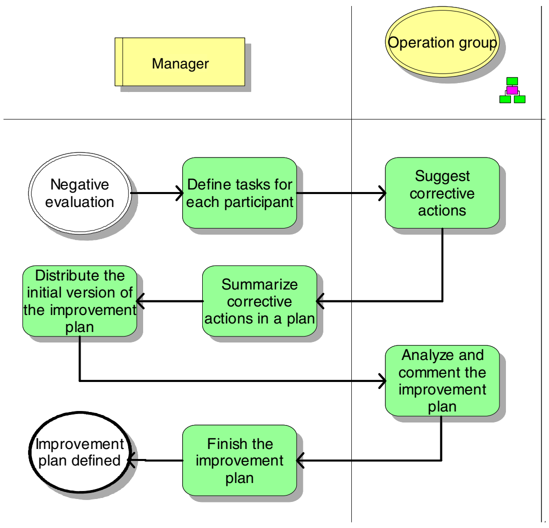

Manager
Operation group
Negative evaluation
Define tasks for each participant
Suggest corrective actions
Distribute the initial version of the improvement plan
Summarize corrective actions in a plan
Analyze and comment the improvement plan
Improvement plan defined
Finish the improvement plan

Fig. 8. Adding the sub-process Artifact integration in the "Define corrective actions" activity

Following the method rationale for the aware collaborative maturity level (3), we suggested a new activity “Track daily results” to track group work. This should be a quick meeting to discuss the results recently obtained and to plan corrective actions. The periodicity of this meeting (twice a week, as in figure 9) was agreed with the process users according to the organization daily routine.

At the end of this meeting, some problems had been solved collaboratively among participants. However, more complex and urgent problems may last. The solutions for these problems rely on the interaction among specialists of other areas, which are three hours away from the field. Our suggestion is that another meeting should follow for discussing remaining problems, in which specialists participate (figure 9). It is important to notice that all new meeting activities were proposed in order to make use of the collaborative rooms. Therefore, these meetings may be virtually conducted, using all the available collaboration resources and infrastructure available, such as videoconference technology and smartboards.

The first meeting will be executed by the operation group which is composed by the Manager, Assistant, Technician, supervisor (fig. 10). The second meeting is conducted by a group of specialists, which is composed of Managers, Technicians, Supervisors, Specialist and the Engineer, as shown in figure 10.

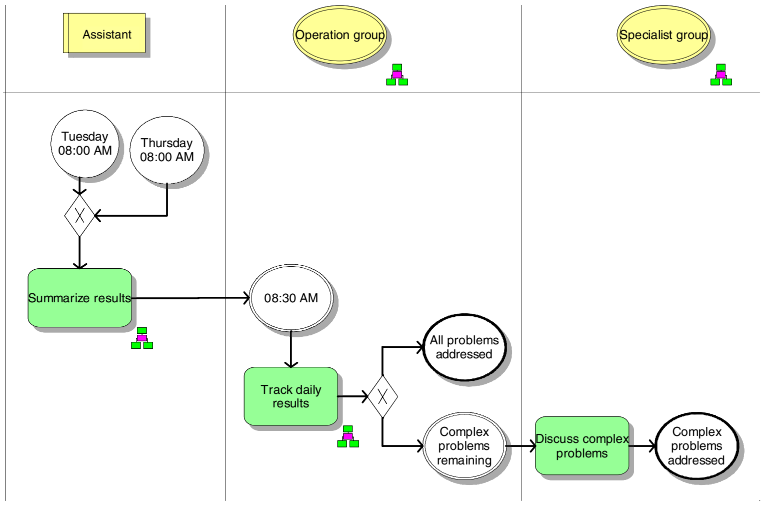


Fig. 9. Another meeting to discuss complex problems with specialists

Since process artifacts are already stored in a network drive, the network itself works as an established centralized repository, available for all process actors. Also, the process model is published in the organizational intranet, using the BPM tool.

Finally, to complete the reflexive collaborative maturity level (4), CollabMM method suggests that group members interact to analyze success and challenges, share and discuss lessons learned and new ideas. Considering that the business process already deals with analyzing and discussing problems during its execution, we understand this interaction is achieved using the same meetings already proposed for the process. Thus, no additional modification is needed.

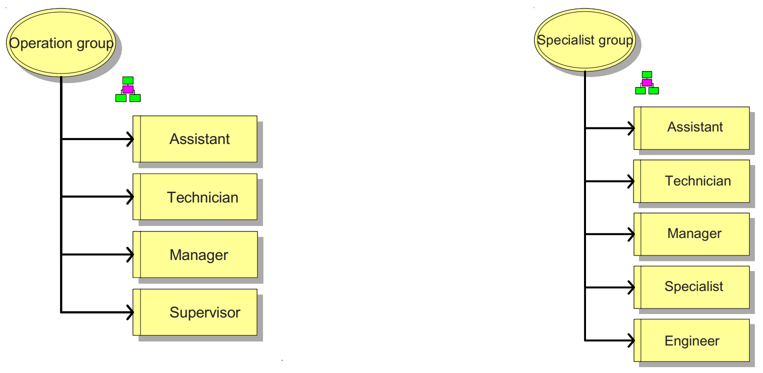


Fig. 10. Meeting group's composition

This concludes the method application, and resulted in a TO-BE Business process model (figure 10) that is CollabMM-level 4 compliant.

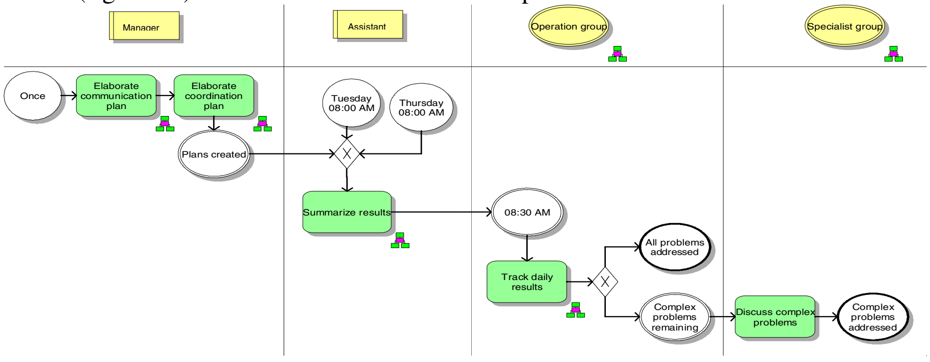


Fig. 11. The final business TO-BE Process

3.3 Evaluating the Proposed Approach for Designing Collaboration in the PETROBRAS Process

Our approach was composed of three simple, yet effective, phases. First, we established the organization collaborative maturity level by answering a set of proposed questions regarding the analyzed process. The set of questions guided the process modeler in evaluating whether a practice was already implemented or not in the organization, and precisely identified specific points of the process that should be modified in order to enhance collaboration between process actors. Second, the desired collaborative maturity level was determined taking organization needs and the potential benefits resulting from the collaboration level into account. Finally, we identified which activities should be included in the process so as to achieve the desired collaborative maturity level. It is important to say that those activities may be inserted in any (AS-IS, TO-BE)-based BPM methodology.

The result of our approach was a to-be business process model with specific activities to plan, apply and evaluate collaboration between professionals of PETROBRAS that are involved in the addressed business process.

The analysis of the resulted process shows that it explicitly defines groups of professionals that should collaborate for conducting some activities. We may also notice that collaboration is enforced between specialists that are geographically separated in order to discuss solutions for complex problems. This collaboration did not exist in the AS-IS process and it will certainly improve the quality of the decisions taken, as well as knowledge sharing among specialists.

Moreover, the CollabMM played an important role in our approach since it pointed specific collaboration issues that should be considered in the TO-BE process modeling. Also, the set of questions we have proposed certainly accelerated its modeling. Without either the method or the set of questions, the design of the TO-BE model would certainly be more complex, would take much longer, and would be much more difficult to be conducted in other process modeling projects.

4 Conclusion

This work presented an approach to design collaboration among professionals involved in a real scenario of the PETROBRAS petroleum company in Brazil. The GEDIG project is an initiative towards improving decision-making during some PETROBRAS processes, and establishing inter-professionals collaboration through information sharing. The GEDIG Project involved reorganizing information, implementing new requirements in software applications, acquiring new technology and improving work process by including collaborative activities in daily work.

Our approach combined the use of a BPM methodology [8] with the CollabMM collaboration maturity model and its corresponding method [7]. The BPM methodology contributed with the definition of the main activities; while CollabMM helped in identifying opportunities for collaboration and guiding the business process modeler in designing a more collaborative version of the business process.

Our approach guides the process modeler in designing business processes that enhance collaboration among actors involved in its execution. Without it, the process modeler is left with no orientation in which aspects are important to consider, which steps should follow in order to plan, conduct and evaluate collaboration in processes.

This work presented an experience in following this approach in a real scenario. The resulted TO-BE process included activities specifically for planning and evaluating collaboration between team members. The final process clearly reached the expected results, since without this collaboration the actions to correct problems that arise during operational activities miss various aspects and were based on one particular view or with partial information.

We highlight the importance of selecting the right process to be benefited from our approach, since the addition of collaboration activities may sometimes make the process costly and time consuming. For this reason, the selected process should be one in which collaboration plays a major role for improving results. The business process addressed in this paper fits this requirement.

As future work, we intend to investigate system requirements elicitation and process automation, by integrating our proposed approach with other approaches for identifying system requirements from business processes [18] [19] and for implementing workflows [20]. Future investigations will also address metrics for calculating process indicators and formally defining organization collaborative maturity levels.

References

1. Scholtes, P.R., Joiner, B.L., Streibel, B.J.: The Team Handbook. 2nd edn. Oriel. Madison (1996)
2. Telleria, K.M., Little, D., Macbryde, J.: Managing processes through teamwork. Business Process Management Journal 8, 338–350 (2002)
3. Staniforth, D.: Teamworking, or individual working in team? Team Performance Management 2, 37–41 (1996)
4. Khoshafian, S., Buckiewicz, M.: Introduction to Groupware, Workflow, and Workgroup Computing. John Wiley & Sons, New York (1995)
5. Borelli, G., Cable, J., Higgs, M.: What makes teams work better? Team Performance Management 1, 28–34 (1995)
6. Rugullies, E.: Team Collaboration Best Practices: Getting People to Share Their Knowledge, Forrester (2003)
7. Magdaleno, A.M., Araujo, R., Borges, M.: Designing Collaborative Processes. In: BPMDS. International Workshop on Business Process Modeling, Development and Support, Trondheim, Norway, pp. 283–290 (2007)
8. Sharp, A., Mcdermott, P.: Workflow Modeling: Tools for Process Improvement and Application Development. Artech House, Norwood (2000)
9. Paulk, M.C., Curtisl, B., Chrissis, M.B., Weber, C.: Capability Maturity Model for Software. Carnegie Mellon University, Software Engineering Institute. Pittsburgh (1993)
10. CMU-SEI: Capability Maturity Model Integration. Carnegie Mellon University, Software Engineering Institute. Pittsburgh (2006)
11. Ehms, K., Langen, M.: Holistic Development of Knowledge Management with KMMM. In: American Productivity & Quality Center (APQC) - Showcasing Successful Knowledge Management Implementation, San Antonio (2000)
12. Fisher, D.M.: The Business Process Maturity Model: A Practical Approach for Identifying Opportunities for Optimization. Business Process Trends (2004)
13. Rosemann, M., Bruin, T.: Application of a Holistic Model for Determining BPM Maturity. Business Process Trends (2005)
14. Araujo, R., Borges, M.R.S.: Extending the Software Process Culture – An Approach Based on Groupware and Workflow. In: Bomarius, F., Komi-Sirviö, S. (eds.) PROFES 2001. LNCS, vol. 2188, pp. 297–311. Springer, Heidelberg (2001)
15. Ellis, C., Gibbs, C.J., Rein, G.I.: Groupware: some issues and experiences. Communications of the ACM 34, 39–58 (1991)
16. Dias, M.S., Borges, M.R.S.: Development of groupware systems with the COPSE infrastructure. In: CRIWG. International Workshop on Groupware, Cancun, Mexico, vol. 1, pp. 278–285 (1999)
17. Santoro, F.M., Borges, M.R.S., Santos, N.: Learning to Plan the Collaborative Design Process. In: Shen, W.-m., Lin, Z., Barthès, J.-P.A., Li, T. (eds.) CSCWD 2004. LNCS, vol. 3168, pp. 33–44. Springer, Heidelberg (2005)

18. Araujo, R.M., MKnight, D., Borges, M.R.S.: A Systematic Approach for Identifying System Requirements from the Organization´s Business Model. In: Brazilian Workshop on Information Systems, vol. 1, pp. 1–10 (2005)
19. Miranda, I.S., Araujo, R.M., Borges, M.R.S.: Discovering Group Communication Requirements. In: IDEAS. Workshop Iberoamericano de Ingenieria de Requisitos y Ambientes de Software, Venezuela, pp. 107–120 (2007)
20. Magdaleno, A.M., Nunes, V.T., Araujo, R.M., Borges, M.R.S.: Increasing Flexibility in Process Deployment with the Process Beans Composer. In: CSCWD. International Conference on Computer Supported Cooperative Work in Design, vol. 2, pp. 1229–1234 (2006)

Modeling Requirements for Value Configuration Design

Eng Chew, Igor Hawryszkiewycz, and Michael Soanes

Department of Information Technology, University of Technology, Sydney, Australia
{Eng.Chew, Igor.Hawryszkiewycz, Michael.G.Soanes}@uts.edu.au

Abstract. Breadth and depth complexity are key challenges in achieving business process fusion as the enabler for value configuration design. The PARM framework is proposed as the requirement to address breadth and depth complexity through the independent but integrated operation of the process, activity, resource and management viewpoints. The operational scenarios for each viewpoint result in varying process modeling extension requirements. Existing process modeling constructs have varying support for these requirements. The PARM framework solution is an extension and integration of existing modeling constructs rather than a solution in its own right. Using the MDA approach of abstracting a platform independent model from a platform specific implementation, it is the goal in future papers to define process modeling extensions to support the PARM framework and map these into existing implementation architectures.

Keywords: business process design, value configuration design, process architecture.

1 Introduction

Porter[1] introduced the concept of the value chain as a series of activities that add value in contributing to the delivery of customer requirements. The value chain concept was later extended by Stabell and Fjeldstad[2] into value configuration, *defined as a network of value chains*. Value configuration denotes the fact that in practice, an enterprise commonly networks with several partners in servicing its customers. The value configuration models the enterprise-wide business process as a *network* of interdependent core processes. Designing individual core business processes in isolation, without the enterprise-wide view, can lead to a sub-optimal process design when aggregated into the total value network.

Dynamic reconfiguration of the value configuration is gaining momentum as a new competitive advantage. Gartner Group [3] has labeled this trend "business process fusion" and defines it as "the transformation of business activities that is achieved by integrating previously autonomous business processes to create a new scope of management capabilities." Gartner Group [4] says that through a new operating and management focus on enterprise wide processes and technology integration, business process fusion will enable an enterprise to increase its agility and improve efficiency.

There is a recent trend to advocate business process management (BPM) as a key enabler for business process fusion. This paper (as part of a series of papers) proposes value configuration design, as the next evolution of business process design, is the enabler for business process fusion.

Value configuration design requires a *process engineering methodology* which ensures the resultant value configuration will deliver the customer value (requirements) in line with the business strategy. Our research aims to develop the methodology, which will serve as a practical process innovation tool for process managers.

The first paper [5] in the series introduced the breadth / depth complexity matrix as the challenge in addressing value configuration design. The second paper [6] introduced the Process / Activity / Resource / Management (PARM) framework as the requirements framework to address breadth / depth complexity. This paper proposes a set of requirements for each viewpoint of the PARM framework as input to identifying the process modeling constructs extensions necessary to address these requirements. The next paper will propose extensions to existing modeling languages to support the PARM framework and map them to proposed architectural implementations. The final paper will propose a process engineering methodology that leverages the PARM framework solution to achieve the objectives of value configuration design.

2 Breadth / Depth Complexity as the Challenge

Soanes [5] introduced the concept of the breadth / depth complexity matrix to describe the inadequacy of individual business process designs.

Breadth complexity is defined as the range of activity types within a business process ranging from highly structured systemic to unstructured ad-hoc activities.

Depth complexity is defined as the abstraction levels of process logic within a business process ranging from very coarse process logic (e.g. work passing from one resource to the next) to very granular process logic (e.g. navigation between fields on a data capture screen).

The combination of breadth and depth complexity results in the following matrix:

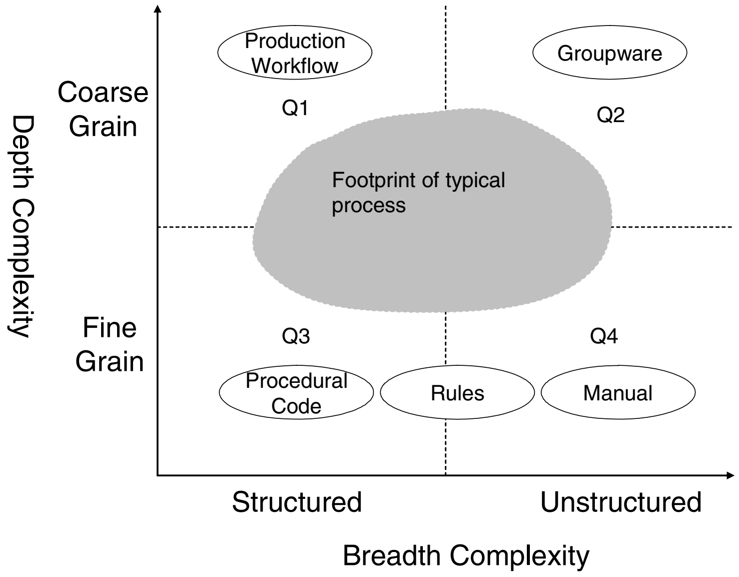


Fig. 1. Process Breadth / Depth Complexity Matrix

Soanes proposed that the footprint of typical processes crosses multiple breadth / depth quadrants of the above matrix. Soanes concluded that existing process design strategies and toolsets tend to specialise in one quadrant of the matrix. An example mapping of toolsets to quadrants is illustrated in Figure 1. Given individual processes can span multiple breadth/depth segments, this specialisation strategy can result in multiple process design strategies and toolsets being used within the one process. This fragmenting of business process logic across multiple toolsets results in a more complex task to maintain process logic as business requirements change – an obstacle to achieving business process fusion.

3 PARM Framework as the Requirement

Chew et al [6] introduced the PARM framework as the definition of the requirement for addressing breadth and depth complexity as per the following diagram:

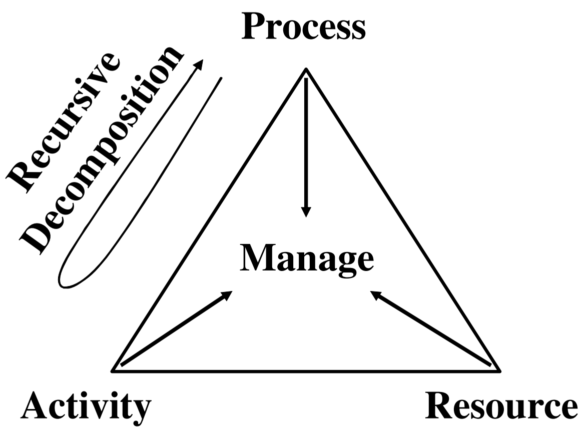


Fig. 2. Process / Activity / Resource / Management (PARM) framework

The Process / Activity / Resource / Management (PARM) framework defines four viewpoints of business processing that need to be integrated and managed as part of the design considerations (in response to *stakeholder requirements*) for each core business process:

- The Process viewpoint focuses on controlling, guiding and restricting the flow of activities performed for specific process instances. Its measurable objective is to meet the *customer's* end to end service delivery expectations.
- The Activity viewpoint focuses on the facilitation of an environment to manage human activity with the recognition that human resources will prioritise their own execution of multiple activities across multiple processes simultaneously based upon their own individual work practices. The execution sequence is not assumed to be deterministic – contrary to conventional process design which assumes that

activities will be executed as prescribed by the design. Its measurable objective is to provide the most effective (both productivity and quality) environment for the completion of all work across all processes – reflecting the *process / knowledge worker's* cognitive decision making behaviour which is unstructured.

- The Resource viewpoint forecasts, plans, schedules and assigns resources to activities. Its measurable objective is to maximize the utilization and therefore the efficiency of the total resource pool. This viewpoint captures the *resource planner's* requirements.
- The Management viewpoint integrates the process, activity and resource viewpoints through balancing the tension between service, cost and quality expectations. It reflects the requirements of the *business owner* of the process.

Breadth complexity requirement is modeled by the alignment and integration of the process, activity and resource viewpoints, with the "breadth" being accentuated by the activity viewpoint which explicitly models both structured and unstructured behaviours.

The recursive decomposition of the framework parameters (an activity at one level of abstraction can be decomposed as a process at the next lower level of abstraction), enables breadth complexity to be managed at multiple levels of depth complexity.

Process design completeness is achieved through a consistent and integrated approach to modeling and managing the process, activity and resource viewpoints.

Chew et al's key observation was that existing process design strategies and BPMS toolsets assume that the process viewpoint is the driver of activities and resources. The PARM framework recognises that in highly structured processes, the process viewpoint can control the activity and resource viewpoints. However, the nature of unstructured processes implies that activity initiation and resource allocation can be initiated and controlled independently from the overall process.

In the following sections, the requirements of each viewpoint will be expanded.

4 Process Viewpoint Requirements

This reflects the scenario where the process viewpoint is the controller and is directly instantiating activity commencement and resource assignment. This scenario is particularly relevant to automated system processing but also reflects the traditional production line human centric business processing.

This is the traditional BPM viewpoint and thus no attempt is made in this paper to define the total requirements of BPM. The following are specific requirements relating to addressing the breadth / depth complexity challenge:

- o Decomposition of the process design to multiple levels of abstraction is a critical enabler for depth complexity (a requirement that most BPMS's address). There is some dispute whether decomposition process design is a good practice. For example Ould [7] in describing Riva as a process design methodology, specifically discourages decomposition. It is the view of this paper that decomposition is essential.
- o Traditional process design involves the use of highly structured graphical modeling constructs such as sequencing, concurrency branching, conditional branching etc. An additional requirement to address breadth complexity is the ability to define a

set of activities with no sequencing control. By default this can be achieved with graphical constructs by a concurrency branch for each activity. However, by adding the capability for an activity to decompose at the next level to a set of declarative rules, provides a richer ability to define complex sequencing control. For the common scenario of no sequencing control, the set of rules in effect become an action item list of actions that must be completed to achieve the activity being decomposed.

- o The process viewpoint is the key communication viewpoint. To address different audiences, it is ideal to have the ability to produce graphical models at varying depth abstractions. Thus for senior management, the model will be filtered to a higher level whereas for process participants it may be at a detailed level. Ideally this depth filtering could also be applied to selective segments so that where appropriate some segments of the process could be shown in detail and others in summary.
- o Where the process viewpoint is not the controller (e.g. a knowledge worker is performing activities based upon their own preferences), the process design needs to define wait states where the process awaits an external event (e.g. completion of an activity that the process was not in control of). Most modeling constructs already support this requirement through the use of defining states and then as a physical implementation of that state, tie the state to awaiting an event. Role Activity Diagrams for example provide the capability to define a state as well as event triggers.

5 Activity Viewpoint Requirements

This represents the scenario where the activity viewpoint is the controller whereby an individual resource based upon their own individual work practices, multi tasks across multiple activities across multiple processes utilising or interacting with multiple other resources. This scenario is particularly relevant to knowledge worker environments and exception handling within production processes.

The modeling requirements of this scenario are:

- o The ability to define what cannot happen as constraints to the dynamic ordering of activities. These would be defined as declarative rules global to a process and all its sub processes.
- o The ability to link activity events (start or completion of an activity) back to a process to trigger process state transition. This is to cater for activities that are performed outside of the process viewpoint control.
- o Ideally, the ability to define suggested activity flow sequence to prompt resources to execute activities in the optimal order.

In addition to the above modeling requirements, the activity viewpoint would ideally be supported by an activity or work portal that is an integration of work sources (e.g. BPMS, email etc) and work tools (e.g. application systems, groupware etc) to facilitate the knowledge worker to multitask across multiple activities across multiple processes. The detailed requirements of a work portal are beyond the scope of this paper. However it is worth noting that portals is a very active research focus. Gartner [8] have defined six generations of portal evolution and call portals "the Swiss army knife of enterprise computing."

6 Resource Viewpoint Requirements

In this scenario, the resource viewpoint is the controller based upon a pool of outstanding and forecasted work (i.e. activities to be performed on specific process instances). A resource optimisation strategy (whether centralised or distributed) assigns resources to outstanding work.

The modeling requirements of this scenario are:

- To facilitate the greatest range of options for runtime optimisation, it is best to minimize unnecessary prescriptive activity flow definition allowing the optimisation algorithm to decide the best activity flow. Thus the declarative rule definition requirement described for the process viewpoint that defaults to a set of action items is encouraged to enable greater resource optimization flexibility.
- Resource requirements per activity per process type need to be defined to enable the resource optimizer to predict resource requirements.

The resource viewpoint is an area poorly supported in current BPMS. Most BPMS tools will support simulation as a means of identifying the optimisation of resource allocation. However as Reijers and van der Aalst [9] highlight, a simulation model typically focuses on a single process while the people involved distribute their time over multiple processes.

The benefit of applying resource management to BPM is subject to the accuracy of the defined resource requirements per activity per process type. This definition consists of two components (using labour hours as the example resource)

- The expected time each individual activity will take.
- The volume mix of the frequency of each activity within the total process based upon the percentage of cases that follow conditional routings within the process.

Both of these components need to be measured over some sample size and sample timeframe. It would be sensible to have the BPMS track these components as the basis for re-calculating these on a periodical snapshot basis for each process.

On initial analysis, one could conclude that the resource management dimension has no bearing on the breadth / depth complexity challenge. However, in a practical implementation of this strategy, the abstraction level of the activity tracked is an important factor in ensuring the accuracy of the time estimate per activity. At too high an abstraction level, the time estimate is too broad an average with a high standard deviation. Too low an abstraction level, results in an onerous exercise to define a very fine grained process flow and the resulting tracking sample size may be too small to provide a statistically accurate result.

Thus, the depth of activity decomposition is an important driver for accurate resource management and as a result has an influence on the depth complexity of the process design.

A critique could also be made that this viewpoint is oriented towards production process management and not relevant to the less structured processes where activities are more difficult to estimate. As a business management culture, it is proposed that the practice of setting a plan, measuring an actual and identifying the variance (as represented in the Plan Do Check Act (PDCA) management philosophy originally proposed by Deming) is applicable to the whole continuum of activity structure and predictability. For less predictable activities, the abstraction level the plan is pitched may be higher, the timeframe projected forward may be shorter and the plan may be unique for each process / activity instance. A classical example within the normally highly structured manufacturing environment is repairing a machine. The PDCA philosophy would advocate that a plan with target time is made for assessment of the fault and then having identified the fault, a plan and estimate is made for the repair. Thus resource management in this scenario needs to be more dynamic and emergent.

7 Management Viewpoint Requirements

A key driver of the PARM framework is the ability for each viewpoint to operate independently with integration both at the definition phase as well as based upon execution feedback. The Management viewpoint is the integrator of the other viewpoints through balancing the tension between service, cost and quality expectations.

The modeling requirements of this viewpoint are:

- A common meta-model that each viewpoint's requirements can be mapped to, facilitating the integration of the requirements before mapping to the desired implementation architecture.
- An event logging model that through process mining facilitates the execution feedback to each viewpoint.
- A measurement model that utilises the event logging information to evaluate operational performance in service, quality and cost as feedback to evolution of each viewpoint's design.

8 PARM Framework Implementation

The scope of this paper is to outline the PARM framework requirements. A detailed description of the PARM framework implementation is the scope of the next paper in this series. However it is appropriate to define an architectural context in which the requirements need to be implemented.

Model Driven Development (MDD) is a software engineering approach consisting of models and model technologies to raise the abstraction level at which software is created. Although there are many implementations of MDD (as per Hailpern et al. [10]), the most prevalent is the Object Management Group's (OMG's) implementation of MDD called Model Driven Architecture (MDA)[11].

MDA defines a viewpoint as abstracting to a selected set of parts, connectors and rules in order to focus on a particular concern. It defines three viewpoints:

- A computation independent viewpoint focuses on the context and requirements of the system hiding structure and processing of the system.
- A platform independent viewpoint focuses on the operation of the system while hiding the implementation details for a specific platform.
- A platform specific viewpoint focuses on the implementation of the platform independent viewpoint to a specific platform.

MDA advocates raising the level of application definition abstraction by producing models at each viewpoint where ideally the next level of abstraction is generated automatically from the higher level.

From a PARM framework implementation perspective, the goal is to provide extensions to existing techniques within each of the abstraction levels rather than mandating the use of specific techniques. Applying this to specific MDA abstraction levels: the goal is to provide modeling extensions to existing process modeling approaches as the platform independent viewpoint; that then map into existing implementation architectural styles as the platform specific viewpoint.

The next two sections will define the context within which the PARM framework requirements need to be mapped to process modeling approaches and implementation architectural styles.

9 Process Modeling Approaches

From a platform independent viewpoint, many techniques and standards have evolved for abstracting to a higher level the modeling of process logic.

There have been multiple schemes proposed for categorising modeling languages. For example Giaglis [12] proposes four perspectives: functional, behavioural, organisational, informational. Huff [13] identifies four different categories of process modeling language (PML) paradigms: Non-executable, State-based PMLs, Rule-based PMLs, Imperative PMLs.

For the purposes of defining how the PARM requirements impact process modeling approaches, this paper proposes a process modeling categorization that amalgamates characteristics of both categorisation schemes above.

Two fundamental logic expression paradigms can be leveraged to model process logic: procedural logic and declarative logic.

Procedural logic prescriptively defines a predetermined flow of activity control. Procedural logic techniques vary in their level of abstraction attained ranging from the low level of abstraction achieved by procedural code (using languages such as Java for example) ranging up to the high level of abstraction of unstructured techniques like use case scenarios. In the middle of this range is the multitude of graphical techniques and their associated notations.

Given the PARM framework's de-emphasis of fine grained prescriptive activity flow definition, the capabilities of existing mainstream graphical process modeling languages are adequate to meet the prescriptive process modeling requirements of the PARM framework.

Declarative logic defines the constraints (via the use of rules) within which the process can execute and the actual activity flow sequence is determined dynamically at execution of the rules. As per procedural logic, the level of abstraction of declarative logic techniques varies ranging from the low level abstraction of expressing rules directly in a rules language to the mid level technique of decision tables (that are then translated into a rules language) and the high level abstraction of structured English. The rules language can be either a proprietary vendor rules language or desirably based upon a standards based rule language such as OMG's Semantics of Business Rules and Vocabulary (SBVR).

As defined by Ross [14], declarative logic can be used to define all process logic. Although declarative logic is more powerful in its ability to express more complex unstructured process logic, it's more difficult to interpret and communicate the process flow. Consequently, procedural logic is a more desirable technique for the predictable activity flow of structured processes.

The breadth complexity challenge advocates that procedural logic techniques by themselves, lack the expressive power to handle the breadth of unstructured process logic. Consequently to achieve PARM framework breadth complexity, requires integrating procedural logic techniques with declarative logic techniques.

The PARM framework requirements described above translate into two uses of declarative rules integrated within a prescriptive modeling approach:

- As global constraints that are defined for a specific process and its entire sub processes providing the "stop what must not happen" requirement.
- As the basis for implementing the required set of action items representing the ability to define activities as an unstructured non deterministic sequence of actions.

10 Implementation Architectural Styles

From a platform specific viewpoint, many architectural styles have evolved as a means of implementing the operation of a system and specifically process logic.

A classification of architectural styles is proposed by Fielding [15]. Fielding bases his categorisation on the constraints inherent in the communication of components of the system. Fielding defines twenty-two styles on this basis with the recognition that there are further possible styles.

As zur Muehlen [16] documents in his standards landscape, there are a multitude of standards from multiple standards organizations targeted at varying objectives within the total implementation architecture domain. There is much debate over which abstraction level of standard is more important and whether one standard replaces another.

It is beyond the scope of this paper to define in detail the multitude of architectural styles and standards in the implementation domain.

However, given the role events are advocated to play in the PARM framework's requirement to integrate the process, activity, resource and management views, it is appropriate to propose the Event Based Architecture (EBA) style as a strong candidate for implementation of the proposed PARM framework approach. Seybold [17] defines EBA based upon business events occurring inside or outside an organization that is then notified to interested parties who then evaluate the event and optionally take action.

Gartner in a research report on the role of events in enterprise applications [18], state that event driven processes is the key underlying factor that will enable the revolutionary improvements in business processes and application systems as advocated by the business process fusion concept.

11 Conclusion

Conceptually we want to support the mapping of any process modeling approach to any implementation architectural style. The goal of the PARM framework is not to edict a mandatory choice of either. The goal is to provide extensions to modeling approaches to address breadth and depth complexity and illustrate the ability to map these extensions into multiple architectural implantation styles. However from a realistic starting point, a subset of both must be chosen.

In a real implementation of the PARM framework, the choice of modeling and implementation styles will be influenced by specific business domain requirements and the existing legacy implementation environment.

The PARM framework solution is an extension and integration of existing solutions rather than a solution in its own right.

The next step is to propose the extensions required to existing modeling approaches to support the PARM framework requirements and map these to implementation architectural styles.

A subset of modeling approaches and implementation architectural styles will be chosen as a realistic scope for illustrating the PARM framework solution.

References

1. Porter, M.E.: Competitive Advantage (1985)
2. Stabell, Fjeldstad: Configuring value for competitive advantage. On chains, shops and networks. Strategic Management Journal 19(5), 413–417 (1998)
3. Hayward, S.: Gartner Group. Business Process Fusion: Enabling the Real-Time Enterprise (October 16, 2003)
4. McDonald, M., Rowsell-Jones, A.: Gartner Group. Agility and Efficiency: Business Process Fusion (April 2004)
5. Soanes, M.G.: Process Design Strategies to Address Breadth and Depth Complexity. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102, Springer, Heidelberg (2006)
6. Chew, E., Hawryszkiewycz, I., Soanes, M.: Value Configuration Design – an evolution in adequate business process design. In: Krogstie, J., Opdahl, A., Sindre, G. (eds.) CAiSE 2007 and WES 2007. LNCS, vol. 4495, Springer, Heidelberg (2007)
7. Ould, M.: Business Process Management: A Rigorous Approach (2005)
8. Gootzit, D., Valdes, R., Phifer, G.: Gartner Group. RAS Core Research Note (December 13, 2006)
9. Reijers, H., Van Der Aalst, W.: The effectiveness of workflow management systems: Predictions and lessons learned. International Journal of Information Management (2005)
10. Hailpern, B., Tarr, P.: Model-driven development: The good, the bad, and the ugly. IBM Systems Journal 45(3), 451–462 (2006)
11. Object Management Group. MDA Guide V1.0.1, http://www.omg.org
12. Giaglis, G.M.: A Taxonomy of Business Process Modeling and Information Systems Modeling Techniques. The International Journal of Flexible Manufacturing Systems 13, 209–228 (2001)
13. Huff, K.E.: Software Process Modeling. In: Fuggetta, A., Wolf, A. (eds.) Trends in Software Process, pp. 1–24. John Wiley & Sons, Chichester (1996)
14. Ross, R.G.: Principles of the Business Rules Approach (2003)
15. Fielding, R.T.: Architectural Styles and the Design of Network-based Software Architectures. Doctoral dissertation, University of California, Irvine (2000)
16. zur Muehlen, M.: Standards Landscape, at http://www.workflow-research.de
17. Michelson, B.: Patricia Seybold Group. In: Event Driven Architecture Overview, 2nd edn. (February 2006)
18. Schulte, R.W.: Gartner Group. The Growing Role of Events in Enterprise Applications (July 9, 2003)

CBP Workshop

Introduction to the First Workshop on Collaborative Business Processes

(CBP 2007)

Preface

Recent years have seen the trend of business globalization, which urgently requires dynamic collaboration among organizations. The business processes of different organizations are integrated seamlessly to adapt to the continuously changing business conditions and to stay competitive in the global market. Though current business process technologies have achieved a certain level, there is still a large gap between the current supports and the requirements from real collaboration scenarios. Especially in a loosely coupled collaboration environment, many non-functional yet crucial aspects, such as privacy and security, reliability and flexibility, scalability and agility, process validation, QoS guarantees, etc., are with a great lack of sufficient supports. This gap in turn obstructs the further advancement and wider application of business process technologies. Therefore, more academic research, facilitating infrastructure, protocols and standards are being expected to shift current business process management for supporting collaborative business processes.

This workshop attracted 23 high-quality international submissions from 61 authors across the world including Oceania, Europe, America and Asia. Each paper was carefully reviewed by at least two experts from our International Program Committee. Based on the quality of the submissions and their relevance to the workshop themes, we accepted 11 papers to be included in the workshop proceedings. This indicates an acceptance rate as low as 47% for this year.

The accepted papers covered the main areas of collaborative business process management, namely, process modelling, QoS management, privacy and security, and implementation technologies. We are very grateful to all authors for writing, revising and presenting their papers. Due to the dynamic nature of business collaboration, the modelling for collaborative business processes has to capture the flexibility of business collaboration, while rigorously preserving the internal business logic. In these workshop proceedings, various research approaches were presented, covering traditional Petri nets, event-driven mechanisms, to concurrent algebraic term nets (CATN) and business process management notations (BPMN), to address the modelling issue. To tune the quality of collaborative business process management, research on transaction reliability and quality analysis through fault injection were discussed in this workshop. Privacy and security are particularly important in a collaborative business process environment, and consequently this workshop included papers on privacy-preserving collaborative business process management, and delegating revocations and authorizations to highlight the research efforts towards this issue. Finally, some modern implementation technologies, including a toolkit for service interaction patterns, and a phased deployment scheme, were also introduced in this workshop.

We appreciate the indispensable support of the Program Committee members and external reviewers, who provided excellent feedback and valuable directions.

We thank the BPM 2007 Organization Chair, Marlon Dumas, the Workshop Co-chairs, Arthur ter Hofstede and Boualem Benatallah, and the Proceedings Chair, Hye-young Helen Paik, for their support in the organization of the workshops and the proceedings. Our thanks also go to Queensland University of Technology, Australia, and Swinburne University of Technology, Australia, for providing facilities to organize this workshop.

Finally, we would like to express our sincere appreciation to the ARC Research Network in Enterprise Information Infrastructure (EII) for the generous sponsorship.

September 2007

Chengfei Liu
Qing Li
Yanchun Zhang
Marta Indulska
Xiaohui Zhao

Workshop Organization

Workshop Organizers

Chengfei Liu
Centre for Information and Technology Research
Faculty of Information and Communication Technologies
Swinburne University of Technology, Australia

Qing Li
Department of Computer Science
City University of Hong Kong, China

Yanchun Zhang
School of Computer Science & Mathematics
Victoria University, Australia

Marta Indulska
Business School
The University of Queensland, Australia

Xiaohui Zhao
Centre for Information and Technology Research
Faculty of Information and Communication Technologies
Swinburne University of Technology, Australia

Workshop Program Committee

Wasana Bandara (Queensland University of Technology, Australia)
Boualem Benatallah (University of New South Wales, Australia)
Mark Cameron (ICT CSIRO, Australia)
Fabio Casati (University of Trento, Italy)
Sandy Chong (Curtin University of Technology, Australia)
Maya Daneva (University of Twente, The Netherlands)
Asuman Dogac (Middle East Technical University, Turkey)
Schahram Dustdar (Vienna University of Technology, Austria)
Aditya K. Ghose (University of Wollongong, Australia)
Jaap Gordijn (Free University, The Netherlands)
Paul Grefen (Eindhoven University of Technology, The Netherlands)
Yanan Hao (Victoria University, Australia)
Igor Hawryszkiewycz (University of Technology, Sydney, Australia)
Willem-Jan van den Heuvel (Tilburg University, The Netherlands)
Patrick C. K. Hung (University of Ontario Institute of Technology, Canada)
Kwang-Hoon Kim (Kyonggi University, Korea)
Michael zur Muehlen (Stevens Institute of Technology, USA)

Andreas L. Opdahl (University of Bergen, Norway)
Helen Paik (University of New South Wales, Australia)
Zhiyong Peng (Wuhan Universty, China)
Manfred Reichert (University of Twente, The Netherlands)
Shazia Sadiq (University of Queensland, Australia)
Wasim Sadiq (SAP Research, Australia)
Michael Schrefl (University of Linz, Austria)
Markus Stumptner (University of South Australia, Australia)
Hongbing Wang (Southeast University, China)
Hua Wang (University of Southern Queensland, Australia)
Jiacun Wang (Monmouth University, USA)
Minhong Wang (University of Hong Kong, China)
Jian Yang (Macquarie University, Australia)
Yun Yang (Swinburne University of Technology, Australia)
J. Leon Zhao (University of Arizona, USA)
Xiaohui Zhao (Swinburne University of Technology, Australia)

External Reviewers

Christian Eichinger
S. Berger
Cheng Zeng
Xuhui Li
Lai Hokyin
Jan Recker
Seung Ryu

Collaborative e-Business Process Modelling: Transforming Private EPC to Public BPMN Business Process Models

Volker Hoyer, Eva Bucherer, and Florian Schnabel

SAP Research CEC St. Gallen, Switzerland
Institute for Media and Communication Management,
University of St. Gallen, Switzerland
{volker.hoyer,eva.bucherer,stephan.florian.schnabel}@sap.com

Abstract. Introducing process orientation to overcome the functional-oriented organizational structure was the main concern within enterprises during the last decade to improve process quality. The next wave of process-oriented enterprises deals with integrating private tasks and processes into cross-organizational business processes characterized by high automation effort and supported by a wide penetration of e-Business technologies. In this work, we propose a concept for transforming internal private processes to publicly visible processes in a semi-automatic way. This transformation will be done by hiding the modelling complexity from the users. Evaluated on the basis of Event-Driven Process Chains (EPC) for private process view and the Business Process Modelling Notation (BPMN) for public process view, this article identifies the challenges both on semantic and syntactic level regarding the integrated mapping between different modelling layers as well as modelling languages.

Keywords: Collaborative e-Business Process, Event-Driven Process Chain (EPC), Business Process Modelling Notation (BPMN), Business Process Modelling Layers.

1 Introduction

1.1 Motivation and Structure

In the early 1990ies companies focused on the Business Process Reengineering (BPR) approach [1]. Now a changed business environment characterized by borderless enterprises [2] and seamless processes as well as real-time businesses [3] leads to the next wave of process-oriented efficiency improvement. Adaptive Business Networks instead of linear wired value chains spawn new challenges in the context of modelling business processes both on internal (private) and public level. According to popular business opinion, Information Technology (IT) will thereby be transformed into a commodity meaning a common infrastructure like telephone or power grids [4]. On top of a Service-Oriented Architecture (SOA) [5] [6] Business Process Management (BPM) [7] [8] will act as an intermediary between IT infrastructure and strategy layer [9]. It links further to business partners processes to create inter-organizational collaboration processes. This will allow enterprises not only to react flexible, on demand and on time, on technical changes but on business environment changes as well.

As shown in previous research activities of the authors [10] the next wave of process orientation is characterized by new challenges regarding private and public e-Business processes. On the one hand **process transparency** across enterprise borders allows seamless information flows reducing coordination problems. Positive side effects are a flexibility to react to changes of the business environment and economization on financial as well as personnel resources. On the other hand concepts of **information hiding** are necessary to protect critical internal information, building the foundation for competitive advantages. Additionally, the increasing dynamics in the business environment requires an **highly automated transformation** between the private and public view on processes to handle a Continuous Improvement Process (CIP). In contrast to the BPR approach [1] the process improvement is achieved by a permanent adaptation and improvement in small steps and not by one radical procedure. In this article, we propose such a transformation concept between private and public business processes that considers the three above-mentioned business challenges and identifies the outcoming challenges both on syntactic and semantic level.

The reminder of the paper is organized as follows. After a brief introduction on electronic collaboration among enterprises in Chapter 2 including the presentation of the business process modelling layer concept (private, public and collaborative), we present in Chapter 3 a two-step transformation concept by means of the business-oriented Event-Driven Process Chains (EPC) on private level and the Business Process Modelling Notation (BPMN) on public level. Based on practical experiences taken in the EU-funded project GENESIS [11], Chapter 4 illustrates identified challenges transforming private and public business processes. An overview about related work in Chapter 5 and a short summary close this work.

2 Collaborative e-Business Processes

2.1 e-Business and Collaboration

During the last years, e-Business has become widely accepted in many different industry segments. By adopting systems that allow for business transactions to be conducted electronically rather than paper-based, enterprises can significantly reduce the effort for data-processing, increase business data accuracy and may even discover new business models or partners [12]. However, due to huge technical complexity and missing globally accepted e-Business standards [13], electronic collaboration is currently limited to portal technology on presentation level. Thereby users are confronted with diverse user interfaces and working processes provided by the business partners (so called 1:n relation). New standardization approaches like ebXML or United Nations Centre for Trade Facilitation and Electronic Business (UN/CEFACT) Core Component Technical Specification (CCTS) [14] try to integrate business processes and semantic aspects on the application level [15] enabling real networkability and n:m relation. As a result standardized business processes lead to next generation e-Business frameworks [16] built on SOA and BPM concepts.

In the frame of the EU-funded project GENESIS [11], a consortium of several partners from across Europe proposes such a holistic framework for performing seamless Business-to-Business (B2B) and Business-to-Government (B2G) processes focused on the business environment in Eastern Europe. Typical business processes considered in GENESIS are transactional processes like order, invoice, VAT declaration or bank transfer. The main goal is the research, development and pilot application of the needed methodologies, infrastructure and software components creating a living evolutionary e-Business platform.

2.2 Business Process Modelling Layers

Inter-organizational business processes are performed by multiple independent parties. Since organizational borders usually represent boundaries for system interactions and information flows, a number of process particularities arise in comparison to company-internal (private) business processes. To achieve seamless business processes across enterprise borders the heterogeneity of different terminologies and modelling notations used within the organizations have to be overcome. However, autonomy of the different business partners has to be taken into account meaning that an organization should be able to flexibly participate in business relations. Important contributions to handle these challenges with regard to inter-organizational business processes come from workflow management, e.g. the Public-To-Private Approach [17] and the Process-View-Model [18]. As depicted in Figure 1 these approaches distinguish between the internal process (private process) and the cross-organizational interaction (collaborative process).

On **private process level**, organizations model their internal business processes according to a modelling approach or notation that is most suitable for internal demands independently of the modelling methodologies used by the business partners. In the example shown in Figure 1, enterprise A uses the

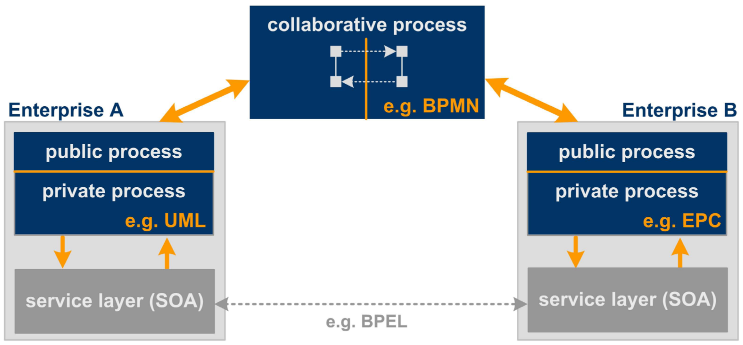


Fig. 1. Business Process Modelling Layers

Unified Modelling Language (UML) for modelling internal processes whereas enterprise B models with Event-Driven Process Chains (EPC). A comparison addressing the heterogeneity of business process models can be found in [19]. As a response, abstraction concepts hide details of the internal business process from external business partners in the **public process view**. According to the SOA paradigm [5] public process views are comparable to web service descriptions in the Web Service Definition Language (WSDL) and can be interpreted as a mediator and interface. By hiding the internal process implementation and protecting also critical internal information [20] public process views provided by an organization connect private processes to a **collaborative business process**. This level defines the interactions of two or more business entities taking place between the defined public processes. One possible language for modelling collaborative processes could be the Business Process Modeling Notation (BPMN) which consolidates ideas from divergent notations into a single standard notation. Examples of notations or methodologies that were reviewed are: UML Activity Diagram, UML EDOC Business Processes, ebXML Business Process Specification Scheme (BPSS), Activity-Descision Flow (ADF) Diagram, RosettaNet, and EPC [21].

Underlying the three presented business process modelling layer a **service layer (SOA)** represents the technical implementation of the business processes within the enterprises. Thereby, we adhere to the OASIS Reference Model for SOA within this work [6]. It defines SOA as "... a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. It provides a uniform means to offer, discover, interact with and use capabilities to produce desired effects consistent with measurable preconditions and expectations." [6]. Especially the widespread SOA implementation Web Services [5] and the respective orchestration language Business Process Execution Language (BPEL) describing the process flow by loose coupling of services play a major role on the executable technical SOA-layer. The mapping of private business processes to BPEL is regarded by several research activities, i.e. EPC/ BPEL [22], BPMN/ BPEL [21], or UML/ BPEL [23].

In the following we abstract from the service layer and focus on the private and public business process layers.

2.3 Business Process Modelling Languages

A business process modelling language is an artificial language used to map existing processes or to design new ones. Each modeling language is defined by a set of rules which consists of a syntax (the notation) and a semantic (the meaning) [24].

Both textual and graphical modelling languages exist. In the domain of BPM graphical models are often preferred, as they allow for an overview of the whole process and the presentation of timely and factual interrelations. The decision to employ a specific modelling language should be made in regard to the modelling purpose. Purposes range from pure documentation or auditing to the automation of processes. The construction of a model can reduce the complexity of an actual situation as a model usually just represents the relevant aspects. This fact is applied in the idea of hiding private information on a public process level as proposed in this paper.

For an exemplary model transformation we employed extended Event-Driven Process Chains (eEPC) on the private process level. This decision was made with respect to the wide industry dissemination especially in German-speaking regions. The popularity of EPCs is due to their usage within the Architecture for Integrated Information Systems (ARIS) concept [25]. EPCs were developed on the basis of Petri nets during the early 90ies and are especially focused on business and processes. Temporal and logical sequences are described by the use of functions (active elements), events (passive elements), and logical connectors such as "‘AND"’, "‘OR"’, "‘XOR"’ to define the control flow. These objects are linked via control flow arcs. By adding additional modelling elements the extended version of EPCs (eEPC) allows in addition for modelling organization and data views.

On the public process level we propose the implementation of the Business Process Modeling Notation (BPMN). BPMN is now a standard administered by the Object Management Group (OMG). It was developed by the Business Process Management Initiative (BPMI) in 2003 as an approach to ease graphical process modelling. It incorporates ideas and experiences of prior standards such as UML activity diagrams and ebXML Business Process Specification Scheme (BPSS). A small number of core elements permit a coherent process modelling. The usage of BPMN for public models is justifiable because of several reasons. According to [7] the basic requirement for collaboration is an open process modelling standard. This aspect disqualifies proprietary standards such as EPC. Apart from being "open" the implemented standard has to be relatively strong so that it is adopted by the industrial community. Applying the workflow pattern of [26] confirms the mightiness of the notation. Also BPMN is a relatively new standard the integration into commercial modelling tools is proceeding quite fast. Furthermore, additional mapping guidelines of BPMN elements to BPEL ease the implementation of BPMN modelled business processes [21]. The use of so-called black and white boxes allows modelling public and private process views.

Especially the last point is one reason to favor BPMN over UML on the public level. UML does not explicitly assist the discrimination of different modelling layers. Furthermore, BPMN is easier to understand for people with a non-technical background.

3 Transformation Concept

After building the foundation of this research article the following chapter presents a two-step concept transforming private EPC business processes to public BPMN business processes.

3.1 Overview

As proven in [10] the interrelation between private and public processes is characterized by a high complexity regarding the before-mentioned three business challenges process transparency, information hiding and automated transformation. Additionally, the heterogeneity of the used business process modelling notations requires a concept reducing the complexity for the person modeling processes within the enterprise. Experiences taken in the GENESIS project [11] have demonstrated that especially the employees of Small and Medium-sized Enterprises (SME) characterized by a missing modelling know-how are overcharged to map business processes simultaneously between different modelling views as well as between different modelling notations.

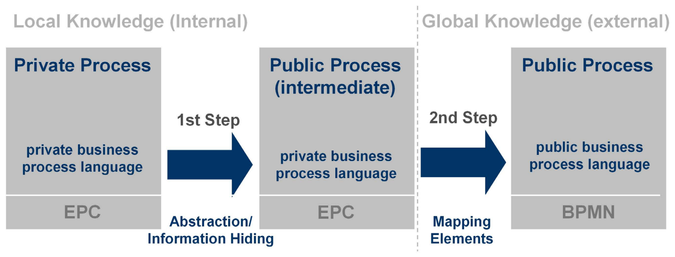

Local Knowledge (Internal)
Global Knowledge (external)
Private Process
private business process language
EPC
1st Step
Abstraction/ Information Hiding
Public Process (intermediate)
private business process language
EPC
2nd Step
Mapping Elements
Public Process
public business process language
BPMN

Fig. 2. Two-Step Transformation Concept

To handle these challenges we propose a two-step transformation concept as depicted in Figure 2. The first step abstracts from the private process flows to an intermediate public process layer by using the internal modelling notation. Both process transparency on the one side and information hiding on the other side have to be taken into account in this transformation step. Based on the intermediate layer using the private modelling language (i.e. EPC) the second step maps the abstract public process to the public modelling notation (BPMN). Challenges in this transformation step exist in the differences between the numerous modelling notations using diverse modelling elements, process flow or abstraction representations. An excellent comparison of the two relevant modelling notations in this work, EPC and BPMN, can be found in [27].

3.2 1st Transformation Step: Abstraction

As mentioned above, on the private business process layer EPCs are used due to the wide dissemination within the industry. Thereby it can be seen as a placeholder for other adequate modelling notations. By taking into consideration the EPC modelling guidelines [28] the EPC models on private business process layer have to follow the specified EPC rules to apply the EPC abstraction rules of the first transformation step. Figure 3 shows these six abstraction rules which are explained in the following:

- **EPC Abstraction Rule 1.** All trivial events have to be eliminated on the public process layer. These passive elements connect the active elements, the functions, and serve as intermediate state allowing a simple understanding for business people. In terms of collaborative e-Business processes these events become no longer necessary.
- **EPC Abstraction Rule 2.** Initiated events located in the process flow before logical connectors such as "‘AND"’, "‘OR"’, or "‘XOR"’ have to be deleted. The process flow including the linked functions is self-explained and sufficient on the public process layer.
- **EPC Abstraction Rule 3.** Every function of a private process is executed by an assigned organization unit of the enterprise. In consideration of hiding from this internal competitive relevant information, organization units are not visible on public process level.
- **EPC Abstraction Rule 4.** As mentioned in Chapter 2 collaborative e-Business processes focus on the information exchange. Only functions sending or receiving a message (data object) are part of the public business process layer. All other functions have to be dropped.

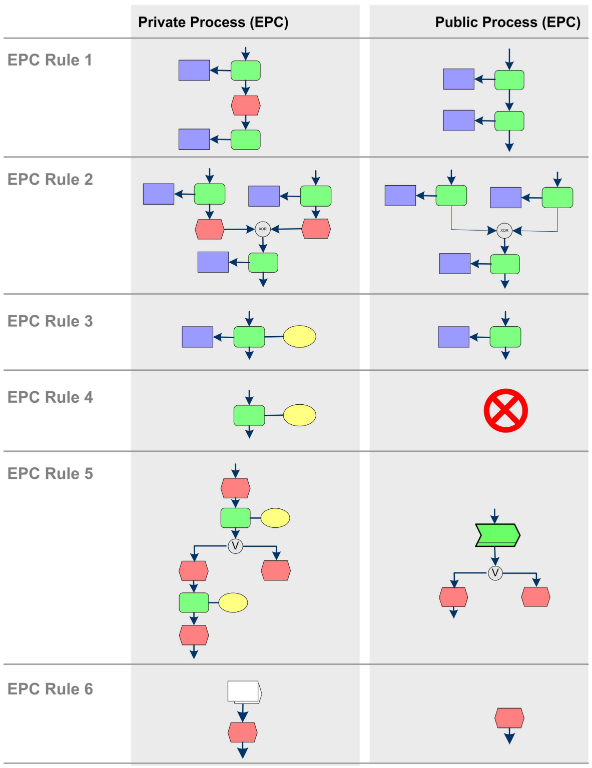


Fig. 3. EPC Abstraction Rules (1st Transformation Step)

– **EPC Abstraction Rule 5.** Sensitive critical process steps not interacting directly with external business partners have to be abstracted by means of process modules. Public relevant events have to follow the process modules to describe the public process flow.
– **EPC Abstraction Rule 6.** Process interfaces can be used to point from a function to a refining sub-process. This hierarchical representation does not exist on the public process layer.

3.3 2nd Transformation Step: Mapping to the Public Business Process Notation

On basis of the intermediate layer abstracting from the private business process and using the internal process modelling notation (in case of this work EPC) the second step maps to the public business process notation. Reasons for the approprioateness of BPMN were evaluated above.

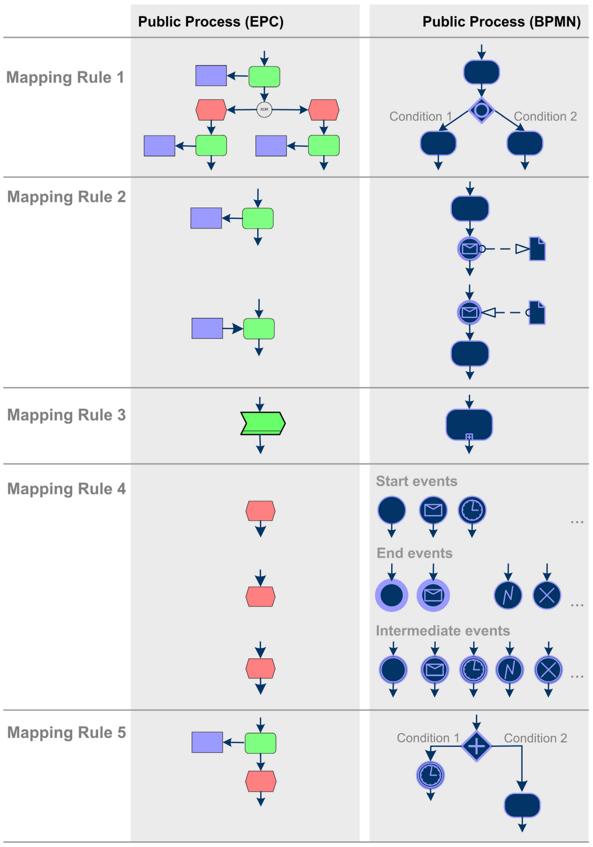


Fig. 4. Mapping Rules (2nd Transformation Step)

– **Mapping Rule 1**. According to the semantic meaning of the EPC events the connectors in BPMN have to be added with constraints to represent the conditioned process flow.
– **Mapping Rule 2.** In addition to the sending or receiving EPC functions the BPMN message element has to follow the functions to play by the modelling rules of BPMN.
– **Mapping Rule 3.** EPC process modules which abstract from internal process flows correspond to BPMN sub-functions. Therefore they can be directly mapped and hidden from internal process steps like a black box.
– **Mapping Rule 4.** According to their semantic meaning EPC events have to be transfered to BPMN events. In contrast to the EPC notation with only one event type there exist three basic event types in BPMN (start, intermediate and end events) with several sub types like message, timer, rule, link, multiple, error, cancel, or compensation.
– **Mapping Rule 5.** Semantic related functions and events on private level have to be transfered to an adequate BPMN representation.

4 Challenges

The combination of the two transformation steps leads from private EPC to public BPMN business process models. The envisioned bi-directional process interfaces can not be achieved completely due to the different representation types and granularity of the two modelling languages. But these **syntactic** challenges (i.e. mapping rule 1) play only a tangential role due to the fact that the introduced intermediate layer minimizes such syntactic differences.

The actual large challenges exist on **semantic** level of the second transformation step. As already indicated event mapping between the two modelling notations is limited to uni-directional relation (mapping rule 4) [29] and requires additional user interactions. Another challenge is related to a slight variance of semantic representation of a process flow as can be seen by mapping rule 5. Experiences taken in the GENESIS project have demonstrated how unavoidable user interactions lead to a wide variance of process interpretation. Without standardized mapping guidelines to generate a common understanding public processes will not be identically interpreted by the users and will not allow a flexible collaboration. Finally the users decide about success or failure of a concept [10]. That is why these identified semantic challenges have to solve with only a low number of "'semantic mapping rules"'.

5 Related Work

In literature modelling collaborative e-Business processes is discussed from several perspectives. Existing approaches are mostly limited to either private, public or collaborative processes like [18], [30], or [17].

A methodology dealing with collaborative processes is the UN/CEFACT Modelling Methodology (UMM) [31]. Based on Unified Modelling Language (UML) and Rational Unified Process (RUP) UMM is a methodology similar to a software process and supports components to capture business process knowledge. According to the Open-EDI reference model [32] the UMM specifies collaborative business processes involving information exchange in a technology-neutral, implementation-independent manner. It combines an integrated process (UML) and data (Core Component Technical Specification, CCTS) modelling approach [14] as well as modelling business collaboration in context [33].

An approach focusing on modelling inter-organizational processes is described in [34]. According to [35] the coordination between the different organizations requires an agreement on how to interact and exchange information. Therefore a public process model is built detailing the process interfaces between the organizations. In a further step private processes are aligned to the public process. [34] assume an existing public process the private processes could be aligned to. In contrast the approach of this paper will derive a public process out of private processes.

Another approach on inter-organizational interoperability is described by [29]. It states that the partners' private processes modelled in different notations have to be integrated into one collaborative process model. This is done by a model transformation via an intermediate representation. [36] define an intermediate representation as a model which is exported into a standard form. In [29] the model transformation via an intermediate representation is revealed by an example of a horizontal model transformation. A horizontal transformation means the transformation on the same abstraction layer. As realization of a horizontal transformation a XML-based mapping of an eEPC model to a BPMN model is regarded. Based on the graphically modelled private processes a public view is created and exported into a XML-based intermediate representation. Such an intermediate representation is the EPC Markup Language (EPML) for EPCs and the BPMN Markup Language (BNML) for BPMN. In [29] the goal notation for the collaborative process is BPMN. Therefore the models in EPML notation are transformed into the BNML using a XSLT-script. The resulting process in BNML notation can then be transformed into a BPMN model. The last step of the transformation is to manually combine the public processes to a collaborative process.

6 Conclusion and Further Work

In the frame of this article, a transformation concept is presented introducing an intermediate layer to map private business processes to public business processes by means of the two modelling languages EPC and BPMN. Taking into account the two challenges process transparency on the one side and information hiding on the other side the concept reduces the complexity users are facing with regard to modelling collaborative e-Business processes across enterprises. First experiences taken in the GENESIS project have proven the applicability on conceptual level.

Further work will deal with the implementation of a script transforming the private and public business processes in a semi-automatic manner eliminating manual user activities that leads to work more on strategic issues and thus increase the value-added productivity [10]. Also, the investigation about how much transformation work could be freed from the user will be part of further research activities to clarify the saving potential in terms of manual activities as well as time aspects.

Acknowledgments. This paper has been created closely to research activities during the EU-funded project GENESIS (Contract Number FP6-027867) [11].

References

1. Hammer, M., Champy, J.: Reengineering the corporation. Brealey, London (1993)
2. Picot, A., Reichwald, R., Wigand, R.T.: Information Organization and Management: Expanding Markets and Corporate Boundaries. John Wiley and Sons, New York (1999)
3. Alt, R., Oesterle, H.: Real-Time Business. Springer, Berlin (2004)
4. Carr, N.G.: IT doesn't matter. Harvard Business Review 81(5), 41–49 (2003)
5. Alonso, G., Casati, F., Kuno, H., Machiraju, V.: Web Services Concepts, Architectures and Applications. Springer, Berlin (2004)
6. McKenzie, C.M., Laskey, K., McCabe, F., Brown, P.F., Metz, R.: Reference model for service oriented architecture 1.0 (2006), http://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf
7. Smith, H., Fingar, P.: Business process management: the third wave. Meghan-Kiffer Press, Tampa, Fla. (2003)
8. Harrington, H.: Business Process Improvement: The Breakthrough Strategy for Total Quality, Productivity, and Competitiveness. McGraw-Hill, New York (1991)
9. Oesterle, H., Back, A., Winter, R.: Business Engineering. Springer, Berlin (2004)
10. Hoyer, V., Christ, O.: Collaborative e-business process modelling: A holistic analysis framework focused on small and medium-sized enterprises. In: Abramowicz, W. (ed.) BIS 2007. LNCS, vol. 4439, pp. 41–53. Springer, Heidelberg (2007)
11. GENESIS: EU project GENESIS (FP6-027867) (2007), http://www.genesis-ist.eu
12. United Nations Economic Commission for Europe: A roadmap towards paperless trade (2006), http://www.unece.org/cefact
13. Zhao, K., Xia, M., Shaw, M.J.: Vertical e-business standards and standards developing organizations. Electronic Markets 15(4), 289–300 (2005)
14. UN/CEFACT: Core component technical specification v2.01 (2003), http://www.untmg.org
15. Janner, T., Schmidt, A., Schroth, C., Stuhec, G.: From EDI to UN/CEFACT: An evolutionary path towards a next generation e-business framework. In: Proceedings of the 5th International Conference on e-Business (NCEB2006) (2006)
16. Hoyer, V., Janner, T., Mayer, P., Raus, M., Schroth, C.: Small and medium enterprise's benefits of next generation e-business platforms. The Business Review, Cambridge (BRC) 6(1), 285–291 (2006)
17. van der Aalst, W.M.P., Weske, M.: The P2P approach to interorganizational workflows. In: Dittrich, K.R., Geppert, A., Norrie, M.C. (eds.) CAiSE 2001. LNCS, vol. 2068, pp. 140–156. Springer, Heidelberg (2001)

18. Shen, M., Liu, D.R.: Coordinating interorganizational workflows based on process-views. In: Mayr, H.C., Lazanský, J., Quirchmayr, G., Vogel, P. (eds.) DEXA 2001. LNCS, vol. 2113, pp. 274–283. Springer, Heidelberg (2001)
19. Mendling, J., Neumann, G., Nuettgens, M.: A comparison of XML interchange format for business process modelling. In: Proceedings of EMISA 2004 - Information Systems in E-Business and E-Government (2004)
20. Parnas, D.L.: On the criteria to be used in decomposing systems into modules. Communications of the ACM 15(12), 1053–1058 (1972)
21. Object Management Group: Business process modeling notation specification, OMG final adopted specification (2006), http://www.bpmn.org
22. Mendling, J., Ziemann, J.: Transformation of BPEL processes to EPCs. In: Proceedings of the 4th GI Workshop (2005)
23. Hofreiter, B., Huemer, C.: Transforming UMM business collaboration models to BPEL. In: Meersman, R., Tari, Z., Corsaro, A. (eds.) OTM-WS 2004. LNCS, vol. 3292, pp. 507–519. Springer, Heidelberg (2004)
24. Harel, D., Rumpe, B.: Modeling languages: Syntax, semantics and all that stuff. Technical paper number mcs00-16, the weizmann institute of science (2000)
25. Scheer, A.W.: ARIS, Business Process Frameworks. Springer, Berlin (1999)
26. van der Aalst, W.M.P., ter Hofstede, A.H.M., Kiepuszewski, B., Barros, A.P.: Workflow pattern. Distributed and Parallel Databases 14(3), 5–51 (2003)
27. Vanderhaegen, D., Zang, S., Scheer, A.W.: Inter-organizational process management through model transformation. Working paper 182, university of saarbruecken, institute for information systems (2005), http://www.iwi.uni-sb.de/download/iwihefte/iwiheft_178.pdf
28. IDS-Scheer: ARIS Method Version 7.0. IDS-Scheer (2005)
29. Vanderhaeghen, D., Zang, S., Hofer, A., Adam, O.: XML-based transformation of business process models - enabler for collaborative business process management. In: Proceedings of 11th Conference
30. Medjahed, B., Benatallah, B., Bouguettaya, A., Ngu, A., Elmagarmid, A.: Business-to-business interactions issues. The VLDB Journal 12, 59–85 (2003) (Digital Object Identifier (DOI) 10.1007/s00778-003-0087-z)
31. UN/CEFACT: UN/CEFACT Modeling methodology (UMM) (2006), http://www.unece.org/cefact/umm
32. ISO/IEC: Information technology - Open-EDI reference model, ISO/IEC standard 14662:2004(e) (2004), http://www.iso.org
33. Hofreiter, B., Huemer, C.: Modeling business collaborations in context. In: Meersman, R., Tari, Z. (eds.) OTM-WS 2003. LNCS, vol. 2889, pp. 829–844. Springer, Heidelberg (2003)
34. Legner, C., Vogel, T., Loehe, J., Mayerl, C.: Transforming inter-organizational business processes into service-oriented architectures - method and application in the automotive industry. In: VDE Verlag (2007)
35. ATHENA: Cross-organizational business process requirements and the state-of-the-art in research, technology and standards (2005)
36. Sendall, S., Kozaczynski, W.: Model transformation: the heart and soul of model-driven software development. Software (IEEE) 20(5), 42–45 (2003)

Transforming XPDL to Petri Nets

Haiping Zha[1,3,4], Yun Yang[2], Jianmin Wang[1,3,4], and Lijie Wen[1,3,4]

[1] School of Software, Tsinghua University, Beijing, P.R. China, 100084
{chp04,wenlj00}@mails.tsinghua.edu.cn, jimwang@tsinghua.edu.cn
[2] Faculty of Information and Communication Technologies
Swinburne University of Technology, Melbourne, Australia, 3122
yyang@it.swin.edu.au
[3] Key Laboratory for Information System Security, Ministry of Education,
P.R. China, 100084
[4] Tsinghua National Laboratory for Information Science and Technology,
P.R. China, 100084

Abstract. As a textual specification for process definition, XPDL lacks formal semantics which hinders the formal analysis and verification of business processes. In this paper, we provide a method for translating XPDL processes into Petri nets for the formal analysis of XPDL processes. The algorithm validity has been proved, and has also been verified by experiments on artificial and practical processes.

1 Introduction

The XML Process Definition Language (XPDL) is a formal standard process definition language proposed by the Workflow Management Coalition (WfMC) [14]. The purpose is to serve as an exchange language between different modeling languages. Today there are over 50 major business process management and application vendors that support the XPDL standard, including IBM, Oracle, BEA, Fujitsu, Tibco, and Global 360 [8]. However, As a textual specification of process definition, XPDL lacks formal semantics which serves as the foundation of formal analysis and computer aided verification.

To analyze process models without formal semantics, usually we can first transform them into formal models. A lot of such efforts have been dedicated to analyze BPEL, e.g., to finite state machines [4,5], to process algebra [3], to Petri nets [11,6], and our former work to OWL-S [10], etc. And a Petri net is a rather desirable target language of such transformation, with its formal semantics and the availability of many analysis techniques and tools [1].

Although XPDL is a competitive standard to BPEL, no attempts on transforming XPDL have been published so far. In this paper, we present a mapping method from XPDL to Petri nets. For each XPDL process model, we can get a Petri net with equivalent structure and behaviors. The Petri net can be exported to a PNML [13] file for further analysis or simulation purposes.

The remainder of this paper is organized as follows. Section 2 gives a brief overview on XPDL and process meta-model. Section 3 presents a mapping semantics between XPDL and Petri nets. Section 4 presents the algorithm step by step and its validity proof. Section 5 gives case study details. Section 6 discusses the related work and our contributions. Section 7 concludes the paper and discusses future work.

2 A Brief Introduction to XPDL

The WfMC has identified five functional interfaces to a workflow service as part of its standardization program. XDPL forms part of the documentation relating to *Interface one* - supporting process definition import and export. XPDL is a common meta-model for describing the process definition. The purpose is to serve as an interchange of process definition between different tools and also different vendors. The first version of a standard interchange language was the Workflow Process Definition Language (WPDL), published by the WfMC in 1998. The growing popularity of XML and its use for defining document formats for the Internet, combined with some years of accumulated experience using WPDL in workflow and BPM tools, led to the creation of XPDL 1.0, which was officially released in 2002. XPDL retained the semantics of WPDL but defined a new syntax using an XML schema. Neither WPDL nor XPDL 1.0 proposed a specific graphical representation. Intended to be used as a file format for Business Process Modeling Notation (BPMN), XPDL 2.0 was published in 2005, which is back compatible with XPDL 1.0 [9].

Figure 1 is an XPDL process meta-model that describes the top-level entities contained within a process definition, with their relationships and attributes. We have skipped XPDL package meta-model, for XPDL package is just a container for XPDL processes. For more details of XPDL meta-model, we can refer to [14].

Currently dozens of vendors have announced conformance of XPDL. A more detailed list of vendors and products can be found on the WfMC website [1].

3 Mapping from XPDL to Petri Nets

Our goal is to translate an XPDL process model into a Petri net. However, we do not pursue a complete Petri net semantics to XPDL. We focus on structure and behavior equivalence. As shown in Figure 1, a process meta-model includes several elements, e.g., activities, participants, applications, transitions and data fields, etc. We skip some elements which are not related to our concerns, e.g., applications, data fields and other extended attributes of various tool vendors, etc. In this section, we introduce the mapping semantics in three aspects, namely, activities, links and routing structures.

3.1 Activities

An XPDL activity can have attributes and child elements, such as *Id*, *Name*, *Performer*, and *Join/Split type*, etc. An activity is either primitive or structured

[1] $http://www.wfmc.org/standards/conformance\_chart.htm$

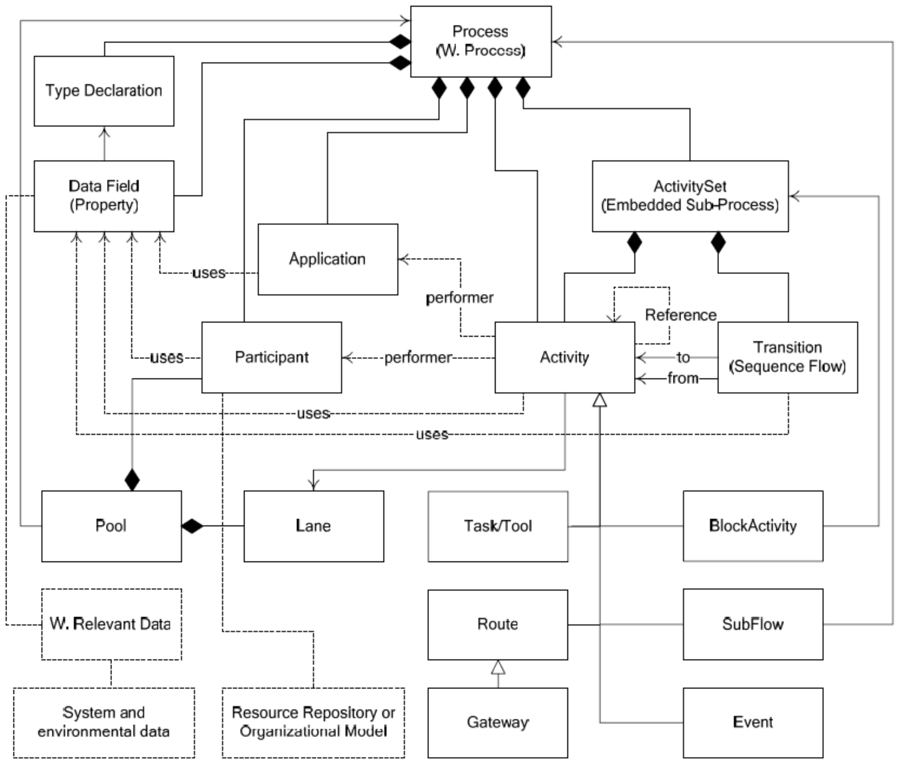


Fig. 1. XPDL Process Meta-Model

(activity sets). An sample of the primitive activity element is shown as follows.

$$
\begin{array}{l}
< Activity\ \ Id = \text{“}Act1\text{”}\ \ Name = \text{“}A\text{”} > \\
\quad + < Implementation > \\
\qquad < Performer > Par1 < /Performer > \\
\quad + < StartMode > \\
\quad + < FinishMode > \\
\quad + < TransitionRestrictions > \\
\qquad\quad < TransitionRestriction > \\
\qquad\qquad + < Join\ \ Type = \text{“}AND\text{”} > \\
\qquad\qquad + < Split\ \ Type = \text{“}XOR\text{”} > \\
\qquad\quad < /TransitionRestriction > \\
\qquad < /TransitionRestrictions > \\
\quad + < ExtendedAttributes > \\
< /Activity >
\end{array}
$$

It is instinctive to map *XPDL activities* to *Petri transitions*. A PNML Petri model is ready to integrate limit activity attributes, e.g., *Id*, *Name*. Other attributes and child elements need an extended Petri net model. As to the Join/Split type attributes, they are no longer a part of Petri net transition. The Join/Split type will be discussed in Section 3.3.

3.2 Links

Links are literally *transitions* in XPDL process. To avoid confusing with the term *transition* in Petri nets, we use links instead throughout this paper. A link element is shown as follows.
$< Transition\ \ From = \text{“}Act1\text{”}\ \ Id = \text{“}Tra2\text{”}\ \ To = \text{“}Act2\text{”} >$
$+ < ExtendedAttributes >$
$< /Transition >$

A link is a relatively simple element, which maps to an arc in a Petri net. A link can have attributes, such as *Id*, *From* (Source activity) and *To* (target activity).

There is a challenging problem when we map a *XPDL link* to a *Petri net arc*. In the XPDL context, the source object and target object of a link are both activities. But according to the Petri net syntax, an arc cannot connect two transitions directly. So a *Petri net place* must be inserted between two transitions. Roughly speaking, an *XPDL link* maps to two *Petri net arcs*, with one arc's target on the inserted place, and the other arc's source on the same one.

However, the problem is only partly resolved. In a Petri net, a routine structure depends on the connections of arcs between transitions and places. Just inserting a place between two neighboring transition is only correct syntactically, the semantic correctness cannot be guaranteed. In fact the situation is even worse, because the semantic incorrectness may be different under different context. Hence we cannot find a simple method to remedy.

To avoid the uncertainty of semantic incorrectness, we introduce a mapping technique which considers the correctness both syntactically and semantically. In this step, we add an input place for each transition. We map an XPDL link to a Petri net arc between the source transition and the target transition's input place, regardless of the join/split conditions of both XPDL activities. Then we derive a Petri net with correct syntax. Although semantic incorrectness still exists, this time it becomes simpler to fix, because the semantic incorrectness can only be from two routing structures, i.e., AND-join structure and XOR-split structure. The correct measures are presented in Section 3.3.

3.3 Routing Structures

In an XPDL process model, there are explicit routing structures other than links. As mentioned in Section 3.1, each activity can have two kinds of join and split types, i.e., AND-join, XOR-join, AND-split and XOR-split, as shown in Figure 2 (a). However, in a Petri net there is no corresponding structure. A Petri net implements routing structure through right connection between transitions and places, as shown in Figure 2 (b). In Section 3.2 we derived a Petri net without AND-join and XOR-split structure. A correcting adjustment of the derived model is necessary, i.e., (1) correcting an XOR-join structure to an AND-join structure, and (2) correcting an AND-split structure to an XOR-join.

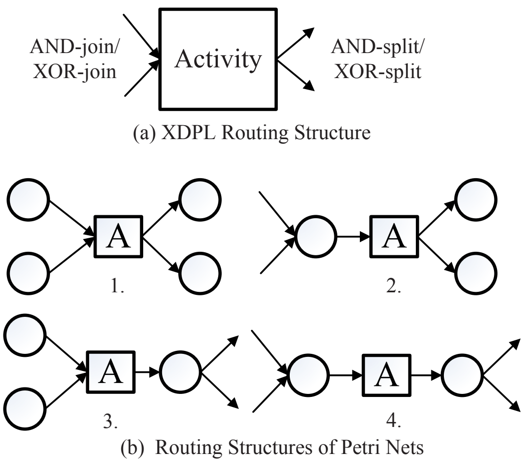

AND-join/
XOR-join
Activity
AND-split/
XOR-split
(a) XDPL Routing Structure
A
1.
2.
3.
4.
(b) Routing Structures of Petri Nets

Fig. 2. Routing Structures Mapping

As shown in Figure 2 (b), 2 and 4 have an XOR-join structure, and 1 and 3 have an AND-join structure. In our algorithm, there is no other output arc of a transition's input place, except the arc to the transition itself. So the adjustment impact is local which means the operation will not involve other transitions, as shown in Figure 3 (a). For each input arc of the input place, the first step splits a new place for each input arc. The second step adds an input arc linking the new place and the transition. The final step deletes the old input place and the input arc.

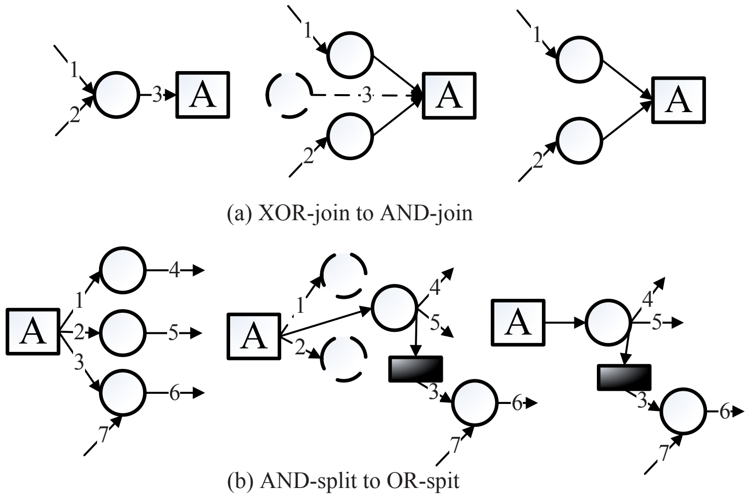


Fig. 3. Correction of Routing Structure

To correct an AND-split structure to an XOR-split it is similar, just through incorporating output arcs and output places. But there is an exception when the output place's in-degree is more than 1. This case needs to introduce a routing transition, as shown in 3 (b). If we do not distinguish Arc 6 from Arc 4 and Arc 5 in 3 (b), behaviors of the transformed Petri net will be changed.

4 Algorithm

The following steps are the mapping algorithm from an XPDL process model to a Petri net.

1. Map each *activity* in XPDL to a *transition* in Petri nets, along with their *attributes*.
2. Add an input *place* and an input *arc* for each transition, except for which has no input links in XPDL.
3. Map each *link* to an *arc*, linking source transition and target transition's input place.
4. Correct *AND-join* structure.
5. Correct *XOR-split* structure.
6. Add a *source place* and a *sink place*.

The details of each step have been discussed in Section 3 except Step 6. The purpose of Step 6 is to construct a Wf-net [1], because a Wf-net is more desirable for the analysis purpose. The only one source place serves as the input place for all transitions without any input places. And the only one sink place serves as the output place for all transitions without any output arcs.

Theorem 1. *The Petri net is equivalent with the XPDL model in structure and observed behaviors, i.e., the transforming algorithm is correct.*

Proof. 1. Step 1 guarantees that the two models have the same activity set.
2. Step 2 and Step 3 make that all activity ordering relations are the same in the two models.
3. Since the algorithm treats any *join structure* as XOR-join and any *split structure* as AND-split, the original XOR-join and AND-split structures remain unchanged after transformation.
4. In the previous steps, AND-join structures are treated as XOR-join structures and XOR-split structures as AND-split structures improperly. Step 4 and Step 5 remedy these flaws respectively.

Therefore, the transforming algorithm guarantees the same structure of the two models. Consequently, it guarantees the same observed behaviors. □

5 Case Study

In this section, we present our case study. The sample process is modelled by JaWE [7], which is a compatible process editor to XPDL, as shown in Figure 4. In

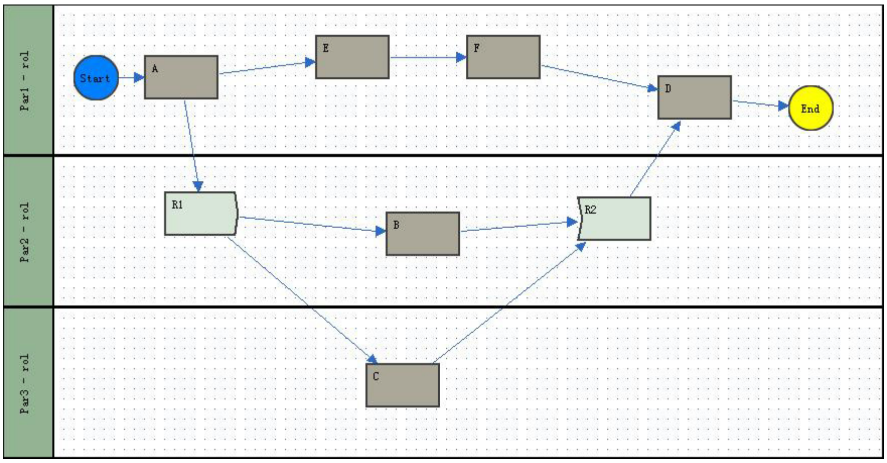


Fig. 4. Sample Process Definition in XPDL

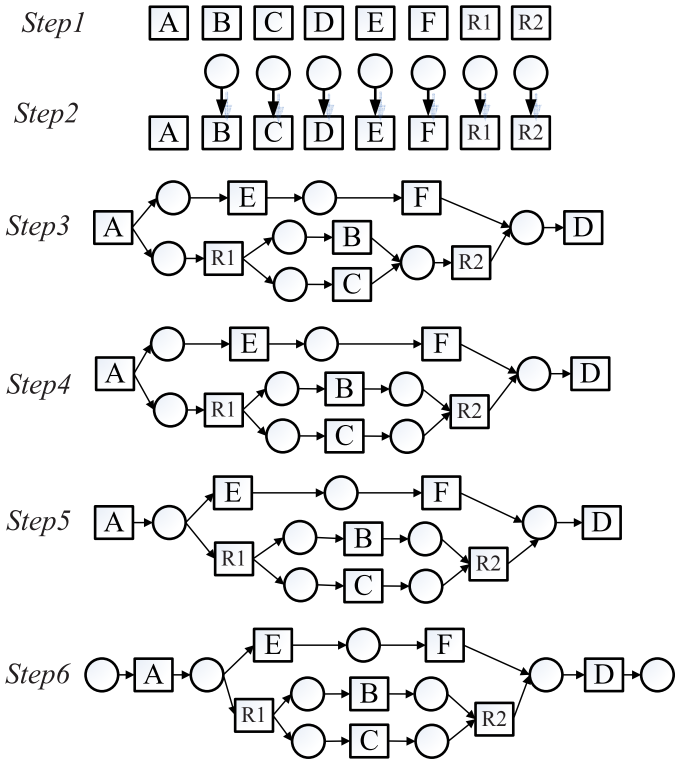


Fig. 5. Illustration of XPDL Transformation Algorithm Steps

the process, there are eight activities including two routing activities. There are three participants named par1 to par3. The routing structures include sequence structure (e.g., EF), split structure (e.g., AE or AR1), and parallel structure (e.g., BC). Correspondingly, there exist all possible Join and Split types, i.e., XOR-split (e.g., A), AND-split (e.g., R1), XOR-join (e.g., D) and AND-join (e.g., R2).

Figure 5 illustrates the transformation steps. In Step 1, each activity in the XPDL process model maps to a transition of a Petri net, with the same Name. In Step 2, each transition is added with an input place and a connect arc except Transition A, because corresponding activity of Transition A has no input link in the XPDL process. After Step 3, a Petri net with correct syntax is generated, but it is not correct semantically. Note that there are only one type of the join structure, i.e., the XOR-join structure, and only one type of the split structure, i.e., the AND-split structure. Therefore, the following two steps are necessary to correct the Petri net with AND-join structure (in Step 4) and XOR-split structure (in Step 5). Finally, Step 6 is an additional step to construct a Wf-net.

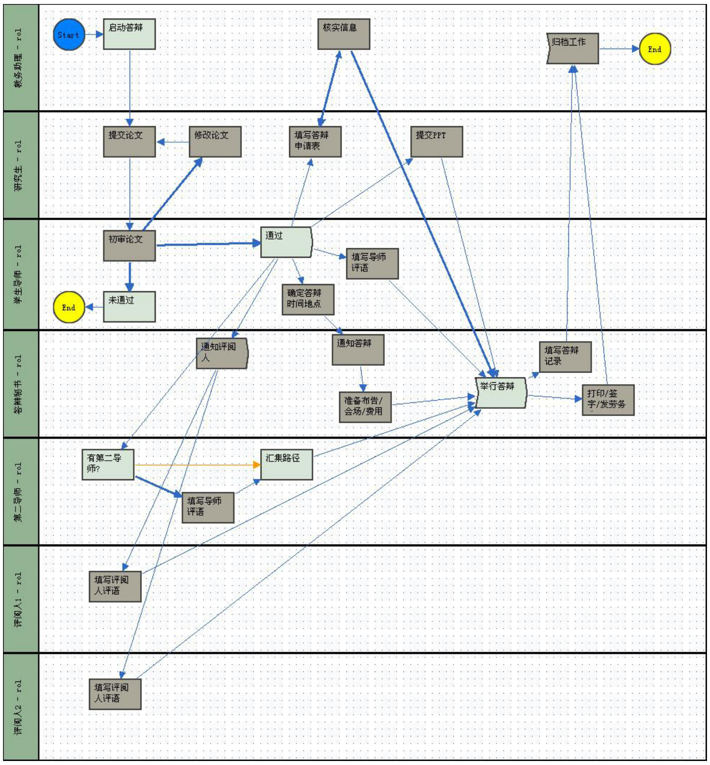


Fig. 6. XPDL Definition Model of Graduate Dissertation Defense Process

Currently we have implemented our algorithm as part of our *ISEflow* project. In the project, we can import XPDL definition file generated by an XPDL conformance process editor, e.g. JaWE [7]. Then the transformed Petri net model can be exported to a PNML [13] file for further analysis, or to a lightweight Petri net based engine to collect execution logs. Since XPDL is popularly used in China, we can study rich practice process models in a sound manner through the transformation.

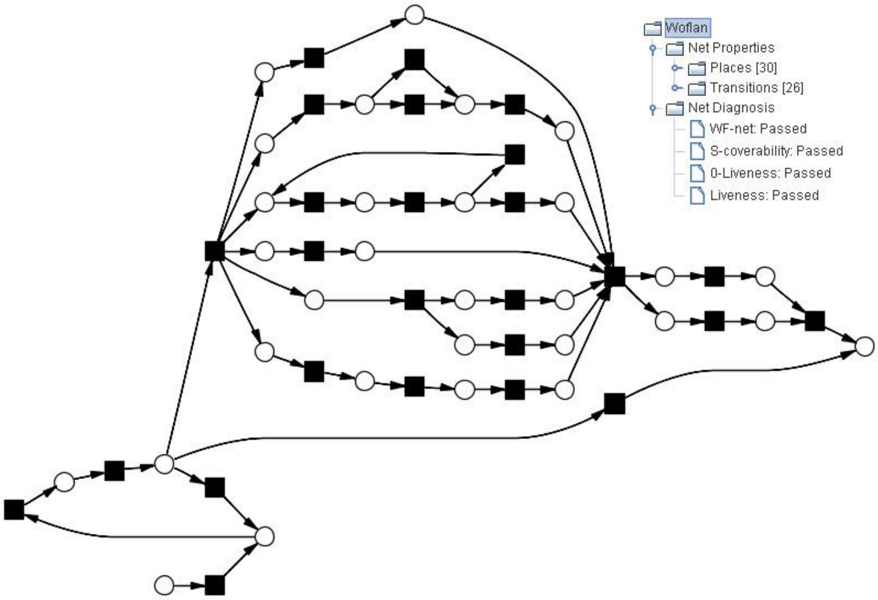


Fig. 7. Petri Net Corresponding to the XPDL Model in Figure 6 with Woflan Analysis Result

We have also conducted a lot of experiments on practical processes. In Figure 6, a real world XPDL model of a graduate dissertation defense process is shown. We reserve the activity labels in original Chinese for it does not matter to our understanding of the process structure and control flow. And in Figure 7, the corresponding Petri net model is presented with the analysis result by Woflan [12] in ProM [2].

6 Related Work

There are several publications that define transformations between different business process modeling languages, with the same motivation to analyze and verify business process. However, most of these efforts are for BPEL [4,5,3,11,6,10].

Petri nets (including Wf-nets) are a desirable target language of transformation, for its formal foundation and lots of available analysis theory and tools.

Similar efforts as ours can be found in efforts to transform BPEL to Petri nets. In [11] authors make an attempt to map a BPEL process model on a Wf-net for verifying process soundness. They classify activities into different categories according to their different semantics. And make a Wf-net mapping for each type activities respectively.

In [6] each construct of BPEL language is separately mapped into a Petri net. Each pattern has an interface for joining it with other patterns as is done with BPEL constructs. The collection of these patterns forms the Petri net semantics for BPEL.

The contribution of our approach lies that: (1) It is the first attempt to transform XPDL to Petri nets for formal analysis. In spite of the comparability between BPEL and XPDL, they are different language with different specification [8]. As to the best of our knowledge, there is no publication on formal transformation of XPDL yet. (2) We focus on structure and behavior equivalence. It enable us to develop an concrete algorithm whose validity can been proved, rather than present a set of sematic mapping rules.

7 Conclusion and Future Work

Formal semantics is the foundation of analysis and verification of process definition. This paper aims to automatically transform XPDL process models to Petri nets with equivalent structure and behaviors. We have presented the mapping semantics as well as concrete transforming algorithm. The algorithm has been proved and demonstrated experimentally.

As future work, we will seek to integrate the algorithm into ProM workflow process mining framework, so that XPDL models can be directly imported to ProM, then they can be analyzed by Petri net analysis tools in ProM. In addition, to support various analysis and simulation purposes of XPDL process, we will improve mapping semantics to cover more XPDL 2.0 specification.

Acknowledgements

This work is supported by the 973 Project of China (No. 2002CB312006), the Project of National Natural Science Foundation of China (No. 60473077 and No. 60373011), and the Program for New Century Excellent Talents in University of China.

References

1. van der Aalst, W.M.P.: The Application of Petri Nets to Workflow Management. The Journal of Circuits, Systems and Computers 8(1), 21–66 (1998)
2. van Dongen, B.F., de Medeiros, A.K.A., Verbeek, H.M.W., Weijters, A.J.M.M., van der Aalst, W.M.P.: The ProM framework: A New Era in Process Mining Tool Support. In: Ciardo, G., Darondeau, P. (eds.) ICATPN 2005. LNCS, vol. 3536, pp. 444–454. Springer, Berlin (2005)

3. Ferrara, A.: Web Services: a Process Algebra Approach. In: Proceedings of the 2nd international conference on Service oriented computing, pp. 242–251. ACM Press, New York (2004)
4. Fisteus, J.A., Fernández, L.S., Kloos, C.D.: Formal Verification of BPEL4WS Business Collaborations. In: Bauknecht, K., Bichler, M., Pröll, B. (eds.) EC-Web 2004. LNCS, vol. 3182, pp. 76–85. Springer, Berlin Heidelberg (2004)
5. Fu, X., Bultan, T., Su, J.: Analysis of Interacting BPEL Web Services. In: Proceedings of the 13th international conference on World Wide Web, pp. 621–630. ACM Press, New York (2004)
6. Hinz, S., Schmidt, K., Stahl, C.: Transforming BPEL to Petri Nets. In: van der Aalst, W.M.P., Benatallah, B., Casati, F., Curbera, F. (eds.) BPM 2005. LNCS, vol. 3649, pp. 220–235. Springer, Berlin Heidelberg (2005)
7. ObjectWeb. Enhydra JaWE: Open Source Java XPDL Editor, http://www.enhydra.org/workflow/jawe/index.html
8. Pyke, J.: Is XPDL the Silent Workhorse of BPM? (2007), http://www.ebizq.net/hot_topics/bpm/features/7852.html
9. Shapiro, R.M.: XPDL 2.0: Integrating Process Interchange and BPMN. In: 2006 Workflow Handbook, WfMC, pp. 183–194 (2006)
10. Shen, J., Yang, Y., Zhu, C., Wan, C.: From BPEL4WS to OWL-S: Integrating E-Business Process Descriptions. In: SCC2005. Proceedings of IEEE International Conference on Services Computing, Orlando, Florida, USA, pp. 181–188 (2005)
11. Verbeek, H.M.W., van der Aalst, W.M.P.: Analyzing BPEL Processes using Petri Nets. In: Proceedings of the Second International Workshop on Applications of Petri Nets to Coordination, Workflow and Business Process Management, pp. 59–78 (2005)
12. Verbeek, H.M.W., Basten, T., van der Aalst, W.M.P.: Diagnosing workflow processes using Woflan. The Computer Journal 44(4), 246–279 (2001)
13. Weber, M., Kindler, E.: The Petri Net Markup Language. In: Ehrig, H., Reisig, W., Rozenberg, G., Weber, H. (eds.) Petri Net Technology for Communication-Based Systems. LNCS, vol. 2472, pp. 124–144. Springer, Berlin Heidelberg (2003)
14. WfMC. Workflow Process Definition Interface – XML Process Definition Language (XPDL) (WFMC-TC-1025). Technical report, Workflow Management Coalition (2005)

Interaction Modeling Using BPMN

Gero Decker[1] and Alistair Barros[2]

[1] Hasso-Plattner-Institute, University of Potsdam, Germany
gero.decker@hpi.uni-potsdam.de
[2] SAP Research Centre, Brisbane, Australia
alistair.barros@sap.com

Abstract. Process choreographies describe interactions between different business partners and the dependencies between these interactions. While different proposals were made for capturing choreographies at an implementation level, it remains unclear how choreographies should be described on a conceptual level. While the Business Process Modeling Notation (BPMN) is already in use for describing choreographies in terms of interconnected interface behavior models, this paper will introduce interaction modeling using BPMN. Such interaction models do not suffer from incompatibility issues and are better suited for human modelers. BPMN extensions are proposed and a mapping from interaction models to interface behavior models is presented.

1 Introduction

The Business Process Modeling Notation (BPMN [1]) is the de-facto standard for business process modeling. It is mainly used for capturing activities, decision responsibilities, control and data flow in business process *within* one organization. However, in cross-organizational settings we concentrate on the interaction behavior between the different partners involved. The individual partners can internally implement processes as they like as long as their interaction behavior conforms to the *choreography* that is agreed upon. Especially when relying on electronic messages as means for interaction between different partners, an exact definition of message formats and interaction sequences is of major importance.

BPMN can already be used for choreography modeling by expressing interconnected interface behavior models. However, this modeling style leads to redundant control flow dependencies and the danger of incompatible processes. An example for such incompatibility would be a supplier who waits for the payment to arrive before delivering the purchased goods. The buyer, on the other hand, waits for the goods to be delivered before actually paying for them. Both partners would wait endlessly – a classical deadlock situation. Interaction models avoid these problems by describing control flow dependencies between interactions. This means that a particular control flow dependency is not explicitly assigned to any of the partners in the model.

Another drawback of redundancy is that modelers need more time for creating and understanding the models. It has turned out that interaction modeling allows faster creation and understanding by human modelers.

There are different language proposals for interaction modeling, e.g. the Web Service Choreography Description Language (WS-CDL [9]) and Let's Dance [11]. WS-CDL operates on an implementation level and only comes with a textual syntax. Let's Dance has a graphical notation, however, it is very different to that of established process modeling languages. The motivation for extending BPMN is to reuse a very popular notation as many process modelers are already trained in this notation.

The remainder of this paper is structured as follows. The next section will revisit choreography modeling with standard BPMN, before section 3 introduces the extensions for interaction modeling. Section 4 shows how interface behavior models can be generated out of interaction models. Section 5 will report on related work and section 6 concludes and points to future work.

2 Choreography Modeling Using Standard BPMN

A bidding scenario is going to be used as sample choreography throughout this paper. Three types of participants are involved in this scenario: a seller, several bidders and an auctioning service. The seller initiates an auction with the goal to sell her goods for the highest possible price. She does not operate the auction by herself but rather outsources this to an auctioning service. Different bidders can join in if they are interested in the goods and place their bids accordingly. Figure 1 shows the structural view on this collaboration scenario using BPMN.

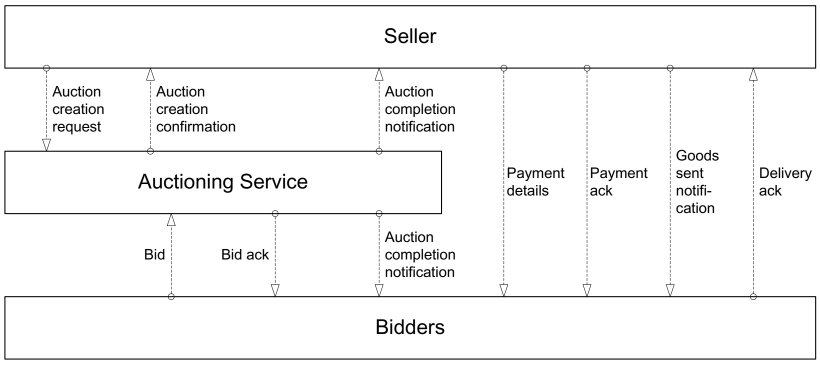


Fig. 1. Bidding scenario: Structural view

The participant types are represented by pools and message flows between the pools indicate which messages might be sent from a participant of one type to a participant of another type. Ordering constraints between the message exchanges are not expressed in the diagram.

Figure 2 depicts the complete choreography consisting of interconnected behavioral interfaces. For every message flow message send and receive events are

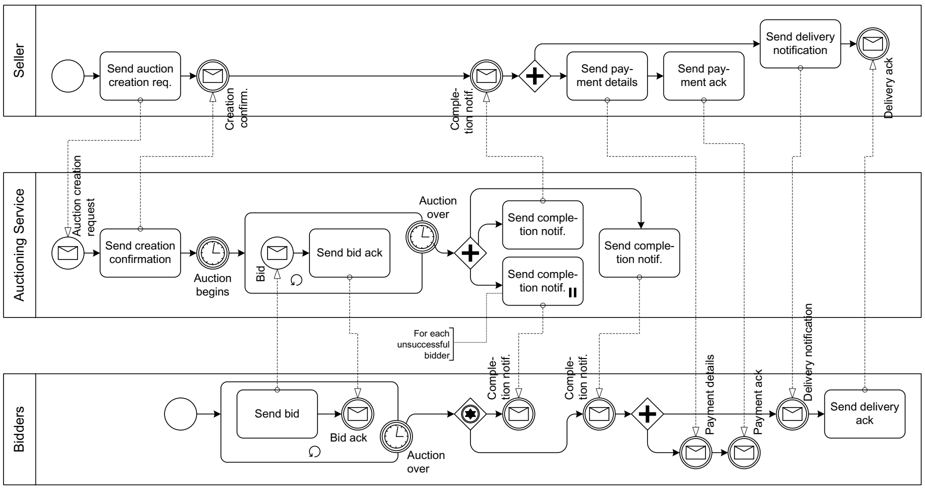


Fig. 2. Interconnected behavioral interfaces for the bidding scenario

introduced. The control flow within each pool connects these communication activities and therefore defines the behavioral dependencies between the different message exchanges.

First, the seller sends an auction creation request to the auctioning service who acknowledges it with a confirmation message. As soon as the auction begins (depicted by an intermediate timer event), bidders can place bids that are in turn confirmed by the auctioning service. The auction ends at a given point in time and the auctioning service notifies the seller about which seller has placed the highest bid and how high the corresponding amount is. The bidder who has won the auction also gets a notification. All other bidders with lower bids are informed, too. Finally, payment and shipment can happen in parallel. The seller sends payment details containing e.g. the bank account number to the successful bidder and acknowledges the payment as soon as the money has arrived. On the shipment side, the seller sends a notification to the bidder as soon as the goods are sent and the bidder acknowledges the delivery.

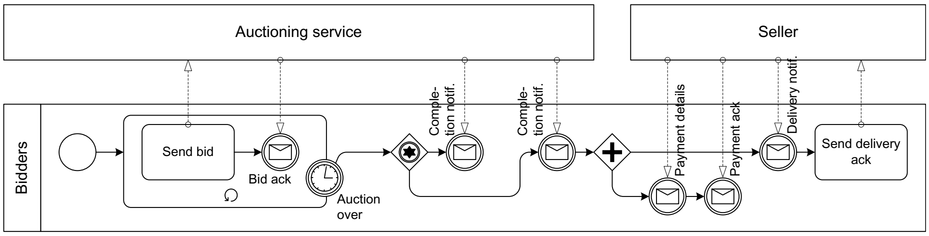


Fig. 3. Participant behavior description for bidders

While the choreography model contains all relevant interactions and dependencies, interface behavior models are the individual views on the choreography from the perspective of one of the participants. Figure 3 shows how such a model looks like for the bidders. Only the communication actions of the bidders are included, while the other participants are depicted as black boxes.

Modeling choreographies in terms of interconnected interface behavior models has two drawbacks:

1. **Redundancy.** As an example, the ordering constraint between the auction creation request and the creation confirmation appears twice: in the interface behavior models of the seller and the auctioning service. Parallelism, branching, loops and timeouts are duplicated in the model, too. This redundancy involves unnecessary modeling effort and often lead to invalid models.
2. **Potentially incompatible behavior.** If sequencing structures do not match properly, we might run into deadlocks. An even more common modeling error occurs in the case of branching: While modelers immediately understand the semantics of data-based XOR-gateways, we often find misunderstanding in the case of event-based XOR-gateways, resulting in erroneous models.

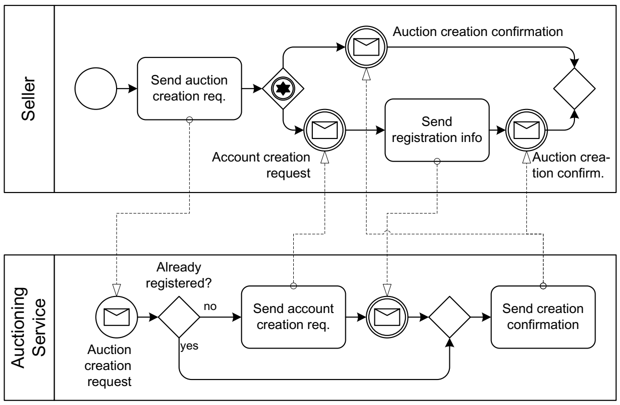

Seller
Send auction creation req.
Auction creation confirmation
Send registration info
Account creation request
Auction creation confirm.
Auctioning Service
Already registered?
no
yes
Send account creation req.
Send creation confirmation
Auction creation request

Fig. 4. Matching branching structures

Figure 4 shows properly matching branching structures. The data-based XOR-gateway on the auctioning service's side indicates that it decides whether registration is needed or not. The event-based XOR-gateway makes the choice on the seller's side dependent on which message comes in. Process modelers often use data-based gateways instead of event-based gateways, ignoring the location of where the choice is made. Furthermore, we see in this example that the receipt action for the auction creation confirmation needs to appear twice in the seller's interface behavior model. Such problems compound when further parties are involved. Also, looping and multiple instances are typical sources for mismatches between the different interface behavior models.

Interaction models do not suffer these drawbacks. As interactions are the basic building blocks, less nodes are needed to express the same choreography as an interaction model. Especially the distinction between local choices and choices made by the environment is often not needed. Incompatibility does not occur as control flow dependencies are not duplicated. While observing human modelers using the different choreography modeling styles, it turned out that interaction modeling leads to faster model creation and understanding.

However, interaction models come with their own anomalies. Locally unenforceable choreographies, i.e. choreographies where the individual participant cannot collectively enforce global control flow constraints without additional synchronization messages, can be expressed. Imagine e.g. that an interaction between the seller and a bidder must only occur after a certain interaction between the auctioning service and another bidder has taken place. In this case the seller and the first bidder cannot know when the second interaction has actually happened. This property of local enforceability is reported in [12] and [5].

3 BPMN Extensions for Interaction Modeling

This section is going to introduce the BPMN extensions for interaction modeling, which we will call "iBPMN". Atomic interactions are going to be the basic building blocks of these models and control and data flow are defined between them. I.e., we do not use separate send and receive activities in the models.

Each elementary interaction (represented as message event) is attached to a message flow in iBPMN, as shown in Figure 5. All control flow constructs that are available in BPMN also apply in these interaction models. E.g. we see that parallelism and timers appear in the choreography. Elementary interactions

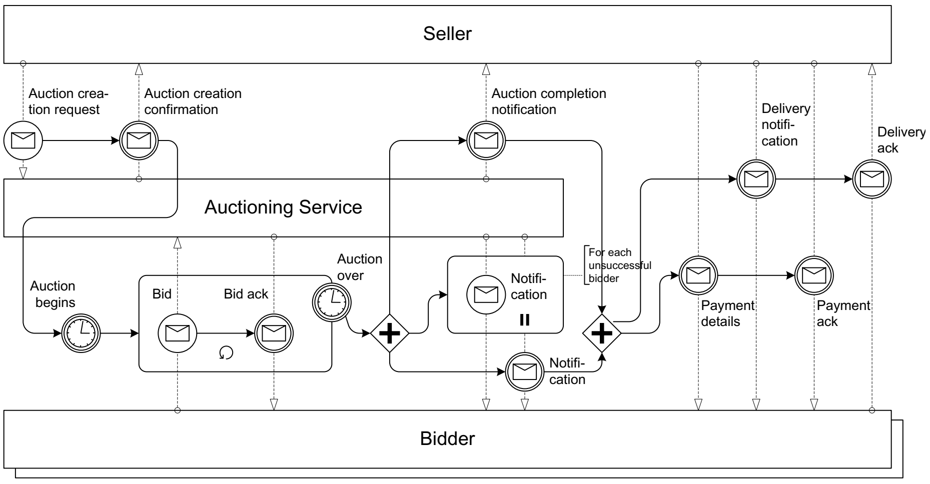


Fig. 5. iBPMN interaction model for the bidding scenario

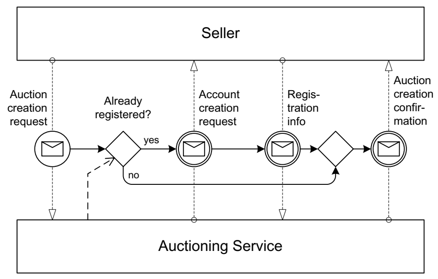


Fig. 6. Explicit choice

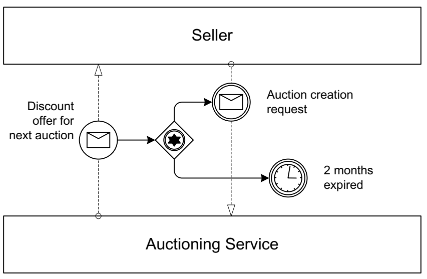


Fig. 7. Racing choice

can be composed to form complex interactions, enabling loop interactions and multiple instances interactions as shown in the figure.

In iBPMN, pools are empty, i.e., the internal behavior of the participants is completely hidden. Therefore, the interaction model is a refinement of the structural diagram in Figure 1.

A recurrent scenario in choreographies is that several participants of the same type are involved in one conversation. Our bidding scenario also includes this case: Several bidders participate in an auction. In order to make a clear distinction between the case with only one participant of a type vs. potentially many participants, we introduce shadowed pools as shown in Figure 5.

In interaction models we make a distinction between explicit choices and racing choices. In the case of explicit choices one participant decides which branch to take. This is represented by a data-driven XOR-gateway. We further add an association between the gateway and one of the pools in order to define who actually carries out the choice. In the case of racing choices one among a set of events can happen and the event occurring first inhibits the others from happening and determines which branch is taken. Event-based XOR-gateways depict this. Figures 6 and 7 illustrate the two types of choices.

Another recurrent scenario in choreographies is passing on participant references. Imagine the payment between bidder and seller is carried out using a payment service and the seller can choose which service to use. The seller needs to pass on the reference to this service to the bidder so that the bidder can issue the payment with that service. Figure 8 illustrates how this is represented in iBPMN: A data object is attached to the message flow and the object is in turn associated with the corresponding participant.

We are now going to validate the suitability of iBPMN for choreography modeling by investigating which of the Service Interaction Patterns are directly supported. These patterns describe recurrent scenarios in choreographies and have already been used to assess Let's Dance [11] and WS-CDL [4].

Some of these patterns appear in our sample choreography. E.g., *Send*, *Receive* and *Send/receive* can be found in the first two interactions where the seller initiates the auction.

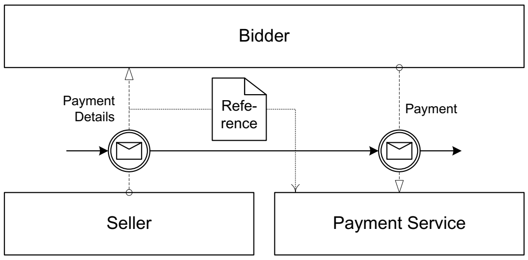


Fig. 8. Participant reference passing

In the case of *Racing incoming messages* a participant processes the first out of a set of messages that he receives. This can be modeled using an event-based XOR-gateway. *One-to-many send* occurs in the choreography where the auctioning service sends out notifications to all the unsuccessful bidders, which is modeled through a multiple instances send activity. However, the fact that a message is sent to all unsuccessful bidders is only captured by the annotation. It might be desirable to more directly integrate such a "for each" into iBPMN. A general drawback of BPMN is that it cannot be specified which particular participant a message is sent to, only the participant type is defined. We have said that simple pools indicate that there must be at most one participant of that type in one choreography instance. Therefore, we can be sure that all messages sent to a participant of type auctioning service are actually sent to the same concrete participant, if all messages belong to the same choreography instance. It becomes difficult in those cases where we have many participants of the same type in one choreography instance. In our example we do not directly see that there is a distinction between the bidder with the highest bid and the remaining bidders.

One-from-many receive can be found during the bidding phase: The auctioning service does not know in advance how many bidders are going to take part in the auction. As there are potentially many bidders a bid from any sender is received and processed. *Multi-responses* is a bi-lateral pattern where several responses are sent back as result of a single request. This can easily be modeled using loop interactions in iBPMN. *Contingent requests* involves a list of recipients for requests. If the first recipient does not respond within a given timeframe, the request is sent to the second and so on. Loop interactions with corresponding annotations express this in iBPMN. However, late responses from previous participants are discarded. Therefore, this pattern is only partially supported in iBPMN. *Atomic multicast notification* is not supported in iBPMN.

Relayed request requires that a participant observes a conversation between two other participants. This can easily be modeled using two parallel interactions in iBPMN. *Request with referral* alludes the notion of link passing mobility. Figure 8 showed how this is modeled in iBPMN.

We see that all Service Interaction Patterns are directly supported in iBPMN (except *atomic multicast notification*). In this sense iBPMN provides the same pattern support as Let's Dance. iBPMN provides better pattern support than

WS-CDL, as scenarios where multiple participants of the same type are involved and the exact number of participants are only known at design-time are fully supported in iBPMN.

4 Generation of Interface Behavior Models

While deriving individual interface behavior models from classical BPMN choreographies is trivial (see Section 2), deriving these models from iBPMN choreographies is more complex. A typical approach to generating interface behavior models out of interaction models is by means of model reduction (cf. [12]). Those interactions where the corresponding participant is not involved in are marked as τ-actions and they are removed from the model while preserving control flow dependencies. This section presents an algorithm for interface behavior models out of simple iBPMN models.

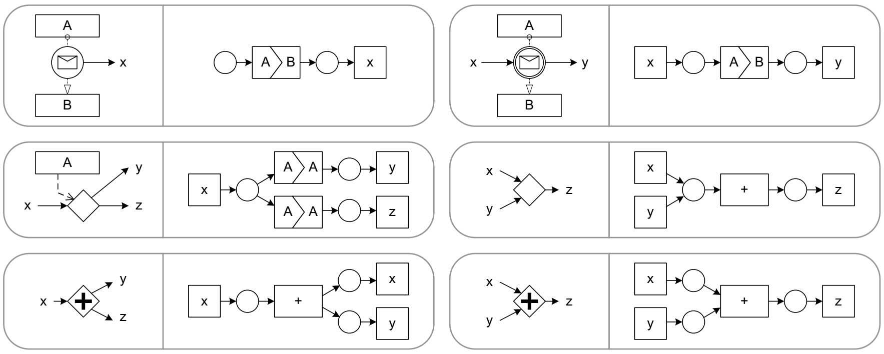


Fig. 9. iBPMN constructs and their interaction Petri net representation

We restrict the algorithm to a very small subset of iBPMN models. We only allow elementary interactions as well as AND- and data-based XOR-gateways. In [5] we have already presented a reduction algorithm for interaction Petri nets, an extension to classical place / transition nets for interaction modeling. We are going to reuse this algorithm in the following way:

1. Translate the simple iBPMN model to an interaction Petri net. Figure 9 shows the translation rules for the allowed constructs. We mark those interactions as τ-actions where the participant who we generate the interface behavior model for does not participate. We also mark those transitions representing exclusive choices being made by another participant as τ. The transitions representing AND-gateways are labeled "+".
2. Apply the reduction algorithm from [5]. This removes all τ-transitions from the model. Those "+"-transitions that are involved in choices, i.e. sharing a common input place with another transition, are relabeled to τ and are reduced as well.

3. Transform the resulting interaction Petri net in such a way that only those patterns appear that can be directly translated back to BPMN. Optionally, the net can be reduced by removing redundant places or removing "+"-transitions with at most one preceding and at most one succeeding transition.
4. Translate the interaction Petri net to BPMN.

The resulting BPMN interface behavior models will contain message send activities and event-based gateways, in addition to those constructs allowed in the input iBPMN model. The interaction Petri net patterns that are translated back to BPMN are shown in Figure 10.

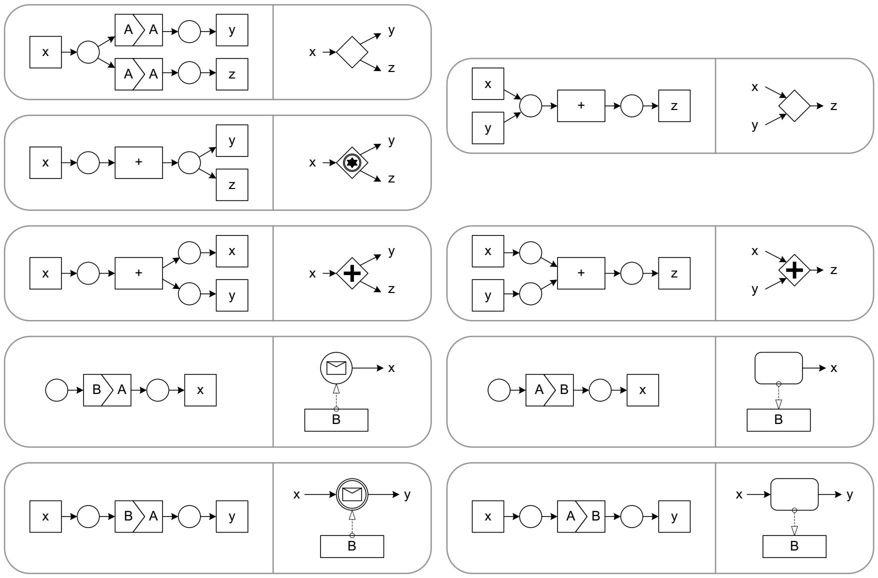


Fig. 10. Interaction Petri net patterns translated to BPMN

The transformation in step 3 leads to adding "+"-transitions in those cases where interactions have more than one input or output place or where input or output place are shared with other transitions. The "+"-transitions will be translated to gateways later on. Only the patterns depicted in Figure 10 can be translated. Therefore, the introduction several "+"-transitions might be necessary in some structures.

Figure 11 shows the result of the generation algorithm. The two interactions $m3$ and $m5$ do not occur in the interface behavior model for A, as this participant is not involved in $m3$ and $m5$. The choice made by B results in the occurrence of an event-based gateway for A. A only knows which branch B has chosen as soon as it gets one of the messages $m2$, $m4$ and $m6$. Furthermore, $m4$ and $m6$ needed to be sequentialized as BPMN requires that an event-based gateway is followed by events as opposed to other gateways. In this case, the parallelism from the

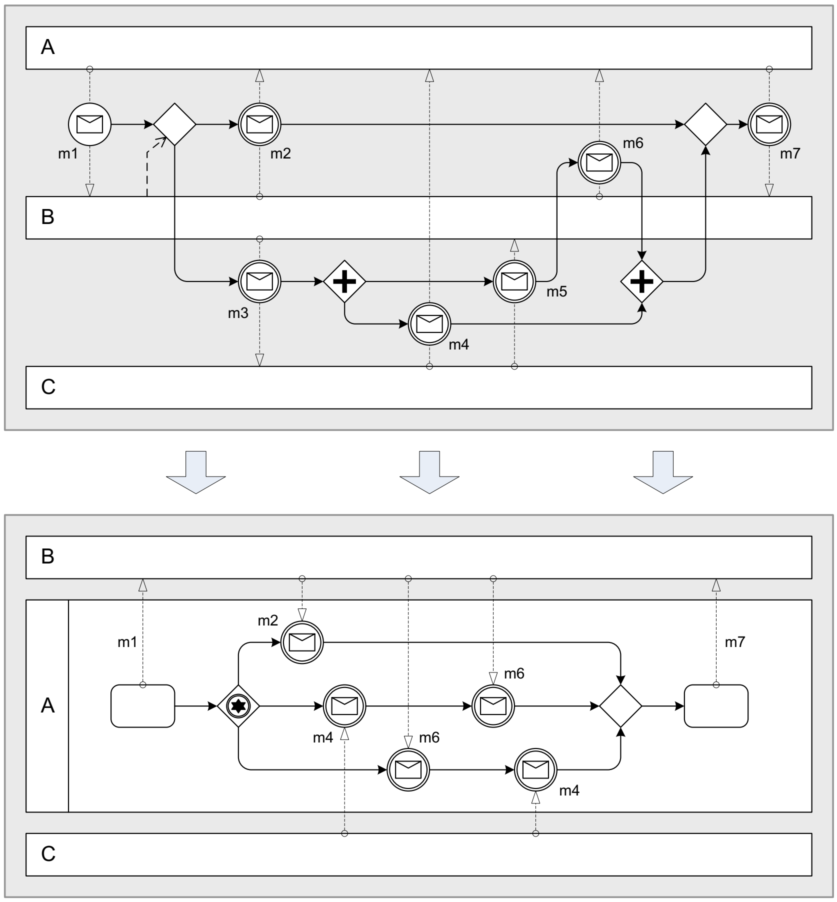


Fig. 11. Generation of a BPMN interface behavior model from an iBPMN model

interaction model completely disappears in A's interface behavior model (while it would be preserved in the interface behavior model for C).

It is not possible to generate valid BPMN interface behavior models out of every valid simple iBPMN model. This is due to the fact that the reduction algorithm might produce non-free-choice interaction Petri nets. In free-choice nets, every transition that shares an input place with another transition t, has the same set of input places like t (cf. [6]). A non-free-choice net would need to be represented by a BPMN model, where an event-based XOR-gateway is followed by an AND-gateway. This is not allowed.

5 Related Work

In [11], Zaha et al. identify the need for describing choreographies on a conceptual level and introduce the choreography language Let's Dance. It provides direct support for most of the Service Interaction Patterns [2], a catalog of common scenarios in choreographies and therefore a benchmark for assessing choreography languages. Let's Dance follows the interaction modeling approach, i.e. interactions are the basic building block in choreography models. Let's Dance comes with a set of own control flow constructs different to those e.g. known from BPMN or UML 2.0 Activity Diagrams. BPMN [1] has been assessed for its suitability for process modeling in [10]. However, choreography modeling was not discussed and the Service Interaction Patterns were not considered.

Message Sequence Charts (MSCs [8]) can also be used for describing choreographies following the interconnected interface behavior modeling approach. However, they are rather suited for describing mere sequences of interactions in contrast to full choreographies: conditional branching, parallel branching and iterations are not supported.

WS-CDL [9] and BPEL4chor [3] are proposals for describing choreographies at an implementation level. Both approaches allow to specify choreographies of web services and do not come with a graphical representation. While WS-CDL follows the interaction modeling approach, BPEL4chor allows to specify interconnected behavioral interfaces. BPEL4chor distinguishes between three different artifact types: Participant topology, behavioral interfaces and participant grounding. The topology describes the structural aspects of the choreography, the behavioral interfaces describe the control and data flow dependencies between the communication activities within the participants and the participant grounding introduces web-service-specific configurations, e.g. the mapping of message links to WSDL port types and operations.

Dijkman et al. have defined a mapping from BPMN to Petri nets in [7]. They consider more constructs than we have used in section 4 including subprocesses, timer events and intermediate events attached to activities (cancellation).

6 Conclusion

This paper has introduced iBPMN, a set of extensions to the Business Process Modeling Notation for interaction modeling. Following an interaction modeling approach as opposed to modeling interconnected interface behavior models, it can be expected that choreography designers can understand models better, introduce less errors, such as incompatibility, into the models and are faster at creating the models. However, a detailed survey validating these hypotheses is left to future work.

We have shown that most Service Interaction Patterns can be expressed using iBPMN and we have presented an algorithm for deriving interface behavior models from simple interaction models. The algorithm is validated through ongoing implementation.

It turns out that some choreographies are not *locally enforceable*, i.e., it is possible to introduce control flow dependencies between interactions that cannot be collectively enforced by the participants without the addition of synchronization interactions. This property was reported in [12]. Verifying the absence of such anomalies in choreographies is beyond the scope of this paper and are also left to future work.

References

1. Business Process Modeling Notation (BPMN) Specification, Final Adopted Specification. Technical report, Object Management Group (OMG) (February 2006)
2. Barros, A., Dumas, M., ter Hofstede, A.: Service Interaction Patterns. In: van der Aalst, W.M.P., Benatallah, B., Casati, F., Curbera, F. (eds.) BPM 2005. LNCS, vol. 3649, pp. 302–318. Springer, Heidelberg (2005)
3. Decker, G., Kopp, O., Leymann, F., Weske, M.: BPEL4chor: Extending BPEL for Modeling Choreographies. In: Proceedings International Conference on Web Services (ICWS) (2007)
4. Decker, G., Overdick, H., Zaha, J.M.: On the Suitability of WS-CDL for Choreography Modeling. In: EMISA 2006. Proceedings of Methoden, Konzepte und Technologien für die Entwicklung von dienstebasierten Informationssystemen, Hamburg, Germany (October 2006)
5. Decker, G., Weske, M.: Local Enforceability in Interaction Petri Nets. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, Springer, Heidelberg (2007)
6. Desel, J., Esparza, J.: Free Choice Petri Nets. Cambridge Tracts in Theoretical Computer Science, vol. 40. Cambridge University Press, Cambridge (1995)
7. Dijkman, R.M., Dumas, M., Ouyang, C.: Formal semantics and automated analysis of BPMN process models. In: Preprint 7115. Queensland University of Technology, Brisbane, Australia (2007)
8. ITU-T. Message sequence chart. Recommendation Z.120, ITU-T (2000)
9. Kavantzas, N., Burdett, D., Ritzinger, G., Lafon, Y.: Web Services Choreography Description Language Version 1.0, W3C Candidate Recommendation. Technical report (November 2005), http://www.w3.org/TR/ws-cdl-10
10. Wohed, P., van der Aalst, W.M., Dumas, M., ter Hofstede, A., Russell, N.: On the Suitability of BPMN for Business Process Modelling. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102, Springer, Heidelberg (2006)
11. Zaha, J.M., Barros, A., Dumas, M., ter Hofstede, A.: A Language for Service Behavior Modeling. In: CoopIS 2006. Proceedings 14th International Conference on Cooperative Information Systems, Montpellier, France (November 2006)
12. Zaha, J.M., Dumas, M., ter Hofstede, A., Barros, A., Decker, G.: Service Interaction Modeling: Bridging Global and Local Views. In: EDOC 2006. Proceedings 10th IEEE International EDOC Conference, Hong Kong (October 2006)

CoBTx-Net: A Model for Reliability Verification of Collaborative Business Transaction

Haiyang Sun and Jian Yang

Department of Computing, Macquarie University
Sydney, NSW2109
{hsun,jian}@comp.mq.edu.au

Abstract. The collaborative business process can be unreliable when business partners collaborate in a peer-to-peer fashion without central control. Therefore, an important issue that needs to be dealt with for any generic solution to manage collaborative business transaction is reliability verification. In this paper, we propose a novel **Choreographical Business Transaction Net (CoBTx-Net)** to specify collaborative business transaction and manages the collaboration by individual participants. Furthermore three reliability properties named *Time-embedded dead marking freeness*, *Inter-organizational dead marking freeness*, and *Collaborative soundness* are defined based on CoBTx-Net to verify (1) the violation of time constraint, (2) collaborative logic conflicts, and (3) the improper termination from individual organizations.

Keywords: Collaborative Business Transaction, Reliability Verification, Peer Based Collaboration, Hierarchical Colored Petri Net(HCPN).

1 Introduction

Collaborative business transaction is about coordinating the flow of information among organizations and linking their business processes, and providing solutions to ensure the eventual generation of the consistent outcomes. Such business collaboration can be unreliable especially when it is conducted in a loosely coupled distributed environment such as web services [1].

Let us take an example in automotive industry. An ordering process is a business collaboration involving multiple business partners-customers, dealers and manufacturers, and communicating through the autonomous and heterogeneous business applications that include quote inquiry and purchase order process, etc. The ordering process begins with a quote inquiry broadcasting from a customer. After receiving the inquiry, dealers will validate the status of the customer. A quote will be returned if the customer has a valid status. Then the customer will choose a dealer who offers the best deal. The selected dealer will receive a purchase order from the customer. After checking the stock, the dealer will send the customer an order acknowledgement together with invoice and payment details. The dealer will deliver the car after receiving the payment.

This collaboration can become unreliable if one of the following happens:

- Violation of time constraint: If the dealer can not deliver the vehicles in time to the customer, the transaction may be suspended temporarily or may eventually lead to the cancellation from the customer.
- Conflict beliefs in collaboration logic: The customer may believe that only 10% deposit is needed to pay before receiving the vehicle, while the dealer requires that the customer pay the total prize before a vehicle can be delivered. Hence the dealer is waiting for another 90% money without any action on delivering the vehicle while the customer is expecting the delivery.
- Improper misbehavior by a participant: If any of the participants, i.e., customer, dealer, does not act as agreed, collaboration cannot proceed.

We can refer *reliability* in the context of business collaboration as that the collaboration is carried out as planned, each partner behaves as agreed, and the transaction completes at the right time and in the right place. As shown in the above example, collaborative transaction is more prone to unreliability than single processed transaction due to its peer based nature, i.e., participating partners have their own business policies and internal processes that are agnostic to each other. Furthermore, the proper completion of a business collaboration depends not only on its own correct behavior, but also the behaviors of the participants perceived through complex interactions.

Therefore we need properly model collaborative business transaction in the peer based execution environment so that its reliability can be verified and managed. The challenge we are facing lies in three facts: (1) We cannot predict and specify every possible situation in a collaboration, which may depend on the emerging situation during runtime, especially when faults and exceptions occur. Therefore it is impossible to pre-define control flow for collaboration; (2) On one hand, there will be no central controlled coordination in place. On the other hand, each participant need to know 'enough' runtime information about its partners to proceed; (3) Reliability properties need to be defined for collaboration so that they can be verified.

Petri Net provides a set of verification mechanisms[7], and its graphically and mathematically founded modeling formalism with various algorithms for design and analysis[2] makes it a good candidate for modeling business transactions. Especially the properties of the Petri Net, including *reachability*, *dead marking* etc, are well presented in the literature. These properties have already been used to verify the reliability of service composition and workflow[9, 11, 12, 13]. However these properties can not be applied directly to business collaboration because the semantics of the properties of Petri Net are not as same as those of reliability in business collaboration. For example, *reachability* as net property to describe the successful connection of the beginning and the ending of the net can not completely specify the transactional semantics concerning the proper termination of a collaboration, because the guarantee of the structural juncture is only part of requirements for achieving the collaboration goal.

In this paper we propose a **Choreographical Business Transaction Net (CoBTx-Net)** for modeling business collaboration, which is based upon

HCPN[3, 4]. Three properties for the purpose of reliability verification are defined based on CoBTx-Net: *Time-embedded dead marking freeness*, *Inter-organizational dead marking freeness*, and *Collaborative soundness*. Relative verification methods originated from approaches for verifying the execution of normal Petri Nets are also refined to cater for the characteristics of CoBTx-Net.

The rest of paper is organized as follows. In section 2, we introduce the specification of Choreographical Business Transaction Net(CoBTx-Net) model. The verification on the desired reliability properties based on CoBTx-net is presented in section 3, followed by the discussion on related work. A conclusion is presented in last section.

2 Specification of Business Collaboration Model

CoBTx-Net is developed from our individual **Business Transaction Net (BTx-Net)**. A BTx-Net[19] models the behavior of an individual participant in a business collaboration. A collaborating process in BTx-Net is separated into three layers as in Fig. 1, each of which is a subnet of BTx-Net. The *Execution (Exe) subnet*, *Abstract (Abs) subnet* and *Communication (Com) subnet* correspond to the generic stratified structure of web service in business collaboration in terms of internal business process, service interface, and business protocol [5]. They are linked by *Refinement function* acting on tokens at each subnet. BTx-Net can enforce organizational transaction consistency through the correlation of the tokens sent out and received, which will be discussed in detail in Section 2.1.

CoBTx-Net is an infrastructure that specifies the dynamic behaviors of business transaction at design time and manages the reliability at runtime. A CoBTx-Net is specified for individual participants to understand the behavior of their business partners. It consists of two components: the BTx-Net of its process and the publicly visible part of its collaborators.

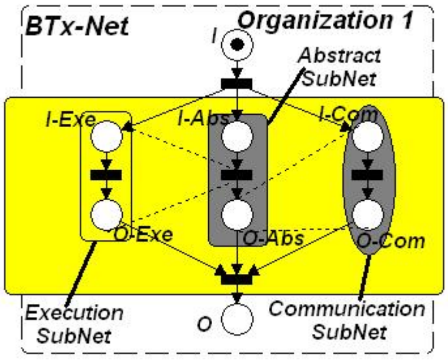


Fig. 1. BTx-Net

2.1 Specification of BTx-Net Model

BTx-Net models the business transaction execution in one organization. It has the following two features:

- It models the collaborating service in terms of three layers that are related through refinement.
- It separates the control flow from message flow so that the control flow can dynamically direct the message flow executing in business transaction according to the properties of the message at runtime, and detect and prevent the advent of the unexpected messages by correlating the elements of control flow and message flow. The properties of the messages are defined based on organization policies and business rules.

The Structure of BTx-Net. As mentioned before, there are three levels in BTx-Net, each of which is a subnet of the BTx-Net as shown in Fig.1:

- At execution level, internal business tasks are formed as an *Exe-subnet*.
- At abstract level, the input/output messages of the operations defined for the service, and the control ability that can transfer messages to and from other subnets form an *Abs-subnet*.
- At communication level, different communication patterns such as request-response, notify, form a *Com-subnet*.

T1he Execution Policy of BTx-Net. There are two types of tokens that are operated within a BTx-Net: the Application-Oriented Token (AO-Token) and the Management-Oriented Token (MO-Token), which movements correspond to the message flow and the control flow respectively. Each MO-token is correlated to a specific AO-token. Here we explain the firing rules for the movement of tokens of the model:

- Token movement at individual level: An AO-Token can move in each level ***iff*** the correlated MO-Token in each level exists and moves with it.
- Cross-level token movement: Every MO-Token is split into each level at the beginning of the business transaction and converged in the end. It indicates the *successful* execution of the transaction in each subnet and that the whole business transaction is successfully terminated. Such split MO-Token can only move within the specific level at runtime. However, the AO-Token moves between levels according to the *refinement functions* defined in BTx-Net.
- Cross-organization token movement: the MO-Token can not move out of the organization, which follows the principle of peer-to-peer collaboration, i.e., no central control exists. However the AO-Token can be exchanged between organizations for the purpose of business collaboration. The MO-Token will be unbound from the correlated AO-Token when it moves out of the organization and bound with it again when the correlated AO-Token returns.

The Advantage of BTx-Net. Now we can explain how consistency can be enforced for business transaction through token movements in the stratified service infrastructure based on the control of intra- and inter-organizational message transferring:

- For the inter-organizational message transferring, the MO-Tokens ensure that the returned AO-Token from other parties is what the organization is expecting. For example, it is not acceptable that the dealer receives the payment message as an AO-Token before accepting the order, because the relevant management token still resides in the *accept order service*, which is a service shall be executed before the *payment service*. Therefore the payment message will be abandoned as an unexpected message. There exists two types of interactions:
  1. For synchronous interaction, the MO-Token will wait for the return of the corresponding AO-Token and examine it to see if it is what the MO-Token expects.
  2. for the asynchronous interaction, the MO-Token needs to examine the AO-Token from asynchronous interaction based on the stored information on previous matched AO-Token. For example the MO-Token needs to correlate the received invoice to the matching purchase order.
- For intra-organizational message movement, the firing rules defined based on organization polices can guarantee that the tasks or operations associated with the AO-tokens movement within and between levels are executed as expected. For example it is impossible to send order final acknowledgement through the communication level for a service when the organization is still processing the order in the order feedback service to decide whether the order should be accepted or not.

Definition 1. *A BTx-Net in one organization* G_1 *is a tuple* $N_{G1}=(P_{G1}, T_{G1}, F_{G1}, \Pi_{G1}, II_{G1}, IO_{G1})$*, Where:*

- P_{G1} *is a set of place graphically represented as circle.* P_{G1}^{Exe}, P_{G1}^{Abs}, *and* P_{G1}^{Com} *are sets of places at each subnet.* $P_{G1}=P_{G1}^{Exe}\cup P_{G1}^{Abs}\cup P_{G1}^{Com}$, *where* $P_{G1}^{Exe}\cap P_{G1}^{Abs}\cap P_{G1}^{Com} = NULL$.
- T_{G1} *is a set of transitions graphically represented as dark bar in Fig. 1, where:* T_{G1}^{Exe}, T_{G1}^{Abs} *and* T_{G1}^{Com} *are sets of transitions at each level.* T_{G1}^{τ} *is set of empty transitions for transferring, distributing, and collecting tokens.* $T = T_{G1}^{Exe}\cup T_{G1}^{Abs}\cup T_{G1}^{Com}\cup T_{G1}^{\tau}$, *where* $T_{G1}^{Exe}\cap T_{G1}^{Abs}\cap T_{G1}^{Com}\cap T_{G1}^{\tau} = NULL$, *and* $P_{G1}\cap T_{G1} = NULL$.
- $F_{G1}=(P_{G1}\times V\times T_{G1})\cup(T_{G1}\times V\times P_{G1})$ *is the flow relation between places and transitions, where* V *is the sets of variables* $V=\{x,y,...\}$ *to represent the tokens.*
- Π_{G1} *is a group of functions at the three subnets of a collaborating service, which includes refinement function, MO-Token exam function, and Colored token map function etc. See [19] for the details of the functions in each subnet.*
- II_{G1}, IO_{G1} *are the sets of in and out places of BTx-Net and their subnets including* $II_{G1}=\{I_{G1}, i_{G1}^{Exe}, i_{G1}^{Abs}, i_{G1}^{Com}\}$, *and* $IO_{G1}=\{O_{G1}, o_{G1}^{Exe}\ o_{G1}^{Abs}\ o_{G1}^{Com}\}$

2.2 Specification of CoBTx-Net

In order to model business collaboration without central control, and provide exclusive methods to verify the reliability of the structure and behavior of the collaborative business transaction, we develop a **CoBTx-Net** model.

The Structure of CoBTx-Net. Now a CoBTx-Net is consisted of five subnets (see Fig. 2). Besides the original three subnets of the BTx-Net, another two subnets as the perceived public part of the BTx-Net of its collaborating partners are included. It has the following additional features compared to a BTx-Net: (1) ExCom-net is a Com-net in the BTx-Net of the collaborating partners which interacts with the organization through links between the Com- and ExCom-nets to implement the business interactions. (2) ExAbs-net represents the public interfaces of the BTx-Net in the collaborating partners which indicates required service(s) for collaboration.

The Execution Policy of CoBTx-Net. The AO-Token will be the only token permitted to transfer out of organizations. Nevertheless, the movement of AO-Token within organization can be controlled by the refinement function resided in each transition or place. When AO-Token is operated out of an organizational boundary, there is no function applied on them since the AO-Token is controlled by the collaborating organization at the time as message is processed. The organization can only observe the movement of AO-tokens on these public interfaces of other organizations.

Another token named ExMO-token is introduced which is operated on the public parts of the collaborating organizations. Unlike the MO-Token(called InMO-Token in CoBTx-Net) in BTx-net which represents the control scope of the organization on its business process, the ExMO-Token only presents the observation that the organization has on its collaborating participants.

Now we can define the firing rules of a CoBTx-Net as:

- The token movements within organization are same as their movements in BTx-Net.
- The token movements out of organizational boundary are as follows:
  1. The AO-Tokens can move outside the organization to the public part of its collaborating participants.
  2. The ExMO-Token is operated at ExCom-net and ExAbs-net respectively, and indicates the sequence of the execution in the public parts of the BTx-Nets of the collaborating participants. The AO-token will move associated with the specific ExMO-Token by correlation which is similar to the binding of MO-Token and AO-Token in BTx-Net. However, no function is used to control their movements which is merely an observation of their movements from the organization point of view.

The Advantages of CoBTx-Net. The CoBTx-Net elaborates an infrastructure to manage the whole business transaction from individual organization point of view: (1) The movements of ExMO-Token let an organization observe the public behavior of the collaborating organizations and guarantee that the correct services are activated to transfer AO-Tokens between organizations. (2) The inclusion of business interactions in the model of CoBTx-Net makes it possible to detect the collaborative logic conflict so that deadlock can be avoided. (3) Adding the public collaborating view part to the organization transaction model facilitates the possible solutions for the improper termination from the collaborating organization.

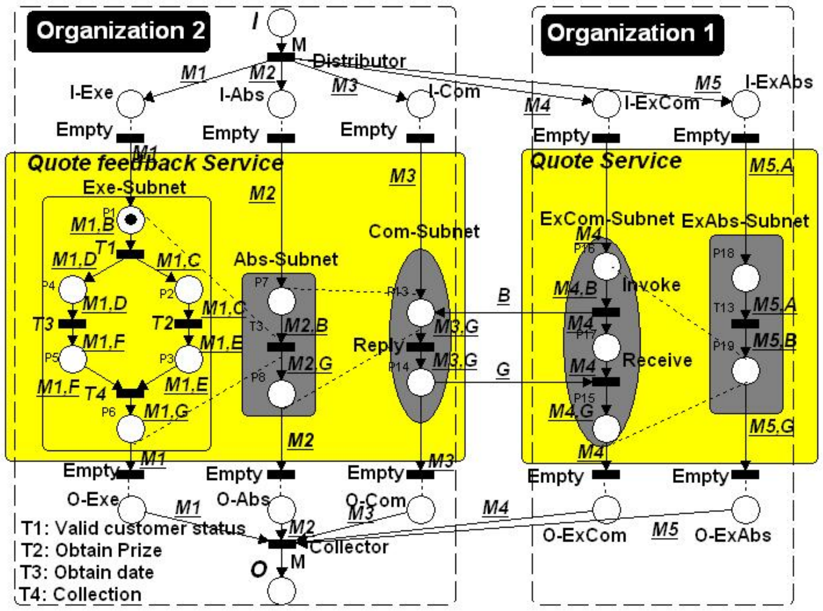


Fig. 2. CoBTx-Net of Organization 2

In Fig. 2, we present a CoBTx-Net of organization 2 (a dealer) to answer the *Quote Request* from Organization 1 (a customer). Other services are omitted in this diagram due to space limit. The CoBTx-Net of dealer is composed by two components, the BTx-Net of dealer and public part of BTx-Net of customer. Refinement function is used to link each subnet for the movement of AO-Token, such as B, D,etc. MO-Token is split into five parts firstly as M_1 to M_5 and moved only in each specific subnet. Finally they will converge at the end of the net as the successful termination of the whole transaction.

Definition 2. *CoBTx-Net for organization* G_1 *is a tuple* $CoN_{G1} = (P_{G1}^{CoN}, T_{G1}^{CoN}, F_{G1}^{CoN}, \Pi_{G1}^{CoN}, II_{G1}^{CoN}, IO_{G1}^{CoN})$*, where*

– $P_{G1}^{CoN} = P_{G1} \cup P_{Gi}^{Com} \cup P_{Gi}^{Abs}$ *where*

$$P_{Gi}^{Com} = \begin{cases} \theta(T_{G1}^{Com}) & if\ (P_{Gi}^{Com}, T_{G1}^{Com})^* \in F_{G1}^{CoN} \\ \bullet\theta(P_{G1}^{Com}) = \bullet T_{Gi}^{Com} & if\ (\bullet T_{Gi}^{Com}, T_{G1}^{Com})^* \in F_{G1}^{CoN} \\ \theta(P_{G1}^{Com})\bullet = T_{Gi}^{Com}\bullet & if\ (T_{Gi}^{Com}\bullet, T_{G1}^{Com})^* \in F_{G1}^{CoN} \end{cases}$$

$and \quad P_{Gi}^{Abs} = \rho(P_{Gi}^{Com}),$

- $T_{G1}^{CoN} = T_{G1} \cup T_{Gi}^{Com} \cup T_{Gi}^{Abs}$ *where*

$$T_{Gi}^{Com} = \begin{cases} \theta(P_{G1}^{Com}) & if\ (P_{G1}^{Com}, T_{Gi}^{Com})^* \in F_{G1}^{CoN} \\ \bullet\theta(T_{G1}^{Com}) = \bullet P_{Gi}^{Com} & if\ (T_{G1}^{Com}, \bullet T_{Gi}^{Com})^* \in F_{G1}^{CoN} \\ \theta(T_{G1}^{Com})\bullet = P_{Gi}^{Com}\bullet & if\ (T_{Gi}^{Com}\bullet, T_{G1}^{Com})^* \in F_{G1}^{CoN} \end{cases}$$

$$and \quad T_{Gi}^{Abs} = \begin{cases} \bullet\rho(\bullet T_{Gi}^{Com}) \\ \bullet\rho(T_{Gi}^{Com}\bullet) \\ \rho(T_{Gi}^{Com}\bullet)\bullet \\ \rho(\bullet T_{Gi}^{Com})\bullet \end{cases},$$

- $F_{G1}^{CoN} = F_{G1} \cup (T_{G1}^{Com}, P_{Gi}^{Com})^* \cup (P_{G1}^{Com}, T_{Gi}^{Com})^* \cup F_{Gi}^{\{Com,Abs\}}$,
- $\Pi_{G1}^{CoN} = \Pi_{G1} \cup \Pi_{Gi}^{\{Com,Abs\}}$,
- $II_{G1}^{CoN} = II_{G1} \cup II_{Gi}^{\{Com,Abs\}}$ *and* $IO_{G1}^{CoN} = IO_{G1} \cup IO_{Gi}^{\{Com,Abs\}}$,

In definition 2, we define several functions and annotations to construct a CoBTx-Net model from BTx-Net:

- θ:$p_1 \rightarrow t_i$ or θ:$t_1 \rightarrow p_i$. By operating the function, we can get the transitions or places of the public parts of the partners' BTx-Net.
- ρ:$P^{Com} \rightarrow P^{Abs}$. We can always obtain the P^{Abs} if we can identify relative P^{Com} by function $\rho()$.
- $(t, p)*$ or $(p, t)*$=(p×t)∪(t×p). It represents the flow relation between places and transitions
- $\bullet t$={y∈P|(y,x)∈F ∩ x∈T}, $t\bullet$={y∈P|(x,y)∈F ∩ x∈T}. $\bullet t$ and $t\bullet$ indicate the pre-places and post-places of a transition respectively.
  $\bullet p$={x∈T|(x,y)∈F ∩ y∈P}, $p\bullet$={x∈T|(y,x)∈F ∩ y∈P}. $\bullet p$ and $p\bullet$ illustrate the pre-transitions and post-transitions of a place respectively.

3 Verify CoBTx-Net for Business Collaboration Reliability

In this section, we will introduce a set of properties used for reliability verification for business collaboration based on CoBTx-Net.

3.1 Time-Embedded Dead Marking Freeness

The *labeled transitive matrix* used in Petri-Net expresses the relationship between $\bullet t$ and $t\bullet$ based on transition t. However, it does not elaborate the time relationship between $\bullet t$ and $t\bullet$. We extend the transitive matrix by associating it with the time impact called *Labeled time-embedded transitive matrix* L_{BP}^{*t} and use it to detect the property named *time-embedded dead marking freeness.* The primitives of the method is analogous to the verification method in [6]. The variation lies in the fact that the novel detect equation on time constraints: $M_k^{*t} = M_k \cdot L_{BP}^{*t}$ is included, where the M_k^{*t} is called *timed marking* and M_k is the ordinary marking. We can draw a conclusion that the business transaction is executing without violation on any time constraints, if the CoBTx-Net is Time-embedded dead marking free. It means that given time Ti there always exists transition t which is executing and satisfying any time constraint of itself.

This approach can be used to detect the time issue at design time if the time constraint of each transition can be predicted, including the transition in the view part such as at ExCom- and ExAbs-net. Meanwhile, it also can be implemented to verify the time issue during runtime, such as that an activity is suspended longer than expected.

3.2 Inter-organizational Dead Marking Freeness

The *Collaborative Logic Conflict(CLC)* presents a type of deadlock, in which each involving organization requires feedback from other organizations and eventually a cyclic chain of feedback requests is generated and each organization is stepping into a suspending status to wait for feedback from each other. Since services are autonomous, normally detecting CLC from an individual organizational view is not as easy as examining CLC from global view. However, due to the presence of the public part of the partners in the individual organization's CoBTx-Net, we can verify CLC in organization's CoBTx-Net by introducing a new property: *Inter-organizational Dead Marking Freeness.*

Lemma 1. *A CoBTx-Net is inter-organizational dead marking free:*
(1) If there are no dead markings in the organization's CoBTx-Net and its collaborating organizations' CoBTx-Nets simultaneously.
(2) If there exists dead marking in each CoBTx-net simultaneously, but there are no cyclic feedback requests from dead marking organizations to others that are also stepping into dead marking status.
(3) If the cyclic feedback requests exist, but there is no dependency between the service unit requiring feedback from others and the service unit required feedback by others in one organization.

The *lemma 1* clearly provides the conditions for the CoBTx-Net to detect the inter-organizational dead marking. Based on the lemma 1, we introduce the concept of *Notify-Token* that passes from organization's CoBTx-Net which suffers dead marking to the collaborating ones to detect such cyclic requests. If the cyclic request exists, the CoBTx-Net is in Inter-organizational dead marking.

The method can be used to detect CLC in design time as well as runtime. In the design time, the CLC can be generated by orthogonal transaction policies between organizations. In the runtime, the CLC may occur when exceptions or faults happen. Due to the space limit, we only introduce a method to detect the CLC from individual organizational point of view.

3.3 Collaborative Soundness

In order to detect the organizational improper termination, we must identify the conditions on which a CoBTx-Net terminates properly. Here we introduce a property named *collaborative soundness* to evaluate the termination condition of CoBTx-Net.

Lemma 2. *A CoBTx-Net is collaboratively sound iff:*
(1) The CoBTx-Net is reachable from the beginning place **i** *to the ending one* **o***,*
(2) Only one ending place **o** *exists and the token deposited at the beginning place* **i** *will finally return to the ending place,*
(3) No active transition (a transition still processes and deposits token(s) to the post-place(s)) at Com-net, Abs-net, and Exe-net of CoBTx-Net when the MO-Token resides in ending place **o***.*

Lemma 2 specifies the conditions for CoBTx-Net to verify the property of collaborative soundness. How they relate to the organization's proper termination is discussed as follows: in (1), the lemma stresses that a reachable path from beginning to ending is necessary for an organization to execute the collaboration. Condition (2) specifies that the process must start from the beginning place and terminate at the only ending place. All other situations can not be acceptable. In (3), the lemma states that no active transition is allowed in the part of the net that belongs to the specified organization when the organization has already terminated its processes. However, active transition is permitted in the public view part of its collaborating partners when the organization's processes are terminated. The MO-Token residing in the ending place ***o*** as the termination of the organization means that the left case in the view part will not have any impact on the termination of the business transaction in this organization.

4 Related Work

Research has been done in the area of business transaction reliability verification. Petri Net is a widely used technique for business process modeling and verification. We shall look into some of the representative work in the area.

In [9], the authors introduce PNML (Petri Net Markup Language) to transfer the composite service into Petri Net model and implement algorithm on the model to verify the reliability of composition. However, the authors did not take into account the interactions between services of different organizations. As a result they can not provide methods for detecting the properties with business collaboration semantics. In [11], the authors provide a model based on Hierarchical CPN to detect the reliability issues of web service workflow. However the model is constructed from a centralized global view which includes all the detailed information of participants. This assumption can not be held in the peer-to-peer loosely coupled business collaboration environment. In [12], the authors construct a verification framework for web service composition. *protocol conformance* as the requirement of business interaction is presented to check the correlation of the complex conversations among multiple organizations. However, the proposed model is too abstract to handle the business collaboration verification without including the internal business process details. The authors in [13] introduce a property *soundness* for verifying the reliability of WF-Net, a Petri-net based workflow model. However without the presence of the cross-organizational service interaction, the work can not be used by individual organizations for managing the reliability of business collaboration.

Several standards in SOC are also presented for describing and managing collaborative business transaction, some of which can be transferred into Petri Nets model for verification as well [8]. BPEL4WS [10] with WS-C [14] and WS-T that includes Atomic Transaction[15] and Business Activity [16] provides some basic support for coordination and exception handling. BPEL4WS is a business process language that orchestrates services. WS-C and WS-T are two associated protocols to provide coordination and transaction support for BPEL4WS. WSCI [17] is a protocol focus on the service choreography, which defines the message sequence interaction and uses *connecter* to link services across organizational boundary. WS-Reliability [18] is another protocol to guarantee the message delivery and order. All above SOC standards provide transactional support from certain aspects, especially in coordination. However, our work specifically focuses on reliability verification of collaborative business transaction. We believe our work can be incorporated into WS standards such as WS-Transaction.

We need models to specify collaborative business transaction as well as mechanisms to detect unreliable problems caused by design errors, exceptions or faults in runtime in the context of SOC before solutions can be provided. However:

1. The existing model for managing business transaction is lacking support in:
   - describing service interactions (not composition),
   - an integrated peer organizational view on its internal business process, service interface, and protocol as well as the interfaces and protocols of its collaborating partners.
   - and their relationships in terms of message flows and control flows.
2. The existing approaches associated with their verifying properties are lack of collaborative transactional semantics support.

Therefore, CoBTx-Net as a model for individual peer organization is then constructed with the corresponding reliability verification methods.

5 Conclusion and Future Work

Collaborative business transactions are prone to unreliability since they are normally executed in loosely-coupled environment. In this paper, we propose a CoBTx-Net model based on Hierarchical CPN as a platform to verify reliability of business collaboration. CoBTx-Net takes the loosely-coupled environments and peer based collaboration into account. Based on CoBTx-Net, three reliability properties defined for verification purpose are introduced to verify the collaborative business transaction. In the future, detailed algorithms on detecting such reliability properties will be developed.

References

[1] Alonso, G., Casati, F., Kuno, H., Machiraju, V.: Web Service: Concept, Architectures and Applications. Springer, Heidelberg (2004)

[2] Girault, C., Valk, R.: Petri Nets for System Engeneering:A Guide to Modeling, Verification and Application. Springer, German (2003)

[3] Huber, P., Jensen, K., Shapiro, R.M.: Hierarchies in Colored Petri nets. In: Advances in Petri Nets 1990, pp. 313–341 (1991)
[4] Stork, D.G., van Glabbeek, R.J.: Token-controlled Place Refinement in Hierarchical Petri Nets with Application Active Document Workflow, In: The Int. Conf. on Application and Theory in Petri Nets, pp. 394–413 (2002)
[5] Newcomer, E., Lomow, G.: Understanding SOA with Web Service, Pearson Education, USA (2005)
[6] Song, Y., Lee, J.: Deadlock Analysis of Petri Nets Using the Transitive Matrix. In: The SICE Annual Conf., pp. 689–694 (2002)
[7] Murata, T.: Petri Nets:Properties,Analysis and Application. In: Proceedings of the IEEE, vol. 77(4), pp. 541–580. IEEE, USA (1989)
[8] Hinz, S., Schmidt, K., Stahl, C.: Transforming BPEL to Petri Nets. In: The Intl. Conf. on Business Process Management (2005)
[9] Chi, Y., Tsai, M., Lee, C.: A Petri-net Based Validator in Reliability of a Composite Service. In: The IEEE Int. Conf. on e-Technology, e-Commerce and e-Service, pp. 450–453 (2005)
[10] Jordan, D., et al.: Business Process Execution Language for Web Service (BPEL4WS) 2.0 (August 2006), http://docs.oasis-open.org/wsbpel/2.0/
[11] Yang, Y., Tan, Q., Xiao, Y., Yu, J., Liu, F.: Exploiting Hirachichical CP-Nets to Increase the Reliability of Web Services Workflow. In: The Symposium on Application and the Internet (2006)
[12] Yi, X., Kochut, K.J.: A CP-nets-based Design and Verification Framework for Web Services Composition. In: The IEEE Int. Conf. on Web Services (2004)
[13] van der Aalst, W.M.P.: Verification of workflow nets. In: The Int. Conf. on Application and Theory in Petri Nets, pp. 407–426 (1997)
[14] Newcomer, E., et al.: Web Services Coordination 1.1 (WS-Coordination) (August 2006), http://docs.oasis-open.org/ws-tx/wstx-wscoor-1.1-spec-pr-01.pdf
[15] Newcomer, E., et al.: Web Services Atomic Transaction (WS- Atomic Transaction) (August 2006), http://docs.oasis-open.org/ws-tx/wstx-wsat-1.1-spec-pr-01.pdf
[16] Newcomer, E., et al.: Web Services Business Activity (WS-Business Activity) (November 2006), http://docs.oasis-open.org/ws-tx/wstx-wsba-1.1-spec-pr-01.pdf
[17] Arkin, A., et al.: Web Service Choreography Interface (WSCI) 1.0 (August 2002), http://www.w3.org/TP/wsci/
[18] Iwasa, K.: Web Services Reliable Messaging 1.1 (WS-Reliability) (November 2004), http://docs.oasis-open.org/wsrm/ws-reliability/v1.1
[19] Sun, H., Yang, J.: A Token Based Dynamic Model for Supporting Consistent Collaborative Business Transactions. In: The IEEE Int. Conf. on Service Computing (2007)

Towards Analysis of Flexible and Collaborative Workflow Using Recursive ECATNets

Kamel Barkaoui and Awatef Hicheur

CEDRIC-CNAM 292, Rue Saint-Martin Paris 75003 France
{barkaoui,hicheur}@cnam.fr

Abstract. In this paper we define a model, namely the recursive ECATNets (Recatnets) based on a sound combination of Extended Concurrent Algebraic Term nets and Recursive Petri nets, allowing a concise modeling of dynamic reconfiguration mechanisms of flexible workflow processes. The descriptive power of Recatnes is well-suited for collaborative workflows modeling. Moreover the use of model checking techniques to prove liveness and safety properties becomes possible due to their semantics defined in conditional rewriting logic.

Keywords: Recursive Petri nets, Flexible collaborative workflow, Rewriting logic.

1 Introduction

Over the last decade, the workflow management systems (WfMS for short) have been increasingly adopted by most organizations for automating, monitoring and improving their critical business processes [1]. A workflow schema is used to represent the structure of a business process in terms of tasks as well as temporal and data dependencies between tasks. Numerous WfMSs are available in the marketplace, but, some limitations have been encountered in their design. Among these limitations, one can note that the workflow schemas used by many WfMSs are often not formally specified which makes it hard to test the correctness of theses workflow definitions before putting them into production (e.g. we need to be able to check that the workflow eventually terminates). Moreover, only few of theses systems provide efficient ways of managing workflows which require dynamic adaptation of their structure at the occurrence of exceptional situations and failures [2], [3]. We refer to workflow flexibility as the ability to dynamically reconfigure workflow schemas (i.e. to create, to extend or to suppress (sub)processes in a structured way) during their execution [3], [4]. More recently, the need for workflow flexibility is particularly enhanced by the current trend of business globalisation where multiple parties belonging to different enterprises are involved in a business process. The modeling and managing of these collaborative business processes that span multiple organisations brings new challenges regarding the decentralization and the flexibility [5],[6]. On one hand, these processes are not executed by a centralized workflow engine, but by multiple workflow engines collaboratively (within the same organization or over multiple organisations). On the other hand, the structures of these processes are extremely dynamic, driven by external conditions, changing user requirements and business partners. If we want to describe, in a faithful manner, flexible workflow processes (i.e. workflows that can adapt their structure during their execution), we need modeling formalisms which offer mechanises allowing us to manage dynamic structural changes. Moreover, the particularity of collaborative business processes enhance the need for modeling notation able to describe and to integrate in an adequate way both the execution order of the process activities (i.e. the control flow perspective) and the distributed execution of the process over collaborative partners and their inter-process interaction (i.e. the organisational perspective). In this paper, we extend the description power of the basic recursive ECATNet model (RECATNets)[7], in order to cope with workflow flexibility problem. This model offers practical mechanisms for a direct and intuitive support of dynamic creation and suppression of processes. This ability allows to model in a concise way workflow processes with a dynamic structure and so to introduce a correct flexibility description in workflow planning and execution (e.g. alternate planning, modification of planning and execution of workflow processes). The descriptive power of recursive ECATNets is relevant in modeling collaborative business processes. Indeed, a current state of a concurrent system is expressed in RECATNet as a dynamical tree of threads where each thread has its own execution context. This dynamical hierarchical structure is well suited to describe the hierarchy of collaboration between subprocesses within an organisation or across multiple organizations and their respective execution environments. Recursive ECATNets model is defined on the basis of a sound combination of the classical ECATNets formalism [8], [9] and the recursive Petri nets (RPNs) [10]. We remind that ECATNets (Extended Concurrent Algebraic Term Nets) are a kind of algebraic nets which allow to describe, in a compact manner, complex systems characterized by synchronisation constraints and algebraic abstract types for specifying the data structures. The ECATNets concurreny semantics [8] is expressed in terms of rewriting logic [11], giving them a solid mathematical foundation. The RPNs, for their part, are introduced as a strict extension of the ordinary Petri nets. They offer a practical *recursive* mechanism to model the dynamic creation of processes and their synchronization. The rest of this paper is organized as follows: In Section 2 we introduce the RECATNets model. Then, we illustrate how RECATNets can be used in modeling collaborative workflow processes with a dynamic structure. In section 3, we show how the semantics of RECATNets can be expressed naturally in terms of conditional rewriting logic. Finally, section 4 concludes the paper.

2 Workflow Modeling Based on Recursive ECATNets

2.1 ECATNets Review

An ECATNet is a high level net $\mathcal{E} = (Spec, P, T, sort, IC, DT, CT, Cap, TC)$ where: $Spec = (\Sigma, E)$ is an algebraic specification of an abstract data type given by the user (with E its set of equations and Σ the set of operations and sorts) and in which places marking are multisets of Σ-terms [8], [9]. Note that $T_{\Sigma,E}(X)$ is the Σ-algebra of the equivalence classes of the Σ-terms with variables in X, modulo the equations E. $MT_{\Sigma,E}(X)$ represents the free commutative monoid of the terms $T_{\Sigma,E}(X)$ endowed with the internal operator $\oplus$ and having $\varnothing$ as the identity element. CATdas(E, X) is the structure of equivalence classes formed from the multisets of $MT_{\Sigma,E}(X)$ modulo the associative, commutative and identity axioms for the operator $\oplus$ (See Fig. 1).

- P is a finite set of places; T is a finite set of transitions (with $P \cap T = \varnothing$).
- *sort* : $P \rightarrow S$ (with S the set of sorts of *Spec*);
- *Cap*: $P \rightarrow$ CATdas(E,$\varnothing$), (Places Capacity);
- *IC*: $P \times T \rightarrow$ CATdas(E,X)*, (Input Condition) such that CATdas(E,X)* = $\{\alpha^+ / \alpha \in$ CATdas(E,X) $\} \cup \{\alpha^- / \alpha \in$ CATdas(E,X)$\} \cup \{\alpha^0 / \alpha \in$ CATdas(E,X)$\} \cup \{\alpha_1 \wedge \alpha_2 / \forall i\ \alpha i \in$ CATdas(E,X)$^*\} \cup \{\alpha_1 \vee \alpha_2 / \forall i\ \alpha i \in$ CATdas(E,X)$^*\}$. For a given transition t (Fig. 1), the expression $IC(p, t)$ specifies conditions on the marking of the input place p for the enabling of t. It takes one of the following form (Table1):

Table 1. The different forms of the expression $IC(p, t)$ for a given transition t

IC(p, t)	Enabling condition
α^0	The marking of the input place p must be equal to α ($IC(p, t) = \varnothing^0$ means the marking of p must be empty)
α^+	The marking of the place p must include α ($IC(p, t) = \varnothing^+$ means condition is always satisfied)
α^-	The marking of the place p must not include α (with $\alpha \neq \varnothing$)
$\alpha_1 \wedge \alpha_2$	conditions α_1 and α_2 are both true
$\alpha_1 \vee \alpha_2$	α_1 or α_2 is true

- *DT*: $P \times T \rightarrow$ CATdas(E, X), (Destroyed Tokens); the expression $DT(p, t)$ specifies the multiset of tokens to be removed from the marking of the input place p when t is fired. It is obvious that the multiset $DT(p, t)$ must be included in the marking of p (i.e. $DT(p, t) \subseteq IC(p, t)$). For notation convenience, $IC(p, t)$ (or $DT(p, t)$) is omitted in the graphical representation of ECATNets, when $IC(p, t) = DT(p, t)$.
- *CT*: $P \times T \rightarrow$ CATdas(E, X), (Created Tokens); the expression $CT(p', t)$ specifies the multiset of tokens to be created in the output place p', when t is fired.
- *TC*: $T \rightarrow$ CATdas(E, X)$_{\text{bool}}$, (Transition Condition); the expression $TC(t)$ is a boolean term which specifies an additional enabling condition for the transition t. This condition is not expressed as a term which has to be included or not included in the marking of one of the input places of t but it specifies some conditions on the values taken by local variables of t (variables related to the all input places of t). Note that when $TC(t)$ is omitted, the default value is the term *True*.

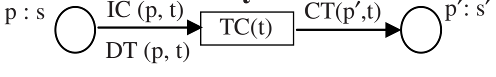


Fig. 1. A generic representation of an ECATNet

An interesting feature of ECATNets is that there is a clear distinction between the firing condition of a given transition *t* and the tokens which may be destroyed during the firing action of *t* (respectively specified via the expression IC(p, t) and DT(p, t)). A transition *t* is fireable when several *conditions* are satisfied simultaneously:

1) Every *IC*(*p*, *t*) is satisfied for each input place *p* of *t*.
2) The transition condition *TC*(*t*) is true.
3) The addition of the tokens *CT*(*p*′, *t*) to the output place *p*′ of *t* must not result in *p*′ exceeding its capacity when this capacity is finite.

When t is fired, DT(p, t) is removed from the input place p and simultaneously CT(p′, t) is added to the output place p′.

The ability of transitions in ECATNet to check for context conditions (positive contextual conditions i.e. read arcs and negative contextual conditions i.e. inhibitor arcs) gives a high power of description. One can easily model concurrent processes managing advanced data structures via the algebraic specifications. However, due to their static structure, the ECATNets are not adequate for the modeling of processes which structures are reconfigurable as it is required in the context of flexible and collaborative workflow systems. In order to deal with this limitation, we have extended the ECATNet formalism with the *recursion* concept [10].

2.2 Recursive ECATNets

A recursive ECATNet has the same structure as an ordinary ECATNet except that the transitions are partitioned into two categories: *abstract* transitions (represented by a double border rectangle, see Fig. 2 and *elementary* transitions (see Fig. 3).

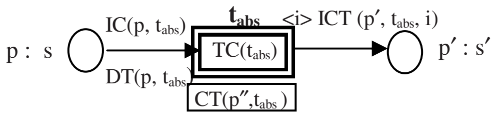


Fig. 2. A generic abstract transition

In a RECATNet a transition *t* (elementary or abstract), is fireable when several conditions are satisfied simultaneously: (1) every *IC(p, t)* is satisfied for each input place *p* of the transition *t* and (2) the condition *TC(t)* is true. In contrast with ordinary ECATNets, the execution of a recursive ECATNet generates a *dynamical* tree of threads (denoting the fatherhood relation) where each of these threads has its own activity. One can note that all the threads of this tree can be executed simultaneously.

- When a thread fires an abstract transition t_{abs}, it consumes the multiset of tokens *DT(p,* t_{abs}*)* from the input place *p* and simultaneously it *creates* a new thread child which starts its execution with the initial marking (indicated in a frame) associated to this abstract transition. Note that for the enabling of an abstract transition t_{abs}, the addition of tokens specified by the initial marking to the places of the newly created thread must not result in one of theses places exceeding its capacity (when this capacity is finite).

- A family of boolean terms Υ is defined and associated to a RECATNet in order to describe the *termination conditions* (i.e. final markings) of the created threads. This family is indexed by a finite set whose items are called *termination indexes*. The set *I* of the termination indexes is simply deduced from the enumeration of all the defined final markings. So, if a thread reaches a final marking Υi (with $i \in I$), it *terminates* aborting its whole descent of threads. Then, it produces (in the token game of its father) and for the abstract transition t_{abs} which gave birth to it, the multiset of tokens $ICT(p', t_{abs}, i)$ in the output place p' of t_{abs}. Such a firing is called a *cut step* and denoted τ_i (with $i \in I$). Therefore, the production of tokens in the output places of an abstract transition is delayed until the thread child, generated by the firing of this transition, reaches a final marking. An arc from an abstract transition t_{abs} to its output place p' is labeled by the following algebraic expression: $<i>$ $ICT(p', t_{abs}, i)$ which means that the produced tokens $ICT(p', t_{abs}, i)$ depend on the final marking Υi reached in the terminating thread child (<i> is the index of this termination). Such an arc can be omitted if the term $ICT(p', t_{abs}, i)$ is null (i.e. an empty marking). Note that if a cut step occurs in the root of the tree of threads, it leads to an empty tree denoted $\perp$.

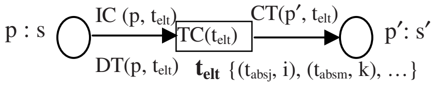


Fig. 3. A generic elementary transition

- The behavior of an elementary transition t_{elt} is extended (in this version of the model) and depends on a partial function *K* which associates to it a set of abstract transitions to interrupt and for each of these transitions a termination index. In the graphical representation of a RECATNet, the name of an elementary transition t_{elt} is followed by the set $K(t_{elt})$ when this set in non empty (in Fig. 4, $K(t_{elt}) = \{(t_{absj}, i), (t_{absm}, k), \ldots\}$). Basically, if a thread fires an elementary transition $t_{elt,}$ it updates its internal marking as a transition of ordinary ECATNets. Moreover, if the function k is defined, the firing of this elementary transition performs the appropriate cut step to each subtree generated by the abstract transitions specified by *K*. So, all threads which are generated by one of the abstract transitions specified by *K* are aborted and, depending of the termination index associated to it, the output tokens of these abstract transitions are produced in the thread where the firing takes place.

Definition (Recursive ECATNets). A Recursive ECATNet is a tuple *RECATNet = (ECATNet, I, Υ, K, ICT)* where :

- $T = T_{abs} \cup T_{elt}$ ($\cup$ denotes the disjoint union) is the set of RECATNets transitions, partitioned into abstract and elementary ones;
- *I* is a finite set of indexes; Υ is a family, indexed by *I*, of boolean terms defined in order to describe the termination conditions of threads. These conditions can be specified by a system of linear inequalities or equalities on the places marking. In order to obtain decidability results, we require that *Spec* = (Σ, *E*) is a many sorted

algebra with finite number of sorts. So, determining the truth value of a termination condition becomes decidable.

- $K : T_{elt} \rightarrow T_{abs} \times I$, is a partial function which associates to an elementary transition the set of interrupted abstract transitions and their associated termination index.
- ICT: $P \times T_{abs} \times I \rightarrow$ CATdas(E, X), (Indexed Created Tokens). The created tokens in the output places of an abstract transition.

2.3 Modeling Flexible Workflow Processes Using RECATNets

The recursive ECATNets inherit from all the modeling capabilities of ECATNets and so of all their advantages in modeling workflow processes. Moreover, via the introduced *recursion* concept, the RECATNets captures, in a concise way, the main dynamic reconfiguration mechanisms such as the ability to create or suppress subprocesses from a running workflow. Consequently, the descriptive power of RECATNets allows to model flexible workflow processes in such a way that the structure of theses workflows can be modified, extended or reduced dynamically during their execution. We may introduce, at this end, two types of tasks in workflow processes: *Elementary tasks* (represented by elementary transitions) and a*bstract tasks* (represented by abstract transitions). The execution of an abstract task generates dynamically, as a lower level thread, a new plan of action from the previous one (i.e. higher level plan). When a plan reaches a final marking, it terminates and the whole descent of action plans generated by it are aborted (i.e. a cut step is executed). Consequently, the structure of a workflow process is described by a dynamical tree of action plans. The exceptional situations which may occur, during the execution of a workflow, can be reflected (in a RECATNet) in two manners: (1) by the execution of *cut steps* (when final markings Υ are reached or when elementary transitions are fired) or (2) by the firings of abstract transitions (i.e. dynamic creation of threads). So, a workflow process may handle exceptions, respectively, by terminating the current process or generating a new action plan (subprocess). Moreover, the descriptive power of RECATNets seems particularly adequate for modeling collaborative business processes. The advantage of the recursive ECATNets approach is to have a formalism which can integrate in a simple way both the control-flow and the organizational perspectives of this type of processes while each perspective is considered in a distinct manner. In this context, we interpret the elements of the RECATNets as follows: The global organisation structure of a collaborative business process (i.e. the distributed execution of this process over the collaborating process engines and their inter-process interaction) is represented by the dynamical structure of the tree of threads which describes the current state of the RECATNet. Each thread in this tree represents the local execution and control of each subprocess in their respective execution environments and partitions. So, the event flows which coordinate and link up together these distributed subprocesses are reflected by the firing of *abstract transitions* (call for a subprocess) or the execution of *cut steps* (termination of a subprocess and result return to the caller). If we compare the descriptive power of RECATNets with other classes of hight level Petri nets having the ability to model the dynamic creation of processes like Nested nets [12] proposed for modeling flexible workflow systems [8] and object Petri nets [13] used for modeling collaborative workflow systems [6], we can say that the main advantage of

RECATNets lies in the following points: First RECATNets allow a more advanced description of complex data structures. Secondly, the true concurrency semantics is naturally specified in RECATNets in contrast to the other two formalisms. In a RECATNet, all the threads of the tree can be executed in parallel. Consequently, the parallel execution of different parts of a process (independently) within multiple business units (within one organization), sites or external organizations can be faithfully described. Also, the hierarchy of created processes in RECATNets is unbounded as in Nested nets and RPNs. This is not the case for object Petri nets where the depth of the hierarchy is limited to two levels. We illustrate the suitability of RECATNets in modeling flexible workflow processes in the particular field of e-commerce collaboration through the following example: A simplified *online computer shopping* workflow specified as a RECATNet model is depicted in Fig. 4.

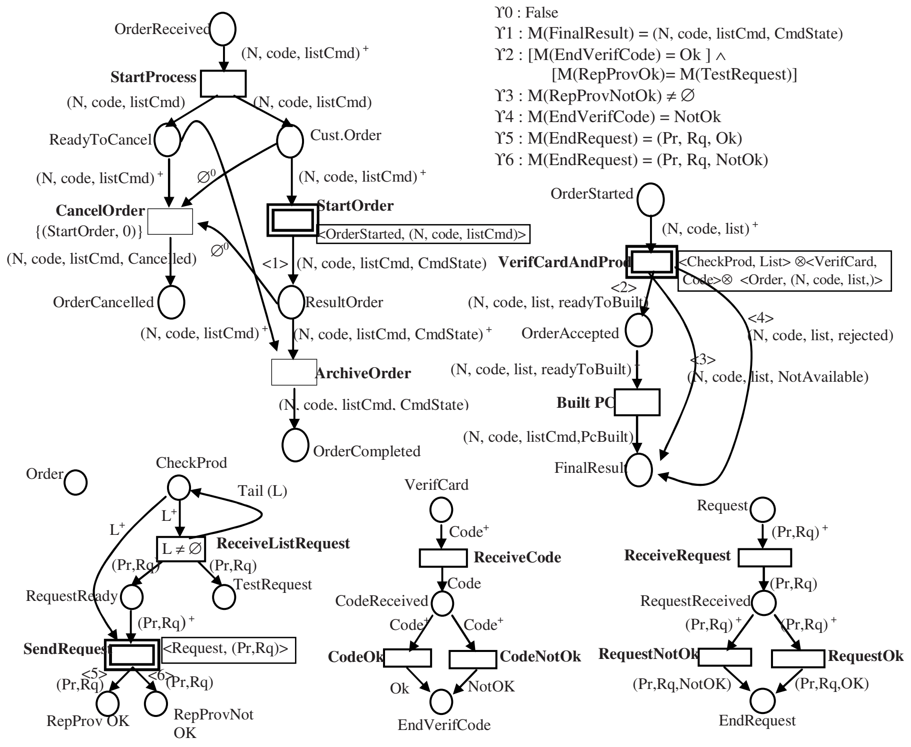


Fig. 4. Online computer shopping workflow

In this workflow example the main company which offers the online shopping service coordinates the execution of the different subprocesses (invocated web services) supported by its collaborating partners. Let us note that the initial state of this net is a tree containing only the root thread with a token *(N, code, listCmd)* in the place *OrderReceived.* This token represents the waiting order which contains, respectively, the order ID number, the customer's code and the list of requested components. The workflow process starts by the firing of the transition "*StartProcess*". Then, the abstract transition "*StartOrder*" initialises the order handling subprocess by creating, dynamically, a new thread in the tree of threads with the associated starting marking. In this subprocess the abstract task "*VerifCardAndProd*" calls two services in parallel, *Check products* and *Validity of credit card* (offered by the collaborating partners) respectively, to check the availability of the requested computer components and the customer's credit worthiness. When the service *Check products* is invoked, the task "*ReceiveListRequest*" looks for the name of the providers offering each requested component. This task produces, at each firing, a couple *(Pr, Rq)* which corresponds, respectively, to the name of the provider and the associated requested component. Next, the abstract task *SendRequest* initialises (at each firing) for a couple (*Pr*, *Rq*) a new instance of the service *Research in stock providers,* i.e., a new thread child is created dynamically in the tree of threads with the initial marking <*Request, (Pr,Rq)*>. The number of running instances of this invoked service is not known at design time. It depends, in fact, of the number of requested products. Each instance (i.e. thread) represents the local execution of the invoked service in the environment of the associated provider *Pr*. The termination of one instance is indicated by a token in *EndRequest* (see Υ_5 and Υ_6). Depending on the value of the token produced in this place, the place *RqProvOk* or *RqProvNotOk*, respectively, is marked in the previous recursion level (after execution of the appropriate cut step). Next, another cut step is enabled at this level of recursion in one of the following cases: (1) if one of the requested computer components is not available (Υ_3 reached), (2) if the credit customer is not OK (Υ_4 reached), or (3) if the credit card and availability of all computer components are both OK (Υ_2 reached; the availability of all the requested components is specified by the state *M(RepProvOk) = M(TestRequest)*). When a cut step is executed, this level of action plan terminates and all the threads generated by it are aborted. Then, depending on the index of this termination (i.e. <2>, <3> or <4>) the outputs of the abstract transition *VerifCardAndProd* are created. When the place *FinalResult* is marked, a cut step (with the index <1>) is executed which reduces the tree of threads to the root level with a token in the place *ResultOrder*. Furthermore, during the processing of the order, the customer has the possibility to cancel his order as long as the corresponding order is not completed. The elementary transition "*CancelOrder*" interrupts the abstract transition "*StartOrder*" with the index <0>.

When the transition "*CancelOrder*" is fired, the thread generated by the transition "*StartOrder*" is aborted and a token *(N, code, listCmd, Cancelled)* is produced in the place *OrderCancelled* but no token is produced in the output place of "*StartOrder*" (i.e. *ICT*(*StartOrder*, *ResultOrder*, 0) = ∅). In the firing sequence given in Fig. 5, the black node in the tree of threads denotes the thread in which the following step is fired. For the sake of clarity, each thread is associated to its internal marking, whereas, the general term of sort `Thread` which describes the current tree of threads (i.e. the distributed state of the RECATNet) is noted just below the tree, in a gray frame (See the next section for a description of the specification of RECATNets semantics and their distributed states in terms of rewriting logic). We note by (*t1* Seq. *t2*) the sequential firing of transitions *t1* and *t2* and by (*t1* Para. *t2*) the parallel firing of *t1* and *t2*. Note that with the rewriting semantics given to RECATNets, the parallel execution of the different subprocesses, independently supported by the collaborative

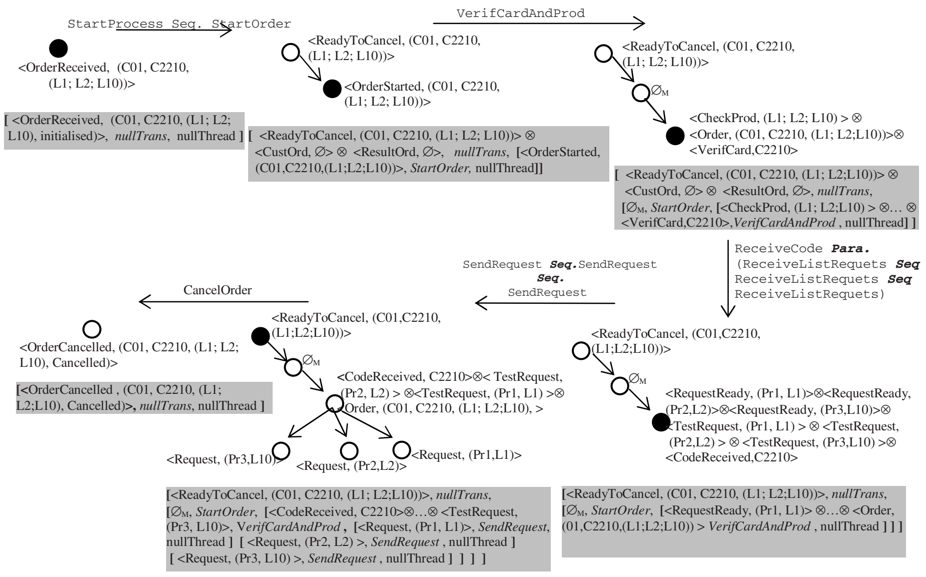

StartProcess Seq. StartOrder
<OrderReceived, (C01, C2210, (L1; L2; L10))>
[<OrderReceived, (C01, C2210, (L1; L2; L10), initialised)>, nullTrans, nullThread]
<ReadyToCancel, (C01, C2210, (L1; L2; L10))>
<OrderStarted, (C01, C2210, (L1; L2; L10))>
[<ReadyToCancel, (C01, C2210, (L1; L2; L10))> ⊗ <CustOrd, ∅> ⊗ <ResultOrd, ∅>, nullTrans, [<OrderStarted, (C01,C2210,(L1;L2;L10))>, StartOrder, nullThread]]
VerifCardAndProd
<ReadyToCancel, (C01, C2210, (L1; L2; L10))>
∅M
<CheckProd, (L1; L2; L10) > ⊗ <Order, (C01, C2210, (L1; L2;L10))>⊗ <VerifCard,C2210>
[<ReadyToCancel, (C01, C2210, (L1; L2;L10))> ⊗ <CustOrd, ∅> ⊗ <ResultOrd, ∅>, nullTrans, [∅M, StartOrder, [<CheckProd, (L1; L2;L10) > ⊗… ⊗ <VerifCard,C2210>,VerifCardAndProd , nullThread]]
ReceiveCode Para. (ReceiveListRequets Seq ReceiveListRequets Seq ReceiveListRequets)
SendRequest Seq. SendRequest Seq. SendRequest
CancelOrder
<OrderCancelled, (C01, C2210, (L1; L2; L10), Cancelled)>
[<OrderCancelled , (C01, C2210, (L1; L2;L10), Cancelled)>, nullTrans, nullThread]
<ReadyToCancel, (C01,C2210, (L1;L2;L10))>
∅M
<CodeReceived, C2210>⊗< TestRequest, (Pr2, L2) > ⊗<TestRequest, (Pr1, L1) >⊗ <Order, (C01, C2210, (L1; L2;L10), >
<Request, (Pr3,L10)>
<Request, (Pr2,L2)>
<Request, (Pr1,L1)>
[<ReadyToCancel, (C01, C2210, (L1; L2;L10))>, nullTrans, [∅M, StartOrder, [<CodeReceived, C2210>⊗…⊗ <TestRequest, (Pr3, L10)>, VerifCardAndProd , [<Request, (Pr1, L1)>, SendRequest, nullThread] [<Request, (Pr2, L2) >, SendRequest , nullThread] [<Request, (Pr3, L10) >, SendRequest , nullThread]]]]
<ReadyToCancel, (C01,C2210, (L1;L2;L10))>
∅M
<RequestReady, (Pr1, L1)>⊗<RequestReady, (Pr2,L2)>⊗<RequestReady, (Pr3,L10)>⊗ <TestRequest, (Pr1, L1) > ⊗ <TestRequest, (Pr2,L2) > ⊗ <TestRequest, (Pr3,L10) >⊗ <CodeReceived,C2210>
[<ReadyToCancel, (C01, C2210, (L1; L2;L10))>, nullTrans, [∅M, StartOrder, [<RequestReady, (Pr1, L1)> ⊗…⊗ <Order, (01,C2210,(L1;L2;L10)) > VerifCardAndProd , nullThread]]]

Fig. 5. A firing sequence of the online shopping workflow example

partners, is well described. Such a construction describes, adequately, the flexible structure of the online shopping workflow process and that a business partnership is often created dynamically and maintained only for the required duration such as a single transaction.

3 Recursive ECATNets Semantics in Terms of Rewriting Logic

Each RECATNet is defined as a conditional rewrite theory where transition firing (elementary or abstract) and cut step execution are formally expressed by rewrites rules. A conditional rewrite rule of the form t: $s \rightarrow s'$ if C (where t is the abstract or elementary transitions or the cut step associated to this rule) means that a fragment of a RECATNet's state that is the pattern s can change to the corresponding instance of s' concurrently with any other state change, if the condition C holds. The global distributed state of a RECATNet is described by a *dynamical tree Tr* of threads marking. Each thread (node) *Th* of the tree *Tr* is expressed as a term [*MTh*, *tabs*, *ThreadChilds*], where: *MTh* represents the marking of the thread *Th* which is expressed as multi-sets of pairs of the form <p, $[m]_{\oplus}$>, where p is a place of the net, $[m]_{\oplus}$ a multi-set of algebraic terms and the multi-set union on the pairs <p, $[m]_{\oplus}$> is denoted by the operator ⊗ (associative, commutative and has *Null* as the identity element). The sub-term t_{abs} represents the name of the abstract transition whose firing (in the thread father) gave birth to the thread *Th*. *ThreadChilds* represents the threads generated by this thread *Th* in the current tree *Tr*. They are described as a finite multiset of terms of sort *Thread*. The constant *nullThread* represents the empty thread and the operator _ _ (the underline indicates the position of the parameter) is the corresponding multiset union operator which is associative, commutative and has the constant *nullThread* as the identity element. The state space of RECATNets is axiomatized by the following equational theory (given in Maude syntax [14]).

```
fmod THREAD is Protecting Marking . sorts  Thread Trans .
op  nullTrans :-> Trans .  op  nullThread :-> Thread .
op [_,_,_] Marking Trans Thread -> Thread .
op _ _: Thread Thread -> Thread [assoc comm id : nullThread].
op Initial : Making -> Thread .
var T : TransAbs . var L : List . vars mth mthf : Thread .
var M : Marking . vars Minit Moutput mts : multiset .
Eq Initial (M) = [M, nullTrans, nullThread] .
*********************************************************************
op DeleteThread : Thread List -> Thread .
eq DeleteThread(nullThread, L) = nullThread .
eq DeleteThread(Thf [M, T, Th], L) = if find(T, L)
then DeleteThread(Thf,L) else [M,T,Th] DeleteThread(Thf,L) fi .
*********************************************************************
op CreateTokens : Multiset Thread Trans Multiset -> Multiset .
eq CreateTokens(Minit, nullThread, T, Moutput)= Moutput.
eq CreateTokens(Minit, Thf [M, T, Th], T, Moutput) =
createTokens(Minit, Thf, T, Moutput ⊕  Minit).
eq CreateTokens(Minit, Thf [M, Tf, Th], T, Moutput) =
createTokens(Minit, Thf, T, Moutput) [owise] .
endfm
```

The rewrite rules associated to a RECATNet are partitioned into four distinct types which are the abstract rules, the elementary rules, the extended elementary rules and the pruning rules. The general form of these rules is given by the following theory.

```
mod RECATNet-BEHAVIOR is  pr  THREAD SPEC .   …
vars M Mf : Marking . vars mTh mThf : Thread.
vars mp mp′ mp_1…mp_n mp_final: Multiset. vars T  Tf : TransAbs .
*** Elementary rules: For each elementary transition t_elt with K(t_elt)=∅
crl [t_elt]:  <p, mp ⊕ DT(p, t)>⊗<p′, mp′>  →  <p, mp> ⊗ <p′, mp′⊕CT(p′,
t)>  if (Nbr(mp′ ⊕ CT(p′,t))≤Cap(p′)) and (InputCond) and (TC(t)).     …

*** Extended elementary rules: For each elementary transition with
**** K(t_elt)={(t_absj,i),(t_absm,k), …} ***********************************
crl [t_elt]: [M⊗<p, mp  ⊕ DT(p, t)>⊗<p′, mp′>⊗<p′_j, mp′_j>⊗<p′_m, mp′_m>…,
T, mTh] →[M⊗ <p, mp > ⊗ <p′, mp′⊕CT(p′, t)>⊗  <p′_j, mp′_j ⊕
CreateTokens(ICT(p′_j,t_absj,i), mTh, t_absj, Ems)> ⊗
<p′_m, mp′_m ⊕ CreateTokens(ICT(p′_m,t_absm,k), mTh, t_absm, Ems)>… , T,
DeleteThread(mTh, t_absj ;; t_absm ;; …)] if (InputCond) and (Nbr(mp′ ⊕
CT(p′,t))≤Cap(p′)) and (TC(t)) .
…
********  Abstract rules *********************************************
crl [t_absi]:[M  ⊗<p, mp ⊕ DT(p, t)>,  T, mTh ]→[M ⊗ <p, mp>, T, mTh
[<p″,CT(p″,t_absi)> ⊗ <p_1, Ems> ⊗…⊗ <p_n, Ems>,t_absi, nullThread]] if
(InputCond) and (TC(t)) .
…
**** Pruning rules ** A pruning rule associated to a cut step occuring
in a thread(not the root),generated by an abstract transition t_absj:
crl [τi]: [Mf ⊗ <p′, mp′>, Tf,[M ⊗ <p_final, mp_final>,t_absj, mTh] mThf] → [Mf
⊗ <p′, mp′ ⊕ ICT(p′,t_absj,i)>, Tf, mThf] if [ϒ_i →True] and (Nbr (mp′ ⊕
ICT(p′, t_absj, i)) ≤ Cap(p′)).        …
*** A rule associated to a cut step occuring in the root thread
crl [τi]:[M⊗<p_final, mp_final>,nullTrans,mTh] → nullThread if [ϒi].
…
endm
```

In a rewrite rule associated to a transition t (elementary or abstract), the component *InputCond* is determined from the expression $IC(p, t)$ as follows (where mp is the marking of the input place p of t): $InputCond = \left\{ \begin{array}{l} mp \equiv \alpha \ \text{ if } IC(p,t) = [\alpha]^0 \\ \alpha \ \textbf{Inclu}\ mp \ \text{ if } IC(p,t) \equiv [\alpha]^+ \\ \text{not } (\beta\ \textbf{Inclu}\ mp) \ \text{ if } IC(p,t) \equiv [\beta]^- \end{array} \right\}$

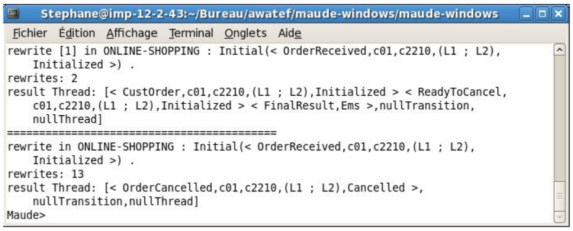
In what follows, we give few rewrite rules from the rewrite theory *ONLINE-SHOPPING* descibing the RECATNet given in Fig. 5.

```
mod ONLINE-SHOPPING is pr THREAD SPEC . subsorts Exp Data < Token .
ops  OrderReceived … EndRequest: -> Place .
ops StartOrder VerifCardandProv SendRequest :-> TransAbs .
vars M  Mf: Marking . vars mTh  mThf : Thread.
vars T Tf : TransAbs . Mp_fr … Mp_co: multiset .
vars N  Pr Rq Code CmdState : Data . vars ListCmd L : List.

********* elementary rules*******************************************
crl [ReceiveListRequest]: <CheckProd, mp ⊕ L> ⊗ <TestRequest, mp_tr> ⊗
<RequestReady, mp_rr>  → <CheckProd, mp ⊕ tail(L)> ⊗ <TestRequest, mp_tr
⊕ (Pr,Rq)> ⊗ <RequestReady, mp_rr ⊕ (Pr,Rq)>  if (L=/=emptyList) and
(FindInList(head(L), ListProviders) =/= errorelt) and Head (L) := Rq
and FindInList (head(L), ListProviders ) := Pr .

********* Extended elementary rule **********************************
rl [CancelOrder]: [M ⊗ <OrderCancelled, mp_OC> ⊗ <ReadyToCancel, mp_RC ⊕
cmd,code,listCmd)> ⊗ <CustOrder, mp_co> ⊗ <ResultOrder, mp_fr>,T, Th] →
[M ⊗ <ReadyToCancel, mp_RC > ⊗ <OrderCancelled, mp_OC ⊕ (cmd, code,
listCmd, Cancelled)>, T, DeleteThread(Th, StartOrder)]if mp_co == Ems and
mp_fr == Ems.
…
endm
```

Since we express each RECATNet *RN* as a conditional rewrite theory, we can prove that there is a formal correspondence between concurrent firing of transitions in a RECATNet *RN* and deductions in rewriting logic [15]. Besides, we can benefit from the Maude system [14] (a high-performance interpreter and language based on rewriting logic) as a simulation environment for RECATNets where the models can also be analysed and model checked using the LTL Model-Checker (of Maude) which allows to prove liveness and safety properties related to (finite) workflow schemas.


Stephane@imp-12-2-43:~/Bureau/awatef/maude-windows/maude-windows
Fichier Édition Affichage Terminal Onglets Aide

```
rewrite [1] in ONLINE-SHOPPING : Initial(< OrderReceived,c01,c2210,(L1 ; L2),
    Initialized >) .
rewrites: 2
result Thread: [< CustOrder,c01,c2210,(L1 ; L2),Initialized > < ReadyToCancel,
    c01,c2210,(L1 ; L2),Initialized > < FinalResult,Ems >,nullTransition,
    nullThread]
==========================================
rewrite in ONLINE-SHOPPING : Initial(< OrderReceived,c01,c2210,(L1 ; L2),
    Initialized >) .
rewrites: 13
result Thread: [< OrderCancelled,c01,c2210,(L1 ; L2),Cancelled >,
    nullTransition,nullThread]
Maude>
```

Fig. 6. A simulation of the online shopping workflow example in Maude environment

In Fig. 6, we give an execution of the example of Fig. 4. using as platform the version 2.1.1 of the Maude system under Linux. The `rewrite` command of Maude rewrites a given expression using the default strategy until no more rules can be applied (one can specify, between brackets, the maximum number of rule applications).

4 Conclusion

The main goal of this paper is to show the pertinence of RECATNets in modeling reconfigurable workflow processes. Moreover, due to their concurrency semantics which is defined in terms of rewriting logic [3], RECATNets allow the formal analysis of flexible and collaborative business processes which operate on distributed and independent execution environments. Indeed using a rewriting logic language implementation such as system Maude [4], it is possible to create rapid prototype on which one can apply formal verification methods such model checking techniques.

References

1. Van der Aalst, W.M.P., Van Hee, K.M.: Workflow Management: Models, Methods, and Systems. MIT press, Cambridge (2002)
2. Casati, F.: A discussion on approaches to handling exceptions in workflows. In: Conference on Computer-Supported Cooperative Work 1998, Seattle (1998)
3. Halliday, J.J., Shrivastava, S.K., Wheater, S.M.: Flexible workflow management in the OPENflow system. In: 4th IEEE Int. Enterprise Distributed Object Comp. Conf., pp. 82–92. IEEE Computer Society, Washington (2001)
4. van Hee, K., Lomazova, I.A., Oanea, O., Serebrenik, A., Sidorova, N., Voorhoeve, M.: Nested nets for adaptive systems. In: Donatelli, S., Thiagarajan, P.S. (eds.) ICATPN 2006. LNCS, vol. 4024, pp. 241–260. Springer, Heidelberg (2006)
5. Sadiq, W., Sadiq, S., Schulz, K.: Model Driven Distribution of Collaborative Business Processes. In: IEEE International Conference on Services Computing, pp. 281–284. IEEE Computer Society, Chicago (2006)
6. Sarshar, K., Theling, Th., Loos, P., Jerrentrup, M.: Integrating Process and Organization Models of Collaborations through Object Petri Nets. In: Sarshar, K. (ed.) MKWI 2006, contribution to XML4BPM, Passau (2006)
7. Hicheur, A., Barkaoui, K.: A Natural Semantics for RECATNets in Terms of Conditional Rewriting Logic. Internal Technical Report, Cedric Lab (2007)
8. Bettaz, M., Maouche, M.: How to Specify Non Determinism and True Concurrency with Algebraic Terms Nets. In: Bidoit, M., Choppy, C. (eds.) Recent Trends in Data Type Specification. LNCS, vol. 655, pp. 164–180. Springer, Heidelberg (1993)
9. Zeghib, N., Barkaoui, K., Bettaz, M.: Contextual ECATNets semantics in terms of conditional rewriting logic. In: 4th ACS/IEEE Int. Conf. on Computer Systems and Application, pp. 936–943. IEEE Press, Sharjah (2006)
10. Haddad, S., Poitrenaud, D.: Modeling and analyzing systems with recursive Petri nets. In: 5th Workshop on Discrete Event Systems, pp. 449–458. Kluwer Academic, Belgium (2000)

11. Bruni, R., Meseguer, J.: Semantic foundations for generalized rewrite theories. J. Theor. Comput. Sci. 360(1–3), 386–414 (2006)
12. Lomazova, I.A.: Modeling Dynamic Objects in Distributed Systems with Nested Petri Nets. J. Fundam. Inform. 51(1–2), 121–133 (2002)
13. Valk, R.: Object Petri Nets-Using the Nets-within-Nets Paradigm. In: Desel, J., Reisig, W., Rozenberg, G. (eds.) Lectures on Concurrency and Petri Nets. LNCS, vol. 3098, pp. 819–848. Springer, Heidelberg (2004)
14. Clavel, M., Duran, F., Eker, S., Lincoln, P., Martı-Oliet, N., Meseguer, J., Talcott, C.: Maude manual (version2.3) (2007), http://maude.cs.uiuc.edu/maude2-manual/
15. Hicheur, A., Barkaoui, K., Boudiaf, N.: Modeling Workflows with Recursive ECATNets. In: 8th SYNASC 2006, pp. 389–398. IEEE Computer Society, Los Alamitos (2006)

Quality Analysis of Composed Services through Fault Injection

Maria Grazia Fugini, Barbara Pernici, and Filippo Ramoni

Politecnico di Milano, Department of Electronics and Information
{fugini,pernici,ramoni}@elet.polimi.it

Abstract. Web service composition can be adopted to integrate information systems as complex composed processes. While interfaces of services are known at composition time, the quality of the composed process may depend on the ability of component services to react to unforseen situations, such as data quality problems and coordination problems. In this work, we propose an approach to analyze the quality of composed services using fault injection techniques, by inspecting the reaction to injected faults in the composed process to assess its quality in terms of fault tolerance capabilities. The component services are analyzed either as black-boxes, when only input and output messages are considered or as white-boxes, when data sources used by the service are considered.

Keywords: information systems quality, process testing, composed services, fault injection.

1 Introduction

The service oriented approach is being adopted more and more in Cooperative Information Systems (CIS) to support service interoperability and composition. Web service technology allows the creation of complex processes, composed of simple services, that exchange messages to each other to form complex conversation schemas [1].

The quality of CIS depends both on the quality of the composition process and on the quality of component services. One of the obstacles to the adoption of the Web service paradigm in CIS is the problem of assessing their overall quality, since services are inherently distributed and heterogeneous, and often invoked without a complete knowledge of their very nature and quality, in particular in mobile and context-aware compositions [2]. In these applications, the dynamic nature of service-oriented composition, where services are discovered, selected, and composed, possibly also at runtime, precludes the possibility to foresee the application behavior in front of faults and unexpected situations.

Since the probability of fault occurrences in large scale, distributed CIS increases with the number of cooperating services, methods and tools to assess and manage faults in composition are needed.

The aim of the WS-Diamond EU Project[1] is to support composed service-based processes with self-healing properties, that is, the ability to self-react to anomalous events, such as faults. In the project, mechanisms to identify possible faults and repair strategies are being studied, with methods based on the diagnosis of causes and repair actions on services and composed processes [3]. Such a self-healing environment supports process execution with the capability of dynamically reacting to faults and to repair their consequences, in order to achieve a correct termination of the composed process. In such an environment, techniques are needed to assess systematically whether a given process behaves correctly in presence of faults, detecting them and reacting with recovery actions.

The goal of this paper is to present a systematic testing method for service-based CIS processes. The method is based on fault injection during process execution. Two types of faults are considered: data faults and delays. *Data faults* are frequent in a CIS, since both data redundancy, which is a possible cause of faults due to misalignment of data, and the probability of out-of-date values increase [4]. The *Execution time* of the complex service depends on the performance of the single Web services and on their composition, and failures might be cause by delays in process execution. The aim of the method is to assess how the process reacts to a systematic set of faults being injected during its execution. The analysis of fault consequences can be used at design time to suggest possible process or service improvements or to support service management with appropriate repair actions.

A common approach to validate the dependability and fault tolerance of a software system is *fault injection*, allowing software faults to be discovered. In a distributed computing environment, research efforts have developed systematic ways to identify faults at design time to design fault management mechanisms. In [5], the motivation of fault injection in software systems is discussed, and the system reaction is studied. As summarized in [6], software injection faults are categorized into compile time and runtime injections. The former technique modifies the source code to inject simulated faults. The latter consists of triggers that inject faults at execution time.

In [7,8], the differences between Web service testing w.r.t. traditional software testing are discussed: lack of user interfaces, scarce knowledge of the service code (WSDL interfaces only are available), dynamic integration, and heterogeneity due to their being located in different server containers. In Web services, [9] describes a tool for generating and validating test cases. In particular, the authors start from the WSDL schema types, and generate a WSDL request with random data and a test script, that allows the user to manipulate the parameters of a Web service request. [10] proposes a technique for testing Web services using mutation analysis: the authors define and apply mutation operators to the WSDL document, in order to generate mutated Web service interfaces that will be used for Web service testing. The WSDL Test tool [7] generates Web Service requests from the WSDL document, and tunes them in accordance with pre-conditions specified by the tester. [11] bases Web service testing on a data perturbation

[1] Project Web Site: http://wsdiamond.di.unito.it

technique. XML messages are modified on the basis of rules defined on the message grammars; these are used to test the Web service behavior. [9] introduces a set of SOAP perturbation operators, that are XML schema independent and that modify the SOAP message directly. In [12], a framework to intercept and perturb SOAP messages is proposed. The injected faults corrupt the encoding schema address, discard messages and insert extra text in the SOAP Body. The work described in [13] helps service requestors to automatically create test cases to select Web services from public registries. WS-FIT [14] intercepts and modifies SOAP messages using scripts. The authors use WS-FIT also to inject latency in messages [15]. The common characteristic of the above mentioned approaches is their focus on testing single services in isolation. In this paper, we propose an approach that, using techniques used for testing single Web services, combines them to test composed processes. In addition, while previous approaches inject simple errors in data to test the services, we create data faults based on data quality properties.

In Sect. 2, we analyze process faults. In Sect. 3 we describe fault injection in complex processes. In Sect. 4 we describe the fault injection tool. Finally, in Sect. 5 a case study is illustrated and commented.

2 Process Faults

In the paper, we analyze processes where the service composition structure is fixed, with a well defined process schema, and we assume that faults may occur only one at a time. Concerning services, we assume that the WSDL description of the service interface is available, and consider: a) black-box testing, when the service implementation code is not accessible, and b) white-box testing, when service code and data sources are accessible.

2.1 Process Quality Analysis Scenario

Figure 1 shows an example process (a detailed example is discussed in Sect. 5). Web service 1 (hereafter WS1) exchanges messages with Web service 2 (WS2). We assume the source code of WS1 is available, as well as the local databases used to store WS1's persistent data (white-box view). Instead, WS2 is only invoked, no information being available about it, except information published in its WSDL document (black-box view). In the quality analysis method, we inject delays in the exchanged messages and in the executed code of WS1, and data faults in (all) the data used by WS1, but we can inject faults only in messages exchanged by WS2, since its internal structure is not visible.

2.2 Data Faults

Starting from the literature on data quality, we have studied the data fault types that can occur in a service composition scenario [16], starting from data quality dimensions, i.e., accuracy, completeness, consistency, and timeliness [17,18].

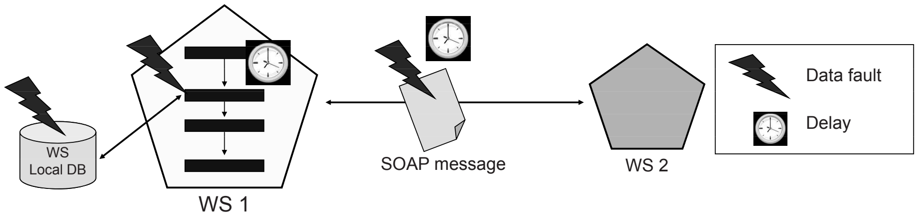


Fig. 1. A sketch of our testbed scenario

Along these quality dimensions, it is possible to classify faults in:

1. *Value mismatch*, due to:
   - Typos: e.g., Jhon instead of John
   - Different format: e.g., date expressed as dd/mm/yyyy instead of mm/dd/yyyy
   - Semantic conflicts in data values: e.g., City: London, Country: Italy
   - Delay in update operations.

   Typos affect data accuracy, while conflicts in data values affect data consistency. Different formats can affects both accuracy (the value is an incorrect representation of real-world value), and representation consistency. Delay in update operations between two databases that contain the same values can affect both timeliness, since the value is no longer valid, and accuracy, since that the value is an incorrect representation of the real-world value.
2. *Missing data* can be caused by value unavailability or by a delay in update operations. The former is related to the completeness dimension. Delay in update operations between two databases that contain the same values can be related to the timeliness dimension.

Our method includes a set of data perturbations algorithms that modify a data item of a given type (string, number, date, and so on) according to the desired faults.

2.3 Time Analysis of a Composed Service

Execution time needed to complete the process increases in the presence of slow simple services. The contribution of each service to the global performance value depends on the structure of the process. Our analysis has the goal of identifying the services that affect the quality in a critical way.

The presence of *timeouts* adds a large variability to the execution time and may cause process *failures*, that are the observable effects caused by a fault, if invoked services do not respond within the time constraints set in the timeouts. Since the performance of a complex service is difficult to evaluate at design time, run time simulation is employed to estimate the contribution of each service to the complex service. Our approach consists in injecting *delays* in the exchanged messages, computing the different overall execution time, w.r.t. normal executions, and in evaluating the results of the execution.

3 Fault Injection

Three different approaches to data fault injection are proposed and illustrated in this section. Then, we illustrate how the effects of faults are analyzed, observing different failures, depending on where data faults are injected (in the exchanged message, in the service operation or in the databases) and delays injection.

3.1 Injecting Data Faults

In our approach, we choose to perturb either exchanged messages or data contained in databases used by the services.

While the former perturbation is always feasible, the latter can be applied only in the white-box Web service view. Under the white-box hypothesis, in a complex service, data are located in databases, used by Web services and exchanged between different services through messages. These data could be replicated in other databases used by other services in the process, or be present only in a service local database. For example, in the e-commerce scenario (see Sect. 5), customer information is stored locally at the Shop, Warehouse, and Shipper services.

Therefore, data perturbation actions can take place on databases or on the exchanged messages. It is worth noting that database perturbation requires to access the service code, besides the databases. This second requirement arises because it could be necessary to inject faults on the used data; hence the *result set* of each query needs to be accessible.

Figure 2 shows our testing approach. Precisely, our approach works as follows.

– If the service databases are available (white-box hypothesis), we introduce data faults following statistical distributions. This kind of perturbation operations are performed off-line, and generate out-of-date and misaligned databases. This method allows one to discover the effects of low quality data, along several simulation runs (Fig. 2.a).
– If the services code is available (white-box), we modify it by introducing the invocation of a defined script. This script acts on database records interested by the query instructions, introducing data faults, either persistent or volatile. This kind of perturbations acts in on-line mode, and only on the used data. This method enables a more specific analysis than in the first case, since faults can be injected on specific data and the databases affected by the perturbation can be selected (Fig. 2.b).
– If both service code and databases are unavailable (black-box), we rely only on the XML structure of the exchanged data. Here, we introduce data faults on the data contained in the messages (Fig. 2.c).

3.2 Reactions to Fault Injection

After the data fault injection phase, corrupted data are contained in the service. A complex service can react to a fault in four different ways:

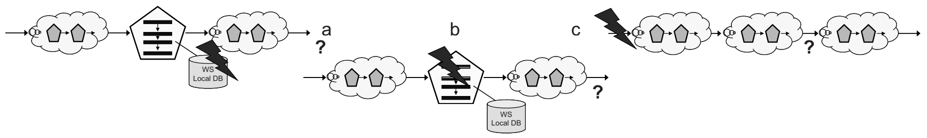


Fig. 2. Testing approach: injection points

- by rising a **fault message**, which is then managed through a fault handler;
- by **stopping** the process execution, because a fault was not foreseen and hence a fault management action is not available;
- by internally recognizing a data fault, which is possibly repaired through a **repair action**; the execution of such action is anyway signalled to the system through a repair message, in order to enable process analysis;
- by presenting **no reactions**, and executing normally, thus demonstrating that the process is not able to recognize the presence of that data fault.

The first case is the best one, from a validation point of view, since the erroneous situation is captured and an appropriate repair plan can be applied. The analysis of the possible repair actions is out of the scope of this paper; however, we mention that possible actions may apply at the process level (for example, as done in the SH-BPEL tool we have developed in WS-Diamond [19]), or may employ a data cleaning approach [20]. If a data fault determines a failure state that a service cannot manage, the service raises an exception and the global process execution stops. In this case, the data fault is detected, but this reaction is not recommendable. In fact, process availability and reliability decrease in front of a simple service stop. If the process has associated repair actions and an adequate action is invoked to repair the faulty data, the service terminates correctly. The last case is obviously the worst and most difficult to manage. In fact, the execution flow is not influenced by the erroneous data, and the final result is not correctly related to the input data.

Evaluating Effects of Data Faults. A problematic aspect is the detection of a data failure. The effects of some data faults may not be immediately visible in the execution flow, but become evident after more interactions. For example, an erroneous price of an item is usually detected by the customer when he receives the wrong item rather then by the shipper. In a complex process, *early detection* of data faults, i.e., a detection close to the source of the wrong data, is appreciated, since wrong data effects can be confined and the possible recovery and cleaning actions can be more efficient.

An issue is how to measure the distance between the source point of the fault and the detection point (usually related to a failure). Different metrics exist to express this distance, such as the elapsed time between the fault and the failure occurrence. However, this is not a good indicator, because it strictly relates to the service duration: a slow service, or a pause waiting for a user input, does not propagate the faulty data to other services and data.

To describe the spread of faulty data, we choose as a metric the number of exchanged messages from the source of the data fault to the instant of the fault detection. This number should be as low as possible, because this means that the fault has been captured close to its source. Hence, starting from the knowledge on where the data fault was injected and on the possible effects, we are able to give a design indication about where to insert a fault handler in the code.

Evaluating Effects of Delays. Two ways are possible to investigate the influence of message delays on the process execution time. A first way is to use a systematic approach using a common fixed delay time for each message. At each execution of the composed service, only one message is delayed. The total execution time is compared to a standard execution time through the *Delay Ratio* (DR) index, defined as $\frac{Computation\ time\ with\ delay}{Computation\ time\ without\ delay}$.

By analyzing DR indexes, we are able to infer which is the influence of the less performing service: the higher the value, the more the message performs as a process bottleneck. Therefore, in order to improve the overall performance of the composed service, the best approach is to focus on the services dealing with messages with the highest values of the DR. Different actions are possible to improve the composed service: a service substitution, or a parallel invocation of different services of the same type. Possible faults induced in the process by delays are due to time constraints in some services: if a service does not complete in a predefined time, the service is considered as failed. This situation is revealed by the non termination of the invoking service, if the interaction is a synchronous request/response. However, handling of asynchronous interactions is more problematic.

The second approach to execution time analysis deals with statistical simulations of the composed service workload. In this case, different messages are delayed during the same execution, using different probability distributions. The result of the simulation is the service execution behavior in real execution environments.

4 A Support Tool for Complex Service Analysis

The quality analysis method is supported by a fault injection tool. The tool allows the creation of fault injection services to perform perturbation on data and messages. These services are invoked in different ways according to different scenarios:

- **stand-alone**, by the tester, in order to introduce data errors in the databases used by a Web service, before test execution;
- from the **Web service code**, in order to change read values of an instance of the service or to delay the execution (white-box testing);
- from a **SOAP handler**, that intercepts SOAP messages and changes data parameters or to delay the execution (black-box testing).

Each service contains the requested fault information and the probability distribution for the fault. Figure 3 depicts the main modules of our tool:

- **Perturbation Manager:** It manages the fault injection service creation and the test execution;
- **Operations Database:** It contains the defined perturbation operations;
- **Data Perturber:** It is invoked from the service code and performs defined changes to process data in the local databases;
- **Message Handler:** It intercepts SOAP messages exchanged between services and applies defined changes to transported data;
- **Fault Generator:** It is in charge of generating a data fault in data, relying on a set of perturbation algorithms;
- **Log Database:** It collects the performed operations and allows fault analysis;
- **Log Analyzer:** It is in charge of analyzing and retrying the injection operations, using the fault detection distance metrics;
- **User Interface:** It allows users (i.e., designers and testers) to compose scripts in an assisted way, to define the automatic test script and to view the log analysis results.

The tool is developed in Java, using Tomcat and Axis as environment and ActiveBpel as workflow engine. The WSDL4J libraries are used to retrieve information from WSDL documents. Axis libraries are used to change the SOAP messages to inject faults.

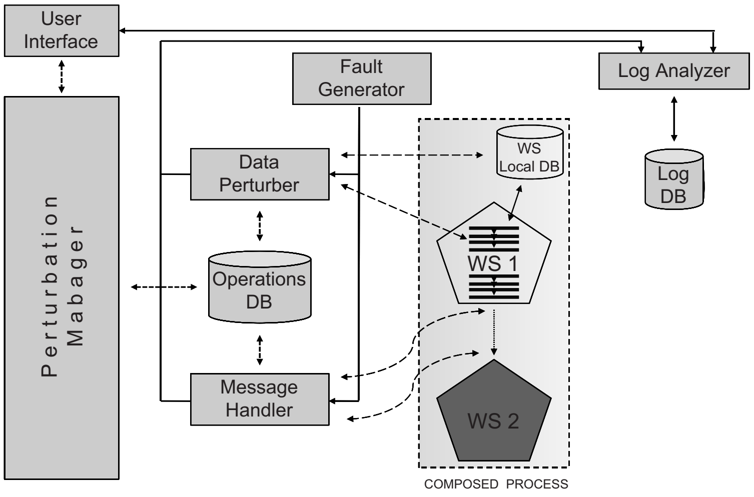


Fig. 3. Fault injector architecture

5 Test Case

In this section, we apply the method to a reference example, in order to show how design quality of a complex service con be tested. The reference example is a foodshop application, composed of three complex services:

– a **Shop** that is in charge of receiving the customer's orders, splitting the item list into a subset of unperishable items (such as pasta, frozen food, ...), and a subset of perishable items (for example vegetables, meat, ...), collecting the answers about the availability of the items, and replying to the customer;
– a set of **Warehouse**s where unperishable items are stored;
– a set of **Supplier**s that supply perishable items.

Each service provides of a set of operations at a fine-grained level. A simplified representation of the process is given in Fig. 4. The Shop is an orchestrated Web service which invokes the operations of both the Supplier and the Warehouse Web services. The Supplier Web service has a timer that terminates the service when the input messages for the Reserve operations after a CheckAvailability operation have been waited for too long. We apply our fault injection method to

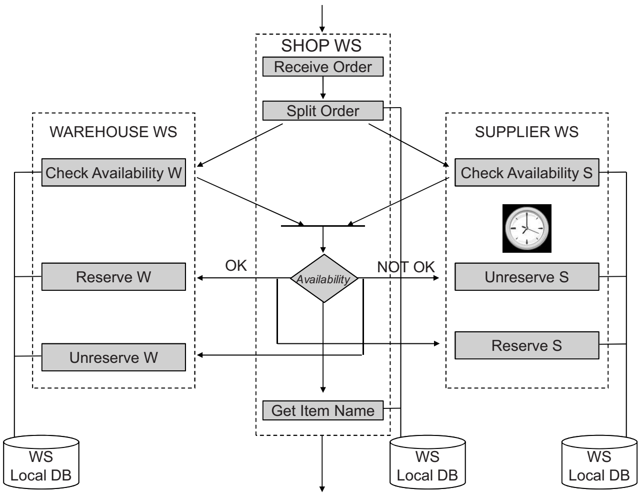


Fig. 4. Reference example

analyze the quality of the Shop process. The first aspect we analyze is the effect of delays. Our tool introduces a delay in one message at a time. Table 1 shows the DR index when each message is delayed with a fixed time of 5, 10 and 20 seconds respectively. When a delay of 5 seconds is injected, we see that DR is similar in all the cases: this means that a short delay in any of the exchanged messages has the same impact on the process. The second delay injection (10s) is more significant: in the parallel invocation of Warehouse and Supplier, the less performing service is Supplier, because the injected delay causes a higher value value of DR when injected into Supplier than when injected into the messages of the Warehouse. A process improvement could be the parallel invocation of several instances of Supplier services or the substitution of the service with a faster one. The last case shows that if the injected delay is 20 seconds, the Supplier Web service terminates in advance and the complex service does not terminate correctly. In this case an unmanaged exception is raised. In order to improve the process quality, a possible suggestion is to add to the Shop the possibility to re-invoke the Supplier service when this terminates before the normal end.

Table 1. Delay Ratio when only one message is delayed of 5, 10 and 20 seconds

		Delay Ratio		
Operation	Message Type	5 s	10 s	20 s
Split order	Request	1.25	1.33	1.54
Check AvailabilityS	Request	1.25	1.65	F
Check AvailabilityW	Request	1.25	1.42	1.75
Reserve S	One Way	1.26	1.29	F
Reserve W	One Way	1.03	1.21	1.42
Get Item Name	Request	1.28	1.50	1.77

The second quality aspect is the effect of data faults on the process, injecting data faults in services and messages. Table 2 collects, for each data and for the operations visible in white-box services, the result of the data fault injection. D means that the fault is detected, and eventually managed (e.g., with data cleaning). The number in brackets is the number of messages exchanged from the fault injection until detection. It is worth noting that a process is better designed if the faults are detected early, so the number of exchanged messages is lower. In case the fault is not detected, the ND symbol is used: this means that the process execution ends without managing the wrong data, so the global results could be uncorrect. The last case, denoted by F, means that an unforseen fault has occurred, and the complex process terminates incorrectly.

In the *Typo* column, we summarize the process reaction when one of the typo data fault is injected. On this test case, different typo types do not cause different reactions, since no more deep data analysis is performed. So, we inject indifferently one of the five typos types. From the experimental results, we can infer that if an inaccurate value is read from the local service databases, the Warehouse service detects the faulty situation by raising a message fault, while the Supplier service does not provide this check and causes a fault. The check of the correspondence between required and sent items is done at the end of the composed process, and this is revealed by the high number of exchanged messages. A possible improvement to the process consists in moving this check closer to response messages: in this way a re-invocation of the service in case of failure is possible, and a better management could be performed. More problematic are the Customer Code data, because its changes are never detected: to improve the process a check could be introduced at the end of the process.

Null values are detected and managed if they are input values, but they cause a fault if they are injected in the composite services interactions. *Null* values management could be added in the process.

Table 2. Reactions of the complex service with respect to data fault injection

Service	Data	Data fault Typo	Null	Misalignment
Shop	Customer Code	ND	D(0)	ND
Warehouse	Item List	D(2)	F	D(18)
Warehouse	Availability	D(2)	F	D(18)
Supplier	Item List	F	F	D(17)
Supplier	Availability	ND	F	D(17)

Operation	Data	Data fault Typo	Null	Misalignment
Shop (ReceiveOrder - Request)	Customer Code	ND	D(0)	ND
Shop (ReceiveOrder - Request)	Item List	D(1)	D(0)	ND
Warehouse (Check availability - Request)	Item List	D(2)	F	D(16)
Warehouse (Reserve - Request)	Item List	D(2)	F	D(16)
Supplier (Check availability - Request)	Item List	F	F	D(15)
Supplier (Reserve - Request)	Item List	D(2)	F	D(15)

Misalignment faults are discovered at the end of the process; only if the input process messages are perturbed, data faults cannot be discovered, because all the data in the process are wrong.

6 Concluding Remarks

This paper has presented a method to analyze quality aspects related to complex services, based on the use of a testing tool. Two aspects have been investigated, namely the effect of delays on complex services and the reactions of the composed service to data faults. A support tool uses fault injection as the basic mechanism to test complex services through our method. Experimental results have been discussed.

In future work, we will consider a larger number of fault types and link them to possible repair actions on the process for data quality problems; we plan to enable the selection of the "best" data cleaning algorithms, based on the type of the detected data fault. This approach can contribute to increase the quality of a complex service. In fact it enables to detect data faults as early as possible and to correct the faulty data. This technique allows repairing the process without modifying the process structure.

Acknowledgements

This work has been partially supported by the WS-Diamond FET STREP project funded by the European Commission under IST FP6.

References

1. Papazoglou, M., van den Heuvel, W.J.: Service-Oriented Design and Development Methodology. Int. J. on Web Engineering and Technology (2006)
2. Pernici, B.: Summary Report on Service Design and Development. In: Cubera, F., Krämer, B.J., Papazoglou, M.P. (eds.) Service Oriented Computing (SOC). Number 05462 in Dagstuhl Seminar Proceedings (2006)
3. Console, L., WS-Diamond Team: WS-DIAMOND: An Approach to Web Services - DIAgnosability, MONitoring and Diagnosis. In: International e-Challenges Conference, The Hague (October 2007)
4. Cappiello, C., Francalanci, C., Pernici, B.: Time-Related Factors of Data Quality in Multichannel Information Systems. Journal of Management Information Systems 20(3), 71–91 (2004)
5. Carreira, J., Silva, J.G.: Why do some (Weird) People Inject Faults? SIGSOFT Softw. Eng. Notes 23(1), 42–43 (1998)
6. Looker, N., Munro, M., Xu, J.: Testing Web Services. In: 16th IFIP International Conference on Testing of Communicating Systems, Oxford (2004)
7. Sneed, H.M., Huang, S.: WSDLTest - A Tool for Testing Web Services. In: WSE, pp. 14–21 (2006)
8. Xu, W., Offutt, J., Luo, J.: Testing Web Services by XML Perturbation. In: ISSRE, pp. 257–266 (2005)
9. de Almeida, L.F.J., Vergilio, S.R.: Exploring Perturbation Based Testing for Web Services. In: ICWS, pp. 717–726 (2006)
10. Siblini, R., Mansour, N.: Testing Web services (2005)
11. Offutt, J., Xu, W.: Generating Test Cases for Web Services Using Data Perturbation. SIGSOFT Softw. Eng. Notes 29(5), 1–10 (2004)
12. Looker, N., Xu, J.: Assessing the Dependability of SOAP RPC-Based Web Services by Fault Injection. In: WORDS Fall, pp. 163–170 (2003)
13. Zhang, J., Qiu, R.G.: Fault Injection-based Test Case Generation for SOA-oriented Software. In: SOLI 2006. IEEE International Conference on Service Operations and Logistics, and Informatics, pp. 1070–1078 (June 2006)
14. Looker, N., Munro, M., Xu, J.: WS-FIT: A Tool for Dependability Analysis of Web Services. In: COMPSAC Workshops, pp. 120–123 (2004)
15. Looker, N., Munro, M., Xu, J.: Assessing Web Service Quality of Service with Fault Injection. In: Quality of Service for Application Servers, SRDS, Brazil (2004)
16. Cappiello, C., Ficiaro, P., Pernici, B.: HIQM: A Methodology for Information Quality Monitoring, Measurement, and Improvement. In: Workshop QOIS 2006, Tucson (2006)
17. Redman, T.: Data Quality for the Information Age. Artech House (1996)
18. Wand, Y., Wang, R.: Anchoring Data Quality Dimensions in Ontological Foundations. Communication of the ACM 39(11) (1996)
19. Modafferi, S., Mussi, E., Pernici, B.: SH-BPEL: a Self-Healing Plug-In for Ws-BPEL Engines. In: MW4SOC 2006. Proceedings of the 1st workshop on Middleware for Service Oriented Computing, pp. 48–53 (2006)
20. Batini, C., Scannapieco, M.: Data Quality. Springer, Heidelberg (2006)

Automated Approach for Developing and Changing SOA-Based Business Process Implementation

Uttam Kumar Tripathi[1] and Pankaj Jalote[2]

[1] Department of Computer Science and Engineering,
Indian Institute of Technology – Kanpur, Kanpur 208016, India
[2] Department of Computer Science and Engineering,
Indian Institute of Technology – Delhi, New Delhi 110016, India
uttam.tripathi@gmail.com, jalote@cse.iitd.ac.in

Abstract. Reducing development time for business process implementation and downtime for Change Management is an important task in today's fast changing market environment. We present an automated approach for developing and changing SOA-based business process implementation. We distinguish two layers: At modeling layer processes are modeled using BPMN and then at the implementation level they are implemented as a BPEL processes, which is further composed of a set of Web Services. At the core of our approach are a set of tools which perform the task of making changes, translating process description from one layer to another and tools for various other tasks of automated code generation. Finally we illustrate our approach for change management using the example of university registration system.

Keywords: Business Process Management, Service Oriented Architecture, Change Management.

1 Introduction

Every enterprise or institution has a goal which it tries to attain through its day to day running. This goal can vary over a broad range, for a financial firm it will be to make maximum profit for itself and its investors, for an public educational Institution the goal will be to provide education to its students (even to the most needy one) and for a non-government organization the goal will be to do social wellbeing (financial wellbeing of organization doesn't matters to them in an ideal situation). Whatever the goal might be in-order to attain them through their day to day running Enterprises use a set of business processes, which are created by the business executives. Executing these business processes in today's era makes heavy use of IT. The use of IT reduces the manual overhead of the execution of the process and also increases the overall efficiency.

Today Internet has become a major tool for enterprises with potential to offer unmatched flexibility, functionality and reach-ability at lower cost. With IT supporting the business processes, the key issue is to translate the business processes specified by business executives into software-enabled or software-implemented systems. This often involves building of software applications & can be time incurring and expensive. For improving the business efficiency, one of the key goals is to implement the business processes efficiently and quickly.

Traditionally, the business processes were often implemented by developing custom software. This is not only expensive & time consuming but also error prone. To promote reuse of software, such that applications can be developed fast, component-based software development approaches evolved. This led to the development of new & very promising approach of web-services based development.

With web services, different software services are provided, potentially by different vendors, over the network. The task of the application builder it to use these services, perhaps builds a few missing ones, & then delivers the application as an integration of these services. In this environment, BPEL[4] (Business Process Execution Language) is one of the standards to build applications using web services.

The business processes themselves are popularly specified using BPMN (Business Process Modeling Notation), which is a graphical-notation to express processes. Many tools are available to represent business processes in BPMN [3] like BPVA[11] and BPEdit[15].

With this the whole processes is simplified to: the business executives create the business process using BPMN, this becomes the input for software development, which may then use the technology of BPEL & web services for implementation. Our work is about automating parts of this life-cycle so that the time involved in the development gets reduced.

Once the Business Process has been successfully deployed and has been running then starts the phase of process management. The most important part of the management phase is to be able to handle changes that happen in the Business Process. As discussed in [1] changes are an ever occurring phenomenon in Business Environment. These changes can be a result of agility of the market, new client needs, or organizational restructuring (e.g. because of mergers & acquisitions etc). They affect the way of doing business because they do influence the design of business processes. Irrespective of what is the reason of the Change, it is important to be able to incorporate these changes in the implementations and bring the system back running with new requirements. Hence, the approach of process management should be such that it is able to accommodate these changes in the model and finally into the implementation as and when they occur.

For the goal of managing Changes in an effective manner we developed on the work mentioned in [1] and came up with an end-to-end solution for accepting changes at the modeling level and thereby initiating the task of changing the final executables in an organized fashion while ensuring that most of the steps continue to remain automatic. Automating the steps results in two major advantages, firstly it reduces the time of development/management and secondly it reduces the need for programmer thereby positively affecting the cost aspect of the project.

Figuring out efficient approaches for business processes management has been an area of study for quite sometime now. These issues have gained significant importance since the deadline periods for projects are shrinking and companies in-order to make maximum use of their available manpower are constantly devising approaches using which they can do so. Problem of change management is of significant importance because it is desired that corporations get their business process model back running and functional with as little downtime as possible. As defined in[1], changes are broadly classified as long-term changes and short-term changes. In [2], the authors have successfully presented an approach for handling short-term changes and managed agility in business process. For this they have used the approach of association of business rules with the overall business process. For the task of managing Long-Term Changes in [1] the authors have introduced a novel approach for handling Changes at the semantic level by making use of a central Change Management System. However previous work that has been done in automated development and change management lacks an integrated approach which can cover the complete life cycle of the development and management of business processes. We present an approach which is in-line with this requirement, is end-to-end and targets the shortcomings on the works done in the past.

In sections to follow we would talk in detail of about our work, section 2 is about approach for automating business process development. In section 3 we introduce the problem of change management in business processes and present an approach for handling it using atomic generic actions. Then we illustrate our approach using an example of Student Academic Registration Business Process in section 4. Finally we conclude and discuss possible future works in section 5.

2 Automating Implementation of BPMN Models in BPEL

In the approach that we have adopted for automating life cycle of business process development we made use of phased approach of development. Phased Approach is very common in Software Engg. [5], it helps in separating various tasks of development as independent sequence of activities. Further the adopted approach helps in creating business processes which are able to reflect the requirements perfectly.

The automated approach for development of business processes implementations that we are present consists of a BPMN modeling phase and a BPEL executable generation Phase. The output of phase-I would be operated by a BPMN to BPEL translator and would then act as an input for phase-II. In our study we used the BPMN2BPEL tool[10]. The approach considers the translator as a black-box so it will continue to work fine even for any other kind of translator or newer versions of translators handling larger subset of BPMN. However the limiting factor for the approach will be the core subsets supported by the translator, details of the core-subset of BPMN2BPEL (which are the limiting factor for our work) can be found in [10]. The overall life-cycle is outlined in Fig. 3, for the descriptions in the next sub-sections please follow this figure.

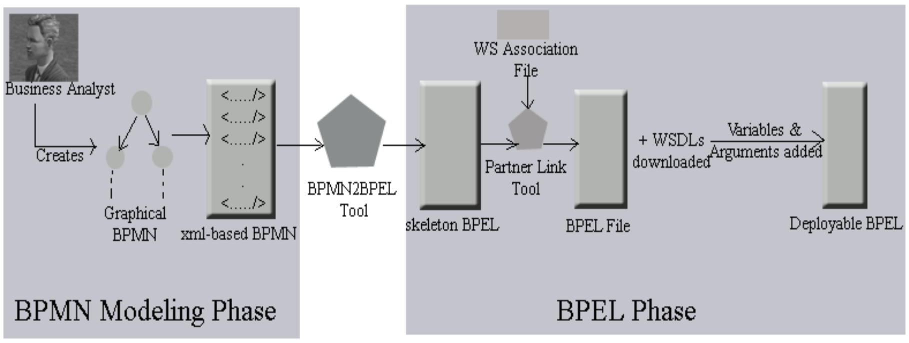


Fig. 1. Automating Phased Approach for Business Process Development

2.1 Requirement Capture and BPMN Model Generation

Development starts with generating BPMN model by the business analysts. During this phase flow/decision based requirements are captured for the target Business Process Implementation to be developed. Business Analysts of the company would meet and would agree on the business process that they want to have for the Enterprise. Then they would formally model the BPMN model using one of the graphical editors available, so that they can convey their needs to the technology team. There are a number of BPMN graphical editors which can be used for this. We have used Business Process Visual Architect [11] for generating graphical BPMN models.

Next using the graphical BPMN model we generate the xml-based BPMN model file, which would act as an input for BPMN2BPEL tool. This file will contain the same information as represented by the graphical model but in an xml format which can be parsed by the tool to extract out the relevant information. For the description of the process flow the tags like "node" and "arcs" are used. Here a node is used to represent a task in the process model and an arc represents the connection between two nodes. We have considered the core-subset of BPMN supported by BPMN2BPEL for our study, details of which can be found in [10].

2.2 BPMN to BPEL Translation

The single most important step of the development process is the translation step is where we generate the BPEL code for process flow from the BPMN model. As mentioned earlier for this step we use BPMN2BPEL tool, input being the xml-based BPMN model file and the output is a BPEL file, which would contain the code for all the process flow in BPEL. Details of how the translation algorithm works for the tool can be found in the reference cited above. Another relevant tool for the translation purpose can be eClarus[13].

The tool we are currently using has limitations and works for only core subset of BPMN artifacts, but in our approach the translator tool acts like a black-box and the approach doesn't depend on how the translator works. Therefore any other translator, which you are using or which might be available in future can easily fit in the overall solution. This step also performs validation of the BPMN model and incase the BPMN model was incorrect the tool raise errors on its invocation. The output file generated by this step is used for preparing the final BPEL file ready for deployment.

2.3 BPEL Executables Generation

As a standard for implementing a service-oriented architecture we rely on web services. To combine web services with processes we use the Business Process Execution Language for web services-BPEL which makes it possible to also define inter-organizational services. Once we have used the translator on our BPMN model file we would obtain a skeleton BPEL file containing the process flow description as described in previous sub-section. This BPEL file needs to be further refined and more information must be appended to it so that it is ready for execution. Most important information that needs to be associated during this phase is the web service association information so that the BPEL Engine knows that to which web service it needs to send a particular query. Incase there is a particular task for which no Web Service already exists for use then the programmer would be required to build and deploy a web service for performing that task.

For automating steps of this Phase we have developed a Partner Link tool which extracts the relevant WSDL for the web services to be used from their point of deployment and then inserts the desired keywords and declarations into the BPEL file. The WSDL files are also downloaded in the local directory as they are required for deployment of the business process. Next step in this phase is updating the variables and arguments and finally the business process is deployed and made running.

2.3.1 Partner Link Information Addition

Business processes execution makes use of an architecture which requires invocation of web services located across the web. Therefore an important step in the development process is associating the business processes with the various partner links that the business process is expected to invoke as a part of its execution. For our approach we build a tool for this step which takes as input the BPEL file received from the BPMN2BPEL tool and a formatted text file containing the information that which Web Service needs to be invoked for which particular task mentioning the deployment point of the web service, which is generally mentioned as the URL.

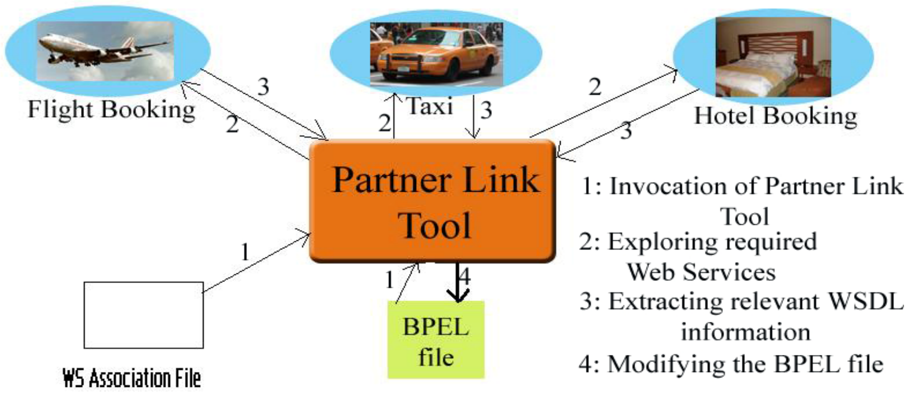

Flight Booking
Taxi
Hotel Booking
Partner Link Tool
WS Association File
BPEL file
1
2
3
4
1: Invocation of Partner Link Tool
2: Exploring required Web Services
3: Extracting relevant WSDL information
4: Modifying the BPEL file

Fig. 2. Outline of the working of Partner Link tool

Upon its invocation the tool explores the WSDL files at the deployment-point of the web services, extracts out the required information from the WSDL and inserts them in the BPEL file at the relevant position. This involves adding *xmlns*, *partnerLinks*, *portType, OperationName* etc. Moreover the tool also downloads all the required WSDL files in the working directory because they are required for the deployment step. UDDI[14] can be used for the discovery of Web Services to figure out which Web Services to use for particular tasks.

2.3.2 Variable Programming

Once all the partner link information is added to the BPEL file we now need to declare variables and prepare them for invocation of various partner link web services. This task in the current approach is handled manually by a programmer writing small snippets of codes for Variable programming but an approach of assuming a well-defined structure of comments in WSDL file just above variable structure declaration can be used for automating this task as well. Details of this will not be discussed in this paper.

Deployment and Running of the Business Process

Once the BPEL file has passed all the previous steps it is ready for deployment and execution. For running the BPEL process we used Oracle Business Process Manager [12] as the Engine. We need to create a *build.xml* file specifying the Business Process to be deployed and the set of web services it is going to use. Finally the invocation of the ***obant*** tool on this build file results in deploying the business process and making it available for usage. Later user can use the administrator tool provided at *http://localhost:9700/BPELConsole* to manage the deployed processes and the respective endpoints for their access.

3 Changes in Business Process Models

Changes are an ever occurring phenomenon in a business environment. These changes might surface because of the organization restructuring resulting from, for example a merger, acquisitions, shift in the client needs, technological changes etc. All these affect the manner in which business is done and therefore require change in the business process model. Since changes happen far too often therefore it is highly desirable that the business process management approach is able to accommodate these

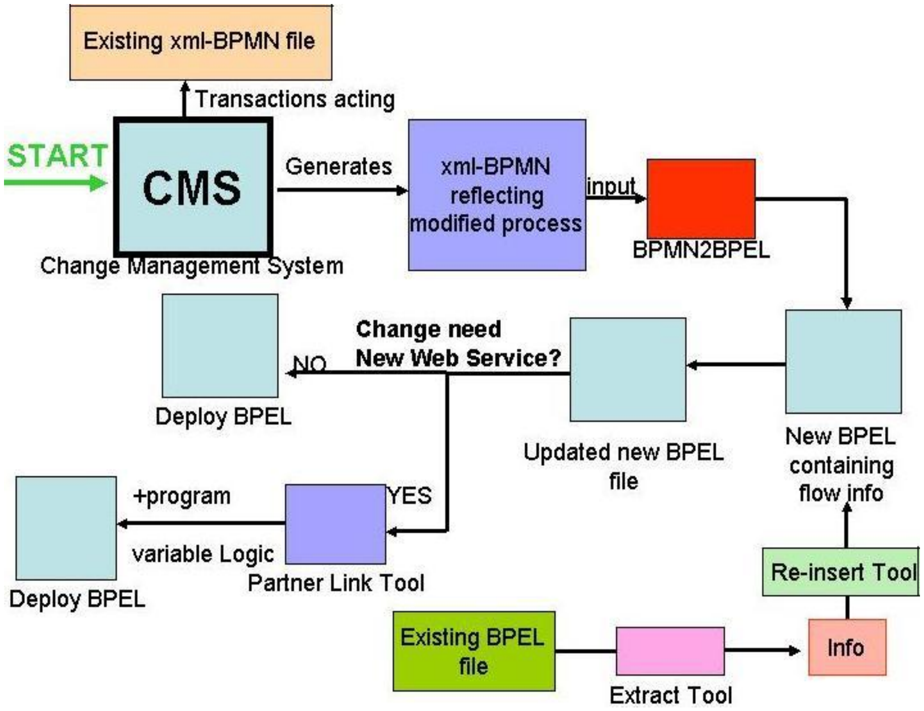


Fig. 3. Life-cycle for Change Management of Business Processes

changes in the model and further into the implementation as and when they happen and as seamlessly as possible. Changes in business processes are accepted at the model level and are then absorbed into the implementation layer. Key issue in change management is to reduce the downtime and cost involved while handling a change. It is clearly not desirable to redo everything and redevelop completely in the case of a change occurring. Our approach makes use of all the existing codes, models and implementations and tries to minimize the changes to be made manually. Furthermore our approach automates many of the tasks in an effort to reduce the time for implementing a change.

3.1 Introducing Changes in the Model

As discussed, the changes are introduced at the BPMN level. Then using a sequence of steps *(mostly automated),* we get those changes absorbed into the BPEL. This is depicted aptly in the figure below.

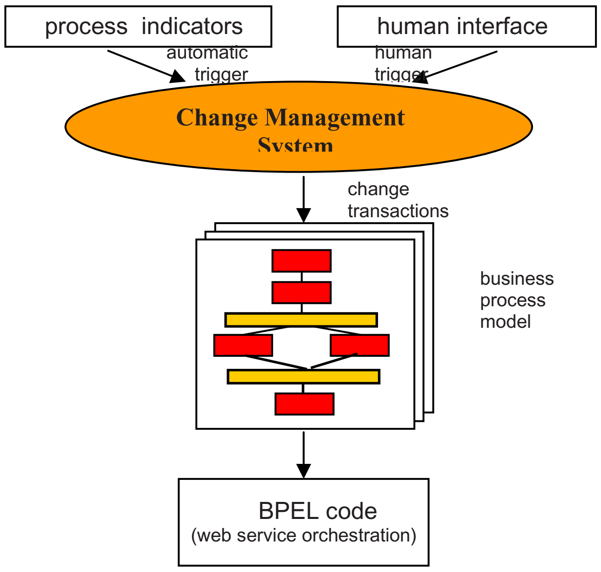


Fig. 4. The Change Management Schema

All the change made to the BPMN process model are made through a central Change Management System (CMS). CMS uses a set of atomic actions for making any change in the process model. For this purpose the desired change would be represented as a combination of these actions. We have 4 generic actions for introducing changes. These are:

- Create Action: This action is used to introduce a new node in the process model. It takes as input the unique ID user wants to have for the new node and the kind of node the user wants to have (*eg. event, task or gateway*). If no node already exists with the requested ID in the process file then the node is created, else a message is displayed informing the user to choose a different ID.

- Delete Action: Delete action takes as input the ID of the node wished to be deleted and results in removing the node from the process model. The action also removes all the arcs involving the deleted node. This ensures that any inconsistencies resulting from the deletion of the node are removed.
- Put In Sequence Action: This action introduces a new arc in the process model. For the invocation user needs to provide the ArcID desired, sourceNodeID and TargetNodeID.
- Remove Sequence Action: Remove Sequence Action takes as input the ArcID to be removed and removes the desired arc from the process model.

Once the model has been successfully modified then the next steps, described below, aim to bring those desired changes into the BPEL, the executable level.

3.2 Capturing Relevant Information from Existing BPEL

Once a change is made in the BPMN model the next steps involves generating BPEL file which would contain modified needs but it is almost certain that lots of existing code and implementation would continue to remain useful even for the modified BPEL. Therefore it is highly undesirable to having to loose on all the refinements made in the previous versions of BPEL and because of this reason we extract out all the implementation specific information from the previous versions of BPEL, store them on a persistent storage and later on as per the requirement we reinsert them in the modified BPEL file.

For this purpose we have built a tool which takes as input a BPEL file, parses it and extracts out the implementation specific information. Towards this we assume a standard BPEL file to be structured as follows:

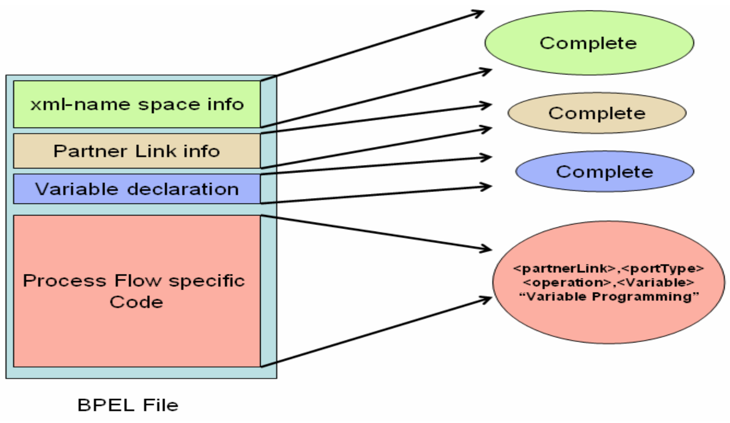

xml-name space info
Partner Link info
Variable declaration
Process Flow specific Code
BPEL File
Complete
Complete
Complete
<partnerLink>,<portType> <operation>,<Variable> "Variable Programming"

Fig. 5. Figure depicting what information is captured from various parts of a BPEL file

The tool on execution captures all the xml-name space info, partner link declaration info, variable declaration. This means that in the process flow specific code (generated by BPMN2BPEL tool for previous version) all the major changes that would have been made during the development process by the partner link tool and refinements made after that are captured by the tool. This is because these are the information which are not captured at the level of BPMN and were added during the later stages. Now we don't want to redo all the steps which have been done during previous cycles of development and capture them before generation of the modified flow-level code of BPEL using BPMN2BPEL tool. The information captured here is mainly stored with the *task_name* acting as an index, which we will use for re-inserting the information later on.

3.3 Generating New BPEL Flow File

After the steps of modifying the BPMN model through generic actions using Change Management System, and capturing the implementation based information from previous BPEL file now is the time to generate the new BPEL flow file using the BPMN2BPEL tool, to reflect the modified requirements. This step is similar to what we have already discussed while talking about automated phased development approach. The BPMN2BPEL tool will take the modified xml-BPMN file as input and will generate flow based BPEL file representing the modified process model.

3.4 Inserting Back the Relevant BPEL Information

Once we have generated the flow-based BPEL for modified process model, then we use a tool for inserting back all the implementation related info captured earlier. Since the information was stored with *task_name* acting as an index therefore the tool first searches for the relevant position for inserting the information in the BPEL file by looking for the *task_name* and then appending the information at the desired location. Incase a task had got deleted during the process of modification then it won't be found during the search and hence its information will not be appended. On the other hand if a new task was added then its information would not had been stored by the extraction tool and hence its details would continue to be set as default in the BPEL file and one would be required to run the partner link tool.

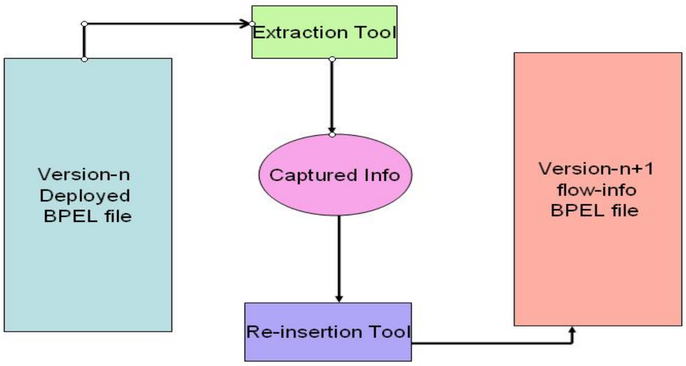


Fig. 6. Working of the Extraction and Re-insertion tool with different versions of BPEL

This two step process of extraction of the implementation information from the previous version of BPEL file and then inserting is back in the modified version of BPEL file helps in automating the passing over of the relevant implementation code from version-i of BPEL onto version-(i+1) of BPEL. Automation reduces both the time of development and the cost of development.

3.5 Running Partner Link Tool if Required

If the changes made in the BPMN did not involve introduction of a new task then after re-inserting the implementation based information from the previous version of BPEL we would be ready to deployment of the new BPEL file. Whereas if we had introduced new tasks while making the change then it would be required to run the partner link tool and associate newer Web Services with the process.

For this similar to what we had discussed during development section we would invoke the Partner Link tool, with inputs being the modified BPEL file and a Web Service association file. This would results in downloading the required WSDL files and adding the required information into the process file from the WSDL files. Also incase of adding new task this step may be also be followed by some variable programming steps.

3.6 Deployment and Running of the Modified Business Process

Finally our modified Business Process is ready for use, we can go ahead and deploy new BPEL file along with the WSDL files. Upon which if the process is deployed with the same endpoint as that of the previous version then for the end-users the change will appear completely transparent such that they can continue using the old Service end-point to start using the new business process implementation!

4 An Example

We would now demonstrate the approach for change management using an example. Registration is a very common process which is carried out by the students of almost every university at the beginning of a new semester. The process involves registering for the courses; payment of the fees and finally when the academic office accepts the registration request the registration is considered to be complete. Following is simplified BPMN model for the registration process.

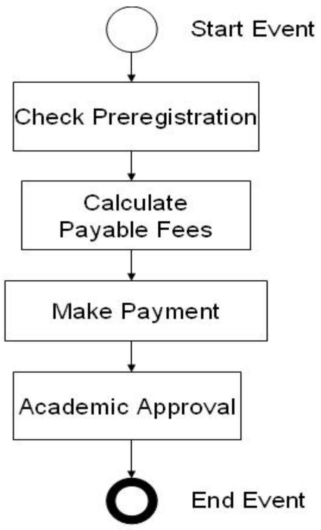
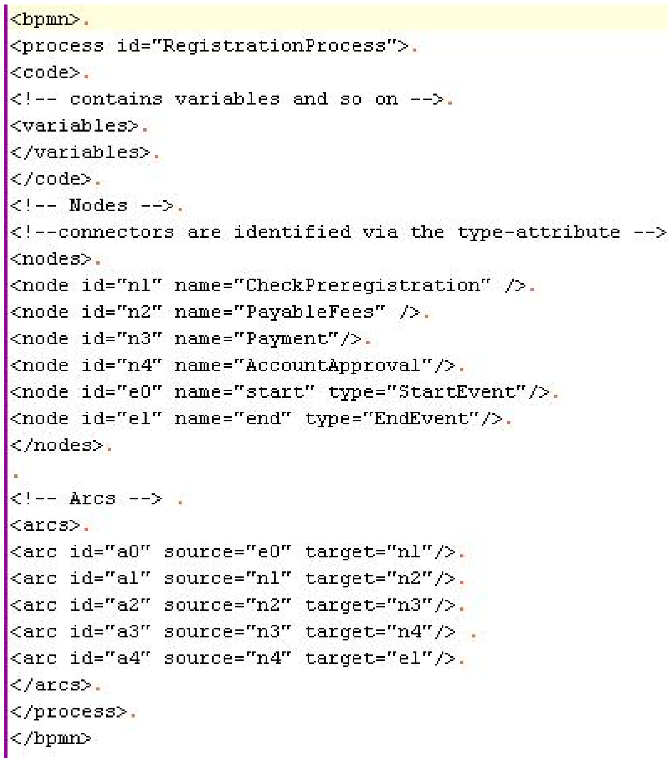


Fig. 7. Simplified graphical BPMN model for registration process at IIT



```
<bpmn>.
<process id="RegistrationProcess">.
<code>.
<!-- contains variables and so on -->.
<variables>.
</variables>.
</code>.
<!-- Nodes -->.
<!--connectors are identified via the type-attribute -->
<nodes>.
<node id="n1" name="CheckPreregistration" />.
<node id="n2" name="PayableFees" />.
<node id="n3" name="Payment"/>.
<node id="n4" name="AccountApproval"/>.
<node id="e0" name="start" type="StartEvent"/>.
<node id="e1" name="end" type="EndEvent"/>.
</nodes>.
.
<!-- Arcs --> .
<arcs>.
<arc id="a0" source="e0" target="n1"/>.
<arc id="a1" source="n1" target="n2"/>.
<arc id="a2" source="n2" target="n3"/>.
<arc id="a3" source="n3" target="n4"/> .
<arc id="a4" source="n4" target="e1"/>.
</arcs>.
</process>.
</bpmn>
```

Fig. 7. (*continued*)

Using this BPMN model the technology team would develop the required web services; make use of the 3rd party web service for credit card payment. Next they would generate the BPEL executables and make the process running. This would result in making the process running and available for use for performing registration online.

IITs are actually national institutes and their aim is to provide quality education to even to the neediest students. They provide scholarship for this purpose but according to the current process the fees has to be paid before hand and they get refund for it later on. This can seem a bit redundant and suppose because of the same reason the institute decides to modify its registration process. It wishes to identify if a student is eligible to fees waiver and if it is so then he does not needs to go through the payment steps and can directly proceed to the final step of academic approval. Using the approach suggested in previous sections this change would be first introduced at the BPMN level using CMS and then would be brought into the BPEL executables.

The changes that need to be done are as follows:

1. Perform a Check after Preregistration: Check if the person is eligible for scholarship
2. IF ELIGIBLE: Go directly to the Academic Approval step
   IF NOT_ELIGIBLE: Perform the fees payment steps.

For making these changes the CMS would be invoked to carry out following actions:

- **<Create> "**Gateway" c1 "XOR-Split"
- **<Create> "**Gateway" c2 "XOR-Join"
- **<RemoveSeq>** a1
- **<RemoveSeq>** a3
- **<PutInSeq>** a5 n1 c1
- **<PutInSeq>** a6 c1 c2 "FEESWAVIED=TRUE"
- **<PutInSeq>** a7 c1 n2 "FEESWAVIED=FALSE"
- **<PutInSeq>** a8 n3 c2
- **<PutInSeq>** a9 c2 n4

This would result in modifying the BPMN model to reflect the desired change. Following this the BPMN2BPEL tool would be invoked to generate the BPEL as per the modified requirement. Since this change didn't require any new service to be used therefore it would not require any manual interference and after the use of extract and reinsert tool the new BPEL file would be ready for deployment and use!

5 Conclusion and Future Work

As stated in the introduction the basic goals that we had set for ourselves while working towards making business process development more efficient and adaptable to changes was to reduce the time that is involved in building new implementations and modifying existing ones. For this purpose we used the approach of automating the life-cycle of both development and change management. Further for change management we adopted the approach which involved reuse of the existing models, code and implementation which reduced and rework involved and thereby again positively affecting the development time. While reducing the time of development and change management we had to ensure that the Quality does not get compromised and it is taken care or very nicely. The most important concern of missing on the requirements while developing the model or making changes is taken care of as follows:

- We separate the model layer and implementation layer thereby giving Business Analysts a level which involves least amount of technical detail and using which they can model/change their business process. This is important because after all they are the driving force in the running of an organization therefore it is important to capture their requirements very efficiently.
- As mentioned in [5] apart from several other best practices documents on Software development it is desired that before going ahead with any development phase first a design of the requirement is agreed upon and is clear. This helps in reducing the possibilities of missing on requirements because bringing new features later on involves heavy costs. We first create a model of the process we have to implement and then we go ahead with its implementation.

Also it would be worth noting that we generate the flow-level BPEL from the BPMN model automatically and ***not*** creating it separately by having BPMN model as a reference. Adopting this approach helps us create exactly the same BPEL code as represented by the BPMN model without any ambiguity. This further enhances the quality aspect of development.

Since our solution involved the usage of BPEL which has strong linkages with the concept of SOA, thereby we do reuse of existing Web Services for the tasks which we desire to perform and do-not develop them on our own. This reduces the development time considerably and also brings in the usage of Component Based Software Engineering (CBSE) in the overall development/Change Management process.

Thereby it is clear that we have proposed a schema which has been able to achieve various goals that we had set for making the process of development of business processes and Change Management highly efficient. These architectures are flexible enough to accept other tools which are able to perform the desired task with higher efficiency. In the research tests performed so far on these proposed solutions the results obtained have been very impressive in reducing the development time for new processes and downtime at the time of making changes.

Future work for this would be to build new interface for Change Management System which can perform changes at the graphical level, which in-turn can be combined by a BPMN-BPEL translator, having input interface accepting graphical BPEL. Moreover currently the WS Association file is generated manually, but by using UDDI and some algorithm for dynamic composition of Web Services likes [16], this can be automated thereby giving the freedom to use the most efficient service available.

References

1. Tripathi, U.K., Hinkelmann, K.: Change Management in Semantic Business Process Modeling. In: IEEE International Symposium on Autonomous Decentralized Systems (March 2007)
2. Hinkelmann, K., Probst, F., Thönssen, B.: Agile Process Management Framework and Methodology. In: AAAI Spring Symposium on Semantic Web Meets e-Government, Stanford University (March 2006)
3. http://www.bpmn.org
4. http://www.ibm.com/developerworks/library/ws-bpel/
5. An integrated approach for software engineering, Pankaj Jalote, Springer press
6. http://www.soa.com
7. http://www.csse.monash.edu.au/~hws/CBSE10/
8. Campbell, S., Mohun, V.: Mastering Enterprise SOA with SAP NetWeaver and mySAP ERP. Wiley, Chichester (2006)
9. Karch, S., Heilig, L.: SAP NetWeaver. Galileo Press (2004)
10. Ouyang, C., van der Aalst, W.M.P., Dumas, M., ter Hofstede, A.H.M.: Translating BPMN to BPEL. BPM Center Report BPM-06-02, BPMcenter.org (2006)
11. http://www.visual-paradigm.com/product/bpva/
12. http://www.oracle.com/technology/bpel
13. http://www.eclarus.com/
14. http://www.uddi.org
15. http://www.bpiresearch.com/Resources/RE_BPEdit/re_bpedit.htm
16. Zeng, L., Benatallah, B., Dumas, M., Kalagnanam, J., Sheng, Q.Z.: Quality Driven Web Services Composition. In: WWW 2003. Presented at World Wide Web Conference, Budapest, Hungary (2003)

A Phased Deployment of a Workflow Infrastructure in the Enterprise Architecture

Raf Haesen[1,2], Stijn Goedertier[1], Kris Van de Cappelle[3], Wilfried Lemahieu[1], Monique Snoeck[1], and Stephan Poelmans[2]

[1] Department of Decision Sciences & Information Management, Katholieke Universiteit Leuven, Belgium
firstName.lastName@econ.kuleuven.be
[2] Vlekho Business School, Belgium
firstName.lastName@vlekho.wenk.be
[3] KBC Bank & Insurance, Belgium
firstName.lastName@kbc.be

Abstract. Many organizations migrate to service-oriented architecture (SOA) since it caters for the demanded flexibility and reusability in information systems. Besides delineating appropriate business services, a mechanism for coordinating these services is needed to support business processes. The current state-of-the-art falls short in realizing that goal since existing standards and software packages tend to neglect existing enterprise architectures. Moreover they assume a central position in the architecture from which they control all services according to prescriptive process models, which makes them rather useless in a realistic setting. Therefore we introduce four dimensions to classify workflow engines that reflect the degree of support for the presented requirements. Subsequently we combine these dimensions to describe a phased roll-out of a solution that fulfills the requirements. That solution is currently deployed at KBC Bank & Insurance Group.

1 Setting the Scene

This paper presents some practical viewpoints on the introduction of workflow management as we derived them at KBC Bank & Insurance Group, hereafter called KBC. The financial group is one of the top three bank-insurers in Belgium with a key position in Central-Europe. It has chosen to adopt a SOA approach as an enterprise-wide architecture that is able to support the bank's internal and external growth for the next decade. While most existing functionality is currently transformed into services, automated support for the coordination of these services is generally missing. KBC has set its eyes on workflow management as a key enabler of such coordination.

At KBC, as in many organizations, enterprise systems are designed according to a layered architecture. When distributed systems where not yet feasible, all layers were present on one single back-end, such as a mainframe. In Fig. 1a, the presentation layer represents access to the business logic layer via a terminal.

With the advent of Internet technology, browser-based clients could be installed virtually everywhere while still using the same mainframe. In order to ensure robustness and consistency, the mainframe continued to support the two-phase commit protocol of transactions. Therefore additional layers were added between the presentation and business logic layer to maintain session state during the execution of one activity that spans multiple transactions. On the one hand, the presentation logic layer maintains the screen flow, while on the other hand the assembling logic layer establishes the connections with the mainframe. These layers are typically realized using a web application server, which is illustrated in Fig. 1b.

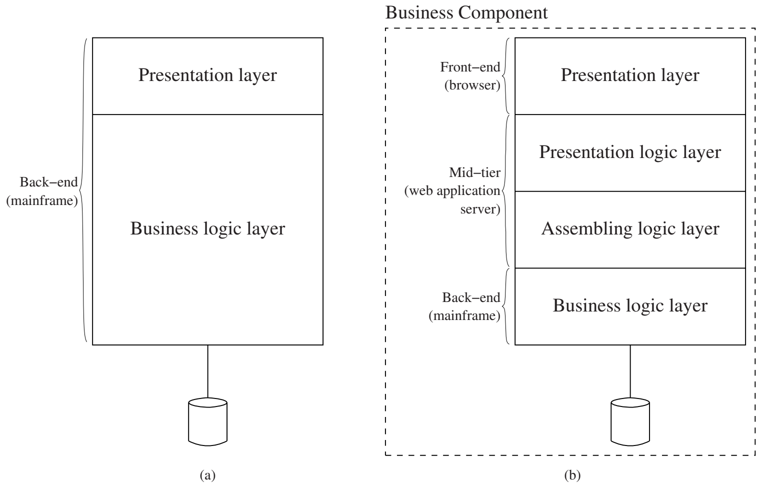


Fig. 1. Architecural layers

The construction of enterprise systems as presented in Fig. 1 clearly adheres to the principles of Component Based Development as presented by Herzum and Sims [1]. Indeed, each business component – itself being composed of distributed components – represents an integrated business concept that interacts as a whole with other business components. SOA goes one step further, in that it also governs interactions across the layers of each business component, by exposing the functionality in each layer as services [2]. This reasoning shows that it seems feasible to deduce services that encapsulate business data and associated business logic (called **business concept services**) and services that support single activity types (called **activity services**). Services that support the coordination of multiple consecutive activities are generally missing. These services are called **workflow services**.

Workflow services are the main point of interest in the remainder of this paper that is structured as follows. Since it is important to clearly define all workflow relevant terms, these terms and their definition are summarized in section 2. Section 3 elaborates on the stringent yet realistic business requirements concerning workflow management, for which current approaches do not offer a satisfying solution. Section 4 introduces four basic dimensions to classify workflow engines, which reflect the degree of support for the presented requirements. Subsequently, these dimensions are used in section 5 to derive a phased introduction of a workflow management system that satisfies the requirements. Finally, section 6 compares our results with some related work and section 7 presents some conclusions.

2 Relevant Terminology

The same workflow-related terminology is often used in many different contexts with different meanings. In this paper we provide a classification of workflow systems. It is therefore important to agree on the meaning of a number of terms that are relevant to workflow. Although the Workflow Management Coalition (WfMC) provides a good starting point [3], several definitions had to be modified for two reasons: firstly, the WfMC standards mainly focus on user-interfacing processes enacted by means of task distribution while they tend to neglect actor autonomy and straight-through processes, i.e. processes that do not require human interaction. Secondly, the definitions generally lack consistency since, for example, the definition of a term is often copied (sometimes with small modifications) in another definition instead of referring to that term.

– A **business process** is a set of one or more **activities** that collectively realize a business objective or policy goal, normally within the context of an organizational structure defining **organizational roles** and relationships.
– **Workflow** is the computerized facilitation or automation of a **business process**, in whole or in part.
– An **activity** is the description of a piece of work that forms one logical step within a **process**. An **activity** may be a manual activity, which requires an **actor**, or an automated activity, which may require an **actor** (user-interfacing activity) or not (fully automated).
– A **process instance** (**activity instance**) is the representation of a single enactment of a **process** (**activity**), including its associated data. Each **instance** represents a separate thread of execution of the **process** (**activity**), which may be controlled independently and will have its own internal state and externally visible identity, which may be used as a handle, for example, to record or retrieve audit data relating to the individual enactment.
– A **workflow participant** is a resource (**actor** or system) that performs an **activity instance**.
– An **actor** is a person who fulfills one or more **organizational roles**.
– **Workflow relevant data** or **context** is data that is used by a **workflow engine** to determine the **state transitions** of a **process instance** and **activity instance**, for example during the evaluation of **transition conditions** or during **workflow participant** assignment.
– **Workflow control data** is data that is managed by the **workflow engine**. Such data is internal to the **workflow engine** and is normally not accessible to applications.

- **Process instance state** (**activity instance state**) is a representation of the internal conditions defining the status of a **process instance** (**activity instance**) at a particular point in time. Most **workflow engines** maintain such status information as part of their **workflow control data**. The **process instance state** can generally be derived from the **state** of the **activity instances** of which the **process instance** is composed.
- A **state transition** is the change of one **process instance state** (**activity instance state**) to another **process instance state** (**activity instance state**).
- A **transition condition** or **process rule** is a logical expression that decides whether a **state transition** can occur or not.
- A **work item** or **task** is the representation of an **activity instance** to be processed by an **actor**.
- An **organizational role** is a collection of attributes, qualifications and/or skills that can be used for authorization or to identify a specific party in a process.
- A **workflow engine** is a software service or component that provides the run time execution environment for **process instances**. It interprets the process definition and interacts with **workflow participants**.
- A **workflow management system (WFMS)** is a system that defines and manages **workflows** (build-time) and enacts these **workflows** by means of one or more **workflow engines** (runtime).

3 Business Requirements

At KBC, as in many information-intensive organizations, the introduction of a vast number of information systems, tools and frameworks creates a complex and heterogeneous environment. Nevertheless, this environment contains many valuable and expensive assets that can not be simply dismissed. We now elaborate on three requirements that are particularly prevalent in such a context. The first requirement encompasses process modeling flexibility while the latter two take an operational point of view.

Firstly, it is generally impossible to model the exact control flow of many processes due to their complex nature. This certainly applies to unstructured processes in which interaction with human actors is required. It is rather complex to explicitly model a set of activities that can be performed in an arbitrary order by the actor. Moreover, it is not always possible to predict every event that might influence the process. On the other hand, structured processes, such as the ordering of office supplies, are suited to be explicitly modeled. Since straight-through processes do not involve any human interaction but only information systems, explicit process models are appropriate to describe these kinds of processes as well.

Secondly, an important yet seldom discussed topic about workflow management systems (WFMS) is the position they assume in the existing enterprise architecture. As discussed in section 1, an average information system already contains – hopefully neatly separated – services that execute business logic, manage business data and offer support for the execution of activities. Nevertheless, a WFMS usually enforces the design of proprietary user interfaces. This is reasonable since the WFMS then has complete control over which data is entered during which activity. However the value of an in-house user interfacing framework is often too high to be abandoned. Moreover, third-party software packages (such as ERP and CRM packages) commonly cover multiple architectural layers of multiple business objects that also have to coexist with a WFMS.

Finally, even if a WFMS can operate in a heterogeneous environment, it will still require a central position in that every activity must be coordinated by the engine. This way the workflow engine can successfully maintain the state of its process instances. Again, this is an undesired situation for two reasons: firstly it should also be possible to perform activities without being guided by a workflow engine, especially for unstructured processes. For example, a senior claim handler with many years of experience knows which activities he has to perform to handle a claim without the assistance of a workflow engine. The obligation to follow a predefined path of activities might even be counterproductive. A second reason reinforces the need to work outside the context of a workflow engine: an actor must be able to perform activities if the workflow engine (temporarily) fails. This allows the workflow engine to be installed on less critical machines with lower Service Level Agreements.

4 Positioning Workflow Management Systems

Existing workflow management systems fail to support the presented business requirements because of their particular view on the modeling and enactment of processes. This section proposes four basic dimensions that can be used to classify these views. On the basis thereof workflow management systems can be assessed.

4.1 Process Modeling Paradigm: Procedural Versus Declarative

In a classical **procedural** approach, a business process model is much like a script since it contains explicit, prescriptive information about how processes should proceed. The most prominent example of a prescriptive process description standard is BPMN [4], which mainly focuses on the temporal ordering of activities: the resulting process models (only) show *when* an activity can be executed, depending on preceding activities and some conditions. The main advantage of these process models is that they can easily be interpreted by humans. However, the disadvantage of these languages is that they assume an explicit, often over-specified process context that leaves no freedom to maneuver at execution time. Consequently, business processes need to be modeled explicitly to the greatest level of detail. In practice, the explicit modeling of business processes is a work of gargantuan proportions.

In a **declarative** approach, business processes are modeled using declarative languages. Such languages contain descriptive information about the constraints that govern the activities in business processes. A minimal yet sufficient set of constraints leaves as much freedom as is permissible at execution time for determining a valid and suitable workflow. Instead of (over-)specifying when a condition must be evaluated, constraints are like invariants that should always hold. The general advantage of this approach is that known process constraints can be predefined at design time whereas requirements that cannot be anticipated are dealt with at runtime. On the other hand, it is generally not straightforward to check whether a set of constraints is complete and correct.

4.2 Degree of Centrality: Middleman Versus Optional

One of the key questions is to which degree the workflow engine will participate in the actual business processes. The two extreme options are the workflow engine as the middleman and the workflow engine as an optional component.

A workflow engine behaves as a **middleman** if no change can occur in any system by any workflow participant without intervention of the workflow engine. Hence such an engine knows all process definitions, all possible events, all actors, all systems etc. Its main advantage is that everything is under control of the engine and no-one or no system can make mistakes or overrule business regulations or policies. On the other hand it has several disadvantages: firstly, many processes can not be managed by a middleman, because it is impossible to define every possible path, including all events that may impact a process. Secondly, many information systems can not participate in a controlled environment like this. Finally, the explicit modeling of all scenarios and exceptions makes the solution too expensive. Benefits will never reach the level of investments.

A workflow engine is **optional** if it is perfectly possible to perform activities without informing or being guided by that engine. The decision to use the engine is left to the actor or may depend on the role of the actor. This option is superior since it does not restrict all access to the information systems to one central point. However, this type of engine is only capable of returning possible activities that can be executed. Other types of process rules that must be obeyed, such as authorization rules, must be moved to the activity (service) level since they are not automatically controlled by the engine.

4.3 Statekind: State-Unaware Versus Stateful Versus Stateless

A workflow engine can be state-aware or state-unaware. A workflow engine is **state-unaware** if it manages no process instances, but merely knows all process definitions. All information about a concrete process instance must be provided as input. A **state-aware** workflow engine on the other hand manages process instances. To do so, the engine can work in a stateful or stateless mode.

A workflow engine is **stateful** if the state of process instances is maintained as part of the workflow control data, i.e. if it resides in the workflow engine itself. It is a huge challenge to keep the state in the workflow engine synchronized with reality. In fact, this is only possible for workflow engines that act as a middleman, i.e. when all activities are covered by the engine. This is impossible when any business object, referenced by a process instance, can be changed by other instances or activities that might not be under control of the workflow engine. Moreover it is unlikely that it is possible to define all possible events, although this is the only way to ensure that the state is correct.

A workflow engine is **stateless** if the state of process instances is derived from the state of business data, i.e. one or more business objects. Every time the workflow engine has to determine the next possible activities, it reconstructs the state through the use of business concept services. This can be seen as the extension of one of the key principles of the case handling paradigm [5] that states that an activity is completed if the associated mandatory data objects are entered.

Although this approach allows the execution of activities outside the control of a workflow engine, it has some disadvantages. Firstly, the reconstruction of state can create serious performance issues. Assume for example that the workflow engine only knows the claim case ID. To reconstruct state it has to query all objects, damages, parties, etc. involved in this claim case to collect their attributes. Therefore it might be better to add "meta-state" to a business object that explicitly indicates the state of that object. For example, the state of an order (requested, shipped, paid, etc.) can be added as an attribute of order. On the other hand, this creates the danger of moving process information in the business objects, only for the sake of the process. This must obviously be avoided since it impedes process flexibility. Secondly, there is a concurrency issue: if multiple actors modify the same business object at the same time, then the system can be left in an inconsistent state. This issue is also identified in [6] as one of the drawbacks of the case handling paradigm. Existing commercial WFMS lock an entire case if one actor starts performing activities, which is a serious weakness. We refer the reader to [7] for a detailed discussion about stateless process enactment.

4.4 Degree of Activity: Active Versus Passive

A workflow engine plays a **passive** role if it contains a set of process rules that are executed upon a request, i.e. an external consumer takes the initiative to call the engine. It determines the next activities that can be executed together with their workflow participants, without actually invoking activities. A passive workflow engine needs state and context as input to execute the process rules. Optionally it can gather additional context by invoking some business concept services.

A workflow engine has an **active** part if takes itself the initiative to enact processes. It makes sure that the processes are really being executed by actually invoking the activity services that require no human interaction. Moreover the engine can start the right activity service (i.e. show the right screens) when the actor selects an activity. Additionally, it takes care of the distribution of tasks for activities that require this.

5 Discussion

As can already be deduced from the previous section, the presented dimensions to position workflow engines are not entirely orthogonal. The following principles and constraints can be summarized:

– An optional workflow engine must be stateless. Since business objects are not exclusively managed by the engine, it must reconstruct the state of process instances when necessary.
– A stateful workflow engine must act as a middleman. If the state of each process instance is part of the workflow control data, the engine must know everything that might change that state.
– A stateless workflow engine embraces declarative process modeling. The fact that a stateless workflow engine derives its state from business objects is a declarative rule that constrains the enabling and completion of activities.
– A workflow engine with an active part can (re)use a (state-aware) passive counterpart. Since the active part enacts processes by invoking applications and distributing tasks, it needs to apply process rules and it needs the state of running instances, hence it can use a potentially existing passive part.

Now it is clarified how the dimensions can be combined, we summarize two combinations that represent two extreme kinds of workflow engine:

– A workflow engine is “Big Brother” if it enacts procedural processes as a stateful and active middleman. The engine keeps the state of all its process instances, which is possible since it acts as a middleman. It becomes the central point in the existing architecture, since every activity, event and state change must be under control of the engine. Despite its disadvantages, this is where existing workflow management systems are to be positioned.
– A workflow engine is an “Help assistant” if it is state-unaware passive. The engine does not interact with any system; it is only capable of returning next possible activities upon request. It is the responsibility of the actor to perform these activities by navigating to and invoking the right activity services. The workflow engine does not “see” what the actor is doing – state transitions within the workflow engine only occur after the actor has manually given feedback to the workflow engine. Its main advantage is its simplicity but its main disadvantage is that such an engine delivers very little added value.

It is obvious that none of these options provide a satisfying solution. One of the aforementioned principles states that a workflow engine must be stateless to be optional, which was one of the business requirements (see section 3). Therefore KBC decided to deploy the following phased solution to create a workflow infrastructure that satisfies the business requirements. Each additional step requires more effort (implementation, changes to existing systems, etc.) but increases the usability of the infrastructure.

Phase 1. Create a state-unaware passive workflow engine: this component, which interprets the process rules, should at least be present. It is important to isolate these rules from the applications, since it increases process flexibility dramatically. The passive workflow engine should be placed on a back-end.

Phase 2. Make the passive workflow engine state-aware: as already discussed, a stateless approach can not be avoided. However, given its disadvantages concerning performance, concurrency and maintenance, it should only be applied to processes that really require it. The state – whether explicitly kept or derived – is managed by business concept services and should consequently be positioned in the business logic layer. This component obviously uses the process rules component to consult the process definitions.

Phase 3. Add an active component that supports user-interfacing processes: this component needs to integrate with the existing user-interfacing framework (UIF) that enables the execution of activities. The active component presents activities to the actor and starts the selected activity (service), i.e. it shows the right screens. Additionally, it invokes the appropriate activity services that do not require input from an actor. Since the UIF already contains backend connectivity and support for creating screen-flows, this component fits perfectly on the mid-tier.

Phase 4. Extend the active component so that it also supports third-party systems: in a later stadium, the active component should be extended so it integrates with other packaged applications. Currently, an EAI framework enables back-end interactions using message-oriented middleware. In this phase the EAI framework should be extended in order to support new communication standards, such as the Web services stack [8].

In some parallel tracks, the other components of the workflow reference model of the WfMC [9] can be built, such as a task manager, administration tools, a history manager, monitoring tools, etc.

6 Evaluation and Related Work

The process modeling dimension is the primary dimension that draws the line between a process-driven and a process-aware SOA approach. Most researchers and commercial vendors consider a *process-driven* SOA approach [10,11,12]. In such an approach, services are generally seen as wrapped legacy systems, which are orchestrated using a prescriptive process description language, such as BPMN [4] or BPEL [13]. This paper clarifies the importance of a *process-aware* SOA approach, in which business processes are modeled using declarative languages. The fact that a stateless workflow engine derives its state from business objects is a declarative rule that constrains the enabling and completion of activities, which can be seen as an extension of the case handling paradigm [5]. Besides catering for an increase in process flexibility, this approach enables the decentralization of the WFMS and enables the execution of activities apart from the workflow engine. Other examples of declarative languages are the ConDec language [14] based on temporal logic and the PENELOPE language [15], which allows the expression of temporal deontic rules.

Apart from case handling systems, commercial workflow management systems do not consider declarative business modeling and enactment. Instead, workflow engines are generally evaluated according to the support they provide for a set of control-flow, data, resource and exception patterns as they are identified by the *Workflow Patterns* initiative [16]. Moreover, the generally underexposed operational perspective [17] on workflow management presented in this paper shows that an existing enterprise architecture heavily influences the choice for a workflow management system.

7 Conclusion

This paper presented some viewpoints on the practical introduction of workflow management in an organization migrating to service-oriented architecture. The pre-existing valuable information systems, tools and frameworks form a heterogeneous environment that puts many yet justly constraints, which are not supported by existing workflow management systems: a WFMS should not become the single access point to all the other systems, an actor should be able to perform activities apart from the workflow engine, and for many processes it is practically impossible to model all activity constraints – such as scheduling, authorization, task distribution, etc. – in an explicit way. The Workflow Management Coalition proposes standards that neither elaborate on the potential existence of an enterprise architecture [9,3]. Therefore we proposed four dimensions to classify workflow engines that reflect the degree of support for the presented requirements.

Based on the four dimensions, a phased roll-out of a workflow infrastructure was derived: firstly, a passive workflow engine should be built to interpret the processes rules. Secondly, that engine should be made state-aware, or more specifically, stateless to make the engine optional. Finally, an active component should be added that effectively enacts business processes by invoking the appropriate activity services.

Acknowledgements

We gratefully thank Pieter De Bruyn and Frank Zwertvaegher for their valuable input during the workshops at KBC. This work was partially funded by the KBC-Vlekho-K.U.Leuven research chair on ‘Service and Component Based Development’ sponsored by KBC Bank & Insurance Group.

References

1. Herzum, P., Sims, O.: Business Components Factory: A Comprehensive Overview of Component-Based Development for the Enterprise. John Wiley & Sons, Inc., New York (2000)

2. Greenfield, J., et al.: Software Factories: Assembling Applications with Patterns, Models, Frameworks, and Tools. John Wiley & Sons, Chichester (2004)
3. WFMC: Workflow management coalition terminology and glossary. Technical Report WFMC-TC-1011, WorkflowManagement Coalition (1996)
4. Object Management Group: Business process modeling notation (bpmn) – final adopted specification. OMG Document – dtc/06-02-01 (February 2006)
5. van der Aalst, W.M.P., Weske, M., Grünbauer, D.: Case handling: A new paradigm for business process support. Data and Knowledge Engineering 53(2), 129–162 (2005)
6. Reijers, H., Rigter, J., van der Aalst, W.: The case handling case. International Journal of Cooperative Information Systems 12(3), 365–391 (2003)
7. Haesen, R., De Rore, L., Goedertier, S., Snoeck, M., Lemahieu, W., Poelmans, S.: Stateless process enactment. In: Pattern Languages of Programming (PLoP 2007) (accepted, 2007)
8. Fremantle, P., Weerawarana, S., Khalaf, R.: Enterprise services. Communincations of the ACM 45(10), 77–82 (2002)
9. WFMC: The workflow reference model. Technical Report WFMC-TC-1003, Workflow Management Coalition (1995)
10. Marks, E.A., Bell, M.: Service-Oriented Architecture (SOA): A Planning and Implementation Guide for Business and Technology. John Wiley & Sons, Inc., New York, USA (2006)
11. Papazoglou, M.P., Georgakopoulos, D.: Service oriented computing: Introduction. Communications of the ACM 46(10), 24–28 (2003)
12. OSOA: Service component architecture – assembly model specification. version 1.00 (March 2007), http://www.osoa.org
13. Leymann, F., Roller, D.: Modeling business processes with bpel4ws. In: Nüttgens, M., Mendling, J. (eds.) XML Interchange Formats for Business Processes, Marburg, GI, pp. 7–24 (2004)
14. Pesic, M., van der Aalst, W.M.P.: A declarative approach for flexible business processes management. In: Business Process Management Workshops, pp. 169–180 (2006)
15. Goedertier, S., Vanthienen, J.: Designing compliant business processes with obligations and permissions. In: Business Process Management Workshops, pp. 5–14 (2006)
16. van der Aalst, W.M.P., ter Hofstede, A.H.M., Kiepuszewski, B., Barros, A.P.: Workflow patterns. Distributed and Parallel Databases 14(1), 5–51 (2003)
17. Jablonski, S., Bussler, C.: Workflow Management. Modeling Concepts, Architecture and Implementation. International Thomson Computer Press, London (1996)

Evie – A Developers Toolkit for Encoding Service Interaction Patterns

Anthony M.J. O'Hagan[1], Shazia Sadiq[1], and Wasim Sadiq[2]

[1] School of Information Technology and Electrical Engineering
The University of Queensland, St Lucia, QLD 4072
Brisbane, Australia
{tohagan, shazia}@itee.uq.edu.au
[2] SAP Research Centre
133 Mary Street, QLD 4000
Brisbane, Australia
wasim.sadiq@sap.com

Abstract. Facilitation of collaborative business processes across organizational and infrastructural boundaries continues to present challenges to enterprise software developers. One of the greatest difficulties in this respect is achieving a streamlined pipeline from business modeling to execution infrastructures. In this paper we present Evie - an approach for rapid design and deployment of event driven collaborative processes based on significant language extensions to Java that are characterized by abstract and succinct constructs. The focus of this paper is to provide proof of concept of Evie through encoding examples that are inspired by service interaction patterns. Where as the patterns provide business semantics, the Evie language provides a rapid means of encoding them at an abstract level, and subsequently compiling them to create a fully fledged Java-based execution environment.

Keywords: Service Interaction, Harmonized Messaging, Event Brokering.

1 Introduction

Process enactment systems traditionally rely on the control flow defined within the process model to drive the process. This approach has been successful in the past. However, it becomes arguable for Collaborative Business Processes (CBPs) that are characterized by asynchronous and highly dynamic business activity. In collaborative processes, it is expected that independent specialized application components both within and across organizational boundaries will be capable of detecting and responding to the events that dictate subsequent process flow. These events can be many, can arise at any time during the overall process and their (time of) occurrence cannot be anticipated by dependent components.

Modeling a collaborative process through the exchange of event data rather than through a rigid control flow between its activities is a significantly different albeit more natural way of capturing the logic behind collaborative processes. Thus, business activity takes place within application components, however the context for the business activity is provided by the event data. How the business activity deals with the data is not the question, instead capturing which business activity may need to be informed about a particular event, and when, is the question at hand.

The critical factor is that the process enforcement system be empowered with sufficient intelligence so that the appropriate action can be taken when a particular event notification arrives. This action basically consists of communicating the relevant data to the right process participant such as, an application component, a business activity performer, or a workflow management system, at the right time.

There have been several developments in this regard to assist in the modeling of business requirements that govern these interactions ([3], [4], [10]). However, it is the translation of business models into executable environments that remain a big bottleneck in the wider adoption of CBP related technologies. This is the basic premise of our work, that is, CBP are mostly designed by technical teams often software engineers, where high level models of limited or tedious functionality may prove unproductive.

Following on this premise, we argue that there is a need to provide developer tools to facilitate the deployment of event driven CBPs. In this paper, we present an approach that attempts to capture the dynamics of CBPs and the underlying event dependencies through a scripting language - Evie. Without compromising on the importance of a model-driven approach, the Java language extensions are intended to provide the power of a programming style language, but at a sufficiently high level of abstraction. The developed program is intended to serve two objectives: to serve as a source for setting up an execution environment; and to serve as a target for a high level model (if available).

We use the work achieved by [3] on service interaction patterns (SIP) as a source of business requirements in the context of event driven CBPs. Although there are a large number of service interaction patterns, we will only use some selected ones in this paper due to space limitations (for further examples see [14]). The purpose of these encodings is to provide proof of concept for Evie as a developers toolkit for rapid development of an executable environment for CBPs based on high level models such as the SIP patterns on one hand, and Java components on the other.

In the remaining paper, we first present the proposed approach based on Evie, and the basic features of the Evie language[1]. The next section presents related work in this area in order to establish the position of this work. Section 4 presents proof of concept encodings of selected SIP patterns. Conclusions and main contributions are summarized in the section 5.

2 Background

During the past several years, there has been extensive research on enterprise architectures in pursuit of the evasive business-IT alignment. The most significant technology development in the recent past impacting on enterprise architectures has been service oriented architectures or SOA [2]. Even though an essential stepping

[1] A more detailed coverage of Evie's syntax and semantics is covered in [14].

stone for *service enablement* of enterprise applications, web services standards do not provide the complete solution for CBPs.

Achieving communication between disparate enterprise applications through messaging is well established in message oriented middleware[2], with recent trends towards solutions that can scale beyond the traditional hub-and-spoke message broker. The extended functionality of the Enterprise Service Bus (ESB) [6] is currently a dominant approach in this respect, providing the ability to store messages and establishing streamlined *service communication*.

Recent developments from business software vendors have identified the need for solutions that go beyond *service enablement* and *communication* capability. These provide tools that allow multiple services both within and across enterprise systems to be collated into value added composite applications (see ESA & CAF from sap.com).

We observe that a critical aspect of current enterprise architectures based on the above approaches is the management of the rules for *service interaction* (serviceinterationpatterns.com). This functionality would naturally reside in middleware components and is the main driver for the approach presented in this paper. While there have been significant developments within the first two phases of service enablement and communication, the last phase of managing service interaction still holds many challenges.

CPBs are typically orchestrated by tools based on control flow such as BPEL. Difficulties in modeling service interactions through typical control flow constructs as found in workflow modeling languages are known to be ineffective in the CBP scenario due to the scale of options. Instead, approaches that utilize event processing have emerged as a more promising alternative [13]. Some operators and related event algebras can be found in: HiPAC [8], Compose [10], Snoop [5], RAPIDE [13], TriGS [16] and [7].

The Evie language is motivated by the need for rapid but reliable development of services that act as message brokers or gateways providing routing, transport and encoding mappings between disparate legacy and business partner servers.

There is significant evidence that such infrastructures are featuring prominently in current enterprise systems (see SAP Exchange Infrastructure, IBM WebSphere, BEA AquaLogic, Oracle Integration and Microsoft Biztalk). Consequently, the need to provide tools for rapid development, testing and deployment for broker and gateway services has increased manifold. The Evie language targets this aspect by delivering an abstract and succinct means of expressing broker business logic. Compiled Evie programs are deployed within an execution framework API and user interface tools that support rapid systems integration development and simulation testing.

The overall process for design and deployment of Evie applications can be summarized as: (1) A high level collaboration process is prepared by a business analyst using a model design tool that creates a set of graphical design artifacts. (2) A model compiler translates these into a skeleton Evie rule script that is augmented by a software engineer into a complete set of executable rules. (3) Finally, an Evie compiler translates Evie rules into Java components to be deployed and executed inside an Evie Framework server that exposes a set of broker services.

[2] See middleware.org

2.1 Evie Language

The Evie language defines a notation for event processing. It leverages existing technology investment by extending the Java language with additional constructs that describe dynamically ECA rules that implement service event behavior.

An Evie **event** is a CBP state change communicated between partner services that is observable at a given point in time. Events provide the impetus for process progression in CBPs. *Event types* identify primitive CBP events.

An **event condition** is a rule precondition that defines when a rule action executes. It may use event operators to express conditions requiring multiple events occurring over a period of time. Conditions may reference attributes of the event type (Payment) or event sender's service type (Customer):

```
receive Payment p from Customer c
  where c.status == "approved" && p.amount <= c.maxPayment
     OR delay +days(5)
```

An Evie **rule action** can execute arbitrary Java statements including conditional and iterative logic, compose and send output messages, read or update context variables and, interestingly, dynamically bind and activate new rule instances. Evie rules do not support fact inference or backtracking.

Rules can include **`when`**, **`repeat`** and **`exclusive`** modifiers and both Java conditions and Evie event conditions. Examples below will illustrate their semantics.

```
ruleStatement     ::= repeatRule ( then repeatRule )*
repeatRule        ::= [repeat] exclusiveRule
exclusiveRule     ::= exclusive exclusiveRuleList | whenRule
exclusiveRuleList ::= "{" ruleStatement ( ruleStatement )* "}"
whenRule          ::= when "(" eventCondition ")" statement
```

Fig. 1. Evie rule statement grammar

2.2 Evie Framework

The Evie execution framework is a Java API class library from which a standalone broker or gateway service is constructed. Its architecture is composed of four tiers that control external communications, event routing, rule execution, and data persistence.

CBP partners each agree that they will expose services that can be characterized by specific event behavior. The behavioral contract between these services will include agreement on ***event message types***, and ***rules*** governing message exchange. The behavioral contract of a service is referred to as its ***service type*** and typically corresponds to a business role. A ***service instance*** is any partner service that exhibits the corresponding service type (role) behavior. An Evie framework server can concurrently implement *multiple* service instances for *multiple* partners.

The **first tier** implements communications to external services. Evie developers can defer until deployment, decisions such as service end point addresses, channel multiplexing of service instances, transport protocols and event message encoding.

In the **second tier**, input and output events are routed between external channels and internal service instances. This isolation ensures that developers describe interactions with abstract event and service types and thus focus on conceptual business logic rather than communications wiring.

The core **third tier** manages the Evie scripts consisting of event-condition-action (ECA) rules [7] containing event conditions and corresponding actions. Compiled event conditions subscribe to input event patterns. As each input event arrives it is matched to a set of ***persistent event subscriptions*** corresponding to active rule instances. Each matching subscription activates a compiler generated ***event procedure*** corresponding to either the rule action or partial evaluation of a multi event condition. Evie supports ***persistent threads*** containing Java context variables. When an event procedure is fired, it is passed the persistent thread that created the original event subscription thus restoring the execution context of a rule instance. Context variables are used to compose and dispatch output events and compose rule instances.

The **fourth tier** employs a transactional object-relational mapping engine to persist and query service instances, events, event subscriptions and persistent threads.

3 Encoding Service Interaction Patterns

In this section, we illustrate the use and expressiveness of Evie through its ability to encode service interaction patterns as given in [4]. As a brief background we quote the basic motivation of their work:

"With increased sophistication and standardization of modeling languages and execution platforms supporting business process management (BPM) across traditional boundaries, has come the need for consolidated insights into their exploitation from a business perspective. Key technology developments in BPM bear this out, with several web services-related initiatives investing significant effort in the collection of compelling use cases to heighten the exploitation of BPM in multi-party collaborative environments. In this setting, we present a collection of patterns of service interactions which allow emerging web services functionality, especially that pertaining to choreography and orchestration, to be benchmarked against abstracted forms of representative scenarios."

The follow Evie code example will be used to illustrate Patterns 1-3.

```
A.1  service input rules User {
A.2    Alerter alerter = Alerter.members.first();
A.3    send new AlertRequest(new Date("15/08"), "Anniversary") to alerter;
A.4    // ...  execute non-blocked actions here ...
A.5    when (receive Alter alert from alerter correlate alert) {
A.6      // ...  execute blocked actions here ...
A.7    }
A.8  }
```

Example 1. Sending a reminder alert to a registered user

Pattern 1. *Blocking and Non blocking Send.* **Synonyms**: *Unicast; point-to-point send.*

Description. A party sends a message to another party.

Discussion: An Evie send statement is never blocked, however it can be used in conjunction with a corresponding receive condition to implement blocking behavior.

A timeout or other arbitrary exit condition can be included in the condition that waits for the reply event. A send statement can delay dispatching the event by the sender.

The recipient of a message is typically unknown at design time. The single recipient reference required for this pattern can be constructed from a run-time service repository query based on recipient service attributes. It may also be obtained from a referenced within a prior message sender or a message body attribute.

All Evie services of a given service type exhibit predictable event behavior implemented by a combination of rules and communications transport guarantees. By default, an Evie service type assumes a communications protocol that provides a reliable *exactly once* transactional message delivery that preserves the order of dispatch. Updates to input and output message queues participate in the same transactional state change associated with rule evaluation. In the example above, the message enqueuing resulting from the *send* and the activation of the *receive* rule are performed within a single transaction. This avoids a potential race condition whereby a reply is received before a corresponding receive rule is activated.

Pattern 2: *Receive*

Description. A party sends a message to another party.

Discussion: Messages are transactionally enqueued until the recipient service is available and ready to consume it via a `receive` event condition. The `receive` condition performs three functions: (1) it can declare message and role variables; (2) it subscribes to observe an input event pattern (3) and finally when a match occurs it consumes and binds input events to the event and role variables.

To facilitate replies, each event message includes a reference to its sender. Event conditions often constrain the sender to match a service type. To support this, service references includes the sender's service type.

The ProcessAfter event attribute can be set by a sender to delay processing by the receiver. A fault reply is sent by the Evie framework if the ProcessBefore attribute is set and has expired before a rule is activated that consumes the message. A sender can use this to ensure timely event processing of event data that can become stale. This is of particular use when an Evie server is down for a prolonged period of time.

Normally, when an input is not consumed by an active rule it is bounced back to the sender. In multi-party interactions, a network propagation delay can cause a race condition whereby a rules is not activated in time to receive an input event. A sender can avoid this by setting the ProcessBefore attribute to ensure that an event is preserved by the recipient and tested against newly activated rules up until the ProcessBefore deadline after which time it will be bounced (as normal) if unmatched.

Pattern 3: *Send/Receive.* **Synonyms:** *Request/Response.*

Description. A party X engages in two causally related interactions: in the first interaction X sends a message to another party Y (the request), while in the second one X receives a message from Y (the response).

Discussion: A response message is identified by its type and correlation with a prior request message. Unlike BPEL, Evie avoids explicit process instances; rather it employs message correlation alone to partition related events. This avoids the maintenance associated with creating and destroying distributed process instances since process threads travel with the event.

The Evie execution framework can reply with a correlated generic FaultMessage that indicates a failure to receive or process a message. Application specific exception message types may also be defined and used to indicate a fault.

Pattern 4: *Racing incoming messages.* **Synonyms:** *Racing messages* [12].

Description. A party expects to receive one among a set of messages. These messages may be structurally different (i.e. different types) and may come from different categories of partners. The way a message is processed depends on its type and/or the category of partner from which it comes.

Discussion: A set of Evie rules that are activated together can be identified as mutually exclusive by surrounding them with an `exclusive { … }` statement block that groups subrules into a new exclusive rule. When one subrule fires, the others are atomically deactivated. If more than one fires concurrently, then the first rule has precedence. A temporal condition can cause an exclusive rule to timeout.

A solution to a case example from [3] quoted below illustrates the expressability of being able to combine **then**, **when**, **exclusive** and **repeat** rule constructs. We use the internal events `Escalation` and `Deescalation` to record the current phase in a cyclic state machine implemented by making an exclusive rule to be repeating.

"The escalation service of an insurance company's call center may receive storm alerts from a weather monitoring service (which typically herald surges in demand), notifications of long waiting times from the queue management service, or notifications of low resourcing levels from the call center's HR manager. The receipt of any of these three types of messages by the escalation service triggers an escalation process (different processes apply to the various types of notifications). While an escalation process is running, subsequent storm alerts, queue saturation or low resourcing notifications are made available to the call center manager but will not trigger new escalations."

```
package evie.alert_escalation;

message Alert { String (50) reason;  }
message StormAlert extends Alert { }
message QueueDelayAlert extends Alert { }
message LowReourceAlert extends Alert { }
message AlertProcessed { }
message Escalation { Alert trigger; }
message Deecscalation { }

role input rule EscalationBroker {
  send new Deescalation() to self;  // Initialise in a Descalated state
  repeat exclusive {  // toggles Deescalation <-> Escalation states
    when (receive Deescalation deescalation) {
      exclusive {
        when (receive StormAlert alert1 from Weather) {
          send new Escalation(alert1) to HR_Mgr, self;
        }
        when (receive QueueDelayAlert alert2 from QueueMgt) {
          send new Escalation(alert2) to HR_Mgr, self;
        }
        when (receive LowResourceAlert alert3 from CCMgr) {
          send new Escalation(alert3) to HR_Mgr, self;
        }
B.25       }
B.26     }
B.27
B.28     when (receive Escalation escalation) {
B.29       send escalation.alert to EscalationProcessor;
B.30       repeat exclusive {
B.31         when (receive Alert alert) {  // matches derived types: C.2-C.4
B.32           send alert to HR_Mgr;
B.33         }
B.34         when (receive AlertProcessed from EscalationProcessor
B.35                       correlate escalation.alert
B.36               OR delay (mins(+60)) {
B.37           send new Deescalation() to HR_Mgr, self;
B.38         } then when (receive Deescalation) { }
B.39       }
B.40     }
B.41   }
B.42 }
```

Example 2. Alert Escalation rules (role declarations not shown)

Pattern 5: *One-to-many send.* **Synonyms:** *Multicast, scatter* [17].

Description. A party sends messages to several parties. The messages all have the same type (although their contents may be different).

Discussion. An Evie rule action can send an arbitrary number of messages of uniform or mixed type using a sequence of send statements. Arbitrary Java statements can be used to compose the events and conditionally execute the send statements. Each input event commences a new transaction context. All outgoing events triggered by that input event are committed to output queues in the same transaction. When an input event fires multiple rules, the transaction scope may encompass multiple rule actions.

Related Patterns: *Broadcast, Publish/Subscribe*. A single send can broadcast a uniform event to a role type or a subset of role members. Services subscribe to receive events by registering in the role (dynamically or at deployment time). Evie extends *publish/subscribe* by requiring services respond in compliance with the role's behavioral contract.

Multiple instances with a priori runtime knowledge [1]: Evie can enumerate a fixed list or dynamically construct the list at run-time. Each outgoing event can potentially initiate a parallel process. A corresponding multi-event condition can be used to synchronize control after full or partial replies from these recipients.

```
// A. Broadcast uniform RFQ to all Suppliers
send new RFQ(standard_terms) to role Supplier;

// B. Broadcast RFQ only to Suppliers who offer the required RFQ service
List<Supplier> suppliers = new List<Supplier>();
// Evie uses Hibernate SQL
suppliers = select("from Supplier where ? in Supplier.services", rfq.service);
send new RFQ(standard_terms) to suppliers;

// C. Broadcast customised RFQ to each supplier
for (Supplier supplier : suppliers) {
   send new RFQ(supplier.negotiated_terms) to supplier;
}
```

Example 3. Message Broadcast

Pattern 6: *One-from-many receive.* **Synonyms:** *Event aggregation* [13], *gather* [17].

Description. A party receives a number of logically related messages that arise from autonomous events occurring at different parties. The arrival of messages needs to be timely so that they can be correlated as a single logical request. The interaction may complete successfully or not depending on the set of messages gathered.

Discussion. Evie `receive` conditions can consume a sequence of events into a Java collection. Events may be filtered and the sequence terminated by an aggregate condition, temporal condition, a separate event condition or any logical combination. A group correlation expression is used to identify related events. This is distinct from event correlation that is used to filter events and identify process instances.

```
// A. Collect products under $10 until total price is $200 or Deadline occurs.
when(receive Order order where order.price < 10 into List<Order> orders
     AND (sum(orders.price) < 200 OR receive Deadline) {
}
// B. Collect next 10 A events into aList and process them unless B occurs
when(receive A into List<A> aList AND count(aList) == 10 AND NOT receive B) {
  // Process aList if B does not occur
}
// C. Collect A events into aList that arrive in the next 5 minutes.
// Only collect events that have the same name as the first event
when(receive A a into List<A> aList correlate a.name AND delay min(5)) {
}
// D. Over 7 days, gather and process lists of correlated A events.
now = new Date();
repeat when (receive receive A firstA into List<A> aList) {
  // Nested receive takes priority in consuming matching A events.
  when (receive receive A a into aList correlate firstA
           AND delay (now + days(7))) {
    // Process each aList at the end of the 7 days
  }
}
```

Example 4. Message Aggregation

Pattern 7: *One-to-many send/receive.* **Synonyms:** *Scatter-gather* [11], [17].

Description. A party sends a request to several other parties, which may all be identical or logically related. Responses are expected within a given timeframe. However, some responses may not arrive within the timeframe and some parties may even not respond at all. The interaction may complete successfully or not depending on the set of responses gathered.

Discussion. The following example demonstrates this pattern in Evie. A Request for Quote (RFQ) process has three service roles: RFQ Manager, Requester and Supplier. The Requester issues a RFQ document requesting quotes for supply of a given product from a set of Suppliers. The RFQ Manager acts as the broker service that supervises the collaboration process.

The RFQ process can be decomposed into a sequence of phases that exhibit different event behavior. Phase rules are joined using a `then` construct (C.13, C.23). Rules and context threads are activated and deactivated as these phases progress.

Phase I: The RFQ Manager receives an RFQ from a Requester (C.3). The RFQ specifies the product and quote submission deadline. Sends and receives are correlated (C.4) with this RFQ creating a new process. RFQ is broadcasts to all Suppliers (C.7) and receives reply Quotes (C.8). Concurrent read/update to `bestQuote` (C9, C10) emulates software transactional memory [15].

Phase II: When the deadline expires (C.13), the `then` construct terminates threads and active rules in the previous phase (B.6 – B.15) before executing actions in the new phase (B.17 – B.27). RFQ Manager notifies the successful Supplier and Requester (B.19-20). Late Quotes are now rejected with a QuoteFailure.

Phase III: (C.24) After 30 days the final phase causes all RFQ correlated rules to be terminated. Subsequent RFQ correlated events will bounced back to their **sender**.

```
C.1   package evie.rfq;
C.2   service input rules RFQ_Manager {
C.3    repeat when (receive RFQ rfq from Requester requester) {// Phase I
C.4      correlate (rfq) {
C.5        Quote bestQuote = null;
C.6        when (true) {
C.7          send rfq to role Supplier; // broadcast RFQ to all Suppliers
C.8          repeat when (receive Quote quote from Supplier) {
C.9            if (bestQuote == null || bestQuote.bidPrice > quote.bidPrice) {
C.10             bestQuote = quote;
C.11           } // if
C.12         } // when
C.13       } then when (delay(rfq.deadline)) {   // Phase II
C.14          if (bestQuote != null) {
C.15             send new QuoteSuccess(bestQuote)
C.16                   to requester, bestQuote.sender;
C.17          } else {
C.18            send new QuoteFailure("No Quotes received") to requester;
C.19          }
C.20          repeat when (receive Quote quote from Supplier) {
C.21            send new QuoteFailure("Not received before deadline")
                   to quote.sender correlate quote;
C.22          } // repeat when
C.23        }then when (delay(days(30)){       // Phase III
C.24        }
C.25      }
C.26    }
C.27   }
C.28  }
```

Example 5. Request for quote – RFQ Manager Rules

Pattern 8: *Multi-responses.* **Synonyms.** *Streamed responses, message stream*

Description. A party X sends a request to another party Y. Subsequently, X receives any number of responses from Y until no further responses are required. The trigger of no further responses can arise from a temporal condition or message content, and can arise from either X or Y's side. Responses are no longer expected from Y after one or a combination of the following events: (i) X sends a notification to stop; (ii) a relative or absolute deadline indicated by X; (iii) an interval of inactivity during which X does not receive any response from Y; (iv) a message from Y indicating to X that no further responses will follow. From this point on, no further messages from Y will be accepted by X.

```
when (receive StartConversation from Role partner) {
  // ... activate other partner rules ...
  repeat exclusive {                        // Detect partner inactivity
    when (delay (mins(10)) { send Terminate to self; }
    when (receive Message from partner) { }  // Observes all Message types.
  }
} then when (receive Terminate from self) {
  send StopConversation to partner;
}
```

Discussion. Evie rules can receive a stream of events of uniform or mixed type and respond to them until an event condition occurs. Evie event conditions and phases directly implement termination of behavior for all except (iii). This special case can be implemented by introducing a rule that detects when the inactivity period is occurs and sending an internal event type that can then be employed in termination event conditions. To detect inactivity we employ a `repeat exclusive` that will reset a time delay each time a message is received from a given partner.

Pattern 10: *Atomic multicast notification*. **Synonyms**. *Transactional Notification*

Description. A party sends notifications to several parties such that a certain number of parties are required to accept the notification within a certain timeframe. For example, all parties or just one party are required to accept the notification. In general, the constraint for successful notification applies over a range between a minimum and maximum number.

```
send new Request(…) to Recipient;
repeat when (receive Reply reply into List<Reply> replies from Recipient
             where count(replies) <= maximum AND delay days(10)) {
  if (count(replies) >= mimimum) {
    // ... success
  }
}
```

Support for dynamic lists of service references within events allows Evie to fulfill many routing patterns. However, space does not permit us to cover all these in depth.

4 Summary and Conclusions

Achieving streamlined support for Collaborative Business Processes is one of the key objectives of enterprise systems, since they offer new business opportunities, benefits of maximizing operational productivity, improved business resource utilization, and supports businesses in gaining competitive advantages.

However, overcoming the complexity that surrounds the interoperation between partners in a CBP is by no means an easy task. Following on two basic intuitions: (1) there needs to be an explicit interconnect between high level business models and underlying execution infrastructures and (2) CBP setups are mostly undertaken by technical teams often software engineers, where high level models of limited or tedious functionality may prove unproductive; we have proposed Evie, as a high level language to form the pipeline between business analysts and software development teams both working towards the common goal of facilitating event driven CBPs.

The Evie approach is well aligned with current trends towards event based architectures for large scale integration systems. However, the proposed approach is distinguished in three respects:

- Providing simple and uniform language constructs that allow the specification of diverse service interaction patterns
- Ability to provide a level of abstraction from the execution details due to the compilation phase that generates the requisite objects and code for execution
- Utilization of an execution model based on event subscription, that provides the ability to cater for high volume and long duration processes with minimal impact on system performance and response latency

An important aspect of this approach is the ability to generate an Evie program from a high level modeling tool. This has not been considered in this paper, but is part of our future work. Further evaluation of Evie's usability in the context of SIP as well as other encodings relating to various business scenarios and examples is anticipated.

References

1. van der Aalst, W., et al.: Workflow Patterns. Distributed and Parallel Databases 14(1), 5–51 (2003), http://www.workflowpatterns.com
2. Alonso, G., et al.: Web Services Concepts. In: Architectures and Applications, Springer, Heidelberg (2004)
3. Barros, A., Dumas, M., ter Hofstede, A.H.M.: Service Interaction Patterns. In: van der Aalst, W.M.P., et al. (eds.) BPM 2005. LNCS, vol. 3649, pp. 302–318. Springer, Heidelberg (2005)
4. Barros, A., Dumas, M., ter Hofstede, A.H.M.: Service Interaction Patterns: Towards a Reference Framework for Service-Based Business Process Interconnection, tech. report FIT-TR-2005-02, Queensland Univ. of Technology (March 2005)
5. Chakravarthy, S., et al.: Composite Events for Active Databases: Semantics, Contexts and Detection. In: VLDB 1994. Paper presented at the Proceedings of 20th International Conference on Very Large Data Bases, Santiago, Chile (1994)
6. Chappell, D.A.: Enterprise Service Bus, 1st edn. O'Reilly Media, Inc., Sebastopol, California (2004)
7. Cao, D., Orlowska, M.E., Sadiq, S.W.: Formal Considerations of Rule-Based Messaging for Business Process Integration. Special Issue of Cybernetics and Systems: An International Journal 37(2) (February/March 2006)
8. Dayal, U., et al.: The HiPAC Project: Combining Active Databases and Timing Constraints. ACM's Special Interest Group on Management Of Data (SIGMOD) 17(1), 51–70 (1988)
9. Decker, G., Puhlmann, F., Weske, M.: Formalizing Service Interactions. Business Process Management, 414–419 (2006)
10. Gehani, N.H., Jagadish, H.V., Shmueli, O.: Event specification in an active object-oriented database. In: SIGMOD 1992. Proceedings of the 1992 ACM Special Interest Group on Management Of Data international conference on Management of Data, San Diego, California, United States (1992)
11. Hohpe, G., Woolf, B.: Enterprise integration patterns: Designing, building, and deploying messaging solutions. Addison-Wesley, Reading (2004), http://eaipatterns.com

12. Kilgore, R., Chase, C.: Testing Distributed Programs Containing Racing Messages. Computer Journal 40(8), 489–498 (1997)
13. Luckham, D.C.: The power of events: an introduction to complex event processing in distributed enterprise systems. Addison-Wesley, Boston (2002)
14. O'Hagan, A., Sadiq, S., Sadiq, W., Orlowska, M.E., Evie.: A Language to Implement Harmonized Messaging to facilitate Collaborative Business Processes. The University of Queensland, School of Information Technology and Electrical Engineering, Technical Report No. 467 (March 2007)
15. Shavit, N., Touitou, D.: Software Transactional Memory. In: Proceedings of the 14th ACM Symposium on Principles of Distributed Computing, pp. 204–213 (August 1995)
16. Retschitzegger, W.: Composite Event Management in TriGS - Concepts and Implementation. In: Quirchmayr, G., Bench-Capon, T.J.M., Schweighofer, E. (eds.) DEXA 1998. LNCS, vol. 1460, Springer, Heidelberg (1998)
17. Snir, M., Gropp, W.: MPI: The Complete Reference, 2nd edn. MIT Press, Cambridge (1998)

Delegating Revocations and Authorizations

Hua Wang[1] and Jinli Cao[2]

[1] Centre for Systems Biology
Department of Maths & Computing, University of Southern Queensland
Toowoomba, QLD 4350, Australia
wang@usq.edu.au
[2] Department of Computer Science & Computer Engineering
La Trobe University, Melbourne, VIC 3086, Australia
jinli@cs.latrobe.edu.au

Abstract. Delegation models based on role-based access control (*RBAC*) management have been known as flexible and efficient access management for data sharing on distributed environment. Delegation revocations are a significant functionality for the models in distributed environment when the delegated roles or permissions are required to get back. However, problems may arise in the revocation process when one user delegates user U a role and another user delegates U a negative authorization of the role.

This paper aims to analyse various role-based delegation revocation features through examples. Revocations are categorized in four dimensions: Dependency, Resilience, Propagation and Dominance. According the dimensions, sixteen types of revocations exist for specific requests in access management: DependentWeakLocalDelete, DependentWeakLocalNegative, DependentWeakGlobalDelete, DependentWeakGlobalNegative, IndependentWeakLocalDelete, IndependentWeakLocalNegative, IndependentWeakGlobalDelete, IndependentWeakGlobalNegative, and so on. We present revocation delegating models, and then discuss user delegation authorization and the impact of revocation operations. Finally, comparisons with other related work are indicated.

Keywords: RBAC, Delegation, Revocation.

1 Introduction

Revocation is a significant function in role-based delegations. For example, Tony delegated role director (*DIR*) to Richard; if Richard moves to another company and does not work at the university, his delegated role *DIR* has to be revoked instantly. Several different semantics of user revocation exist [9]: global and local (propagation), strong and weak (dominance) and deletion or negative (resilience). Propagation refers to the extent of the revocation to other delegated users while Resilience means no time-persistent with negative permissions. Dominance refers to the effect of a revocation on implicit/explicit role memberships of a user. For example, there are two types of revocation in dominance: weak and strong revocation [13]. A strong revocation of a user from a role requires that the user be removed not only from the explicit membership but also from the implicit memberships of the delegated role. A weak revocation only removes the user from the delegated role (explicit membership) and leaves other roles intact. Strong revocation is theoretically equivalent to a series of weak revocations. To perform strong revocation, the implied weak revocations are authorized based on revocation policies.

The purpose of the paper is to describe and analyse a number of revocation schemes and their relationships to one another. Revocation schemes are categorized with four dimensions: dependency, resilience, propagation and dominance. With the help of these dimensions we define sixteen different revocation schemes. The remainder of this paper is organized as follows: Section 2 presents the required technologies for the paper. It includes a delegation example, *RBAC* and role-based delegation structure. Section 3 proposes the details of four revocation dimensions. Rather than formal definition, examples are used to explain the definitions of each dimension. There are sixteen types of revocation based on the dimensions. We do not analyse all sixteen revocation schemes in the paper, instead of, four of them are discussed in section 4. Section 5 compares our work with the previous work on delegation revocation methods. The differences between this work from others are presented. Section 6 concludes the paper and outlines our future work.

2 Basic Technologies

2.1 Delegation Example

In *POIS*, officers might be involved in many concurrent activities such as conducting initial investigations, analysing and confronting crimes, preparing immigration reports, and assessing projects. In order to achieve this, users may have one or more roles such as lead officer, participant officer, or reporter. In this example, Tony, a director, needs to coordinate analysing and confronting crimes and assessing projects. Collaboration is necessary for information sharing with members from these two projects. To collaborate closely and make two projects more successful, Tony would like to delegate certain responsibilities to Christine and her staff. The prerequisite conditions are to secure these processes and to monitor the progress of the delegation. Furthermore, Christine may need to delegate the delegated role to her staff as necessary or to delegate a role to all members of a group at the same time. Without delegation skill, security officers have to do excessive work since the involvement of every single collaborative activity. The major requirements of role-based delegation in this example are:

1. Group-based delegation;
2. Multistep delegation;
3. Revocation schemes;
4. Constraints;
5. Partial delegation.

This paper focuses exclusively on revocation schemes in role-based delegation models. We extend our previous work and propose a delegation framework and analyse how does original role assignment changes impact delegation results.

2.2 Basic Elements and Components

RBAC involves individual users being associated with roles as well as roles being associated with permissions (Each permission is a pair of objects and operations). As such, a role is used to associate users and permissions. A user in this model is a human being. A role is a job function or job title within the organization associated with authority and responsibility [5,16]. RBAC is being considered as part of the emerging SQL3 standard for database management systems, based on their implementation in Oracle 7 [10,12]. Many RBAC practical applications have been implemented [3,11].

A session is a mapping between a user and possibly many roles. For example, a user may establish a session by activating some subset of assigned roles. A session is always associated with a single user and each user may establish zero or more sessions. There may be hierarchies within roles. Senior roles are shown at the top of the hierarchies. Senior roles inherit permissions from junior roles. Let $x > y$ denote x is senior to y with obvious extension to $x \geq y$. Role hierarchies provide a powerful and convenient means to enforce the principle of least privilege since only required permissions to perform a task are assigned to the role.

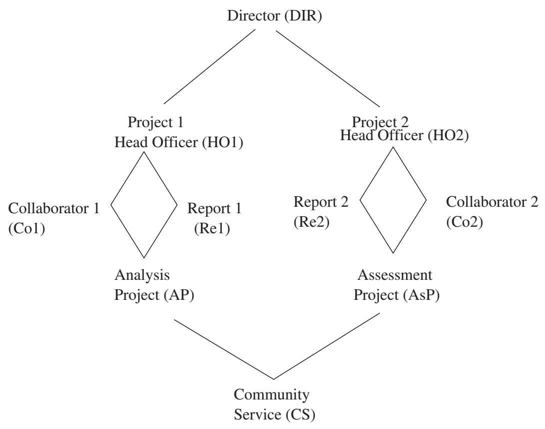


Fig. 1. Role hierarchy in *POIS*

Firgure 1 shows the role hierarchy structure of *RBAC* in *POIS*. An high location role is senior to a low connected location role in the figure, e.g. role *Co1* is senior to role *AP*.

The following Table 1 expresses an example of user-role assignment in *POIS*.

Table 1. User-Role relationship

RoleName	UserName
DIR	Tony
HO1	Babarra
HO2	Mike
Co1	Sam
Re1	John
CS	Ahn

There are two sets of users associated with role r:

Original users are those users who are assigned to the role r;
Delegated users are those users who are delegated to the role r.

The same user can be an original user of one role and a delegated user of another role. Also it is possible for a user to be both an original user and a delegated user of the same role. For example, if Babarra delegates her role *HO1* to Sam, then Richard is both an original user (explicitly) and a delegated user (implicitly) of role *Co1* because the role *HO1* is senior to the role *Co1*. The original user assignment (*UAO*) is a many-to-many user assignment relation between original users and roles. The delegated user assignment (*UAD*) is a many-to-many user assignment relation between delegated users and roles.

We have the following components for RBDM model:

U, R, P and S are sets of users, roles, permissions, and sessions, respectively.

1. $UAO \subseteq U \times R$ is a many-to-many original user to role assignment relation.
2. $UAD \subseteq U \times R$ is a many-to-many delegated user to role assignment relation.
3. $UA = UAO \cup UAD$.
4. Users: $R \Rightarrow 2^U$is a function mapping each role to a set of users. $Users(r) = \{u|(u,r) \in UA\}$ where UA is user-role assignment.
5. $Users\_O(r) = Users\_O(r) \cup Users\_D(r)$
   where
   $Users\_O(r) = \{u|\exists r' > r, (u,r) \in UAO\}$,
   $Users\_D(r) = \{u|\exists r' > r, (u,r) \in UAD\}$.

2.3 Role-Based Delegation and Revocation

The scope of our model is to address user-to-user delegation supporting role hierarchy. We consider only the regular role delegation in this paper, even though it is possible and desirable to delegate an administrative role.

A delegation relation (*DELR*) is existed in the role-based delegation model which includes three elements: original user assignments *UAO*, delegated user assignment *UAD*, and constraints. The motivation behind this relation is to address the relationships among different components involved in a delegation. In a user-to-user delegation, there are five components: a delegating user, a delegating role, a delegated user, a delegated role, and associated constraints. $((Tony, DIR), (Mike, DIR), Friday)$, for example, means Tony acting in role *DIR* delegates role *DIR* to Mike on Friday. We assume each delegation is associated with zero or more constraints. The delegation relation supports partial delegation in a role hierarchies: a user who is authorized to delegate a role r can also delegate a role r' that is junior to r. For example, $((Tony, DIR), (Ahn, Re1)$, Friday) means Tony acting in role *DIR* delegates a junior role *Re1* to Ahn on Friday. A delegation relation is one-to-many relationship on user assignments. It consists of original user delegation (*ORID*) and delegated user delegation (*DELD*). Figure 2 illustrates components and their relations in a role-based delegation model.

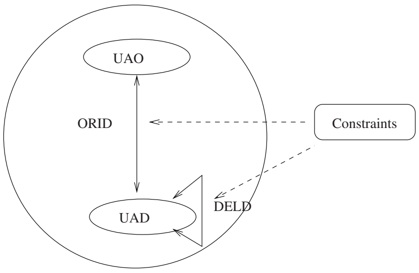


Fig. 2. Role-based delegation model

From the above discussions, the following components are formalized:

1. $DELR \subseteq UA \times UA \times Cons$ is one-to-many delegation relation. A delegation relation can be represented by $((u, r), (u', r'), Cons) \in DELR$, which means the delegating user u with role r delegated role r' to user u' who satisfies the constraint $Cons$.
2. $ORID \subseteq UAO \times UAD \times Cons$ is an original user delegation relation.
3. $DELD \subseteq UAD \times UAD \times Cons$ is a delegated user delegation relation.
4. $DELR = ORID \cup DELD$.

We can find from the components above that delegation relations include original user delegation and delegated user delegation. There are related subtleties concerning the interaction between delegating and revocation of user-user delegation membership and the role hierarchy.

Definition 1. A user-user delegation revocation is a relation $Can - revoke \subseteq R \times 2^R$, where R is a set of roles. ◇

The meaning of Can-revoke (x, Y) is that a member of role x (or a member of an role that is senior to x) can revoke delegation relationship of a user from any role

Table 2. Can-revoke

RoleName	Role Range
HO1	[Co1, CS]

$y \in Y$, where Y defines the *range of revocation*. Table 2 gives the Can-revoke relation in Figure 1.

We analyse revocation dimension in the next section followed by a rich types of revocation schemes.

3 Revocation Dimensions

Different revocation models have been proposed for access control systems [9,20]. For instance, three dimensions were introduced in [9] that are applied for database management system. We gather these dimensions in a unified collections: dependency, resilience, propagation and dominance. Each dimension is defined in this section adopting examples rather than formal descriptions.

3.1 Dependency

Dependency refers to the legitimacy of a user who can revoke a delegated role. Dependent revocation means only the delegating user can revoke the delegated user from the delegated role. Independent revocation allows any original user of the delegating role can revoke the user from the delegated role. For example, with dependent revocation Richard can revoke Christine from the delegated role *Co1* but Tony cannot even though Tony acts as role *DIR* that is senior to and an original role of *Co1*, but Tony can take the delegated role *Co1* from Richard with independent revocation.

3.2 Resilience

This dimension distinguishes revocation through deleting or negative authorization. The effect of a role deleting revocation from a user is acted; another user

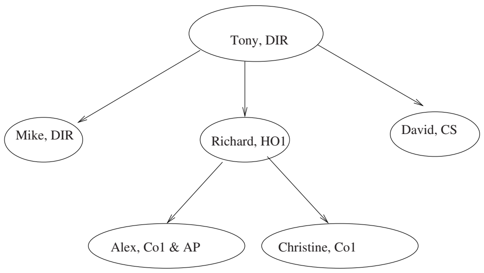


Fig. 3. Revocation relationships

may grant the user the same role that was just revoked, and as a result the revocation has no affection to the user. The negative authorization has high priority in this dimension that means the authorization overrule any other authorizations until the negative one is in turn revoked. As shown in Figure 3, Richard needs to keep Alex from role *Co1*, he can either delete the current delegating relationship, or give a negative authorization of *Co1* to Alex. In the first case, Alex is denied to act as role *Co1*, but only as long as no other users delegate Alex the role. In the second case, Alex can not act as role *Co1* until the negative authorization is revoked. Therefore, negative authorization is stronger than deleting revocation.

3.3 Propagation

This dimension distinguishes revocations according to a delegation structure of role locations. There are local revocation and global revocation in the dimension. The local revocation only happens to the direct delegation relationship while the global revocation effects all other users authorized by the revoked user. We use the Figure 3 to explain the difference between local and global revocations. Suppose Tony wants to revoke Richard from *HO1* but trusts Alex with roles *AP*, *Co1*, and trust Christine with role *Co1* delegated by Richard, local revocation can be applied; otherwise global revocation is applied to take role *HO1* from Richard as well as roles *AP, Co1* from both Alex and *Co1* from Christine.

3.4 Dominance

This dimension deals with role structure in *RBAC*. Due to role hierarchy, a role x' has all permissions of role x when x' is senior to x ($x' > x$). A revocation problem may arise when a user with two roles $\{x', x\}$, the user still has the permissions of x if only to revoke x from the user. The explicit member of a role x is a set of user $\{U|(U, x) \in UA\}$ where $(U, x) \in UA$ means user U has role x and the implicit member of role x is a set of user $\{U|\exists x' > x, (U, x') \in UA\}$. To solve the authorization revocation problem, we need to revoke the explicit member of a role first if a user is an explicit member, then revoke the implicit member. There are two kinds of revocations. The first one is weak revocation which revokes explicit member only; the second one is strong revocation that revokes explicit and implicit members. Alex has two delegated roles *Co1, AP* where role *Co1* is senior to role *AP* in Figure 3. Richard wants to take role *AP* from Alex with strong revocation, both roles *Co1* and *AP* are revoked from Alex, but with weak revocation only role *AP* is revoked.

4 Delegating Revocations

We describe sixteen different revocations based on the dimensions in the previous section. Each scheme, as shown in Table 3, has a unique description with respect to the four dimensions.

Table 3. Revoaction types

No.	Dependency	Resilience	Propagation	Dominance	Name
1	No	No	No	No	DependentWeakLocalDelete(*DWLD*)
2	No	No	No	Yes	DependentStrongLocalDelete(*DSLD*)
3	No	No	Yes	No	DependentWeakGlobalDelete(*DWGD*)
4	No	No	Yes	Yes	DependentStrongGlobalDelete(*DSGD*)
5	No	Yes	No	No	DependentWeakLocalNegative(*DWLN*)
6	No	Yes	No	Yes	DependentStrongLocalNegative(*DSLN*)
7	No	Yes	Yes	No	DependentWeakGlobalNegative(*DWLN*)
8	No	Yes	Yes	Yes	DependentStrongGlobalNegative(*DSGN*)
9	Yes	No	No	No	IndependentWeakLocalDelete(*IDWLD*)
10	Yes	No	No	Yes	IndependentStrongLocalDelete(*IDSLD*)
11	Yes	No	Yes	No	IndependentWeakGlobalDelete(*IDWGD*)
12	Yes	No	Yes	Yes	IndependentStrongGlobalDelete(*IDSGD*)
13	Yes	Yes	No	No	IndependentWeakLocalNegative(*IDWLN*)
14	Yes	Yes	No	Yes	IndependentStrongLocalNegative(*IDSLN*)
15	Yes	Yes	Yes	No	IndependentWeakGlobalNegative(*IDWLN*)
16	Yes	Yes	Yes	Yes	IndependentStrongGlobalNegative(*IDSGN*)

4.1 DependentWeakLocalDelete(*DWLD*)

The DependentWeakLocalDelete(*DWLD*), as the first revocation scheme, is the most easy operation. It does neither have resilience, propagation, nor dominance, and only the direct delegating user can remove the delegation relationship. Suppose Tony as the director wants to revoke role *HO1* from Richard since Richard is not an employee any longer in a company. With the scheme of *DWLD*, role *Ho1* only takes away from Richard and roles of both Alex and Christine are intact; the delegation relationships between Richard and Alex, Richard and Christine come to the relationships between Tony and Alex, Tony and Christine as shown in Figure 4 from Figure 3. The features of scheme *DWLD* when user U_1 wants to revoke role r from user U_2 are:

1. U_1 does not grant role r to U_2;
2. Role r may still stay with U_2 if users other than U_1 delegate r to him;
3. Roles granted by users other than U_1 are intact;
4. The delegation structure may need to update since roles delegated by U_2 have to remain.

4.2 DependentStrongLocalDelete(*DSLD*)

The DependentStrongLocalDelete(*DSLD*) is different from *DWLD* in the dominance aspect. It does not have resilience and propagation, but dominance, and only the direct delegating user can take away the delegation relationship. Suppose Mike acts as role *DIR* wants to remove role *Co1* from Richard since Richard is out of the work as role *Co1*. With the scheme of *DSLD*, not only role *Co1* takes away from Richard by Mike , but role *HO1* has to move from Richard by Tony since role *HO1* delegated by Tony is senior to role *Co1*. New delegation relationships are generated between Mike and Alex, Mike and Christine as shown in Figure 5.

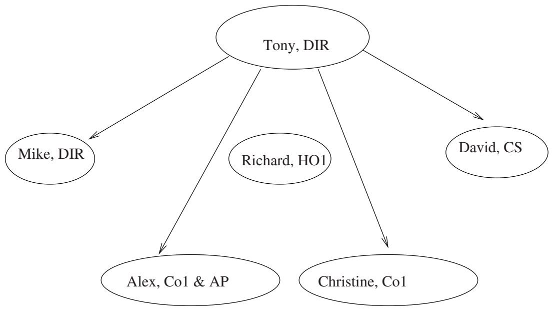


Fig. 4. Delegation relationships after *DWLD*

The features of scheme *DSLD* when user U_1 wants to revoke role r from user U_2 are:

1. U_1 does not grant role r to U_2;
2. Implicit role r' that is senior to role r is removed;
3. Roles other than r and its senior role are intact;
4. The delegation structure may need to update since roles delegated by U_2 have to remain.

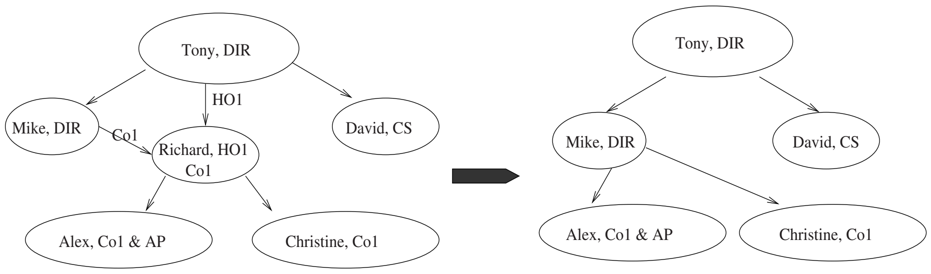


Fig. 5. Delegation relationships after *DSLD*

4.3 DependentWeakGlobalDelete(*DWGD*)

The DependentWeakGlobalDelete(*DWGD*) is different from *DWLD* in the propagation aspect. It does not have resilience and dominance, but propagation, and only the direct delegating user can remove the delegation relationship. Suppose Mike acts as role *DIR* wants to remove role *Co1* from Richard since Richard is out of the work as role *Co1*. With the scheme of *DWGD*, role *Co1* takes away from Richard by Mike , but role *HO1* is intact, role *Co1* is also revoked from Alex and Christine since the delgation authorization is no longer supported by Richard. The new relationships after the *DWGD* are shown in Figure 6.

The features of scheme *DWGD* when user U_1 wants to revoke role r from user U_2 are:

1. U_1 does not grant role r to U_2;
2. role r delegated by U_2 to users is removed;
3. Roles other than r are intact;
4. The delegation structure may need to update since roles other than r delegated by U_2 have to remain.

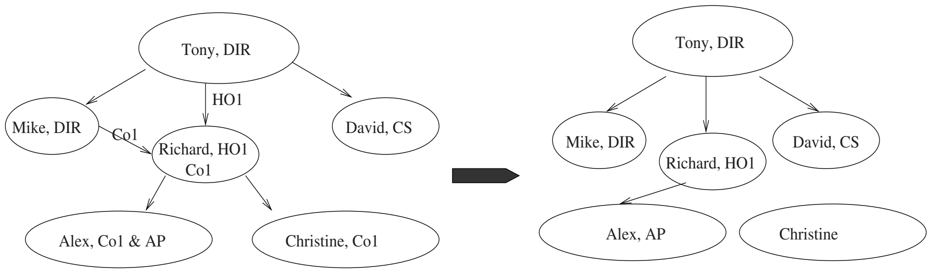


Fig. 6. Delegation relationships after *DWGD*

We do not analyse other revocation operations in the paper due to the paper length limits.

5 Comparisons

There are related work on revocation schemes. The closed work to this paper are a rule-based framework for role-based delegation and revocation [21] and revocations-a classification [9]. The former one was on the delegation framework and the latter one worked on database systems.

Hagstrom et. al [9] have argued that several different semantics are possible for the revoke operation that focus on database systems. Three main revocation characteristics were identified: the extent of the revocation to other grantees (propagation), the effect on other grants to the same grantee (dominance), and the permanence of the negation of rights (resilience). A classification was devised using these three dimensions. However, the dependency was not included in the paper and hence only eight different revocations were discussed. Their work is different from ours in two aspects. First, it focuses on database authorization. No hierarchy structure of database system was analysed. As a result, important features such as role hierarchies and constraints were not supported. By contrast, our work focuses on role-based access model that supports hierarchy structure. Second, they neither gave a discussion of who has the ability to process a revocation operation, which is a critical notion to the delegation model nor discussed the relationships among original user and delegated user. By contrast, the delegation model in this paper is based on original user and delegated user since the delegating relationship in this paper has five components $((u, r), (u', r'), Cons)$ in which (u, r) is original user-role relationship and (u', r') be delegated user-role relationship.

6 Conclusions and Future Work

This paper has discussed role-based delegation revocations and authorization. We have discussed role-based delegation requirements and components in delegation models, and analysed not only revocation dimensions but also a rich revocation schemes. To introduce a practical solution on how to use these revocation ideas, a briefly introduction of a revocation authorization approach is presented. The work in this paper has significantly extended previous work in several aspects, for example, comprehensive revocation dimensions and revocation schemes for various requirements. The work in the paper is helpful for how to build a management system with different revocations.

This is the beginning work on revocation of role-based access control. The future work includes develop algorithms based on the dimensions and revocation schemes and delegation revocation models with constraints.

References

1. Abadi, M., et al.: A calculus for access control in distributed systems. ACM Trans. Program. Lang. Syst. 15(4), 706–734 (1993)
2. Barka, E., Sandhu, R.: Framework for role-based delegation models and some extensions. In: Proceedings of the 16 Annual Computer Security Applications Conference, New Orleans, pp. 168–177 (2000)
3. Barkley, J.F., Beznosov, K., Uppal, J.: Supporting relationships in access control using role based access control. In: Third ACM Workshop on RoleBased Access Control, pp. 55–65 (October 1999)
4. Bertino, E., Jajodia, S., Samarati, P.: A non-timestamped authorization model for data management systems. In: ACM Conference on Computer and Communications Security, pp. 169–178 (1996)
5. David, F.F., Dennis, M.G., Nickilyn, L.: An examination of federal and commercial access control policy needs. In: NIST NCSC National Computer Security Conference, Baltimore, MD, pp. 107–116 (September 1993)
6. Fagin, R.: On an authorization mechanism. ACM Trans. Database Syst. 3(3), 310–319 (1978)
7. Feinstein, H.L.: Final report: Nist small business innovative research (sbir) grant: role based access control: phase 1. technical report. In: SETA Corp. (1995)
8. Ferraiolo, D.F., Kuhn, D.R.: Role based access control. In: 15th National Computer Security Conference, pp. 554–563 (1992)
9. Hagstrom, A., Jajodia, S., Presicce, F., Wijesekera, D.: Revocations-a classification. In: Proceedings of 14th IEEE Computer Security Foundations Workshop, Nova Scotia, Canada, pp. 44–58 (2001)
10. Sandhu, R.: Rational for the *RBAC*96 family of access control models. In: Proceedings of 1st ACM Workshop on Role-based Access Control, pp. 64–72. ACM Press, New York (1997)

11. Sandhu, R.: Role activation hierarchies. In: Third ACM Workshop on RoleBased Access Control, pp. 33–40. ACM Press, New York (1998)
12. Sandhu, R.: Role-Based Access Control. Advances in Computers 46 (1998)
13. Wang, H., Cao, J., Zhang, Y.: Formal authorization allocation approaches for role-based access control based on relational algebra operations. In: WISE 2002. 3rd International Conference on Web Information Systems Engineering, Singapore, pp. 301–312 (2002)
14. Wang, H., Cao, J., Zhang, Y.: Formal authorization allocation approaches for permission-role assignments using relational algebra operations. In: ADC 2003. Proceedings of the 14th Australian Database Conference, Adelaide, Australia, pp. 125–134 (2003)
15. Wang, H., Cao, J., Zhang, Y.: An Electronic Payment Scheme and Its RBAC management. Concurrent Engineering: Research and Application 12(3), 247–275 (2004)
16. Wang, H., Cao, J., Zhang, Y.: A flexible payment scheme and its role based access control. IIEEE Transactions on Knowledge and Data Engineering 17(3), 425–436 (2005)
17. Wang, H., et al.: A framework for role-based group delegation in distributed environment. In: Proceedings of the 29th Australasian Computer Science Conference, Australian Computer Society, pp. 321–328 (2006)
18. Wang, H., et al.: A global ticket-based access scheme for mobile users. Special Issue on Object-Oriented Client/Server Internet Environments, Information Systems Frontiers 6(1), 35–46 (2004)
19. Wang, H., et al.: Achieving secure and flexible m-services through tickets. In: IEEE Transactions on Systems, Man, and Cybernetics, Part A, Special issue on M-Services, pp. 697–708 (2003)
20. Zhang, L., Ahn, G., Chu, B.: A role-based delegation framework for healthcare information systems. In: SACMAT 2002. Proceedings of ACM Symposium on Access Control Models and Technologies, Monterey, CA, pp. 125–134 (2002)
21. Zhang, L., Ahn, G., Chu, B.: A rule-based framework for role-based delegation and revocation. ACM Trans. Inf. Syst. Secur. 6(3), 404–441 (2003)

Privacy Preserving Collaborative Business Process Management*

Sumit Chakraborty[1] and Asim Kumar Pal[2]

[1] Fellowship – MIS (passed), IIM Calcutta, India
[2] Professor, MIS, Indian Institute of Management Calcutta, India
{surya20046@yahoo.co.in, asim@iimcal.ac.in}

Abstract. The basic objective of collaborative supply chain planning (CSCP) is to fulfill the demand of the customers by integrating a network of organizations through mediums such as internet. But, the supply chain (SC) partners are often reluctant to share their strategic information. The exchange of relevant strategic information yet maintaining privacy is a challenging issue for CSCP. It is therefore required to develop privacy preserving coordination mechanisms (PPCM) that can align the business objectives of SC partners. This paper presents a distributed algorithm for PPCM for CSCP based on secure multi-party computation. This requires negotiations for compensation between the buying and the selling firms. We have considered a single buyer and single supplier (SBSS) case along with the process flow logic using a process flow construct for secure computation. We have extended the method to a multiparty negotiation process for a multiple buyer and single supplier (MBSS) case.

Keywords: Coordination mechanism, Distributed algorithm, Multi-party negotiation, Secure multiparty computation, Collaborative supply chain planning.

1 Introduction

The rapid expansion of the global market, the explosive growth of information and communication technologies, aggressive competition and the changing economic and social conditions have triggered tremendous opportunity to conduct business in a collaborative way. The *business processes* of different organizations need to be integrated to adapt the dynamic conditions and to remain competitive in the global market. But, in a loosely coupled collaborative environment, several crucial aspects such as privacy and security do not get sufficient support. *Supply Chain Management* is a well-known management philosophy to boost a firm's competitiveness in today's business environment. The sharing of information is important for efficient coordination of operational processes across a *supply chain* (***SC***). But, the partners of a SC are often reluctant to disclose sensitive strategic information since the information can either be used by the SC partners or can be revealed to their competitors [1]. It is therefore required to develop *privacy preserving coordination*

* This work was partly supported by AICTE Project ISISAMB.

mechanisms (***PPCM***) that can align the business objectives and decision-making activities of the members to optimize the performance of a SC system.

1.1 The Problem

First, we consider a simple case of *collaborative supply chain planning* (***CSCP***), the two party SC - one buying firm B and one selling firm S. The optimization model of the *local planning domain* of B is: min $(o^B)^T \mathbf{x}^B$ s.t. $M^B \mathbf{x}^B \leq b^B$ where $\mathbf{x}^B$ is the vector of decision variables for B; o^B, b^B and M^B are the cost vector, the constraint lower bound vector and the constraint matrix for B respectively. Similarly, that of S is: min $(o^S)^T \mathbf{x}^S$ s.t. $M^S \mathbf{x}^S \leq b^S$. The objective of the present work is to develop a distributed algorithm for the PPCM, which ensures that consistent plans covering the entire SC are generated satisfying various objectives such as minimizing total SC cost, total negotiation time, information disclosure to any party and asymmetry of information disclosure to the parties involved. Both two-party and multiparty cases have been attempted.

The paper is organized as follows. Section 2 gives a brief literature review. Section 3.0 describes privacy preserving collaborative supply chain process. Section 3.1 presents the coordination mechanism for the *single buyer and single supplier*[1] (***SBSS***) case and section 3.2 presents the process flow diagram. Section 4 deals with the multiparty situation. Section 5 gives an overview of secure multiparty computation concepts relevant for the proposed coordination mechanisms. Finally, section 6 concludes the paper by indicating some open problems.

2 Literature Review

The coordination process of autonomous yet interconnected planning domains has been analyzed by various researchers [3,5,6,7,8,11,12]. Our work basically emanates from the coordination scheme proposed by Dudek [3], which proposes a negotiation based coordination scheme combining different aspects of contract design, agent solutions and mathematical programming. The scheme generates good solutions often close to the global optimum, but can not always achieve the delivery plan which has the global minimum SC cost. The algorithm even for only two-party (SBSS) case is time consuming, as it has to conduct a negotiation for a compensation amount in every iteration of *plan generation*. In addition, there is disclosure of cost effects, i.e. increase or decrease of costs over the previous plan.

3 Privacy Preserving Collaborative Supply Chain Planning Process

A centralized SC is considered as a single entity that aims to optimize SC system performance. But, this is not realistic since the members of a SC are often reluctant to

[1] In the paper (also in the literature) 'seller' and 'supplier' terms have been used interchangeably. We have also used 'buying firm', 'buying agent' and 'buyer' terms interchangeably. Similarly for 'selling firm', 'selling agent' and 'seller'.

share their objective functions and constraints directly with the central authority. The members of a decentralized SC act independently to optimize their individual performances [8]. The objective is to devise a distributed mechanism that should be able to coordinate the objectives and decision making activities of independent SC members who are connected over networks such as internet. Optimization of a SC plan can happen in two ways. The first way is to obtain a *global planning domain* by combining the local planning domains (which includes optimizing functions and constraints) of all the partners involved and then optimize the global plan in this domain based on all the domain variables and constraints. The second method is to obtain local planning domain for each partner and generate local plans optimized in these domains given the dynamic set of constraints. The constraints become dynamic because it is not necessary that all the constraints are known to each partner a priori. Actually, often constraints of one partner become dependent on the constraints of others. For example, if a buying firm places an order with a selling firm, the constraints of the selling firm will depend on the constraints of the buying firm expressed through the delivery plan. Again, in the next iteration the buying firm may modify its demand based on the constraint of the selling firm. Finally, global plans which are intended to be optimal for all the parties collectively are attempted to be obtained through an iterative process of optimizing the plans locally. As already indicated, the local planning domain based solution allows the buying firm or the selling firm to locally optimize its plan given the information or condition known to the party at that point of time. Further, *compensation negotiation* (***CN***) is the mechanism through which the conflicts between the respective plans are to be sought, in the process of arriving at the final solution through iterations of bidding and counter bidding.

3.1 Privacy Preserving Single Buyer Single Supplier (SBSS) Algorithm

The coordination of operations along the SC requires a rational, structured plan in order to achieve the optimum choice of objectives and measures to a decision situation for various independent decision making units [3]. The collaborative planning process (***CPP***) spans over multiple planning domains where each planning domain is controlled by an organization. A CPP has six distinct phases: domain planning, data exchange, negotiation and exception handling, execution and performance measurement [4]. Initially *local plans* are generated and relevant data is exchanged among various planning domains in order to generate a common and mutually acceptable plan. This is referred to as a *global plan*. Thus, the quality of a plan and subsequent decision-making process can improve significantly. The next step is to generate a consistent global plan through the modification of internal planning results. The coordination is achieved through negotiation and it results in agreements on objectives, measures and rules. Ultimately, final results are executed and subsequent performance is measured. B bids a plan P to S. S evaluates P and counter bids another plan P’. B in turn evaluates P’ and counter proposes yet another plan P”, and so on. Finally, if the negotiation ends successfully S supplies the order according to the agreed plan. The negotiation for a plan (called *plan negotiation*) consists of *bidding cycles* or *bidding rounds*. In each bidding round, a plan P is bid by either party B or S. The plan negotiation basically consists of plans at various stages as follows:

For any plan P, the cost component of a party B or S (denoted $C^B(P)$, $C^S(P)$ respectively) is private to the party and will not be disclosed to the other party, i.e. what is revealed in the negotiation process are the *order proposal* for B and *supply proposal* for S without any cost implications. Similarly, the total cost (or, total SC cost) for a plan P, $C(P) = C^B(P) + C^S(P)$, is also not revealed to either party. Since P_0 is optimal for B, $C^B(P_0) < C^B(P_i)$ for all $i \geq 1$, i.e. the *cost effect* for B for any plan P_i, $\Delta C^B(P_i) = C^B(P_i) - C^B(P_0) > 0$. Similarly, $\Delta C^S(P_i) = C^S(P_0) - C^S(P_i) > 0$. Cost effect of a buying firm or a selling firm is also referred to as the *local cost effect*. The *global cost effect* or *total cost effect* of a plan P_i is the sum of the local cost effects of all the partners. The objective of the coordination process is to decrease the total cost, not individual costs. However, B is entitled to ask for suitable compensation from S to compensate for the additional cost it has to incur in P_i. Individual cost effect is treated as private information. If B knows the cost effect of S, B will claim compensation accordingly, S may not get any benefit of the cost saving. The CSCP scheme will lose its attractiveness to S. The buying firm does not necessarily have to be informed regarding the full benefit of the selling firm.

B will always ask for a compensation amount, which is at least the cost effect. The CN has basically two purposes: i) to determine whether the current plan P_i is a feasible one, i.e. whether total cost of P_i has increased over the previous plan P_{i-1} (and consequently any other past plan P_j, $j < i-1$); and ii) to determine the amount of savings in costs to be shared between B and S. *Cost implication* of the buying firm for a plan P (denoted $CI^B(P)$) is the cost component of P (i.e. $C^B(P)$) minus the compensation settled (denoted Comp(P)). Similarly, the cost implication for the selling firm $CI^S(P)$ is determined. Note, the total of cost implications for B and S is same as the total SC cost for the plan, C(P). Thus,

Cost implication for B, $CI^B(P) = C^B(P) - \text{Comp}(P)$
Cost implication for S, $CI^S(P) = C^S(P) + \text{Comp}(P)$
Total cost for plan P, $C(P) = C^B(P) + C^S(P) = CI^B(P) + CI^S(P)$

CNs are realistic if the total cost reduces. Compensation will always be settled such that no party loses compared to the previous round, in other words the cost implications for both parties improve. Further, if the CN fails it only means that the total SC cost will increase. When the negotiation ends successfully in the final plan P_f, the total SC cost achieved is nothing but $C(P_f)$. The total savings through the negotiation is $C(P_0) - C(P_f) > 0$, which is apportioned as $\text{Comp}(P_f)$ for B and $C(P_0) - C(P_f) - \text{Comp}(P_f)$ for S. However, the real savings for B is limited to only $\text{Comp}(P_f) - C^B(P_f) + C^B(P_0)$, and for S it is $C^S(P_0) - C^S(P_f) - \text{Comp}(P_f)$.

If a party does not respond, (or responds in the negative) the other party will counter bid next. The other party should get advantage by generating more iterations. One of our assumptions should be that both B and S should be *rational* in exchange of truthful communication and are interested in reducing total SC cost. If none respond, there will be a deadlock. That would mean that neither party is interested in cost reduction, which violates our assumption.

Stopping Criteria: Stopping the negotiation is possible on various counts: total time taken, total number of bidding rounds, number of successive failed biddings, both parties satisfied, etc. If any party wants to withdraw, negotiation ends unsuccessfully.

The following algorithm is proposed reflecting the above discussions:

```
1. The buying firm B bids its optimal plan $P^B$ to the selling firm S.
   Set i = 0; Reference plan $P_i = P^B$.

2. Repeat until the stopping criteria is satisfied:
a.   Set i = i+ 1.
b.   Seller round: S counter bids $P_i^S$ to B; or Buyer round: B counter bids $P_i^B$ to S.
     Set $P_i = P_i^B$ or $P_i^S$ , depending on the bidding round.
c.   B and S compute local cost effects $\Delta C^B(P_i^B)$ and $\Delta C^S(P_i^S)$.
d.   B and S compare local cost effects privately and sets the Reference plan to $P_i$ if
     $\Delta C^S(P_i^S) > \Delta C^B(P_i^B)$.

3. If both parties agree output the agreed final plan, say, $P_f$. B and S jointly settle the
compensation to be given to B through 'compensation negotiation'.
```

As already mentioned in Section 2, in [3] the algorithm for SBSS required CN in every bidding iteration implying enormous loss of time, yet not guaranteeing convergence to the optimal solution (i.e. minimum SC cost), not necessarily even an acceptable solution always.

It should however be mentioned here that even though our algorithm negotiates compensation only once and hence saves time on this count significantly, there is no guarantee of convergence nor is there any surety of a better solution. But, with time saved it could be possible to try out more alternative plans to achieve a better solution. Further, more number of CNs as in [3] would imply indirect disclosure of the cost effect of the buyer in every iteration, as the compensation amount will always be at least the cost effect of B. Note, this is a case of asymmetry in disclosure.

3.2 Process Flow Diagram

Efficient business process management (BPM) aligns organizational business processes with top level business objectives. A comprehensive process model maps all possible execution paths associated with a business process. The process flow diagram associated with the privacy preserving SBSS coordination mechanism has been described below. This diagram uses some basic process flow constructs such as sequence and structured loop. Additionally, we have introduced a process flow construct for secure computation such as private comparison of the local cost effect of each SC partner.

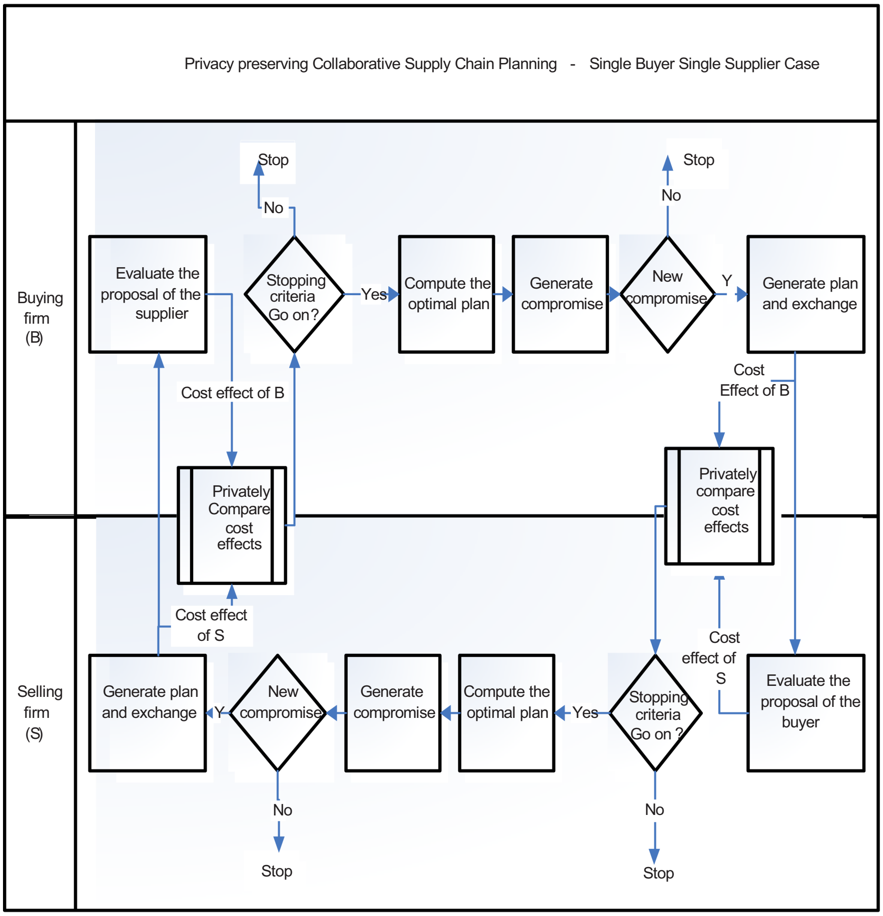

Privacy preserving Collaborative Supply Chain Planning - Single Buyer Single Supplier Case
Buying firm (B)
Selling firm (S)
Evaluate the proposal of the supplier
Stopping criteria Go on?
Stop
No
Yes
Compute the optimal plan
Generate compromise
New compromise
Stop
No
Y
Generate plan and exchange
Cost effect of B
Privately Compare cost effects
Cost Effect of B
Privately compare cost effects
Cost effect of S
Generate plan and exchange
Y
New compromise
No
Stop
Generate compromise
Compute the optimal plan
Yes
Stopping criteria Go on ?
No
Stop
Cost effect of S
Evaluate the proposal of the buyer

Fig. 1. Process flow for privacy preserving SBSS case

4 Multi-party Coordination Mechanism

The aforesaid compensation negotiation based coordination mechanism for CSCP can be extended to 2-tier multi-party SC scenarios: (a) Multiple buying firms and multiple selling firms (MBMS), (b) Single buying firm and multiple selling firms (SBMS) and (c) Multiple buying firms and single selling firm (MBSS). In the following section, we have described a PPCM for the MBSS case.

Let the selling firm S be involved in negotiation with n buying firms B_1, …, B_n. One of the buying firms, say B_1 plays the role of the leader and interacts with other buying firms. Here, the objective is to find the optimal delivery plan of the selling firm, which should result in the minimum SC cost considering all the buying firms. Dudek [3] has treated MBSS case similar to SBSS case. Since all the buying firms give their initial order proposals which are best for themselves, in successive iterations any buying firm can only lose as far as its cost effect is concerned. The selling firm accumulates the demands (via the order proposals) from the buying firms and evaluates these proposals before giving counter bids to each buying firm in a *parallel mechanism*. The following figure shows the *concurrency diagram* of a MBSS model with a single selling firm S and two buying firms B_1 and B_2.

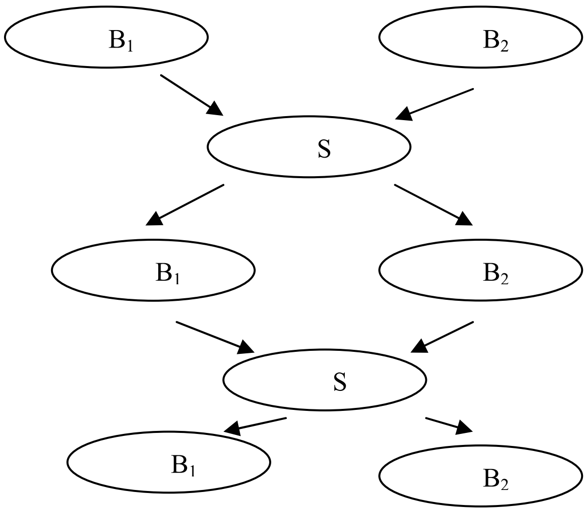


Fig. 2. The concurrency diagram of MBSS

The selling firm computes the total compensation claimed by all the buying firms and compares that with its own cost effect. As in Dudek's work [3] we also propose to extend our SBSS algorithm to the MBSS case. The CN is conducted only once after the final bid has been decided. Here for simplicity we have assumed that the selling firm is able to meet the total demand of all the buying firms. Further, it has been assumed that there is no scope for the selling firm to change the demand or requirement of any buying firm as far as the items and their quantities are concerned. Only, the delivery plans of the buying firms are subject to change. Because of these assumptions it can be safely assumed that whenever a selling firm gives a counter bid the cost of individual buying firms may increase. The *privacy preserving negotiation based coordination scheme* is as follows:

```
1. Each buying firm $B_j$, j=1, …, n, bids its optimal plan $P^{Bj}$ to the selling firm S. S
aggregates $P^B{}_1$, …, $P^{Bn}$ into combined buyer plan $P^B$. Set i = 0; Reference plan $P_i = P^B$.
2. Repeat until the stopping criteria is satisfied:
a. Set i = i + 1.
b. Seller Round:
        For each j, j=1,…,n, S counter bids $P_{ij}^S$ to $B_j$ in parallel;
        S aggregates the plans $P_{ij}^S$ for different j into the combined seller plan $P_i^S$; or
   Buyer Round:
        For each j, j=1,…,n, $B_j$ counter bids $P_{ij}^{Bj}$ to S;
        S aggregates the plans $P_{ij}^{Bj}$ for different j into the combined buyer plan $P_i^B$.
```

Set $P_i = P_i^S$ or P_i^B, depending on the bidding round.
c. S and B_j, j=1,…,n, compute local cost effect $\Delta C^S(P_i^S)$ and $\Delta C^{Bj}(P_{ij}^{Bj})$ respectively.
d. The leader of the buying firms, say, B_1 computes privately the cumulative cost ~~effects~~ effect~~s~~ of all the buyers.

e. S and B_1 (leader) privately compare the cost effects $\Delta C^S(P_i^S)$ and $\sum_{j=1}^{n} \Delta C^{Bj}(P_{ij}^{Bj})$

and sets the Reference plan to P_i if the former is larger.

3. If all the parties agree, the outcome is the finally accepted plan. The selling firm settles the compensation for each buying firm through '*compensation negotiation*'.

MBSS algorithm ensures no disclosure of individual SC costs, individual cost effects, the total SC cost, or the total cost effect of the buying firms and the selling firm. The protocol ensures rapid improvement of cost by removing the expensive plans. It requires the CN for only one iteration to settle compensation corresponding to the plan of minimum SC cost achieved.

A lot of negotiation is saved and the disclosure of information is avoided to a great extent. In the MBSS case this is much more prominent. The CN with multiple buyers is a much more complex and time consuming process, compared to that with a single buyer; also risking more disclosure. Also there will be less chance of success of multi-party CNs which is repeated in the algorithm of [3].

5 Secure Multi-party Computation (SMC) Tools

The aforesaid coordination mechanisms are based on SMC concepts. *Yao's millionaire problem* is to find out who is richer between two parties without disclosing any other information about any one party's value to the other. This problem has been solved in various ways. Yao [10] proposed a solution without using any untrusted third party. The cost of the protocol was exponential in both time and space. Cachin [13] suggested an efficient solution using an oblivious third party. Two inputs can be compared by verifying the most significant bit in which they are different. Similar bits do not affect the result and the effect of unequal low order bits is overshadowed by the high order bits. Based on this principle, Ioannidis and Grama [14] proposed a private comparison protocol using oblivious transfer scheme. Schoenmaker and Tuyls [15] used threshold homomorphic encryption scheme to solve private comparison problem. Blake and Kalashnikovs [16] used the concepts of Q-conditional oblivious transfer and the additive homomorphic Paillier cryptosystem. Two or more agents want to conduct a computation based on their private inputs but neither of them wants to share its proprietary data set to other. The objective of SMC is to compute with each party's private input such that in the end only the output is known. The private inputs are not disclosed in the sense that only information that is revealed is that which can be logically derived from the input and the output. In the study of SMC problems, two models are commonly assumed: *semi-honest* and *malicious*. A semi-honest party follows the protocol properly with correct input. But after the execution of the protocol, it is free to use all its intermediate computations to compromise privacy. A malicious party does not need to follow the protocol properly with correct input; it can enter the protocol with an incorrect input. In this paper, we have assumed that semi-honest agents are involved in PPCM and they act rationally.

6 Conclusion

Collaborative planning, forecasting and replenishment (***CPFR***) is a strategic tool for comprehensive value chain management of an organization. This is an initiative among all the stakeholders of the SC in order to improve their relationship through jointly managed planning, process and shared information [2]. The ultimate goal is to improve a firm's position in the competitive market and the optimization of its own value chain in terms of optimal inventory, improved sales, higher precision of forecast, reduced cost and improved reaction time to customer demands. *Autonomous intelligent agents* can be used to implement the proposed PPCM. Agents are generally reluctant to share their private information in competitive environment of SCM and therefore it is difficult to resolve the conflicts among a group of *decision making agents* for want of proper information. The quality of solution can be improved through private computation since the agents can share their strategic information yet don't have to disclose it to other parties. Our proposed PPCM enable the SC partners to share their strategic information like cost and cost effect privately. Thus, the coordination mechanisms make the negotiation based CSCP process more efficient. Our algorithm both for the SBSS and MBSS cases, particularly for the latter, save time significantly by reducing number of CNs to only one for the entire run as against one for each bidding iteration. It also causes less disclosure of the cost effects. For MBSS even chances of successful negotiation are greater, because the CN is a complex process. However, the convergence of the method, as in [3], is not guaranteed even for the simpler case of SBSS, implying non-optimality of the final plan achieved. One may look for the strategy of improving the solution (for SBSS first), by going back to some intermediate plans generated in the process. For MBSS a strategy for CN between multi-buyer and single supplier needs to be developed.

References

[1] Atallah, M.J., Atallah, E.H.G., Deshpande, V., Schwarz, L.B.: Secure supply chain protocols. In: IEEE International Conference on E-Commerce, Newport Beach, California (2003)

[2] Seifert, D.: Collaborative planning, forecasting and replenishment. Galliers Business (2002)

[3] Dudek, G.: Collaborative planning in supply chain negotiation based approach. Springer, Heidelberg (2004)

[4] Stadtler, H., Kilger, C.: Supply chain management and advanced planning concepts, models, software and case studies, 2nd edn. Springer, Heidelberg (2004)

[5] Bhatnagar, R., Chandra, P., Goyal, S.K.: Models for multi-plant coordination. European Journal of Operational Research 67, 141–160 (1993)

[6] Schneeweiss, C., Zimmer, K.: Hierarchial corordination mechanisms within the supply chain. European Journal of Operational Research 153, 687–703 (2004)

[7] Dudek, G., Stadtler, H.: Negotiation-based collaborative planning between supply chain partners. European Journal of Operational Research 163, 668–687 (2005)
[8] Li, X., Wang, Q.: Coordination mechanisms of supply chain systems. European Journal of Operational Research 179, 1–16 (2007)
[9] Erengüc, S.S., Simpson, N.C., Vakharia, A.J.: Integrated production/distributed planning in supply chain: An invited review. European Journal of Operational Research 115, 219–236 (1999)
[10] Yao, A.C.: Protocols for secure computations. In: 23rd IEEE Annual Symposium on Foundations of Computer Science, pp. 160–164 (1982)
[11] Zhao, X., Liu, C.: Tracking over collaborative business processes. In: 4th International conference on Business Process Management, Vienna, Austria, pp. 33–48 (2006)
[12] Zhao, X., Liu, C., Yang, Y.: An organizational perspective of Inter-Organizational Workflows. In: International Conference on Business Process Management, pp. 17–31 (2005)
[13] Cachin, C.: Efficient private bidding and auctions with an oblivious third party. In: 6th ACM conference on computer and communications security, Singapore, pp. 120–127 (1999)
[14] Ioannidis, I., Grama, A.: An efficient protocol for Yao's millionaires' problem. In: 36th Hawaii International Conference on System Sciences (2003)
[15] Schoenmakers, B., Tuyls, P.: Practical two-party computation based on the conditional gate. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 119–136. Springer, Heidelberg (2004)
[16] Blake, I.F., Kolesnikov, V.: Strong conditional oblivious transfer and computing on intervals. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 515–529. Springer, Heidelberg (2004)

ProHealth Workshop

Introduction to the First International Workshop on Process-Oriented Information Systems in Healthcare (ProHealth 2007)

Manfred Reichert[1], Mor Peleg[2], and Richard Lenz[3]

[1] Information Systems Group, University of Twente, The Netherlands
m.u.reichert@cs.utwente.nl
[2] Department of Management Information Systems, University of Haifa, Israel
morpeleg@mis.hevra.haifa.ac.il
[3] Database Group, University of Erlangen-Nuernberg, Germany
Richard.Lenz@informatik.uni-erlangen.de

Healthcare organizations and healthcare providers are facing the challenge of delivering high-quality services to their patients at affordable costs. A high degree of specialization, prolonged medical care for the aging population, increasing costs for dealing with chronic diseases, and the need for personalized healthcare are prevalent trends in this information-intensive domain. The emerging situation necessitates a change in the way healthcare is delivered to the patients and *healthcare processes* are managed. Business process management (BPM) technology provides a key to implement these changes. Though patient-centered process support has become increasingly important in healthcare, BPM technology has not yet been broadly used in healthcare environments.

The ProHealth 2007 workshop was held in Brisbane in conjunction with the 5th International Conference on Business Process Management. ProHealth 2007 elaborates both the potential and the limitations of IT support for healthcare processes. It further provided a forum wherein challenges, paradigms, and tools for optimized process support in healthcare could be debated. In particular, ProHealth 2007 brought together researchers and practitioners from different communities (e.g., BPM, information systems, medical informatics, E-Health) who share an interest in both healthcare process support and advanced BPM technologies. The workshop dealt with different facets of process-oriented healthcare information systems, and gave insights into the social and technological challenges, applications, and perspectives emerging for BPM in this context.

In healthcare, process-oriented information systems have been demanded for more than 20 years and terms like *continuity of care* have even been discussed for more than 50 years. Yet, healthcare organizations are still characterized by an increasing number of medical disciplines and specialized departments that frequently only focus on their internal processes; i.e., optimization and automation of healthcare processes often stop at the border of healthcare departments.

The patient treatment process, however, requires interdisciplinary cooperation and coordination. The upcoming trend towards *healthcare networks* and *integrated care* further increases the need to effectively support interdisciplinary cooperation along with the patient treatment process. Recent studies discussing the preventability of adverse events in medicine recommend the use of information technology, since insufficient communication and missing information turned out to be among the major factors contributing to adverse events. Yet, there is still a discrepancy between the potential and the actual usage of IT in healthcare.

The ProHealth 2007 workshop focused on research which aims at closing this gap. It elaborated both the potential and the limitations of IT support for healthcare processes and discussed approaches existing in this context. Addressed topics included the modelling of healthcare processes, process-oriented system architectures in healthcare, workflow management in healthcare, IT support for guideline implementation and medical decision support, flexibility and exception handling in healthcare processes, requirements for medical guideline and medical pathway support, process optimization in healthcare organizations and healthcare networks, process interoperability in healthcare and healthcare standards, healthcare process patterns, secure healthcare processes, lifecycle management for healthcare processes, and healthcare process coordination.

Submitted papers were evaluated on the basis of relevance, originality, technical quality, and exposition. Papers had to clearly establish their research contribution as well as their relation to healthcare processes. We accepted six papers as full paper and one as short paper (out of 14 submissions). The seven presentations were complemented by two keynotes. We thank Samson W. Tu (Stanford University, School of Medicine) as well as Robert Dunlop (InferMed Ltd, London, UK) and John Fox (University of Oxford) for taking over this role.

We would like to thank the members of the Program Committee and the reviewers for their efforts in selecting the papers (in alphabetical order): W.M.P. van der Aalst, E. Ammenwerth, O. Bott, P. de Clercq, E. Coeira, J. Fox, Y. Han, S. Jablonski, K. Kuhn, R. Lenz (Co-chair), O. Marjanovic, S. Miksch, B. Mutschler, M. Peleg (Co-chair), S. Quaglini, S. Sadiq, M. Reichert (Co-chair), H. Reijers, H. Schuldt, Y. Shahar, T. Spil, A. ten Teije, P. Terenziani, S. Tu, D. Wang, B. Weber and M. Weske. They helped us to compile a high-quality program for the ProHealth 2007 workshop. We would also like to acknowledge the splendid support of the local organization and the BPM 2007 Workshop Chairs.

We hope you will find the papers of the ProHealth 2007 workshop interesting and stimulating.

Careflow: Theory and Practice

John Fox[1,3] and Robert Dunlop[2]

[1] Department of Engineering Science, University of Oxford, Oxford, UK
[2] *Infer*Med Ltd, London, UK
[3] UCL Medical School and Royal Free Hospital, London, UK

Abstract. In this presentation we shall review different ways of describing the *processes* of delivering patient care, and relate these to traditional workflow in business processes, and the concept of "careflow" proposed by Panzarasa and her colleagues in Pavia. We shall discuss the problem of designing a careflow application as a form of *process modelling*, to be contrasted with older paradigms ranging from rule-based alerts and reminders to Petri nets and Critical path analysis. We shall also consider the need for specialised *formalisms* for describing clinical processes, drawing on experience with workflow languages (e.g. BPEL4WS, BPMN), guideline modelling languages (e.g. GLIF, PRO*forma*), AI planning languages (e.g. PDDL, OCL) and "agent" programming systems (e.g. LALO, 3APL). The adoption of clinical workflow technology will greatly benefit from the availability of appropriate languages for declaratively representing processes of care. To explore some of the requirements for a clinical workflow technology we will review and critique the PRO*forma* process modelling language and the Arezzo® and Tallis toolsets which use it. The discussion will be illustrated with deployed applications and operational prototypes.

Guideline Models, Process Specification, and Workflow

Samson W. Tu

Stanford University School of Medicine
Stanford, CA 94305-5479, USA

Abstract. Many clinical practice guidelines use flowcharts to aid the description of recommendations specified in the guidelines. Similarly a number of computer-interpretable guideline formalisms use graphical task networks to organize knowledge formalized in these models. However, the precise meaning encoded in these graphical structures is often unclear. In this presentation, I will survey some of the computer-interpretable formalisms and analyzed the graphical representations that are used to express process information embodied in clinical guidelines and protocols. I will argue that we can distinguish a number of process types: (1) flowcharts for capturing problem-solving processes, (2) disease-state maps that link decision points in managing patient problems over time, (3) plans that specify sequences of activities that contribute toward a goal, (4) workflow specifications that model care processes in an organization, and (5) computational processes that generates recommendations for specific clinical situations. These process types may be related to each other. A plan of actions, for example, may be mapped to a workflow process when its actions are assigned to healthcare providers playing different roles. A flowchart may depict decisions and actions that are performed over time. Furthermore, a guideline formalism may not make a commitment to the nature of processes being modeled. Its process-specification language may be used to encode different types of processes. Nevertheless, understanding the nature of process being modeled is crucial when it comes to enacting the encoded guidelines and protocols to provide decision support in clinical workflow. A process description that models the problem-solving steps depicted in a narrative guideline, for example, may contain steps that are not appropriate as part of human-computer interactions in a busy clinic.

Restrictions in Process Design: A Case Study on Workflows in Healthcare

Jörg Becker and Christian Janiesch

European Research Center for Information Systems,
Leonardo Campus 3, 48149 Münster, Germany
{becker, janiesch}@ercis.de

Abstract. Automating existing processes is as paving cow path compared to major business process reengineering. However, this rather radical approach is not suitable for all business fields. It requires the freedom to modify organizational structures and free core business processes from non-value adding activities. In sectors like healthcare, there are a variety of legal restrictions and treatment guidelines practitioners have to comply with. Hence, freedom to reorganize the organization and to omit non-value adding activities is heavily compromised. In this paper we present findings from a case study that exemplify restrictions in process reorganization and suggest utilizing more moderate approaches to process management.

Keywords: Process design, design restrictions, process management, workflow management, healthcare, infection control.

1 Introduction

Hammer and Champy [8] see the practice of automating existing processes as “paving cow path” compared to major Business Process Reengineering (BPR). While it is desirable to take the blinkers off to free oneself from restraints of everyday procedures this rather radical approach is not suitable for all business fields. BPR requires the freedom to modify organizational structures and free core business processes from non-value adding activities. This requires introducing radical changes as well as new procedures. In business sectors like healthcare, there are a variety of legal restrictions and treatment guidelines practitioners have to comply with [21, 22]. Hence, freedom to reorganize the organization and to omit non-value adding activities as well as to change mandatory procedures and existing medical information systems (IS) is heavily compromised.

Thus, in healthcare one needs to utilize the less radical principles of Business Process Management (BPM) [1, 25]. Short and precise projects and continuous improvement offer a passable way despite a restrictive environment and legal pre-requisites. Through small iterations potential for process optimization, i.e. reduce cost, free staff from routine work, and improve patient safety without reengineering the company can be achieved. This paper presents a case, performed to show how BPM and commercial off-the-shelf workflow software contribute to lower the frequency of human errors in healthcare [10, 11] by introducing gradual change. The goal of the case study was to improve efficiency of an existing controlling process for hospital acquired infections (HAI).

The structure of the paper is as follows: First, a short literature review summarizes relevant facts on BPR and BPM as well as workflow management. In Section 3 an introduction to healthcare and clinical processes is given as characterization of the project. Section 4 comprises the case study including details on restrictions in process design as well as on the subsequent workflow implementation. The paper closes with conclusions to an outlook.

2 Fundamentals of Processes and Workflows

Processes are generally seen as any activity performed within a company or an organization [14]. In the context of this work, we define a process as "a completely closed, timely and logical sequence of activities which are required to work on a process-oriented business object" [1]. Consequently, a business process is considered as a special kind of process that is directed by business objectives of a company and by the business environment [1]. Business processes can be further classified into value creating core business processes and not value adding supplementary processes. Whereas core business processes are considered to contain corporate expertise and produce products or services that are delivered to customers [9, 16], supplementary business processes facilitate the ongoing operation of the core processes. This distinction is not intended to be always selective as one business process might be a core business process for one product and a supplementary business process for another [1].

The practice of business process reengineering, which emerged in the early 1990s, is seen as fundamental rethinking and radical redesign of business processes. In doing so dramatic improvements in critical, contemporary measures of performance, such as cost, quality, service, and speed can be achieved [8]. This kind of greenfield project, however, does not consider any existing operational sequences or organizational structures during the building of new processes at all. Furthermore, BPR targets the overall process perspective in one single shot rather than iteratively and continuously optimizing process performance. BPM on the other hand serves the planning, controlling, and monitoring of intra- and inter-organizational processes with regards to existent operational sequences and structures, in a consistent, continuous iterative way of process improvement [1].

In dependence to processes, workflows can be seen as part of a work process that contains the sequence of functions and information about the data and resources involved in the execution of these functions [2]. Workflows are an automated representation of a whole or part of a business process. Procedural rules define documents, information or tasks, which are to be passed from one participant to another for action [27].

To-be process models are used as sources to implement workflows. Therefore, process models need to be transformed into workflow models. Process models, however, primarily serve organizational (re-)design whereas workflow models focus on implementing IT support. That is why process models integrate functions in a lower level of granularity than workflow models [2].

While creating workflow models, workflow relevant data is required for the refinement of functions. In consequence, the necessity for a detailed specification of data needed during the execution of activities and data needed to create mathematic routing conditions emerges. Also, criteria when to initiate and when to terminate a workflow and how to handle errors are to be defined [28].

Workflow Management (WfM) aims at providing this automated process execution where the transitions between the individual activities are controlled by a Workflow Management System (WfMS) [28]. If an activity cannot be automated, a WfMS is concerned with demanding input from users while providing all necessary information needed to make a decision.

3 Healthcare and Clinical Processes

Healthcare providers are under constant pressure to reduce costs while the quality of care is to be improved [10]. Expenses for patient treatment and pharmaceuticals are relentlessly rising whereas reimbursements, refunded by insurance providers, are coupled to diagnosis-related groups and fixed [7].

Clinical processes can be classified as generic process patterns or medical treatment processes [12, 15]. Both types of processes may be designed and executed cross-department as well as cross-company. Generic process patterns help to coordinate healthcare processes among different people and organizational units. Medical treatment processes are the representation of an actual care process, which are considered to be the core processes of healthcare facilities. These processes highly depend on medical knowledge and case specific decisions [15]. Clinical process decisions are made by interpreting patient specific data according to clinical knowledge. In order to provide clinical decision support, patient specific data needs to be consolidated and a recallable representation of clinical knowledge needs to be provided in medical IS. The cooperation of clinical knowledge and complex decision support allows the implementation of treatment guidelines in highly flexible processes. Flexibility is required since treatment of patients is likely to differ from patient to patient. In consequence, medical treatment processes need to be quickly adaptable [12]. Medical treatment processes can be further described as a diagnostic-therapeutic cycle. Main components of the diagnostic-therapeutic cycle are: observation, reasoning, and action. These stages are iterated until no further action needs to be taken, i.e. the patient no longer requires treatment [12].

The historical evolvement of heterogeneous IS in healthcare may be due to a lack of expertise in implementing systems, missing investment abilities, but also the development of technology needs to be taken into account [13].

Infection Control (IC) is the process of preventing hospital acquired infections (HAI) by isolating sources of infections and limiting their spread. Nowadays, HAI are by far the most common complications affecting hospitalized patients or intensive care patients [3, 5]. Approximately 2 million patients are affected each year and costs add up to estimated $4.5 to $5.7 billion per year [4]. Identification of HAI typically involves testing of specimen in a laboratory. In addition, nurses working at nurse stations need to get specimen, physician need to order the specimen tests, and finally Infection Control Practitioners (ICP) need to ensure that all precautions have been taken, if a specimen was tested positive.

4 A Case Study on Workflows in Healthcare

4.1 Case Study Scenario

The case was performed in a major healthcare facility in the U.S. The facility consists of multiple independent hospitals. More than 7500 employees are employed at four sites, medical staff counts around 1000 physicians throughout the organization. Overall, almost 1000 beds are available for inpatient care. The scope of the project was to analyze the current IC process, suggest possible improvements through workflow, and finally enhance the current IT solution to increase process efficiency. The uniqueness of this project was rooted in the application service providing (ASP) environment [6, 19].

The team for this subproject consisted of five method experts for process modeling and implementation and six domain experts at the healthcare facility for analysis, test, and evaluation. Staff for technical support (ASP, rule engine) was provided by the overall project management. Process modeling, implementation and pre- and post-metrics took six month; build, test, and integration of the workflow needed to be done in only twelve weeks.

Analog to the theoretically exploration in the previous sections, the actual freedom to restructure processes or the organization was found to heavily compromised by legal restrictions and health care guidelines. Although this became apparent already during the first stage of analysis, consensus was achieved to pursue the project even though potential for optimization could not be fully utilized. It was agreed that a workflow focused pilot project would provide essential knowledge for more complex projects to come.

The software architecture is best described as a three tier, client server architecture built according to principles of service oriented architecture. The architecture consists of the web application tier, the top layer, constituted by web application servers running a user interface. The application tier, the middle layer, is constituted by application servers, a rules engine, and the WfMS.

We used Soarian® as medical IS in this project [18]. The Soarian® environment uses the third part WfMS TIBCO® Staffware Process Suite [23]. The WfMS can use services provided by the medical IS to add and remove items to/from user specific worklists. Users of the medial IS can trigger events, hence invoke the workflow engine to perform actions on demand. Whereas the WfMS evaluates simple routing conditions by itself, complex clinical conditions need to be evaluated with respects to clinical knowledge and patient specific data. Therefore, a rules engine based on Arden Syntax [17] can be used by the WfMS. This rules engine evaluates complex decision in clinical workflows.

4.2 As-Is Analysis

The IC process at the customer consists of two separate process fragments, synchronized over paper reports. The first part of the process starts as soon as a specimen is tested, if the patient to which the specimen is assigned to, is an inpatient at the facility. First, the lab result needs to be checked whether it indicates a positive statement of a HAI. This is done by the laboratory, which has already performed the test of the specimen. Once a lab result indicating a positive infection statement has been identified, an employee working at the laboratory calls the floor the patient is located on. In doing so, the nurse station is notified about an infectious patient and responsibility for putting the patient in isolation is passed to the nurse station. After the nurse station has initiated necessary tasks to isolate the patient, the first part of the IC process ends, as soon as the laboratory has completed the documentation of the lab result in the lab system. The process depicted in Figure 1 illustrates a structured overview of this part of the IC process.

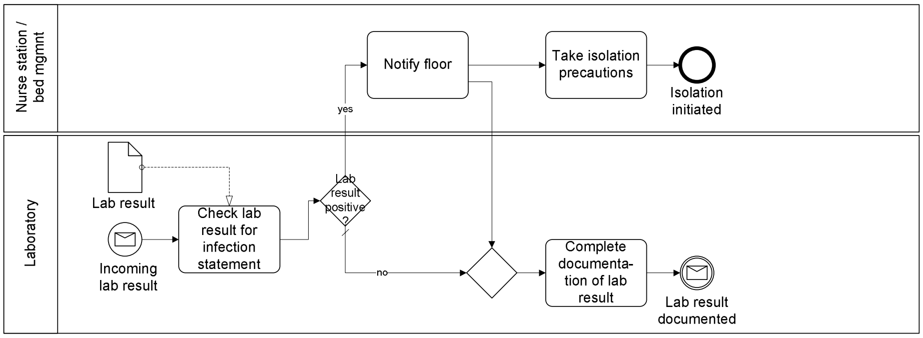

Nurse station / bed mgmnt
Laboratory
Notify floor
Take isolation precautions
Isolation initiated
yes
Lab result
Check lab result for infection statement
Lab result positive ?
Incoming lab result
no
Complete documenta-tion of lab result
Lab result documented

Fig. 1. As-is notification process

The second part of the IC process is a rather a controlling process. U.S. hospitals are required to have ICP supervising the handling of infections at each facility. Furthermore, each hospital must report the amount of infection occurrences on a yearly basis [26]. As for this facility, specific reports for each infectious disease were created on a monthly or even daily basis. These reports are the starting point of the second fragment of the IC process (cf. Figure 2). Once a report of an infectious disease is received by an ICP, it needs to be checked for infection statements. This task results in a list of patients that need a follow-up ensuring that patients who require isolation are actually put in isolation. During daily tours, the ICP does not only check if infection precautions have been taken for infectious patients but also controls, if the infection statement is transferred to the patient's chart. If a patient is not put in isolation, the ICP immediately initiates isolation.

The analysis revealed the following intrinsic problem domains: Since reports of infectious diseases are generated every afternoon, even on weekends, and every Friday afternoon respectively, an ICP does only recognize infections the morning after the report has been generated. However, these reports are triggering the execution of the second part of the IC process. Hence, they are critical in time.

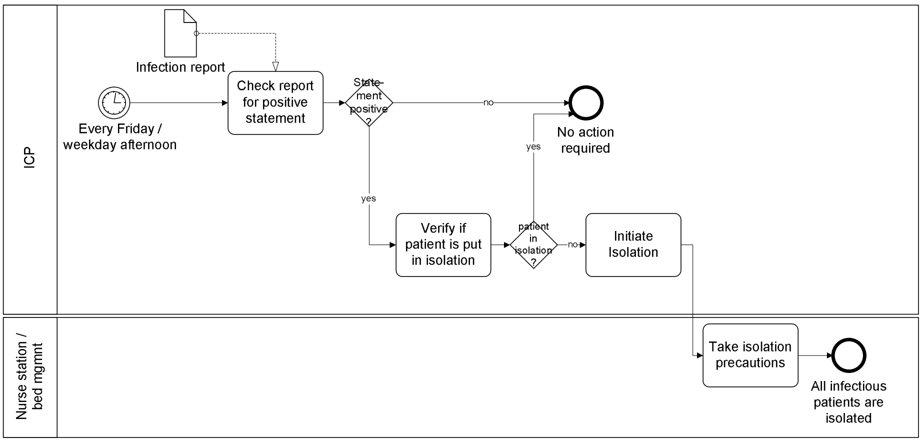

ICP
Nurse station / bed mgmnt
Infection report
Every Friday / weekday afternoon
Check report for positive statement
Statement positive ?
no
yes
No action required
Verify if patient is put in isolation
patient in isolation ?
Initiate Isolation
Take isolation precautions
All infectious patients are isolated

Fig. 2. As-is follow-up process

The analysis of the IC process clearly revealed that no IT is used after the ICP has received infection reports. In addition, further investigations indicated that ICP did not have any access to the WfMS yet. The review of reports and patient charts needs to be done manually as infection reports are printed and corresponding patient charts are not at hand instantly. A sample inquiry performed in collaboration with ICP indicated that screening all necessary documents takes almost 30 % of ICP's daily work time.

Even though ICP have responsibility for the handling infectious diseases, they are not directly participating in the IC process. Further inquiry revealed that the former process was to leave a voicemail for the ICP, assigned to the nurse station the patient was located at, as soon as an infection disease was stated. Once the ICP received the voicemail, the isolation of infectious patients was initiated and controlled by the ICP. However, since ICP do not work nights or at weekends this process was changed to directly call the floor and notify ICP only through reports. In doing so, the customer reported faster turnaround times in putting patients in isolation even though notification of ICP was delayed, i.e. follow-up processes start delayed.

4.3 To-Be Analysis

It became obvious that no change in matters of personnel capacities could be made. Neither could the involvement of ICP in notification tasks be increased nor could the controlling responsibilities of ICP transferred to an IS due to legal prerequisites [26]. The laboratory will still have to notify the nurse station directly, as ICP will, yet, not work during night hours or on weekends. In addition, it was agreed to not work on further improving the turnaround time for putting patients in isolation, but on the elimination of time wasted while generating, delivering, and reading reports as well as screening patient charts. In doing so, it was agreed to optimize IC tasks done by ICP without restructuring the organization or heavily affecting existing clinical and business processes.

Main focus was put on synchronizing the infection notification and the follow-up fragments of the as-is IC process (cf. Figures 1 and 2). The suggestion was to introduce a new workflow supported IC process. It was decided that a workflow should be used to screen new and modified lab results for statements of infectious diseases. Thereby, the lab system and the WfMS have been integrated in a way that every time a lab result is completed in the lab system, data is transferred. In doing so this data is available to users of the system in an instant. For a proper implementation of the integration events defined by Health Level 7 standards have been used. These events are incorporated in the workflow, which evaluates new and modified results and, furthermore, notifies ICP in case a positive infection statement is found. As the case study was a pilot project, it was agreed to limit the workflow to only cover two infections: clostridium difficile (CDIFF) and vancomycin resistant enterococcus (VRE). The workflow can be easily extended to cover other HAI, once clinical conditions have been identified and medical rules are created for an automated evaluation of those conditions.

The uppermost part of Figure 3 illustrates the notification process of the to-be IC process. According to the agreement made with the customer, this part of the IC process was not changed at all. However, the linkage between the notification process and the infection follow-up is made explicit in the to-be process models. A message interface has been added to the notification process and to the follow-up process respectively. The follow-up process is started immediately after notification process and valuable process time is saved by synchronizing both processes parts.

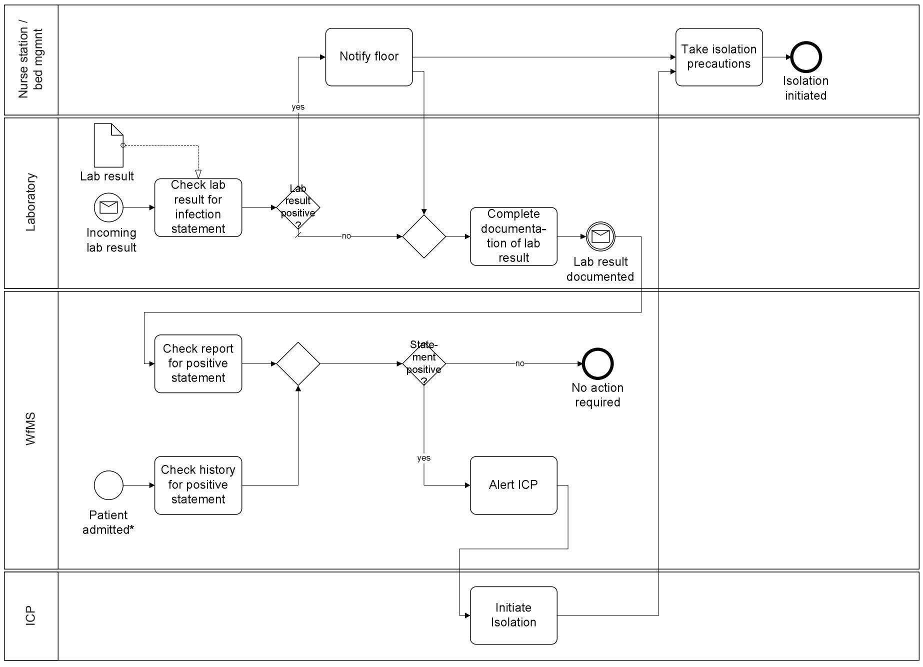

Nurse station / bed mgmnt
Laboratory
WfMS
ICP
Notify floor
Take isolation precautions
Isolation initiated
yes
Lab result
Incoming lab result
Check lab result for infection statement
Lab result positive ?
no
Complete documenta-tion of lab result
Lab result documented
Check report for positive statement
State-ment positive ?
no
No action required
yes
Patient admitted*
Check history for positive statement
Alert ICP
Initiate Isolation
* covering admissions, preadmissions, observation patients, and emergency patients

Fig. 3. To-be notification process and to-be follow-up process

More significant changes have been made to the infection follow-up process that is shown in the lowermost part of Figure 3. The process has been streamlined in order to allow efficient automation through the use of a workflow. To achieve the requested level of automation a WfMS has been introduced to the process. The follow-up process is triggered every time a lab result is documented in the lab system. The lab result is immediately evaluated through workflow. Only if the lab result either indicates a positive CDIFF or VRE statement, the ICP assigned to the nurse station the infectious patient is located at, will be notified. The notification, once again, is performed automatically through workflow. Hence, the ICP will instantly see a new task on his work list. The validation whether a patient has been put in isolation still needs to be done manually, but ICP are now able to access medical records electronically. This enables ICP to work independent of any paper reports or patient charts. Unfortunately, initiating the isolation of patients needs to be executed and monitored manually due to missing integration of participating actors (e.g., bed management).

In consequence of extensively implementing the IC process with the use of IS, new possibilities for further process enhancement have been established. Utilizing new benefits, the to-be infection follow-up process was designed to be executed not only every time a lab result has been documented but also every time a patient is admitted as inpatient, an inpatient is pre-admitted, an outpatient is kept in hospital for observation or a patient requires emergency care. In either case, the last six month of the patient's medical record are screened for an occurrence of an infection. If an infection statement was found within the past six months, the patient is considered to require isolation and infection precautions are taken. Since many HAI (e.g. VRE), are likely to reappear, if the last infection is more recent than six month, these new characteristics enabled the increase of process quality and patient safety in addition to the increase of efficiency.

4.4 Implementation and Controlling

The IC workflow is implemented based on a hierarchical model of procedures and sub-procedures. Thereby, the top level procedure is used to coordinate the overall process flow. The functionality to initiate new workflow instances (e.g., inpatient admit) and terminate existing instances (e.g., patient discharge) is built upon the workflow event handler. So called subscriptions which are basically rules that filter and evaluate selected events allow the definition of case generation and case termination respectively. The workflow has been implemented according to the to-be process models. Thus, a workflow case is initiated the first time a patient is admitted, pre-admitted, put in an observation bed or in the emergency department. The workflow instance will terminate as soon as the patient is sent home (cf. Figure 4).

The purpose of the IC model workflow is to alert users of patients having infectious diseases (e.g., CDIFF, VRE). Therefore, the workflow is designed to immediately check new and modified lab results for patterns indicating a positive statement of an infectious disease. In addition, the workflow checks the medical record for a history of an infectious disease. Alerts on the work lists are created and patients are put on the census list of selected users (e.g., ICP), if a patients has an active indication to be put in isolation. According to the to-be process the IC workflow is required and triggered by multiple events.

Since every event provides a different set of data, the first three activities deal with consolidating data. This ensures that all workflow relevant data is available instantly. The sub-procedures *PT discharge* and *PT discharge cancelled* are used to perform a delayed workflow termination, if the patient was discharged and the discharge was not cancelled within eight hours.

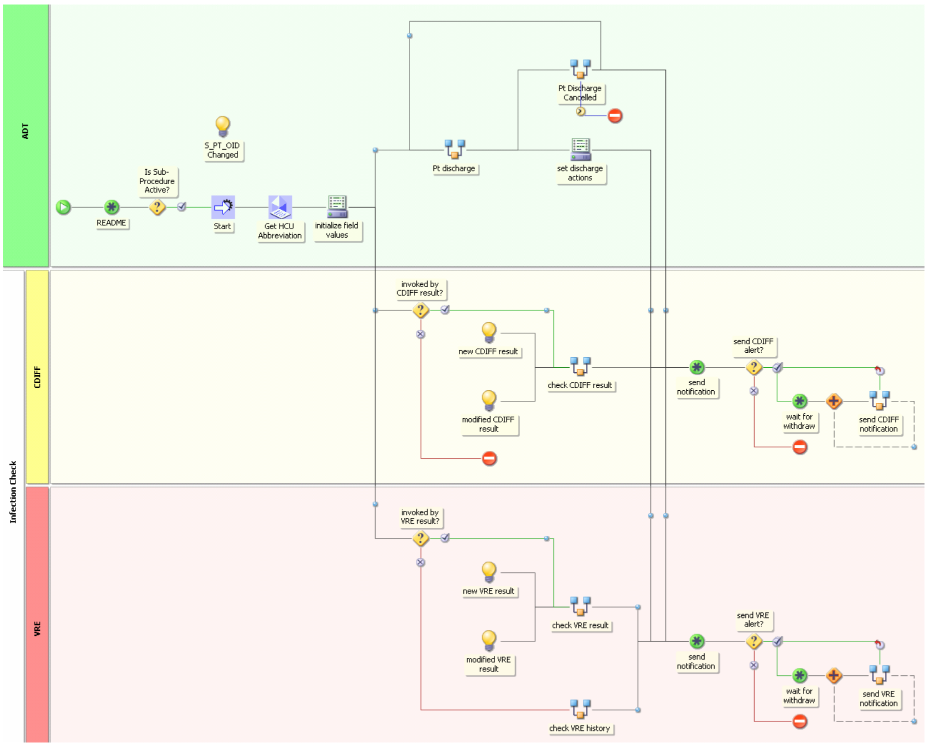


Fig. 4. Infection Control main procedure

Following the path to the VRE infection checking activities, a conditional router (*invoked by VRE result?*) is used to evaluate whether a lab result needs to be checked for VRE patterns or a patients medical record needs to be screened for positive VRE statements. Both, the checking of a lab result and history screen are performed in sub-procedures. Thus, relevant data like patient identifiers, visit identifiers or result values must be passed to the called sub-procedure. Values returned by the sub-procedure must be mapped in the calling procedure. Once the VRE lab result has been evaluated automatically or the medical records has been screened for previous infections without human intervention, another conditional router is used to examine if a user needs to be alerted and if this patient needs to be added to the user's census list. Alerting users and adding the patient to the census is automatically performed by another sub-procedure call.

Since the notification sub-procedure stays active until the alerted user confirms to have recognized the infection alert (e.g., user releases the alert) and new lab results may be submitted in between, there are cases where the alert message needs to be changed or the alert needs to be withdrawn entirely due to new information provided by new lab results. Therefore, functionality to remove or replace alerts is provided. This is done by withdrawing created alerts (e.g., withdraw connection of *send VRE alert?* and *send VRE* notification) before creating a new alert (*wait for withdraw*). Every time a patient is discharged from hospital all alerts will be removed and the patient will be dropped of the user's census list. If a discharge is cancelled the status of the last notification will be restored.

Concurrent to the project realization data was collected for a detailed analysis of the project outcome. The key metric defined in collaboration with the customer is *time to notification*. Time to notification as measurement of time, represents the time spent notifying an ICP of a newly identified infectious patient. In doing so, time measurement started as soon as a positive lab result had been documented by the laboratory. Time measurement stopped once an ICP had been notified. Data ascertainment of notification dates needed to be done manually. Before the implementation of the IC workflow, ICP were asked to write down dates as soon as they had been notified of infectious patients. After the workflow had been implemented the date of notification was considered to be the date when an ICP released the automatically created infection alert.

Independent of the measurements of time to notification more data was collected in order to identify how much time was spent while screening paper reports or patient charts. Therefore, ICP were asked to write down hours spent on reading reports and screening patient charts before the implementation of the workflow. 95 cases were recorded over the course of two month each for pre- and post-measurement.

The comparison of pre- and post-metrics reveals that time to notification was reduced by more than 75 % after the workflow had been implemented (cf. Table 1).

Table 1. Comparison of post- and pre-measurements

	Average (VRE)	Average (CDIFF)	Overall Average
Time to notification: pre-metrics	150.00 h	25.68 h	69.68 h
Time to notification: post-metrics	20.67 h	15.68 h	16.99 h
Difference	129.33 h (86.22 %)	10.00 h (38.94 %)	52.69 h (75.61 %)

Time to notification averaged out at 70 hours before the implementation and has reduced to an average of 17 hours after the implementation of the IC workflow. Although 17 hours still appears to be quite a lot, it is obvious that substituting the old paper reports based IC process for the new workflow supported IC process increased efficiency greatly. Furthermore, ICP spent an average of 30 % of their daily work time on screening infection reports and patient charts. This was decreased to almost zero after the implementation of the workflow, since required patient information is now available instantly through the integration of ICP to the WfMS. It has to be noted that before, VRE reports have only been created every Friday afternoon. This means, that only those VRE occurrences of past Saturday to Friday would appear on Friday's report, which is read on the following Monday at earliest.

5 Conclusion and Next Steps

In this paper we presented reasons for the inappropriateness of greenfield project approaches for the optimization of clinical processes in healthcare.

The main restrictions, which hampered reengineering, originated from judicial and budgetary restrictions. Due to budgetary reasons, it was not possible to increase staff for weekend support or to include all departments (e.g., bed management). Due to judicial restriction, pagers and e-mail were not available at the time because of increased IT test requirements. Furthermore, it was not intended to touch existing IS (e.g., laboratory IS). Several laws or certifications required certain procedures. Significant restrictions impose the certification requirements of The Joint Commission (JCAHO) [20] as it entails implementing the hygiene guidelines of the Center for Disease Control and Prevention (CDC) and the Public Health Service Act [24]. As a consequence, it was, e.g., not possible to transfer authority for ICP to an IS.

Still, significant potential for automating coordination and evaluation task (e.g., calling floors, screening charts) was discovered, utilized and in consequence contributed to patient safety without radically redesigning the organization. Improvements were made through the implementation of HAI history screening, as every inpatient, now, is screened for a positive history. The process quality has been improved through reducing the risk of human errors, since ICP no longer rely on manually generated paper reports or voicemails. The implementation of the workflow greatly contributed to the process of getting health workers, especially ICP, online. It was observed that the automated IC process was functioning as incentive to overcome the negative attitude some health workers might have concerning IT in inpatient care.

References

[1] Becker, J., Kugeler, M., Rosemann, M. (eds.): Process Management: A Guide for the Design of Business Processes, 2nd edn. Springer, Berlin (to appear, 2007)

[2] Becker, J., zur Mühlen, M.: Towards a Classification Framework for Application Granularity in Workflow Management Systems. In: Jarke, M., Oberweis, A. (eds.) CAiSE 1999. LNCS, vol. 1626, pp. 411–416. Springer, Heidelberg (1999)

[3] Borst, F., et al.: Happy Birthday DIOGENE: A Hospital Information System Born 20 Years Ago. International Journal of Medical Informatics 54, 157–167 (1999)

[4] Burke, J.P.: Infection Control: A Problem for Patient Safety. New England Journal of Medicine 348, 651–656 (2003)

[5] Centers for Disease Control and Prevention (CDC): National Nosocomial Infections Surveillance (NNIS) System Report, Data Summary from January 1992 through June 2004, Issued October 2004. American Journal of Infection Control 32, 470-485 (2004)

[6] Dewire, D.T.: Application Service Providers. Information Systems Management 17, 14–19 (2000)

[7] DiMasi, J.A., Hansen, R.W., Grabowski, H.G.: The Price of Innovation: New Estimates of Drug Development Costs. Journal of Health Economics 22, 151–185 (2003)

[8] Hammer, M., Champy, J.: Reengineering the Corporation: A Manifesto for Business Revolution, 1st edn. HarperBusiness, New York (1993)
[9] Harmon, P.: Business Process Change: A Manager's Guide to Improving, Redesigning, and Automating Processes. Morgan Kaufmann, San Francisco, CA (2003)
[10] Institute of Medicine: Crossing the quality chasm: A New Health System for the 21st Century. National Academies Press, Washington, DC (2001)
[11] Kohn, L.T., Corrigan, J.M., Donaldson, M.S. (eds.): To Err is Human: Building a Safer Health System. National Academy Press, Washington, DC (2000)
[12] Lenz, R., Reichert, M.: IT Support for Healthcare Processes. In: van der Aalst, W.M.P., Benatallah, B., Casati, F., Curbera, F. (eds.) BPM 2005. LNCS, vol. 3649, pp. 354–363. Springer, Heidelberg (2005)
[13] Magruder, C., Burke, M., Hann, N.E., Ludovic, J.A.: Using Information Technology to Improve the Public Health System. Journal of Public Health Management and Practice 11, 123–130 (2005)
[14] Object Management Group: Business Process Modeling Notation (BPMN) Specification 1.0 (2006), Available: http://www.omg.org/cgi-bin/apps/doc?dtc/06-02-01.pdf
[15] Panzarasa, S., Stefanelli, M.: Workflow Management Systems for Guideline Implementation. Neurological Sciences 27, 245–249 (2006)
[16] Porter, M.E.: Competitive Advantage: Creating and Sustaining Superior Performance. The Free Press, New York (1985)
[17] Pryor, T.A., Hripcsak, G.: The Arden Syntax for Medical Logic Modules. International Journal of Clinical Monitoring and Computing 10, 215–224 (1993)
[18] Siemens AG: Soarian® (2007), Available: http://www.soarian.com/
[19] Tao, L.: Shifting Paradigms with the Application Service Provider Model. IEEE Computer 34, 32–39 (2001)
[20] The Joint Commission: 2007 National Patient Safety Goals (2007), Available: http://www.jointcommission.org/PatientSafety/NationalPatientSafetyGoals/07_hap_cah_npsgs.htm
[21] The Medical Letter Inc.: Choice of Antibacterial Drugs. Treatment Guidelines from The Medical Letter 2, 13-26 (2004)
[22] The Medical Letter Inc.: Treatment of Clostridium Difficile-Associated Disease (CDAD). The Medical Letter on Drugs and Therapeutics 48, 89-90 (2006)
[23] TIBCO Software GmbH: TIBCO® Staffware Process Suite (2005), Available: http://bpmleader.tibco.com/de/docs/staffware_processsuite_datasheet.pdf
[24] U.S. Food and Drug Administration: Public Health Service Act. (1944), Available: http://www.fda.gov/opacom/laws/phsvcact/phsvcact.htm
[25] van der Aalst, W.M.P., ter Hofstede, A.H.M., Weske, M.: Business Process Management: A Survey. In: van der Aalst, W.M.P., ter Hofstede, A.H.M., Weske, M. (eds.) BPM 2003. LNCS, vol. 2678, pp. 1–12. Springer, Heidelberg (2003)
[26] Weber, S.G., et al.: Legislative Mandates for Use of Active Surveillance Cultures to Screen for Methicillin-Resistant Staphylococcus Aureus and Vancomycin-Resistant Enterococci: Position Statement From the Joint SHEA and APIC Task Force. Infection Control and Hospital Epidemiology 28, 249–260 (2007)
[27] Workflow Managment Coalition: Terminology & Glossary 3.0 (1999), Available: http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf
[28] zur Mühlen, M.: Workflow-based Process Controlling: Foundation, Design and Application of Workflow-driven Process Information Systems. Dissertation. Logos, Berlin (2004)

Declarative and Procedural Approaches for Modelling Clinical Guidelines: Addressing Flexibility Issues

Nataliya Mulyar[1], Maja Pesic[1], Wil M.P. van der Aalst[1], and Mor Peleg[2]

[1] Eindhoven University of Technology
GPO Box 513, NL5600 MB Eindhoven, The Netherlands
{n.mulyar, m.pesic, w.m.p.v.d.aalst}@tue.nl
[2] Department of Management Information Systems, University of Haifa
Mount Carmel, 31905, Israel
morpeleg@mis.hevra.haifa.ac.il

Abstract. Recent analysis of clinical Computer-Interpretable Guideline (CIG) modelling languages from the perspective of the control-flow patterns has revealed limited capabilities of these languages to provide flexibility for encoding and executing clinical guidelines. The concept of flexibility is of major importance in the medical-care domain since no guarantee can be given on predicting the state of patients at the point of care. In this paper, we illustrate how the flexibility of CIG modelling languages can be improved by describing clinical guidelines using a *declarative* approach. We propose a CIGDec language for modelling and enacting clinical guidelines.

Keywords: Clinical guidelines, Computer-interpretable guidelines, flexibility, modelling languages, declarative model specification,temporal logic.

1 Introduction

Clinical Computer-Interpretable Guidelines (CIG) impact clinician behavior (i.e., quality of patient care, costs, etc.) to a great extent when they are implemented as computerized guidelines that deliver patient-specific recommendations during patient encounters [1]. A number of guideline modelling languages have been developed to represent guidelines in a machine and human understandable format that enables guideline execution. Control-flow perspective of guidelines significantly influences the clinical behavior, because it determines the order of actions in medical treatments. Other perspectives (e.g., a model of patient data including temporal data, a model of medical actions and decisions, etc.) add more contextual details to the control-flow perspective, determining the exact favorable clinical behavior. Unfortunately, due to the absence of a single standard for developing CIG modelling languages, the functionality of decision-support systems employing such modelling languages from the perspective of the control-flow differs to a great extent.

We analyzed the suitability of four modelling languages Asbru, PRO*forma*, GLIF and EON for expressing control-flow patterns [2] and revealed that these languages do not offer more control-flow flexibility than process modelling languages employed by the Workflow Management Systems (WFMS) [3]. This is remarkable since one would expect CIG modelling-languages to offer dedicated constructs allowing for more build-time and runtime flexibility. Accommodating *flexibility* into guidelines means that the CIG would be sensitive to the characteristics of specific patients and specific health care organizations [4].

The modelling languages we analyzed explicitly model a care process by specifying the steps and the order in which these steps are to be executed. Although process languages allow for some flexibility by means of modelling alternative paths, any of which could be taken depending on some a-priori available data, they are incapable of handling exceptional or unpredicted situations. Exceptional situations have to be modelled explicitly. However, modelling of all possible scenarios results in complex models and is not feasible since exceptional situations and emergencies may arise at any point in time. This makes it difficult or even impossible to oversee what activity should be performed next. To overcome these problems, i.e. reduce the complexity of models, and to allow for more flexibility in selecting an execution path, *in this paper* we propose CIGDec as a declarative language for modelling clinical guidelines. Unlike imperative languages, declarative languages specify the “what” task should be performed without determining of the “how” to perform it. CIGDec specifies by means of constraints the rules that should be adhered to by a user during a process execution while leaving a lot of freedom to the user in selecting tasks and defining the order in which they can be executed. CIGDec can be considered as a variant of ConDec [5] and DecSerFlow [6].

The remainder of this paper is organized as follows. In Section 2 we introduce CIG modelling languages Asrbu, GLIF, EON and PRO*forma* using a patient-diagnosis scenario. In Section 3 we introduce CIGDec and illustrate a CIGDec model of the patient-diagnosis scenario. We discuss the drawbacks and advantages of the proposed language in Section 4. Related work is presented in Section 5. Section 6 concludes the paper.

2 Computer-Interpretable Guidelines

This section describes the main concepts of four well-known CIG modelling languages: Asbru, EON, GLIF, and PRO*forma*. These have been evaluated from the control-flow perspective using the workflow patterns [7]. We introduce the main concepts of these languages by modelling the following patient diagnosis scenario in the tools Asbru-View, Protege-2000 (for EON and GLIF) and Tallis respectively. A patient is registered at a hospital, after which he is consulted by a doctor. The doctor directs the patient to pass a blood test and urine test. When the results of both tests become available, the doctor sets the diagnosis and defines the treatment strategy.

While specifying the behavior of the scenario, we immediately reflect on the possibilities to deviate from this scenario required for example in an emergency case. In particular, we indicate whether it is possible to skip a patient registration step and immediately start with the diagnosis; to perform multiple tests of the same kind or perform only one of them; and to perform the consultancy by the doctor after performing one of the tests again. Next to it, we indicate the degree of support of the control-flow patterns by the analyzed modelling languages. Table 1 summarizes the comparison of the CIG modelling languages from the perspective of the control-flow patterns [7]. The complete description of the patterns and how they are supported by the analyzed languages can be found in [8,3] respectively.

Table 1. Support for the Control–flow Patterns in (1)Asbru, (2)EON, (3)GLIF, and (4)PRO*forma*

Basic Control–flow	1	2	3	4	New Patterns	1	2	3	4
1. Sequence	+	+	+	+	21. Structured Loop	+	+	+	+
2. Parallel Split	+	+	+	+	22. Recursion	+	-	-	-
3. Synchronization	+	+	+	+	23. Transient Trigger	-	-	-	+
4. Exclusive Choice	+	+	+	+	24. Persistent Trigger	-	-	+	+
5. Simple Merge	+	+	+	+	25. Cancel Region	-	-	-	-
Advanced Branching and Synchronization					26. Cancel Multiple Instance Activity	+	-	+	+
6. Multi-choice	+	+	+	+	27. Complete Multiple Instance Activity	+	-	-	+
7. Structured Synchronizing Merge	+/-	-	-	+	28. Blocking Discriminator	-	-	-	-
8. Multi-merge	-	-	-	-	29. Cancelling Discriminator	+	-	-	+
9. Structured Discriminator	+	+	+	+	30. Structured N-out-of-M Join	+	-	+	+
Structural Patterns					31. Blocking N-out-of-M Join	-	-	-	-
10. Arbitrary Cycles	-	+	+	-	32. Cancelling N-out-of-M Join	-	-	-	+
11. Implicit Termination	+	+	+	+	33. Generalized AND-Join	-	-	-	-
Multiple Instances Patterns					34. Static N-out-of-M Join for MIs	-	-	-	-
12. MI without Synchronization	-	-	-	-	35. Static N-out-of-M Join for MIs with Canc.	-	-	-	-
13. MI with a priori Design Time Knowledge	+/-	+/-	+/-	+/-	36. Dynamic N-out-of-M Join for MIs	-	-	-	-
14. MI with a priori Run-Time Knowledge	-	-	-	-	37. Acyclic Synchronizing Merge	-	-	-	+
15. MI without a priori Run-Time Knowledge	-	-	-	-	38. General Synchronizing Merge	-	-	-	-
State-Based Patterns					39. Critical Section	+	-	+	-
16. Deferred Choice	+	-	+	+	40. Interleaved Routing	+	-	+	-
17. Interleaved Parallel Routing	+	-	-	-	41. Thread Merge	-	-	-	-
18. Milestone	-	-	-	+	42. Thread Split	-	-	-	-
Cancellation Patterns					43. Explicit Termination	-	-	-	-
19. Cancel Activity	+	+	+	+					
20. Cancel Case	+	-	+/-	+					

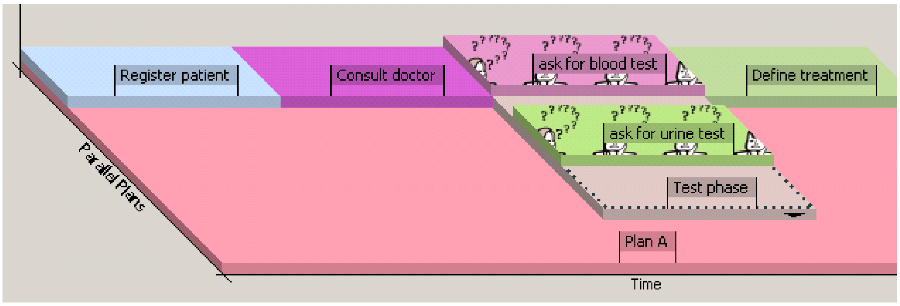


Fig. 1. The patient-diagnosis scenario modelled in Asbru

Figure 1 presents the scenario modelled in AsbruView, which is a markup tool developed to support authoring of guidelines in Asbru [9]. A process model in Asbru [10] is represented by means of a time-oriented skeletal plan. Skeletal plans are plan schemata at various levels of detail, which capture the essence of the procedure, but leave room for execution-time flexibility. The root plan A is composed of a set of other plans that are represented as 3-dimensional objects, where the width represents the time axis, the depth represents plans on the same level of the decomposition (i.e. which are performed in parallel), and the height represents the decomposition of plans into sub-plans.

As the time axis shows, plans *Register patient*, *Consult with doctor*, *Test phase* and *Define the treatment* are executed sequentially. The *Test phase* is a parallel plan consisting of two activities *ask for urine test* and *ask for blood test*. The parallel plan requires all enclosed plans to be completed in order to pass the flow of control to the next plan. In this model, only two types of plans were used: sequential (root plan) and parallel plan (Test phase plan). In AbsruView plans of type: Any-order, Unordered, Cyclical, and If-then-else, and actions of type: Ask and Variable Assignment can be visualized.

Deviations from the modelled scenario are not possible in AsbruView, since all plans are structured and their order is strictly defined. It would be possible to adjust the model and implicitly incorporate all required execution paths. In particular, the Cyclical Plan should be used in order to iterate the execution of a certain task. In order to relax the parallel order of the blood- and urine-tests' tasks, an Any-order Plan could be used. However, the behavior of the model would be still deterministic and not allow for much flexibility. In Asbru there is a concept of plan activation mode. It allows conditions for aborting, suspending and resuming a plan. This can be relevant for the case of registering a patient and not having all the needed data initially: a plan is suspended and later resumed. As the pattern-based analysis showed [3], Asbru is able to support 20 out of 43 control-flow patterns. Asbru uniquely supports the recursive calls and interleaved parallel routing, which are the features not directly supported by other analyzed languages.

Main modelling entities in EON [11] are scenarios, action steps, branching, decisions, and synchronization [12,13]. A scenario is used to characterize the state of a patient. There are two types of Decision steps in EON, i.e. a Case step (select precisely one branch) and a Choice step (select at least one branch). An Action step is used to specify a set of action specifications or a sub-guideline that are to be carried out. Branch and Synchronization steps are used to specify parallel execution. We omit EON model since it is very similar to model created in GLIF (see Fig. 2).

The following features offered by EON can be used in order to make the model of the patient-diagnosis scenario more flexible. A Scenario can be used to model different entry points to the model. This allows to "jump" into the middle of the model and to start execution from that point. This feature is useful for emergency cases where for example a registration step has to be skipped and immediate treatment procedure has to be started. Unfortunately, EON offers not much flexibility with respect to synchronization of multiple branches, i.e. it allows the *define treatment* task to be executed only if a single or all branches have been executed. However, it is incapable of predicting how many branches were selected and performing a partial synchronization after all selected branches were executed. From all analyzed modelling languages, EON supports the lowest number of the control-flow patterns, i.e. only 11 out of 43.

GLIF3.5 [14] is a specification method for structured representation of guidelines. To create a model in GLIF, an ontology schema and a graph widget have to be loaded into the *Protege-2000* environment. Figure 2(a) visualizes the GLIF model of the basic patient-diagnosis scenario. In GLIF3.5 five main modelling entities are used for process modelling, i.e. an Action Step, a Branch Step, a Decision Step, a Patient-State Step, and a Synchronization Step. An Action Step is a block for specifying a set of tasks to be performed, without constraints set on the execution order. It allows for including sub-guidelines into the model. Decision steps are used for conditional and unconditional routing of the flow to one out of multiple steps. Branch and Synchronization steps are used for modelling concurrent steps. A Patient-State Step is a guideline step used for describing a patient state and for specifying an entry point(s) to a guideline.

In order to allow the behavior of the basic patient-diagnosis scenario shown in Figure 2(a) to deviate, all possible pathes have to be explicitly modelled. Figure 2(b)

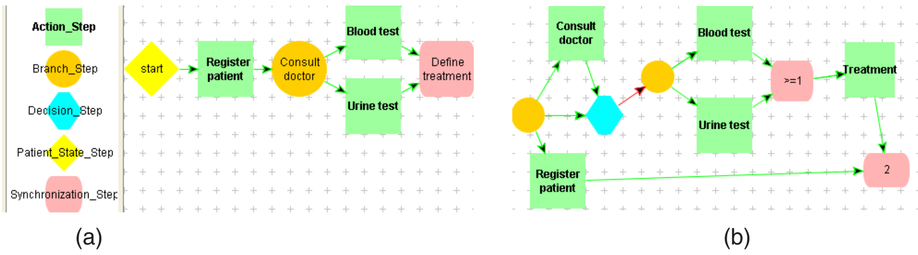


Fig. 2. The patient-diagnosis scenario modelled in GLIF3.5/*Protege*

represents a scenario, in which *Register patient* step can be done in parallel to any other step, but it has to be exactly once to complete the process (if more than once is desired, an iteration condition for *Register patient* step can be added which resembles a while loop: while not all patient data has been entered, repeat Register Patient. In this scenario, a decision can be taken to order tests or to proceed to treatment without tests. However, treatment or ordering of tests cannot be done before consulting with a doctor. One or two tests can be ordered before proceeding to treatment. Figure 2(b) shows how complex the model has become after we introduced several deviations from the basic scenario. Thus, this specification needs to model graphically all the possible paths of execution, and it is not very scalable.

Similar to EON, GLIF allows multiple entry points into the model to be specified by means of the Patient-State step. This allows the execution to start from any point where a patient enters a scenario model while skipping tasks-predecessors. GLIF offers more variants for synchronizing parallel branches, i.e. to synchronize after one, several or all tasks have been completed. However, GLIF is incapable of synchronizing branches when it is unknown which branches and how many of them will be chosen. This explains why the number of control-flow patterns supported by GLIF (17 out of 43) is bigger than in EON but still smaller than Asbru.

PRO*forma* [15] is a formal knowledge representation language for authoring, publishing and executing clinical guidelines. It deliberately supports a minimal set of modelling constructs: actions, compound plans, decisions, and enquiries that can be used as tasks in a task network. In addition, a keystone may be used to denote a generic task in a task network. All tasks share attributes describing goals, control flow, preconditions, and postconditions. A model of the basic patient-diagnosis scenario created in Tallis is shown in Figure 3(a). Note that in PRO*forma* control-flow behavior is captured by modelling constructs in combination with the scheduling constraints. Scheduling constraints are visualized as arrows connecting two tasks, meaning that the task at the tail of the arrow may become enabled only after the task at the head of the arrow has completed. To deviate from the basic scenario, some of the scheduling constraints should be removed as it is shown in Figure 3(b).

In contrast to all examined languages, PRO*forma* allows for late modelling, i.e. if it is not clear in advance what steps exactly should be performed, these steps are modelled by means of keystones, which are substituted by a desired type of the task before the model is deployed. Furthermore, by means of triggers it is possible to specify that a task

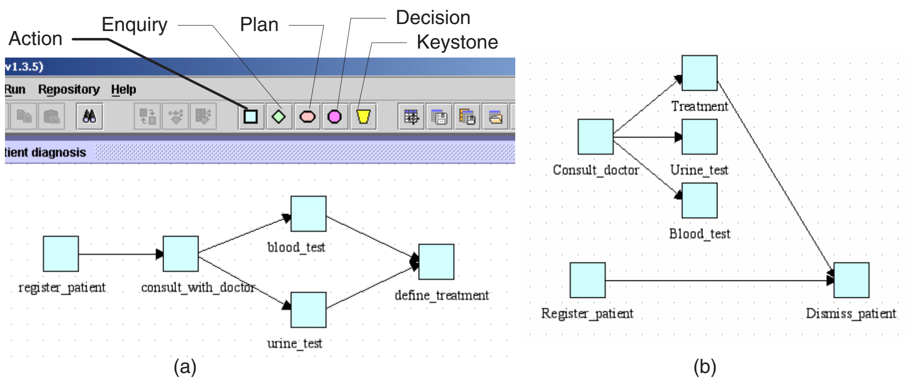


Fig. 3. The patient-diagnosis scenario modelled in PRO*forma*/Tallis

has to be performed even if the task's preconditions were not satisfied. PRO*forma* also allows for more flexibility during the synchronization of multiple paths, thus it is able to predict how many paths from the available ones were selected and to merge them when they have completed. Furthermore, scheduling constraints in PRO*forma* are not obligatory. This means that stand-alone tasks may be activated upon the fulfillment of a pre-condition. PRO*forma* has the highest degree of pattern-support from all analyzed languages, i.e. it supports 22 out of 43 patterns.

The medical community has always emphasized that it is impossible to use workflow formalisms because of specific requirements such as flexibility. However, when we examined guideline modelling languages we didn't find more flexibility than in todays workflow and BPM products. Given a large variety of process modelling languages nowadays it makes no sense to develop more complicated language which would support more control-flow patterns. Instead, we take a completely new approach and propose a CIGDec language for encoding clinical guidelines.

3 Declarative Description of Clinical Guidelines

In this section we present the CIGDec declarative language and show benefits of applying it for modeling clinical guidelines.

Modelers who use traditional CIG modelling languages have to represent all possible scenarios (normal and exceptional) that can occur during the execution. Such a model has to include all possible scenarios that can occur during the execution. This means that CIG modelers have to predict in detail all possible execution paths in advance for the guideline they are modelling. The model itself tends to be very complex and strictly predefines all relationships between all steps in the guideline. Such a model not only prescribes to users what to do, but it also contains a detailed specification about how to do it. Hence, traditional CIG modelling languages are of an imperative nature.

CIGDec is a declarative language, i.e., its models specify what to do and leave it up to the user to decide how to work depending on the case. CIGDec models do not require all possible scenarios to be predicted in advance. On the contrary, the model consists of a set of tasks and some dependencies (relationships) between these tasks. Dependencies between tasks can be seen as some general rules that should always hold in the guideline. Any task in the model can be performed by a user if and only if none of the specified rules is violated. As an extreme example, a CIGDec model that consists only of a set of tasks without dependencies would represent a completely free guideline, where a user can execute any task in any desired order. As more rules in the model as less possibilities to deviate from a certain execution order is given to the user. Therefore, rules constrain the model. Hence, we refer to dependencies between tasks (rules) as to *constraints*.

Any CIG model consists of a set of tasks and some relationships between them specifying the exact order of tasks. Typically, traditional languages use a predefined set of constructs that can be used to define relations between tasks: 1) sequence, 2) choice, 3) parallelism, and 4) iteration. These constructs are used to define the exact control-flow (order of tasks) in the guideline. In CIGDec, this set of constructs is unlimited, i.e., constructs can be added, changed and removed, depending on the requirements of the application, domain, users, etc. We refer to constructs used for defining possible types of dependencies between tasks in CIGDec as to *constraint templates*. Each template has its semantics, which is formally represented by one Linear Temporal Logic (LTL) formula. This semantics is used for the computerized enactment of the guideline [5]. LTL is extensively used in the field of model checking, where the target model is verified against properties specified in LTL [16,17]. For computerized enactment of CIGDec model we use algorithms for translating LTL expressions into automata developed in the model checking field [18,6,5]. Since LTL formulas can be very complex and hard to understand, each template also has unique graphical representation for users. In this way, we ensure that CIGDec users do not have to be LTL experts in order to work with models [5]. Although the set of templates is 'open', we propose a starting collection of eleven templates in [19].

When looking at a traditional CIG model, one usually tries to find the starting point and then follows the control-flow until the end point is reached. This cannot be applied to CIGDec models. Constructs (lines) do not necessarily describe the order of tasks, but rather various dependencies between them. In our starting set of constraint templates we distinguish between two types of templates: 'existence' (unary) templates, and binary templates that can represent a 'relation' or 'negative relation'.

'Existence' templates are unary templates because they involve only one task. Generally, they define the cardinality (possible number of executions) of the task. Binary templates involve two tasks. For example, a special line between tasks might mean that these two tasks include each other (e.g., 'co-existence' template between tasks *A* and *B* specifies that if *A* happens then *B* happens and vice versa, without specifying in which order). The 'responded existence' constraint specifies that if one task is performed then the other task before or after the first one. There are also some binary templates that consider the order of activities. One example is the 'response' template, which specifies that the a given task has to be performed at least once after the other task has been completed. Note that in all these examples it was possible to have an arbitrary execution of other tasks between the two related tasks.

3.1 CIGDec Model for the Diagnosis Scenario

Figure 4 depicts a CIGDec model of our patient-diagnosis scenario. It consists of five tasks. In an extreme case, it would be possible to make and use the model consisting only out of these tasks and without any constraints. This would be a unrestricted model allowing for maximum flexibility, where tasks could be executed an arbitrary number of times ('0..*') and in any order. This model would have an infinite number of execution possibilities (different process instances). However, to develop a model that provides guidance, we add five constraints derived from three constraint templates.

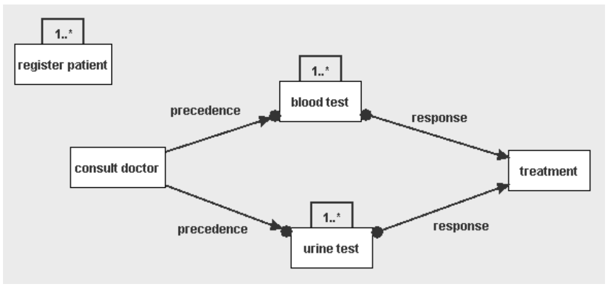


Fig. 4. CIGDec model for the diagnosis scenario

First, there is one unary (involving one task) constraint created from the template 'existence' - constraint presented as cardinality *1..** above the task *register patient*. This constraint specifies that the task *register patient* has to be executed at least once within one process (guideline) enactment.

Second, there are two constraints created from the template 'precedence' as shown in Figure 4: one between tasks *consult doctor* and *blood test* and one between tasks *consult doctor* and *urine test*. Precedence is a binary template, i.e., it defines a dependency between two tasks. A 'precedence' between two tasks *A* and *B* means that task *B* can only be executed after task *A* was executed at least once [6]. It is possible that other tasks are executed between *A* and *B*. Hence, if we want to execute task *blood test* we can do so only after we have executed task *consult doctor*. Note that other tasks from the model can be executed between *consult doctor* and *blood test*. Task *test urine* also has a 'precedence' relation with task *consult doctor* and it can be executed only after task *consult doctor*. Similarly, there could be other tasks between them. Moreover, the doctor may be consulted multiple times before and after doing the tests.

Third, we use a binary template 'response' to create two constraints: one between tasks *blood test* and *treatment* and one between tasks *urine test* and *treatment*. Template 'response' between tasks *A* and *B* defines that after every execution of task *A* task *B* has to be executed at least once while it is possible that other tasks are executed between *A* and *B*. Thus, after every *blood test* at least one *treatment* should follow, and there could be other tasks from the model executed between them. The same holds for tasks *urine test* and *treatment*.

The possibilities given to a user during execution of the model depicted in Figure 4 are defined as a combination of all five constraints in the model. When looking at the models designed by means of the analyzed language Asbru, the execution always had to start with the task *register patient*. This may cause problems in cases of emergency, when there is no time for the registration requiring the procedure with doctor (task *consult doctor*) to start immediately. While in EON and GLIF allow multiple entry-points to a scenario, these entrance steps have to be modelled explicitly. In PRO*forma* a task can be modelled without use of scheduling constraints which allows this task to be executed at any moment. Note however, that the CIG languages assume that a task can be executed once during the model execution or *iteratively* a specified number of times. In CIGDec model a patient-registration step can be performed at any moment during the CIGDec process. Furthermore, CIGDec model allows to perform *register patient* multiple times in case the required data is not available on time.

If we look at the traditional models Figures 1, 2 and 3 (i.e. mode using Absru and EON, GLIF and PRO*forma*), task *consult doctor* was executed exactly once. CIGDec model allows this task not to be executed at all, but it also allows it to be executed multiple times. For example, some patients use medication periodically. For them only the *treatment* task has to be performed either before or after the *register patient* has been executed. On the other hand, in some complex cases, task *consult doctor* can be performed more than once at various points during the CIGDec execution.

If necessary, a doctor can order a *blood test* many times or not at all during the CIGDec process. However, constraint 'precedence' between this task and *consult doctor* makes sure that *blood test* can not be done for a patient that has not seen the doctor before. Note that his holds only for the first *blood test*. Sometimes, the results can be unexpected and doctor can order a different type of *blood test* without having to see the patient again. After every *blood test*, task *treatment* is performed. It is possible that during *treatment* no medication is prescribed due to the good test results. However, it is also possible to wait and to perform several *blood tests* in order to make an informed decision before the task *treatment* is performed. Since task *urine test* has the same relationships as task *blood test* ('precedence' with *consult doctor* and 'response' with *treatment*), the same variants of execution paths hold like for the task *blood test*. However, note that none of the tasks 'blood test' and 'urine test' do not have to execute at all, or each of them can be executed one or more times, or only one of them can be executed one or more times.

CIGDec model from Figure 4 could be used to realize the following scenarios. First, in the 'case A' a periodical medication is prescribed to a chronic patient: only *register patient* and *treatment* tasks are executed. In the 'case B' an urgent visit starts directly with *consult doctor* and only afterwards the task *register patient* is executed. The *urine test* was not necessary. The results of the *blood test* were unclear so the *treatment* is executed only after the results of the second *blood test* became available and an additional *consult doctor* task. In the 'case C', the situation was not urgent, so task *register patient* was performed before the task *consult doctor*. Both *urine test* and *blood test* are performed. However, due to alarming results of the *urine test* an immediate *treatment* was executed to prescribe appropriate medication. The results of *blood test* arrived later, and an additional *treatment* task was executed to handle the *blood test* results as well.

4 Discussion

We have shown that CIGDec can be used to define the degree of flexibility given to a user during the process execution. We have also indicated that a degree of the absolute flexibility can be reached by leaving out all constraints resulting in the freedom given to a user to select any task and execute tasks in any desired order. Since the degree of flexibility has to be controlled in the context of medical care in order to adhere to strict and desirable recommendations, the mandatory and optional constraints have to specified for a modelled guideline. To control the adherence to the specified constraints, the execution engine CIGDec prohibits the violation of the mandatory constraints while allowing the optional constraints to be neglected. All user steps that might result in the violation of constraints are communicated to a user by means of warnings.

The advantages of the proposed CIGDec-based approach over the analyzed modelling languages that employ the imperative approach are as follows:

- CIGDec enables the *flexibility in selection*, meaning that a user executing a model specified in CIGDec gets a freedom in choosing an execution sequence, without requiring this sequence to be thought of in advance and explicitly modelled during the design-time.
- CIGDec enables *late binding*, meaning that it allows to choose an appropriate task at the point of care. This feature is particular important in modelling of CIG since it is not always possible to predict what steps will need to be executed, thus the task selection is case-dependent.
- CIGDec ensures the *absence of change*, meaning that it prohibits choices of users that would violate mandatory constraints.
- CIGDec allows for extendability and allows new LTL formulas to be introduced, thus applicability of CIGDec could be tailored to a specific situation.

The disadvantages of using CIGDec are as follows:

- If a process to be modelled has to be very strict and should allow for flexibility, then the use of CIGDec may result in a complex model.
- CIGDec aims at the modelling of rather small processes, since the description of large processes (containing approximately several thousands of tasks) becomes difficult to understand.

Since both imperative and declarative languages have disadvantages, in order to improve the flexibility of the CIG modelling languages we recommend to augment the CIG languages with the features offered by CIGDec.

5 Related Work

The recent *Workflow Patterns initiative* [2] has taken an empirical approach to identifying the most common control constructs inherent to modelling languages adopted by workflow systems. In particular, a broad survey of modelling languages resulted in 20 workflow patterns being identified [7]. In this paper we have used the revised set of the control-flow patterns [8] to evaluate CIS's modelling languages.

There had been many attempts to enrich the flexibility of workflow (process) management systems. Case-handling systems are systems that offer more flexility by focusing on the whole case (process instance), instead of individual tasks [20]. An example of such a system is FLOWer [21], where users can 'move up and down' the process by opening, sipping and re-doing tasks, rather than just executing tasks. Although users have a major influence on execution in FLOWer, their actions are seen as going backwards or forward in a traditional process model. Moreover, this might some unwanted side-effects. For example, if the user wishes to execute again (re-do) an earlier task, s(he) will also have to execute again (re-do) all tasks that followed it. Unlike in FLOWer, deviations are not seen as an exception in CIGDec but as 'normal' behavior while the process instance unfolds further according to the choices of users.

Flexibility of process enactment tools is greatly increased by their adaptivity. ADEPT is an example of an adaptive system where users can change the process model during the enactment [22]. ADEPT is a powerful tool which enables users to insert, move and delete tasks form the process instance they are currently working on. However, the user has to be a process modelling expert in order to change the model. Moreover, in medical domain cases may have many differences and adaptations would be too frequent and time consuming. CIGDec does not see deviations as changes in the model and a good-designed CIGDec model can cover a wide variety of cases.

One of a promising ways to introduce flexibility is to replace imperative by declarative. Various declarative languages "describe the dependency relationships between tasks, rather than procedurally describing sequences of action" [23]. Generally, declarative languages propose modeling constraints that drive the model enactment [23,24,25]. Constraints describe dependencies between model elements. Constraints are specified using pre and post conditions for target task [25], dependencies between states of tasks (enabled, active, ready, etc.) [23] or various model-related concepts [24].

6 Conclusions

In this paper, we have proposed a declarative approach which could be applied to overcome problems experienced by the imperative languages used for modelling clinical guidelines. In particular, we have shown how by means of applying the CIGDec language more flexibility in selection can be achieved than the considered CIG modelling languages offer. Furthermore, we showed how the model declared in CIGDec can be enacted. In addition, we discussed differences between the proposed declarative and analyzed imperative languages, their advantages and disadvantages, and made a proposition to combine the features of imperative and declarative approaches in order to increase their applicability and usability.

References

1. Peleg, M.: Chapter 4-2: Guideline and Workflow Models. In: Greenes, R.A. (ed.) Medical Decision Support: Computer-Based Approaches to Improving Healthcare Quality and Safety, Elsevier, Amsterdam (2006)
2. Workflow Patterns Home Page, http://www.workflowpatterns.com

3. Mulyar, N., Aalst, W., Peleg, M.: A Pattern-based Analysis of Clinical Computer-Interpretable Guideline Modelling Languages. Technical report, Center Report BPM-06-29, BPMcenter.org (2006)
4. Wald, J., Pedraza, L., Murphy, M., Kuperman, G.: Requirements development for a patient computing system. In: Proc. AMIA Symp., pp. 731–735 (2001)
5. Pesic, M., Aalst, W.: A declarative approach for flexible business processes management. In: Business Process Management Workshops, pp. 169–180 (2006)
6. Aalst, W., Pesic, M.: Specifying, discovering, and monitoring service flows: Making web services process-aware. BPM Center Report BPM-06-09, BPM Center, BPMcenter.org (2006)
7. Aalst, W., Hofstede, A., Kiepuszewski, B., Barros, A.: Workflow Patterns. Distributed and Parallel Databases 14(1), 5–51 (2003)
8. Russell, N., Hofstede, A., Aalst, W., Mulyar., N.: Workflow Control-Flow Patterns: A Revised View. BPM Center Report BPM-06-22, BPMcenter.org (2006)
9. Shahar, Y., Miksch, S., Johnson, P.: The Asgaard Project: a task-specific Framework for the application and critiquing of time-oriented clinical guidelines. Artif. Intell. Med. 14, 29–51 (1998)
10. Seyfang, A., Kosara, R., Miksch, S.: Asbru's Reference Manual, V.7.3. Technical report, Vienna Univ. of Techn., Inst. of SW Techn., Vienna. Report No.: Asgaard-TR-2002-1 (2002)
11. Tu, S., Musen, M.: A flexible approach to guideline modeling. In: Proc AMIA Symp., pp. 420–424 (1999)
12. Tu, S., Musen, M.: From guideline Modeling to guideline execution: Defining guideline-based decision-support Services. In: Proc. AMIA Annu. Symp., pp. 863–867 (2000)
13. Tu, S.: The eon guideline model (2006), http://smi.stanford.edu/projects/eon
14. Boxwala, A., et al.: GLIF3: A representation format for sharable computer-interpretable clinical practive guidelines. Biomedical Informatics 37(3), 147–161 (2004)
15. Fox, J., Johns, N., Rahmanzadeh, A.: Disseminating Medical Knowledge: The PROforma Approach. Artificial Intelligence in Medicine 14(1), 157–182 (1998)
16. Clarke Jr., E.M., Grumberg, O., Peled, D.: Model Checking. The MIT Press, Cambridge (1999)
17. Holzmann, G.: The SPIN Model Checker: Primer and Reference Manual. Addison-Wesley, Boston, Massachusetts (2003)
18. Giannakopoulou, D., Havelund, K.: Automata-based verification of temporal properties on running programs. In: ASE 2001. Proc. of the 16th IEEE int. conf. on Automated sw engineering, vol. 412, IEEE Computer Society, Washington, DC, USA (2001)
19. Mulyar, N., Pesic, M., Aalst, W., Peleg, M.: Towards the Flexibility in Clinical Guideline Modelling Languages. Technical report, Center Report BPM, BPMcenter.org (2007)
20. Reijers, H., Rigter, J., Aalst, W.: The Case Handling Case. International Journal of Cooperative Information Systems 12(3), 365–391 (2003)
21. Athena, P.: Flower User Manual. Pallas Athena BV, Apeldoorn, The Netherlands (2002)
22. Reichert, M., Dadam, P.: ADEPTflex: Supporting Dynamic Changes of Workflow without Losing Control. Journal of Intelligent Information Systems 10(2), 93–129 (1998)
23. Dourish, P., Holmes, J., MacLean, A., Marqvardsen, P., Zbyslaw, A.: Freeflow: mediating between representation and action in workflow systems. In: Proc. of CSCW 1996, pp. 190–198 (1996)
24. Mangan, P., Sadiq, S.: On building workflow models for flexible processes. In: ADC 2002. Proc. of the 13th Australasian database conf. pp. 103–109. Australian Computer Society, Inc. (2002)
25. Wainer, J., de Lima Bezerra, F.: Groupware: Design, Implementation, and Use. In: Favela, J., Decouchant, D. (eds.) CRIWG 2003. LNCS, vol. 2806, pp. 151–158. Springer, Heidelberg (2003)

Managing Socio-technical Interactions in Healthcare Systems

Osama El-Hassan, José Luiz Fiadeiro, and Reiko Heckel

Dept. of Computer Science, University of Leicester, University Road
Leicester, LE1 7RH, United Kingdom
{oehel, jose, reiko}@mcs.le.ac.uk

Abstract. We put forward an architectural framework that promotes the externalisation of the social dimension that arises in software-intensive systems which, like in healthcare, exhibit interactions between humans (social components) and technical components (devices, computer-based systems, and so on) that are critical for the domain in which they operate. Our framework is based on a new class of architectural connectors (social laws) that provide mechanisms through which the biddability of human interactions can be taken into account and the sub-ideal situations that result from the violation of organisational norms can be modelled and acted upon by reconfiguring the socio-technical systems. Our approach is based on formal, algebraic graph-based representations and transformations.

1 Introduction

One of the main sources of difficulties in healthcare oriented process management is the fact that staff may deviate from prescribed medical or organisational workflows. Such deviations are not "errors" but, rather, result from the fact that situations may arise in which humans may need to interact with machines in "sub-ideal" states [17]:

> *"Medical personnel must be free to react and is trained to do so. Therefore, in addition to the customization of a pathway schema at instance creation time, it must be possible to dynamically adapt in-progress treatment cases (i.e., process instances) during runtime. […] As a consequence the actual patient treatment process may deviate from the original treatment plan and the related medical pathway respectively. However, such deviations from the pre-planned process must not lead to errors or inconsistencies".*

Freedom to (re)act matches perfectly the characteristics of *biddable domains* as identified by Jackson [16]: "*[people] can be joined to adhere to certain behaviour, but may or may not obey the injunction*". The problem, as highlighted in the quote above, is to endow systems with the degree of flexibility that allows them to adapt dynamically to changes from ideal to sub-ideal states.

In this paper, we explore the use of architectural modelling techniques for making systems adaptable through dynamic reconfiguration [2]. Such techniques rely on the externalisation of the coordination of the interactions among components of the system in explicit connectors [1]. One can then reconfigure the system by changing the connectors without changing the components themselves. However, in order to be able to address interactions in which some of the components are human, as is the case in healthcare, we need new modelling primitives that extend the notion of connector. For instance, we should be able to model the interactions that, in a healthcare system, are associated with the roles performed by medical staff when engaging with fellow staff, software applications or machines.

Our approach addresses what are sometimes called "socio-technical systems": systems that include a "social dimension" in the sense that people (or groups of people) need to be considered not as external users but as another class of components that, together with software and devices, perform roles that are vital for the "good" behaviour of the system. In order to bring human interaction within the boundaries of systems, we need to be able to refer to normative concepts like permissions, obligations and power, and to model the violations that can take place so that processes and underlying software can be reconfigured to react to non-normative situations in ways that ensure agreed, possibly minimal, levels of service. In this paper, we are particularly interested in capturing sub-ideal situations that are caused by violating an obligation or a permission to perform a certain action or activity.

From this discussion, it should be obvious that our approach differs from the perspective on human interaction taken in HCI (Human-Computer Interactions), which addresses user interface issues, or CSCW (Computer Supported Cooperative Work) [], which tackles person-to-person computer-mediated collaborations. Our focus is not on *user* interaction but on peer-to-peer interactions. Our starting point is a combination of recent work on software architecture [2] and the knowledge that has been accumulated in multidisciplinary areas of research such as deontic logic [22], multi-agent systems [10], role-based access control [21] and other social studies. In [8], we made a preliminary proposal for a new class of architectural primitives and formal configuration modelling techniques based on [24]. These configuration primitives allow us to specify when and how to reconfigure the software architecture, at runtime, in a way that reflects the normative positions of humans as engaged in interactions within organisational settings. In this paper, we provide an overview of this work as applied to healthcare and develop in more detail the semantics of the reconfiguration operations.

Section 2 constitutes a brief discussion of the architectural modelling approach and addresses the new primitives, through which biddable interactions can be coordinated, then after. Section 3 provides a mathematical semantics over graph-based representations and transformations followed by an example.

2 The Architectural Approach

2.1 Architectural Primitives for Causal Interactions

Our architectural framework is based on the CCC architectural approach [2, 3], which includes a business micro-architecture to support engineers and system specifiers to model and implement evolving component-base systems. Architectural primitives like coordination contracts [2] model rules that determine how and when components need to interact in order to fulfil business requirements. The CCC can be classified as a coordination based approach [13] that borrows essential ideas from software architecture [19] in order to externalise interactions from computations, and superimposition as known from parallel program design [12] to support compositional evolution.

Indeed, in software architecture, modelling techniques have been proposed for supporting interaction-centric approaches. More precisely, such techniques promote interconnections to first-class citizens (architectural connectors) by separating the code that, in traditional approaches, is included in the components for handling the way they interact with the rest of the system, from the code that is responsible for the computations that are responsible for the services offered by the components.

The particular architectural approach that is adopted by the CCC builds on event-condition-action (ECA) rules for coordinating the joint behaviour that a group of components need to execute in reaction to a trigger generated by another component or outside the system. A so-called coordination law defines how a number of partners interact. The partners are not named: they are abstracted as coordination interfaces that define types of system entities in terms of operations that instance entities need to make available and events that need to be observed. As an example, consider the coordination of the way a doctor interacts with a respiratory-control system:

```
coordination interface respiratory-control
   partner type DEVICE
   types a:pressure, d:DOCTOR
   operations
       in-charge(d):Boolean
       verify():pressure
       decrease(a): post verify() = old verify()-a
       increase(a): post verify() = old verify()+a

coordination interface doctor-in-charge
   partner type DOCTOR
   types a: pressure
   events plus(a), minus(a)

coordination law restricted-respiratory
   partners d: doctor-in-charge, r: respiratory-control
   types a:pressure
   attributes  min,max:pressure
   rules
       when d.minus(a)
           with r.verify-a≥min and r.in-charge(d)
           do r.decrease(a)
       when d.plus(a)
           with r.verify+a•max and r.in-charge(d)
           do r.increase(a)
```

Each rule of the coordination law identifies, under the "when" clause, a trigger to which the instances of the law will react – e.g. a request by a doctor for an increase or decrease of the pressure. The trigger can be just an event observed directly over one of the partners or a more complex condition built from one or more events. Under the "with" clause, we include conditions (guards) that should be observed for the reaction to be performed: that the changes in the pressure keep it within the specified bounds and that the doctor has been authorised to be in charge of the device. If any of the conditions fails, the reaction is not performed and the occurrence of the trigger fails. Explicit mechanisms can be defined for handling such failures.

There is neither a provision in the CCC approach, nor in any other architectural approach that we know, to model interactions that are only biddable, i.e. situations in which people (social components) are requested to perform given operations but the system cannot cause (force) them to perform these operations. For instance, biddable interaction would occur if the doctor would be requested to alter the current settings. In summary, one needs a richer model of interaction that can capture the fact that coordination in the presence of social components cannot be causal.

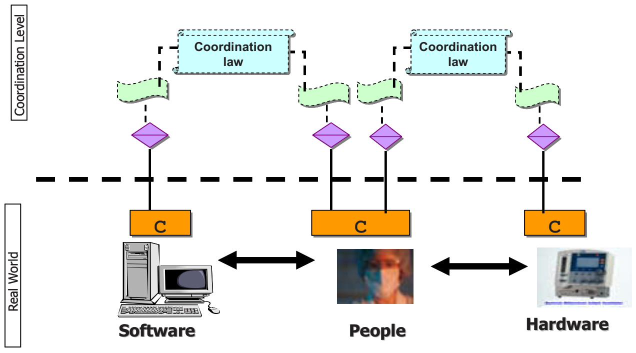


Fig. 1. A CCC-based configuration

2.2 Architectural Primitives for Biddable Interactions

The social layer that is required for healthcare requires integrating a new collection of architectural primitives that support the modelling of human interactions in ways that are flexible and easily amenable to change. More precisely, for the kind of "just-in-time" binding and reconfigurations required for modelling biddability of human participants, we put forward another class of connector types (social laws) defined over a set of social roles, each of which represents the abilities of a social entity within the organisation.

2.2.1 Social Roles

Roles are abstract constructs that specify the behaviour expected of social components by means of operations and ascribed normative aspects that refer to certain institutionalised positions or capabilities. More concretely, we distinguish between having the ability to perform an operation and having the qualification or authorisation to do so: a social component may have the ability to perform an operation and still trigger a role violation if it is not an instance of a role that has the right qualification. Here we use the word qualification to mean, for instance, that the organisation has empowered the social component to perform given operations.

As discussed below, the execution of operations by a component when playing a role without the required qualification is governed by a social law. A social law specifies (social) rules that either impose sanctions or provide a configuration in which the operation can be safely executed, depending on the context in which the violation takes place. However, we need to stress that the execution of operations, even if by qualified components, can be governed by coordination laws and, as such, be refused in certain circumstances for operational reasons, not for deontic ones.

We denote with [+] the operations for which the role is qualified. We can also define a subsumption relation between operations: by declaring $op_1 \supset op_2$ we mean that op_1 can only be executed as part of op_2, in which case a component qualified to do op_2 is also qualified to do op_1.

The general structure of a social role is as follows:

```
social role rolename {specialises rolename}
types {{par}+:datatype}*
operations {
   {'[+]'} opname {⊃ opname}
}*
```

For instance, GPs (General Practitioners) are qualified to perform routine tasks of seeing patients and registering for shifts in wards. A GP can also perform minor operations but will trigger a role violation unless he/she is an instance of a role that is qualified to do so.

We introduce the notion of *task* as an unordered set of events involved in the same activity. Among them, one or two events, corresponding to entry and exit operations, are considered as Behavioural Implicit Communications that need to be managed by the normative reconfiguration view (as captured by social laws) instead of causal coordination context. Our roles and coordination interfaces fit well together in the sense that they capture complementary aspects of human components: organisational and capability-related.

The overall importance of distinguishing between ability and qualification to perform an operation is that it reduces what are normally called normative positions [22] to deontically-governed role transitions. This is because, in the context of an architectural approach to system development, it is easy to model role transition in terms of dynamic reconfiguration. In the absence of such an explicit hierarchy, sub-ideal situations would have to be resolved just by means of sanctions. Instead, we take a more positive and active approach by enabling a reconfiguration if the current context allows such a deviation of the norm to be tolerated. This is precisely the goal of social laws as discussed below.

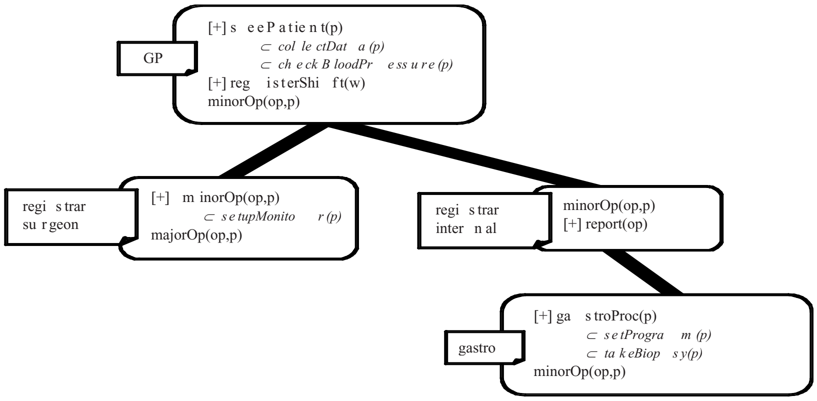


Fig. 2. A role hierarchy

2.2.2 Social Laws

A social law is an architectural primitive through which a software architect models the circumstances in which a system is considered to be in a sub-ideal situation and how the system should respond to such a situation so as to preserve the overall integrity of the system. A sub-ideal situation occurs either as a result of a human deviating from a prescribed routine or by detecting a trigger that refers to a predetermined contextual change in the environment. A human may deviate from his prescribed behaviour either by acquiring a non-granted permission or by not obeying an imposed obligation that urges him/her to execute an action provided that appropriate architectural facilities are granted.

A social law captures interactions of a partner declared as an anchor role. Besides the anchor role, a social law identifies other partners through either social roles or coordination interfaces. The former are useful for reconfiguration operations and the latter for both detecting triggers and reconfigurations as explained below.

```
social law name
anchor role social role
partners {social role, coordination interface}*
types {{par}+:datatype}*
{violation rule
   when trigger
     if condition
     reconfiguration task
     sanction {operation}*
}*
```

In addition, a social law comprises one or more violation rules. We distinguish three kinds of triggers for violation rules: (1) operations of the anchor role that are executed by social components that have no qualification; (2) operations for which the anchor role is qualified but they are initiated in a context in which they are not permitted; (3) operations of the anchor role that are not executed in contexts in which they are required.

The first takes the form:

```
unqualified operation
```

The second takes the form:

```
operation and not enabling state
```

The third are of the form:

```
active state and not operation
```

Notice that, in order to detect the violation of the enabling state (permission), we need a coordination interface that provides an operation that returns a Boolean value and, in order to detect the violation of the obligation, we need a coordination interface that provides an event. The "negated operation" holds in the states in which the operation has not been scheduled for execution by the component that instantiates the anchor role.

2.2.3 A Motivating Example

This example was extracted from a survey undertaken at a Gastroenterology department[1] in UAE. By looking into their documentation we elicited a group of norms that affect the behaviour of doctors: (1) No operation can be undertaken without the patient's permission; (2) Surgical intervention should be carried by Surgeons[2] only; (3) In the case of emergency, a doctor may commit simple surgical interventions if surgeons are not available and it is a life-saving context.

As an example, consider the social law that applies to minor operations. Such procedures involve a social role – a GP – who is the anchor role in the sense that the social laws will apply to the actions performed by instances of this role, e.g. a gastroenterologist. In addition, three coordination interfaces are required to ensure that the GP interacts with the right components: the device that is monitoring the procedure – *monitor-procedure*, and the software component that provides access to administrative data – *administrator*. In the configuration of the system, there will be coordination laws modelling the way these components interact. Because of lack of space, we are not able to provide the definition of the relevant coordination interfaces and laws.

```
social law minor-operation
anchor role d:GP
type p:patient, op:operation
partners a:administrator, m:monitor-procedure
violation rule
   when d.minorOp(op,p) and not a.ensureConsent(op,d,p)
     if m.alarm(p)
     reconfiguration reconfMinor(d,op)
     sanction a.record(d,op,"no_consent")
   when unqualified d.minorOp(op,p)
     if m.alarm(p)
     reconfiguration reconfUnqual(d,op,p)
     sanction a.record(d,op,"unqualified")
```

The social law has two rules triggered by the same event: the moment in which a doctor initiates the operation on the patient. The first rule handles the situation in which there is no record of consent having been given by the patient for the doctor to perform that operation. If the monitor detects that there is an emergency situation, then a reconfiguration of the context is performed to put in place the components and coordination contracts that are required for the operation to proceed. This may involve, for instance, providing access to further information registered on the patient's file, say on allergies. However, if the monitor does not detect an emergency, sanctions apply by recording the violation in the doctor's file.

The second rule is activated if the actual doctor is not qualified to perform minor operations, which is possible because the doctor's role matches one of the roles in the non-surgical branch of the doctor's role hierarchy: GP, registrar internal or gastro. In this case, we have to distinguish again if there is an emergency. For simplicity, we used the same alarm condition provided by the monitor. If an emergency is indeed detected, a reconfiguration of the context is performed to allow the doctor to proceed,

[1] A specialized medical unit in Rashid Hospital, Dubai, U.A.E., that provides treatment for digestive diseases. The unit consists of an Endoscopies suite containing two modern fully equipped endoscopies rooms with ancillary supporting facilities for patient's reception, preparation and post-endoscopies recovery rooms.

[2] A reader should refer to Figure 2 for roles, tasks and permissions definitions.

for instance unblocking actions that, in normative states, should be forbidden to the doctor. Otherwise, sanctions apply. Notice that the reconfiguration operation takes the doctor as a parameter: the hospital may have different rules about the context that should be present during an operation depending on the type of doctor.

Notice that both rules can apply – the doctor may not be qualified and the patient may not have given consent. In the case of an emergency, both reconfigurations apply; otherwise, both sanctions are implemented. The subsequent section highlights reconfiguration operations and the formalism used to reason about them.

3 Modeling Reconfiguration Operations

In order to reason about system reconfigurations at a higher level of abstraction we require a representation of the system's configuration, its graph of components and connections at runtime, as well as of the rules governing their evolution. Graphs and graph transformation [15] provide a visually compelling yet mathematically rigorous formal technique that addresses our needs.

3.1 Modeling Configuration Graphs

Configuration graphs consist of nodes that refer to the system's social, mechanical and technological entities, and edges representing connectors that are superimposed on these entities to coordinate their interactions. More precisely, a configuration graph is a labelled graph where nodes are components labelled with instantiated interfaces and edges are connectors (contracts) labelled with the corresponding law type.

Valid system configurations (instances) are such that their graphs conform to constraints defined by type graphs – "filters" that restrict the allowed types of the instantiated nodes as well as types and cardinality of edges that connect them to populate an architecture instance.

We developed a specific graph typing structure to distinguish between configuration entities (nodes) whose corresponding permissions are fixed and some other entities that hold permissions amenable to change at runtime, e.g. social entities. This typing structure yields a twofold representation of the configuration graph that comprises a *components configuration graph* [20] and a *role configuration graph* that captures instances of roles, tasks and entry actions of social entities. More concretely, the *components graph* is sufficient to reflect casual properties of software, mechanical components and their connections but it falls short of providing a suitable representation of human components that are biddable and subject to organisational norms that can be violated. Conversely, the *role graph* includes a biddable dimension and an organisational dimension; the former addresses the biddable nature of human components, which requires non-causal modelling primitives; the latter is constructed for modelling human capabilities and permissions within an organisation. The bridge between the two-configuration graphs consists of the common human nodes and the edge between the targeted task and its associated coordination interface copy that defines the signature of operations and services included in this task.

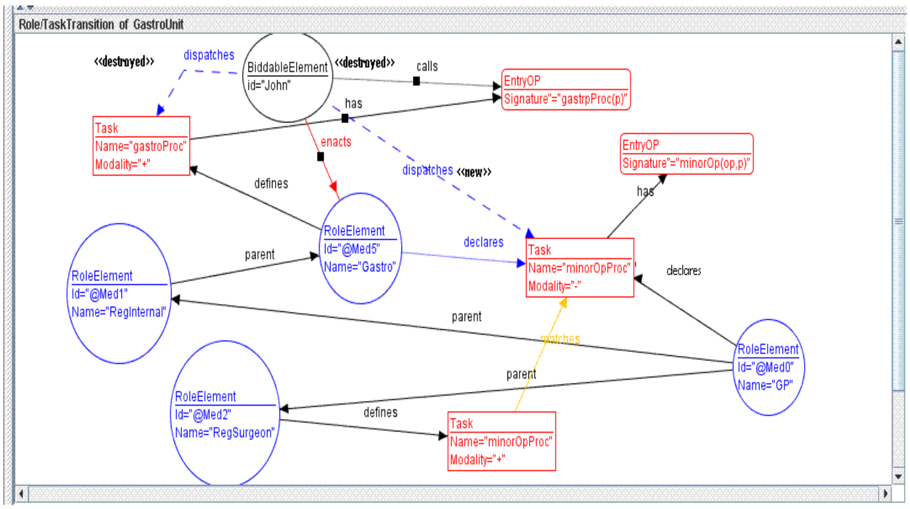


Fig. 3. The configuration Graph

Figure 3 depicts a Configuration graph as a snapshot of the system configuration that preserves the constraints of the type graph: a biddable entity (John) calling an entry operation that is beyond his role's permissions.

3.2 Modeling Reconfiguration Rules Using Graph Transformations

Formally, a typed attributed graph transformation rule r: L→R consists of a pair of instance configuration graphs that conform to a type graph TG, and whose intersection is well-defined (this implies that edges that appear in both L and R are connected to the same vertices in both graphs, that vertices with the same name have to have the same type, and so on). The left-hand side L represents the pre-conditions of the rule while the right-hand side R describes the post-conditions. The AGG[3] tool has been used to demonstrate configurations and reconfigurations as graphs and transformation rules.

We devised two types of reconfigurations: *local reconfigurations* operations that target human (biddable) components by writing his/her permissions or obligations; *global reconfigurations* make a full transition to a new workflow entry (an initial task). For example, *reconfUnqual(role/subject, operation, target)* performs a reconfiguration operation on a human role/task space and its bound permissions, whereas *reconfGloabl(role, Process, entry_operation)* manipulate both software and hardware entities configuration to allow the human participant to perform a new set of operations to achieve new goals as a response of new context settings.

Both reconfiguration operations provide operational semantics for social laws by allowing humans, participating in an instance configuration, to deviate from the prescribed process pattern, e.g. *medical pathways*, at the execution time and/or performing a global reconfiguration: simply model the transition from the current configuration (process) to another.

[3] The Attributed Graph Grammar System: http://tfs.cs.tu-berlin.de/agg/

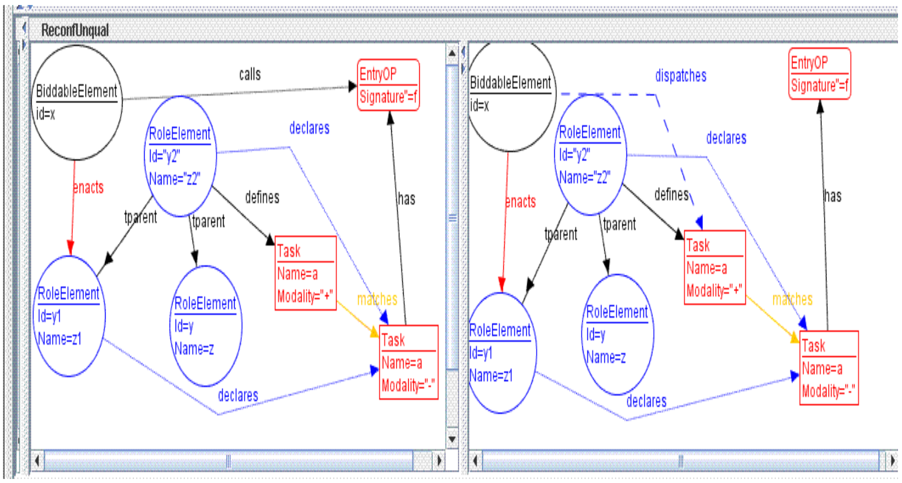


Fig. 4. *ReconfUnequal* Transformation rule

For simplicity and due to space limitations we provide a simple transformation rule (Figure 4) that illustrates a generic and role-based reconfiguration operation – *reconfUnqual* – in which a GP is allowed to perform a minor surgery – *tracheostomy* – in case of emergency, provided that the preconditions are tested by the corresponding social law body when the entry operation is fired. Labels <<*destroyed*>> and <<*new*>> in Figure 3 designate deleted and created nodes/edges imposed by the corresponding transformation rule.

The formalisation provided through graph transformations supports reasoning about reconfiguration steps, brings together computation and coordination with process management in a single model and facilitates the derivation of overall system behaviour by exploiting the notion on operational semantics c.f. GOS [9].

4 Concluding Remarks

The main contribution of our research has been to leverage modelling primitives developed for software architectures to cater for interactions that involve human components. Our approach takes into account the biddable, non-causal nature of human actions, and provides a mechanism for adjusting permissions and obligations on interactions between technical and human components so as to react and adapt to changes in the environment in which they operate. Similarly, adaptation may result from monitoring services [4] and imply a reconfiguration of roles and interactions. Our approach follows a normative system perspective and includes: social laws that express reactions to non-normative situations that may arise from violation of permissions or obligations; social roles that capture humans capabilities within organisations; operations that, within social laws, can reconfigure interactions and/or reassign roles to human components.

In the future, we would like to explore the synergies between our architectural approach and recent research on workflow-based systems, which have been shifting from Workflow Control Patterns and Workflow Data Patterns to that Workflow Resource Pattern [20]. The latter come in line with our focus on modelling resources (human/non human) and their interactions. Recent research on workflow has also shed light on the need for runtime changes – *momentary changes* in [23]. Additionally, Lenz et al. [17] stated that there is a price to be paid for isolating control flow from application logic. They put forward a workflow engine to accommodate ad-hoc changes at different levels of abstractions. Although they distinguish between stable organizational processes and continuously changing clinical treatment processes, they are not able to model how knowledge about medical staff may affect such changes.

References

1. Allen, R., Garlan, D.: A Formal Basis for Architectural Connectors. ACM TOSEM 6(3), 213–249 (1997)
2. Andrade, L.F., Fiadeiro, J.L.: Architecture Based Evolution of Software Systems. In: Bernardo, M., Inverardi, P. (eds.) SFM 2003. LNCS, vol. 2804, pp. 148–181. Springer, Heidelberg (2003)
3. Andrade, L.F., Fiadeiro, J.L., Wermelinger, M.: Enforcing Business Policies through Automated Reconfiguration. In: Proc. of the 16th Intl. Conf. on Automated Software Engineering, pp. 426–429. IEEE Computer Society Press, Los Alamitos (2001)
4. Baresi, L., Ghezzi, C., Guina, S.: Smart Monitoring for Composed Service. In: Proc. of the 2nd International Conference on Service Oriented Computing, pp. 15–19 (2004)
5. Brier, B., Rapanotti, L., Hall, J.: Problem Frame for Socio-Technical Systems: predictability and change. In: WAAPF 2004. Proc. of 1st International Workshop on Advances and Applications of Problem Frames, Edinburgh, Scotland (2004)
6. Castelfranchi, C., Giardini, F.: Silent Agents. Behavioural Implicit Communication for M-A Coordination and HMI. In: Proc. of the 2nd Annual Symposium on Autonomous Intelligent Networks and Systems, Menlo Park, CA (2003)
7. Colman, A. W., Han, J.: Organisational Roles and Players. In: Proc. of AAAI 2005 Fall Symposium on Roles, An Interdisciplinary Perspective, Arlington, VA, pp. 55–62 (2005)
8. El-Hassan, O., Fiadeiro, J.L.: Role-based Architectural Modelling of Socio-Technical Systems. In: CoOrg 2006. Proc. of the 2nd International Workshop on Coordination and Organisation, ENTCS, vol. 181, pp. 5–17 (2007)
9. Engels, G., et al.: Dynamic Meta Modelling: A Graphical Approach to the Operational Semantics of Behavioural Diagrams in UML. In: Evans, A., Kent, S., Selic, B. (eds.) UML 2000. LNCS, vol. 1939, pp. 323–337. Springer, Heidelberg (2000)
10. Falcone, R., Castelfranchi, C.: Level of Delegation and Levels of Adoption as the basis for Adjustable Autonomy. In: Lamma, E., Mello, P. (eds.) AI*IA 99:Advances in Artificial Intelligence. LNCS (LNAI), vol. 1792, pp. 273–284. Springer, Heidelberg (2000)
11. Fiadeiro, J.L.: Designing for Software's Social Complexity. Computer 40(1), 34–39 (2007)
12. Fiadeiro, J.L., Maibaum, T.: Categorical Semantics of Parallel Program Design. Science of Computer Programming 28, 111–138 (1997)
13. Gelernter, D., Carriero, N.: Coordination Languages and their Significance. CACM 35(2), 97–107 (1992)
14. Grudin, J.: CSCW – History and Focus. IEEE Computer 27(5), 19–26 (1994)

15. Heckel, R.: Graph Transformation in a Nutshell. In: Salwicki, A. (ed.) FoVMT 2004. Proc. of the School of SegraVis Research Training Network on Foundations of Visual Modelling Techniques, LNCS, vol. 148(1), pp. 187–198 (1983)
16. Jackson, M.: Software Requirements and Specifications: A Lexicon of Practice, Principles and Prejudices. Addison Wesley, Reading (1995)
17. Lenz, R., Reichert, M.: IT Support for Healthcare processes: premises, challenges, perspectives. Data & Knowledge Engineering 61(1), 39–58 (2007)
18. Padmanabhan, V., Governatori, G., Sadiq, S., Colomb, R., Rotolo, A.: Process Modelling: The Deontic way. In: Stumptner, M., Hartmann, S., Kiyoki, Y. (eds.) APCCM 2006. The 3rd Asia Pacific Conference on Conceptual Modelling (2005)
19. Perry, D.E., Wolf, A.L.: Foundation for the Study of Software Architectures. ACM SIGSOFT Soft. Engineering Notes 17(4), 40–52 (1992)
20. Russel, N.: Workflow Resource Patterns Identification, Representation and Tool Support. In: Pastor, Ó., Falcão e Cunha, J. (eds.) CAiSE 2005. LNCS, vol. 3520, pp. 216–232. Springer, Heidelberg (2005)
21. Sandhu, R.S., Coyne, E.J., Feinstein, H.L., Youman, C.E.: Role-Based Access Control Models. IEEE Computer 29, 38–48 (1996)
22. Sergot, M.: Normative Positions. In: Prakken, H., McNamara, P. (eds.) Norms, Logics and Information systems: new studies in Deontic Logic, pp. 289–310. IOS Press, Amsterdam (1998)
23. van der Aalst, W.M.P., Jablonski, D.: Dealing with Workflow Change: identification of issues and solutions. Comput Syst Sci & Eng. 15(5), 267–276 (2000)
24. Wermelinger, M., Lopes, A., Fiadeiro, J.L.: A Graph based Architectural Re-configuration Language. In: Software Engineering- ESEC/FSE 2001, pp. 21–32. ACM Press, New York (2001)

Adaptive Workflows for Healthcare Information Systems

Kees van Hee, Helen Schonenberg, Alexander Serebrenik, Natalia Sidorova, and Jan Martijn van der Werf

Department of Mathematics and Computer Science
Eindhoven University of Technology
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
{k.m.v.hee, m.h.schonenberg, a.serebrenik, n.sidorova, j.m.e.m.v.d.werf}@tue.nl

Abstract. Current challenges in Healthcare Information Systems (HIS) include supplying patients with personalized medical information, creating means for efficient information flow between different healthcare providers in order to lower risks of medical errors and increase the quality of care. To address these challenges, the information about patient-related processes, such as currently executed medical protocols, should be made available for medical staff and patients. Existing HIS are mostly data-centered, and therefore cannot provide an adequate solution. To give processes a prominent role in HIS, we apply the adaptive workflow nets framework. This framework allows both healthcare providers and patients to get an insight into the past and current processes, but also foresee possible future developments. It also ensures quality and timing of data communication essential for efficient information flow.

Keywords: adaptive workflows, EPR, medical protocols, Healthcare Information Systems.

1 Introduction

The recent study of the Netherlands Health Care Inspectorate [23] has established a number of serious shortcomings in the communication between healthcare providers that can cause risks for patient safety. The typical examples named in the study are a lack of communication between the anesthesiologist and the surgeon involved in the same surgery; repetitive overwritings of variable data, such as blood pressure, with no information when, in which circumstances and by whom the measurements were performed; the use of "no message — good message" principle in the communication between specialists; providing insufficient information for patients about their treatments and expected developments. Another tendency reported is the steady increase of the total amount of patient-related information available, which, on one hand, increases the survival rate, but on the other hand, complicates the overall picture due to the chaotic nature of the information. The chance that the surgeon is aware of all information relevant to the surgery being performed is estimated as very low. The problems mentioned indicate a great need in a new generation of Health Information Systems (HIS) that would satisfy new information and communication demands.

At the same time, the patient should become a focal point of the new generation HIS. On April 30 of 2004, the European Commission adopted an action plan [7] aiming at making healthcare better for European citizens. As opposed to currently available provider-centered health systems, the action plan envisions citizen-centered health systems. A survey of Harris Interactive and ARiA Marketing conducted in 2000 [15] shows that more than 80% of the respondents are interested in obtaining on-line personalized medical information and electronic alerts specific to their medical histories, while 69% would like to have access to the charts that monitor the progress. Therefore, in this paper we aim at the creating a new concept of HIS that would increase the availability of personalized information in HIS both for care providers and for patients.

Currently employed HIS concentrate often on patient-related *data* rather than consider treatments as *processes* producing these data. By getting access to the process information, a patient can obtain a clear personalized picture of ongoing treatments, expectations and risks. Therefore, (s)he is more likely to become an informed decision-maker [8]—note that "participatory" decision-making model is recommended as the preferred model of treatment decision-making [4]. Moreover, shifting the focus of healthcare information systems from data to processes provides a practitioner with information on continuation of a treatment initiated by her and continued elsewhere. That is another reason why we advocate the move from data to processes.

To address the issues raised above, we propose a new approach to HIS. The key idea consists in associating a (number of) process(es) to each patient. By logging into the private web area, the patient gets access to the information related to processes associated to her. This information should include (a view on) data produced by previously performed treatments and a number of likely scenarios for the treatment continuation. Using the same information, a care provider can decide to alter an ongoing process, to abort such a process or to initiate a new one. New processes can either be suggested by a physician, or, more probably, they can be borrowed from a library of protocols, including e.g., hospital policies and medical guidelines.

Our approach relies on the framework of adaptive workflow nets [10,12]. Main advantages of the framework include adaptivity, adaptability and separation of concerns. By *adaptivity* we understand the ability of a process to modify itself as opposed to *adaptability*, which is the ability of a process to be modified by an external party. By *separation of concerns* we understand that every (sub)process has its owner, and the owner is the only actor responsible for performing the steps of the process. Still, the processes are interrelated and communicating.

In addition, we extend the currently used concept of the electronic patient record (EPR) with the notion of history. In other words, rather than overwriting the blood pressure in a database cell, we record the measurement every time it has been taken, by whom (the owner of the process), when and why (note that this information can be automatically generated since we know to which process this action belongs). This history is then used in the decision making process. It should be noted that the availability of history also enables the application of data and process mining techniques [2,17], which can lead to improving the quality of medical care.

2 Motivating Example

As a motivating example we consider the story of Saskia, a thirty-six years old Dutch preschool teacher, pregnant with her first child in the fifth week gestation. Saskia occasionally uses her home computer for chatting with her friends or looking up information on the Internet. During her pregnancy Saskia is assisted by a number of healthcare professionals: midwives, lab assistants, doctors, having (partial) access to Saskia's (electronic) patient record. Below we present the first steps of Saskia's pregnancy in the current situation and discuss possible improvements due to the implementation of our approach.

Currently. Saskia looks up the information on midwifes' offices in an on-line telephone directory. After consulting the web-sites of the offices, she selects one, and calls the office to arrange an appointment with a midwife. During the first visit, the midwife records Saskia's medical, social and gynaecological history and orders a number of standard lab tests: blood type, rhesus, iron deficiency, glucose level and urine. Moreover, since Saskia is thirty-six years old, the midwife briefly informs her on possible age-related risks for the baby and additional tests that can be performed. Upon receiving this information Saskia gets overanxious: she has no time to reflect on the matter and she is not able to decide whether she is willing to take these tests.

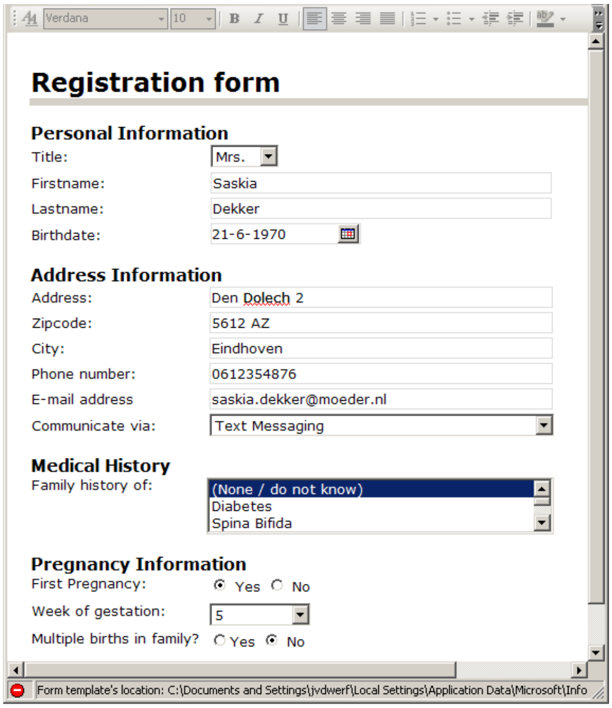
Getting home Saskia talks to her husband and looks for additional information available on-line. Based on this information she decides to undergo a nuchal translucency scan (NT screening). Saskia calls the midwifes' office to arrange another appointment at which she would order the test.

Blood tests show that Saskia is Rh(D) negative, while her husband is Rh(D) positive. Therefore, at twenty eight weeks gestation she gets an anti Rh(D) IgG immunoglobulin injection, which will be repeated after the delivery.

Desired. After selecting the office, Saskia fills an on-line form with her personal data. Being prompted what means of communication (e-mail, text messages, regular mail, phone call) does she prefer, Saskia chooses text messages. Indeed, Saskia is very excited about her pregnancy and wants to get all the information as soon as possible whether she is at home, at her office or out with friends. However, she is a working woman and not every moment might be appropriate for getting the information. Figure 1 presents the completed registration form.

Saskia receives a text message informing her which standard tests she should undergo. She also gets a link to a *personalized web-page* providing her with information on age-related risks and additional tests. Saskia takes her time to study the web-page together with her husband and decides to undergo an NT screening. She indicates this choice on her personalized web-page. A number of test times is proposed to her taking into account the scanner availability and that NT screening is performed between the eleventh and the fourteenth weeks gestation. The time selected and additional information are sent to the laboratory working with the midwifes' office.

Moreover, Saskia gets informed on possible future developments. For instance, she learns that if, according to the NT screening results, a chance of her baby being affected is high, she will be offered to undergo amniocentesis to obtain a conclusive evidence.



Registration form

Personal Information

Title:	Mrs.
Firstname:	Saskia
Lastname:	Dekker
Birthdate:	21-6-1970

Address Information

Address:	Den Dolech 2
Zipcode:	5612 AZ
City:	Eindhoven
Phone number:	0612354876
E-mail address	saskia.dekker@moeder.nl
Communicate via:	Text Messaging

Medical History

Family history of: (None / do not know) Diabetes Spina Bifida

Pregnancy Information

First Pregnancy:	[✓ Yes] No
Week of gestation:	5
Multiple births in family?	Yes [✓ No]

Form template's location: C:\Documents and Settings\jvdwerf\Local Settings\Application Data\Microsoft\Info

Fig. 1. Registration on-line form

The personalized web-page further includes a personalized pregnancy calender. All appointments arranged are automatically added and reminders are sent. Furthermore, depending on Saskia's personal data and outcomes of the preceding tests, new appointments to be scheduled are highlighted in the corresponding periods. For instance, anti Rh(D) IgG immunoglobulin injections will appear on the twenty eighth week and in her delivery procedure description. Note that healthcare providers have access to Saskia's pregnancy calender and get alerted, for example, if she fails to show up at her anti Rh(D) IgG immunoglobulin injection date.

Requirements imposed by the example. In order to support the desired scenario presented above a number of changes has to be introduced to current healthcare information systems. First of all, in order to provide Saskia with information on possible future developments, a healthcare information system should be made aware of the ongoing *processes* rather than only *data* involved. Recall that data relates to information till the current moment, while patients can be eager to know what can happen next. However, assuming one ongoing process is not realistic, as different healthcare providers have their own rules and processes. Sharing these processes might be superficial (a midwife should not necessarily be knowledgeable of the biochemistry of a blood test) or undesirable (lab assistant should not have access to Saskia's personal information). Therefore, rather than considering one process we envision *a series of interrelated but independent processes*.

Another important conclusion that we can draw from our example is that processes should be able to modify themselves on-the-fly. For instance, when Saskia's Rh(D) turns out to be negative, special treatment should be performed, i.e., a special process should be initiated in parallel with the ongoing one. We refer to the ability of a process to modify itself on-the-fly as *adaptivity*. Still, we do not intend to substitute medical stuff by an automatic decision making system. The choice of treatment protocols is certainly made by the care providers together with the patient. Therefore, processes should be not only adaptive but also *adaptable*.

3 Adaptive Petri Nets

As we could see in the Saskia example, HIS is a domain with a great need for adaptivity. In this section we present the theoretical foundations of our approach, so called *adaptive Petri nets*, and we illustrate the concepts introduced by means of a running example: a simplified version of the Dutch age-related prenatal diagnostics protocol, given in Figure 2. The owner of the considered process is the midwife.

A Petri net [18] is a bipartite graph whose nodes are called *places* and *transitions*. Transitions, graphically represented as rectangles, correspond to actions being taken. Places are represented as circles, and they are used to define the process flow. Given a transition we distinguish the *input places of the transition*, i.e., places with an arc going to the transition, and *output places of the transition*, i.e., places with an incoming arc coming from the transition. The Petri net presented in Figure 2 includes such transitions as "select age-related diagnostics, first trimester" and "communicate negative news and select further diagnostics". The name of a transition is written *under* the corresponding rectangle.

Workflow nets [1] are a special class of Petri nets, well-suited for modelling workflow processes. Workflow nets have exactly one place with no incoming arcs, called *the*

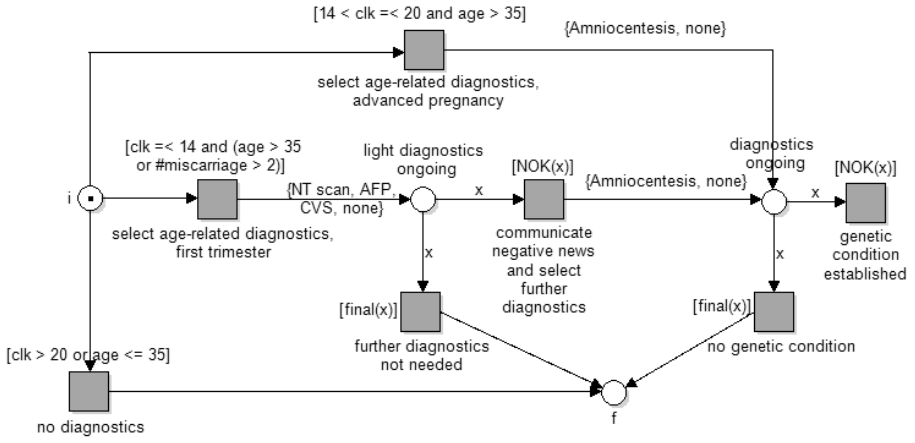

[14 < clk =< 20 and age > 35]
{Amniocentesis, none}
select age-related diagnostics, advanced pregnancy
[clk =< 14 and (age > 35 or #miscarriage > 2)]
light diagnostics ongoing
[NOK(x)]
diagnostics ongoing
[NOK(x)]
{Amniocentesis, none}
i
{NT scan, AFP, CVS, none}
x
x
select age-related diagnostics, first trimester
x
communicate negative news and select further diagnostics
x
genetic condition established
[final(x)]
[final(x)]
further diagnostics not needed
no genetic condition
[clk > 20 or age <= 35]
no diagnostics
f

Fig. 2. Genetic condition related tests

initial place, and exactly one place with no outgoing arcs, called *the final place*. Moreover, every node in a workflow net is on a path from the initial place to the final place. *Extended workflow nets* [10] can be obtained from a workflow net by adding *exception transitions*, which are transitions with at least one input place and no output places. Exception transitions are used for modelling undesirable, abnormal or irregular events of such a nature that the process cannot decide itself how to continue and an assistance of a higher authority/layer is required. The Petri net in Figure 2 is an extended workflow net with the initial place i, the final place f and the exception transition "genetic condition established".

The state of a system is represented by means of *tokens*, drawn as black dots and residing in places. The initial state consists of a single token in the initial place i, while the final state consists of a single token in the final place f. The final state corresponds to the most likely, expected, termination of the process. Dynamics of a process is expressed by means of the *token game*: performing an action corresponds to a *transition firing* removing a token from every input place and adding a token to every output place of the transition. For instance, "select age-related diagnostics, first trimester" removes one token from the initial place i and produces one token in the output place *light diagnostics ongoing*. Figure 2 shows the state of the system before a firing of "select age-related diagnostics, first trimester". In the normal course of events, process terminates when we reach the state consisting of a single token on place f. Firings of exception transitions terminate the execution of the process independently of the process state, disregarding the fact whether there are still tokens left and transitions enabled.

Colored Petri nets [16] extend Petri nets by introducing data and time into the model, i.e., allowing to model a data flow in addition to a control flow. Due to historical reasons data types are commonly referred to as *colors*. Classical Petri nets are extended there by *guards* and *arc inscriptions*. The guard is a logical expression determining whether the transition may fire. Arc expressions at the outgoing arcs define data transformations. The firing of transitions become thus data dependent, and, moreover, transitions modify data.

Global history nets [12] further extend Petri nets by assuming the availability of a history record, registering all firings and the time of firing together with the information which process performed it. Transition guards can depend on the information contained in the history record. Guards are written between square brackets. For instance, "select age-related diagnostics, first trimester" has the following guard: "$\mathsf{clk} \leq 14$ *and* $(\mathsf{age} > 35$ *or* $\#\mathsf{miscarriage} > 2)$". This means that the corresponding EPR contains information that the gestation week is not later than 14 (clk denotes here the pregnancy clock), and either the maternal age exceeds 35 or the history record contains at least three previous miscarriages.

Adaptive nets[10] further extend the formalism by introducing a special color: nets. In other words, a token can be associated with another (adaptive) net, called a *token net*. Token nets are created by using suggested *library nets* or tailor-made nets. So, the firing of "select age-related diagnostics, first trimester" is in fact a procedure where the midwife together with the pregnant woman decides which prenatal test procedure will be carried out, if any at all. The suggested tests (NT scan, AFP, CVS and none) are indicated on the outgoing arc of the transition. By using "any" in the suggested set of token nets, we allow to take an arbitrary, possibly tailor-made, protocol. The considered example does not make use of these freedom of choice, since the set of tests is predefined. Note that the owner of the created token net is not necessarily the midwife. In this example, it will be a lab performing the selected test procedure.

Guards can be used to *synchronize* firings of the (upper layer) net with firings of a token net. The transition "communicate negative news and select further diagnostics" fires if the selected test procedure reports a high risk of genetic condition (NOK), for it is conditioned by the guard NOK(x), where x refers to the token consumed from the "light diagnostics ongoing" place. In other words, NOK in the selected test procedure and "communicate negative news and select further diagnostics" fire *synchronously*, which corresponds to the communication between the lab and the midwife. An additional form of synchronization is illustrated by "further diagnostics not needed". This transition can fire if the token net in the "light diagnostics ongoing" place has reached its final state, i.e., the only token present in the token net is residing in its final place.

4 Modelling Care Processes with Adaptive Nets

In this section we discuss how adaptive nets, discribed in Section 3 can be used to model care processes discussed in Section 2. First of all, we assume that (a view on) the electronic patient record (EPR) is available for all nets. EPR can be seen as a standard data record extended with the history log. Access to the EPR can and typically is further restricted depending on the task being performed: all tasks should be able to consult the pregnancy clock, but it is set during the registration and can be corrected by the midwife only.

Although the theoretical framework of adaptive nets does not restrict in the depth of nesting, we envision that the care applications will typically make use of three-layered processes. The *top-level* process represents the main process flow: registration, the choice of diagnostics and treatment protocols and closing the case. The main process flow can be associated with strategic goals. The second layer is normally a protocol layer. Processes of the second layer can can be carried out by the main care provider herself or delegated to other care providers. The operations can be seen as implementing the tactic goals of the process. The net at Figure 2 provides an example of a second level net. Finally, realization of medical protocols can require services of parties such as medical labs. These basic steps constitute the *third and the last level* of the process.

The deviations from the described process architecture scheme are of course possible: the midwife could for example demand Saskia to consult a cardiologist in case her blood pressure is repeatedly evaluated as too high. The cardiologist process, located at the second layer in this case, is however a strategic-level process, which is still a service for the midwife protocol. The nesting depth in this case could be greater than three.

The top-level process consists typically of the initialization (registration, analysis of the history available), a number of subprocesses taking care of adaptivity aspects, and a process termination (including a semi-automatic post-processing of the EPR to provide consistent data for future use). Figure 3 shows a pattern for a subprocess. The transition "initiate an additional protocol" is mostly often triggered by some event in the environment (e.g. an additional complaint of a patient). In response to this event, an additional

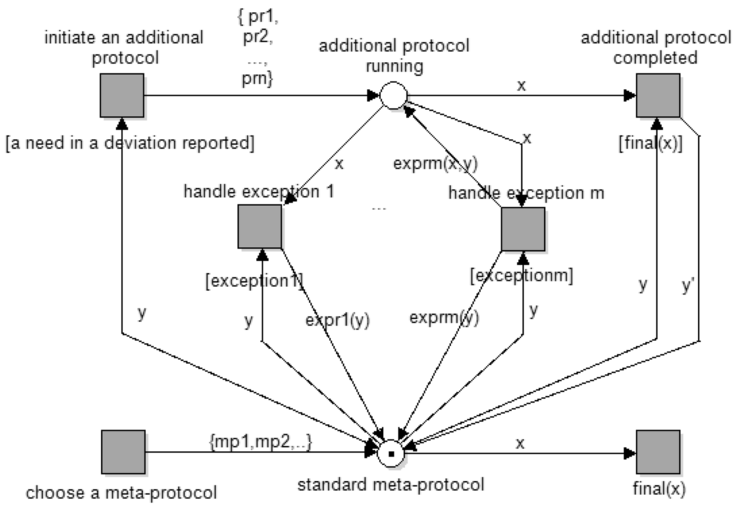
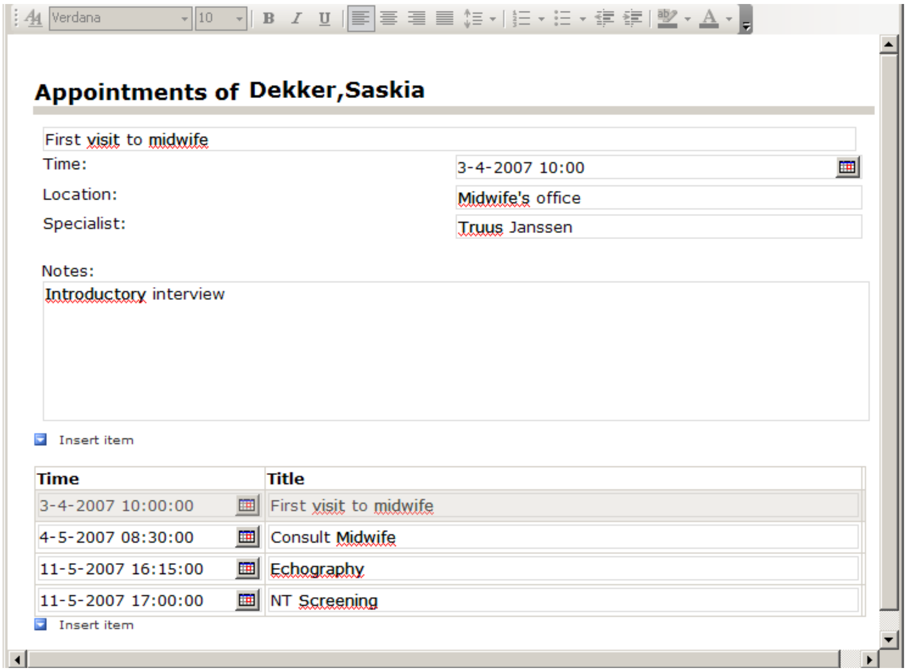


Fig. 3. Generic schema of a subprocess

protocol can be chosen and started (alternatively, or additionally, the token net on the place “standard meta-protocol” can be modified). Exceptions and the termination of this additional protocol are then processed by the corresponding transitions by initiating new processes of modifying the running protocol. Note that there are two kinds of exception-guards possible — the first one are guards demanding a synchronization with a corresponding exception transition in the token net, while the second one are guards specifying some external trigger.

The history record is composed from all actions taken in each protocol and it provides not only the information on the action taken but also specifies who (which process) took it. Also the initiations of additional protocols and modifications in the running protocols are logged. This allows to keep the (otherwise chaotic) history record well-structured, since we can always make a query to get all the information related to some treatment, abstract from unnecessary details (e.g. do not show actions of the third layer protocols), aggregate information related to a class of treatments (e.g. cardiological treatments), even if these treatments were initiated by different care providers (the midwife, the GP, the cardiologist).

Having the information on the processes running, we can construct a forecast view for both care providers and the patient by constructing and exploring a (partial) state space of the running protocols. Here different options are possible. The most simple one is constructing a forecast under the assumption that no exception will happen (e.g. the normal course of pregnancy, taking into account personal ERP information, like negative Rh(D)). More elaborated views allows to look into the future taking into account exceptions that are predefined in the running protocols. For instance, when the age-related diagnostics protocol in Fig. 2 is started for Saskia, it is possible to build a view informing that amniocentesis can be performed in case the NT-screening reports a high risk of a genetic condition.



Appointments of Dekker,Saskia

First visit to midwife

Time:	3-4-2007 10:00
Location:	Midwife's office
Specialist:	Truus Janssen

Notes:
Introductory interview

Time	Title
3-4-2007 10:00:00	First visit to midwife
4-5-2007 08:30:00	Consult Midwife
11-5-2007 16:15:00	Echography
11-5-2007 17:00:00	NT Screening

Fig. 4. Saskia's appointments calendar

The first type of a forecast view (the likely one) can also be used for such a pragmatic thing as planning appointments, tests and treatments. An important issue here is that the interprocess dependencies and restrictions can be handled, so that different protocols would not interfere or damage each other. Another trivial gain is e.g. providing a personalized pregnancy calendar for Saskia (see Figure 4) and providing her with reminders on appointments and tests supplied with additional information, like prerequisites of these tests.

5 Conclusion

The major contribution of our work consists in proposing a process-centered patient-centered framework for the new generation of HIS. While patient-centered systems attract more and more attention of the research community, the current proposals [5,21] concentrate mostly on implementation issues, such as data communication and heterogeneity of the application platforms, rather than on the conceptual ones, such as separation of concerns and adaptivity.

Our proposal is based on a solid theoretical foundation, namely, adaptive (nested) nets [11,10], a subclass of Petri nets. This allows us to perform a number of automatic correctness checks, like soundness and circumspectness. Soundness means that every process can terminate properly from any reachable state, while circumspectness means that every exception can be taken care of by the higher layer.

We do not expect the end user to know what Petri nets are and aim at providing a user-friendly web-interface instead. For this purpose, we developed the tool YasperWe [9] that is designed for prototyping IS. The tool integrates Yasper, which is a Petri net editor including a number of analysis options and compatible with some other analysis tools, with Microsoft Infopath.

Related work. Processes in healthcare are commonly documented in the form of *medical guidelines*. A guideline is not limited to doctors but also covers the workspace of nurses and paramedical personnel. There have been several attempts to formalize guidelines as *flowcharts* and *decision diagrams* and incorporate them into medical decision support systems.

Petri nets have been used for modeling of healthcare workflow, also known as careflow [13,19,20]. The guideline execution system GUIDE [20] translates formalized guidelines to a hierarchical timed colored Petri net. The resulting net can be run to simulate the implementation of the guideline in clinical setting. However, this formalism misses adaptivity and separation of concerns. The idea of adaptivity, i.e., controlled modification, in Petri nets has been considered in [6,14]. However, these approaches were able to model only processes involving two care providers, for instance, a midwife and a general practitioner, which is not sufficient for common healthcare processes.

Currently, there are several approaches that offer some degree of flexibility. Many share the idea of modeling with underspecification, i.e. a model where parts of the process are not given explicitly, but represented by a placeholder. At run-time the configuration is completed by binding the placeholder to a process from a repository, which is also known as late binding. Adaptive nets [10,11], worklets [3], pockets of flexibility [22] are modeling techniques that allow for late binding. Three main advantages of adaptive nets compared to the other approaches are the following:

– the ability to modify the structure of running processes in a controlled fashion;
– the possibility to define explicit synchronization between a token net and its owner;
– verification of a number of correctness properties for a subclass of adaptive nets.

Future work. For the future work, we are going to propose a number of process patterns to facilitate process modelling and a number of query patterns for the creation of different user views.

References

1. van der Aalst, W.M.P.: Verification of workflow nets. In: Azéma, P., Balbo, G. (eds.) ICATPN 1997. LNCS, vol. 1248, Springer, Heidelberg (1997)
2. van der Aalst, W.M.P., Weijters, T., Maruster, L.: Workflow mining: Discovering process models from event logs. IEEE Trans. Knowl. Data Eng. 16(9), 1128–1142 (2004)
3. Adams, M., et al.: Worklets: A service-oriented implementation of dynamic flexibility in workflows. In: Meersman, R., Tari, Z. (eds.) OTM 2006. LNCS, vol. 4275, pp. 291–308. Springer, Heidelberg (2006)
4. Charles, C., Gafni, A., Whelan, T.: Shared decision-making in the medical encounter: What does it mean (or it takes at least two to tango). Social Science and Medicine 44(5), 681–692 (1997)

5. Cheng, P., Yang, C., Chen, H., Chen, S., Lai, J.: Application of hl7 in a collaborative healthcare information system. In: 26th Annual International Conference of the Engineering in Medicine and Biology Society, 2004. EMBC 2004, Conference Proceedings, vol. 2, pp. 3354–3357 (September 2004)
6. Ehrig, H., Padberg, J.: Graph grammars and Petri net transformations. In: Desel, J., Reisig, W., Rozenberg, G. (eds.) ACPN 2003. LNCS, vol. 3098, pp. 496–536. Springer, Heidelberg (2004)
7. European Comission. Communication from the commission to the council, the european parliament, the european economic and social committee and the committee of the regions. e-health - making healthcare better for european citizens: An action plan for a european e-health area, SEC(2004)539 (April 2004)
8. Gerber, B.S., Eiser, A.R.: The patient-physician relationship in the internet age: Future prospects and the research agenda. Journal of Medical Internet Research 3(2), e15 (2001)
9. van Hee, K., Keiren, J., Post, R., Sidorova, N., van der Werf, J.M.: Designing case handling systems. In: Proc. of the International Workshop on Petri Nets and Software Engineering (2007)
10. van Hee, K., Lomazova, I., Oanea, O., Serebrenik, A., Sidorova, N., Voorhoeve, M.: Nested Nets for Adaptive Systems. In: Donatelli, S., Thiagarajan, P.S. (eds.) ICATPN 2006. LNCS, vol. 4024, pp. 241–260. Springer, Heidelberg (2006)
11. van Hee, K.M., Lomazova, I.A., Oanea, O., Serebrenik, A., Sidorova, N., Voorhoeve, M.: Checking properties of adaptive workflow nets. Fundamenta Informaticae 79(3–4), 347–362 (2007)
12. van Hee, K.M., Serebrenik, A., Sidorova, N., van der Aalst, W.: History-dependent Petri nets. In: Kleijn, J., Yakovlev, A. (eds.) ICATPN 2007. LNCS, vol. 4546, pp. 164–183. Springer, Heidelberg (2007)
13. Hoffman, K.: Run time modification of algebraic high level nets and algebraic higher order nets using folding and unfolding construction. In: Hommel, G. (ed.) Proceedings of the 3rd Internation Workshop Communication Based Systems, pp. 55–72. Kluwer Academic Publishers, Dordrecht (2000)
14. Hoffmann, K., Ehrig, H., Mossakowski, T.: High-level nets with nets and rules as tokens. In: Ciardo, G., Darondeau, P. (eds.) ICATPN 2005. LNCS, vol. 3536, pp. 268–288. Springer, Heidelberg (2005)
15. Q. Homan. Harris interactive and ARiA marketing healthcare satisfaction study (October 2000), Available at: http://www.harrisinteractive.com/news/downloads/HarrisAriaHCSatRpt.PDF
16. Jensen, K.: Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use. In: Monographs in Theoretical Computer Science, Springer, Heidelberg (1997)
17. Koh, H.C., Tan, G.: Data mining applications in healthcare. Journal of Healthcare Information Management 19(2), 64–72 (2005)
18. Petri, C.A.: Kommunikation mit Automaten. PhD thesis, University of Bonn, Bonn, West Germany (1962)
19. Quaglini, S., Panzarasa, S., Cavallini, A., Micieli, G., Pernice, C., Stefanelli, M.: Smooth integration of decision support into an existing electronic patient record. In: Miksch, S., Hunter, J., Keravnou, E.T. (eds.) AIME 2005. LNCS (LNAI), vol. 3581, pp. 89–93. Springer, Heidelberg (2005)
20. Quaglini, S., Stefanelli, M., Cavallini, A., Micieli, G., Fassino, C., Mossa, C.: Guideline-based careflow systems. Artificial Intelligence in Medicine 20(1), 5–22 (2000)
21. Ricci, F.L., Serbanati, L.D.: Mobidis: Toward a patient centric healthcare information system. Studies in Health Technology and Informatics 116, 557–562 (2005)

22. Sadiq, S.W., Sadiq, W., Orlowska, M.E.: Pockets of flexibility in workflow specification. In: Kunii, H.S., Jajodia, S., Sølvberg, A. (eds.) ER 2001. LNCS, vol. 2224, pp. 513–526. Springer, Heidelberg (2001)
23. van der Wal, G.: Rapport preoperatief traject ontbeert multidisciplinaire en gestandaardiseerde aanpak en teamvorming. Technical Report 2007-02, Staatstoezicht op de volksgezondheid. Inspectie voor de Gezondheidszorg (February 2007), http://www.igz.nl/15451/475693/2007-02_Rapport_Preoperatie1.pdf

Access Control Requirements for Processing Electronic Health Records

Bandar Alhaqbani[1] and Colin Fidge[2]

[1] Information Security Institute
b.alhaqbani@isi.qut.edu.au
[2] School of Software Engineering and Data Communications,
Queensland University of Technology, Brisbane, Australia
c.fidge@qut.edu.au

Abstract. There is currently a strong focus worldwide on the potential of large-scale Electronic Health Record systems to cut costs and improve patient outcomes through increased efficiency. A number of countries are developing nationwide EHR systems to aggregate services currently provided by isolated Electronic Medical Record databases. However, such aggregation introduces new risks for patient privacy and data security, both by linking previously-separate pieces of information about an individual, and by creating single access points to a wide range of personal data. It is thus essential that new access control policies and mechanisms are devised for federated Electronic Health Record systems, to ensure not only that sensitive patient data is accessible by authorized personnel only, but also that it is available when needed in life-critical situations. Here we review the traditional security models for access control, Discretionary Access Control, Mandatory Access Control and Role-Based Access Control, and use a case study to demonstrate that no single one of them is sufficient in a federated healthcare environment. We then show how the required level of data security can be achieved through a judicious combination of all three mechanisms.

1 Introduction

The healthcare domain—as one of the world's largest hybrid organizations—stands to gain enormously from increased adoption of Information and Communications Technologies. Electronic Health Record (EHR) systems are the latest evolution of healthcare ICT, and countries such as Australia, the United Kingdom and the USA are all working on plans for national EHR systems [9].

An Electronic Health Record is defined by Iakovidis [12] as "digitally stored healthcare information about an individual's lifetime with the purpose of supporting continuity of care, education and research, and ensuring confidentiality at all times". It is a mechanism for integrating healthcare information currently collected in both paper files and Electronic Medical Record (EMR) databases by a variety of separate healthcare providers [16].

Electronic Health Records enable efficient communication of medical information, and thus reduce costs and administrative overheads [5]. Furthermore, EHRs will help to reduce incidents of medication error—in current healthcare systems, medical data is entered and can be interpreted in inconsistent and possibly ambiguous ways. Moreover, a patient's health records are currently often dispersed over multiple sites with no single healthcare professional having access to all of this data. Nationwide EHR systems aim to solve these problems.

However, to achieve these potential benefits, the healthcare industry must overcome several significant obstacles. Currently, medical information is stored in a variety of proprietary formats using numerous off-the-shelf and custom-built medical information systems. This results in a severe inter-operability problem in the healthcare sector [4].

Also, the security of each patient's medical data is a major issue [15] which, if not addressed in both a technically-sufficient and transparent way, will lose the patient's confidence in and trust of the EHR system. In a worst-case scenario, patients may resorting to falsifying information in an attempt to preserve their privacy, thus affecting the integrity of the stored data and potentially leading to life-threatening situations such as inappropriate medication. Chhanabhai et al. [2] showed in their EHR usability survey that 73.3% of participants were highly concerned about the security and privacy of their health records. Their study indicated that consumers are ready to accept the transition to EHR systems, but only provided they can be assured of the system's security.

Several solutions are available to overcome the security concerns associated with EHR systems. Cryptographic technology, through the use of Public Key Infrastructure [3], allows confidential information to be transmitted safely via an insecure communications medium such as the Internet. On its own, however, cryptography merely handles the security of data transmission and does not address the issue of what kind of data is transmitted, or solve the problem of who has access to the data at the sending and receiving ends.

To do this we need to consider access control mechanisms that limit who can see Electronic Health Records and how they can manipulate them. Access control mechanisms have been through many developments [14] in both academia and industry in order to satisfy the needs of healthcare domains. However, progress to date have not been sufficient to meet the security requirements of a federated healthcare environment [8]. Most of the models developed so far have been designed to satisfy healthcare security requirements in a controlled environment, such as the Electronic Medical Record database maintained within a hospital. By contrast, access control mechanisms for EHRs must be safe for use in open networks, such as the Internet, and with peripheral equipment that was not designed for highly-secure operations, such as a patient's home computer.

Discretionary Access Control (DAC), Mandatory Access Control (MAC) and Role-Based Access Control (RBAC) are well-established access control principles and have been recognized as official standards. Each was designed to overcome limitations found in its predecessor. DAC, the first standard introduced, controls each user's access to information on the basis of the user's identity and authorization [17]. MAC, the second standard introduced, governs access on the basis of the security classification of subjects (users) and objects in the system.

RBAC, the third standard introduced, regulates user access to information on the basis of the activities particular types of users may execute in the system.

In this paper, we demonstrate through case studies that none of these three mechanisms in isolation is sufficient for the privacy and security requirements of Electronic Health Record systems. We then explain how a careful combination of all three access control standards can be used to deliver the essential security requirements of a federated EHR system.

2 Related and Previous Work

An access control mechanism is intended to limit the actions or operations that a legitimate user of a computer system can perform [17]. This research area has witnessed numerous developments in the last two decades that have resulted in the widespread adoption of three different access control models. In this section we introduce these three models and point out which of their known limitations would apply in the healthcare domain.

2.1 Discretionary Access Control

Discretionary Access Control is a means of restricting access to objects based on the identity of subjects and/or groups to which they belong [7]. The controls are discretionary in the sense that a user or subject given discretionary access to a resource is capable of passing that capability along to another subject. The identity of the users and objects is the key to discretionary access control. DAC policies tend to be very flexible and are widely used. However, DAC policies are known to be inherently weak for two reasons: granting read access is transitive and DAC policies are vulnerable to “Trojan horse” attacks [10,7].

DAC policies are commonly implemented through Access Control Lists (ACLs) and ‘owner/other’ access control mechanisms, but these mechanisms are difficult to manage because addition and deletion of users or data objects requires discovery and treatment of all dependent entries in the DAC matrix.

In an Electronic Health Record system the access control requirements are more complex than allowed for by Discretionary Access Control because the data in an EHR is nominally ‘owned’ by the patient [9], but is also updated by healthcare professionals, and is stored on infrastructure belonging to health-care providers and regulators. Indeed, a DAC model could create new security problems due to the patient’s mismanagement of their own records [7].

2.2 Mandatory Access Control

A Mandatory Access Control policy, which is known to prevent the “Trojan horse” problem [7,17], means that access control decisions are made by a central authority, not by the individual owner of an object, and the owner cannot change access rights. The need for a MAC mechanism arises when the security policy of a system dictates that protection decisions must not be decided by the object’s owner, and the system must enforce data protection decisions [7].

Mandatory Access Control typically occurs in military-style security. Usually a security labeling mechanism and a set of interfaces are used to determine access based on the MAC policy. For example, a user who is running a process at the *Secret* classification level should not be allowed to read a file with a label of *Top Secret*. This is known as the "simple security rule", or "no read up". By contrast, a user who is running a process with a label of *Secret* should not be allowed to write to a document with a label of *Confidential*. This rule is called the "$\star$-property" or "no write down". Multilevel security models such as the Bell-La Padula Confidentiality [6] and Biba Integrity [1] models are used to formally specify this kind of MAC policy. Nevertheless, unintended information transfer can occur in systems using MAC through covert channels, whereby information of a higher security class is deduced indirectly by intelligently combining information visible to a lower security class [10].

Applying Mandatory Access Control mechanisms in an EHR environment is likely to be very difficult due to the huge number of users who participate in those systems, the wide range of data types, and the desire to give patients ownership and (partial) control over their own medical records. Nevertheless, implementing some form of MAC policy is inevitable in an EHR system, since medical authorities must be ultimately responsible for assigning access rights [5].

2.3 Role-Based Access Control

Role-Based Access Control decisions are based on the roles that individual users have as part of an organization. Users take on assigned roles (e.g. doctor, nurse or receptionist in our case). Access rights (or permissions) are then grouped by role name, and the use of resources is restricted to authorized individuals [10]. Under RBAC, users are granted membership into roles based on their competencies, credentials and responsibilities in the organization. User membership in roles can be revoked easily and new operations established as job assignments dictate. This simplifies the administration and management of permissions since roles can be updated without updating the permissions for every individual user. Moreover, use of role hierarchies provides additional advantages since one role may implicitly include the operations associated with another role. Also, RBAC can satisfy the "least privilege access" requirement [17], which involves granting the minimum set of privileges required for individuals to perform their job functions. Separation of Duty (SoD) can be incorporated into the RBAC model [18] to ensure that a user is not allowed to execute two roles simultaneously, as per the organization's policy.

Role-Based Access Control has gained a lot of attention in healthcare security research thanks to: its ability to provide practical fine-grained access policy administration for a large number of users and resources; it being a neutral policy; and its support for the 'need-to-know' security principle.

However, some access request evaluations are complex, due to the need to consider other contextual parameters in the evaluation phase. To overcome this problem, the Contextual RBAC model adds contextual parameters (e.g. time and location) to the RBAC model [19]. Nevertheless, even Contextual RBAC is insufficient to support the dynamic permission assignments that are needed in the healthcare domain, so Motta et al. [13] extended the model further so that permission assignment is based on specific evaluation mechanisms using contextual attributes that are available at access time, and Wilikens et al. [19] used a trust level as a measurement to assign permissions.

Unfortunately, this extended process would add yet more complexity to an EHR system — which requires establishing a connection between the EHR system and the Hospital Information Systems (HISs) that are responsible for handling administrative work within a hospital — in order to collect those contextual attributes which are not immediately visible, for example, the requestor's current medical role (e.g. as a doctor in an emergency department).

3 Healthcare Access Control Requirements

In the previous section we reviewed the capabilities of Discretionary Access Control, Mandatory Access Control and Role-Based Access Control. In order to better understand what kind of access control solution is needed for an Electronic Health Record system, we summarize in this section the specific access control requirements peculiar to EHR systems, illustrated by a small case study, and review the weaknesses of the existing mechanisms in this situation.

A control mechanism for Electronic Health Record access must satisfy all EHR participants' needs, i.e. patients, medical practitioners and medical authorities. Each participant needs to access certain fields of the health record in order to carry out his job. Also, the various participants need the ability to set specific access controls over the record. The following privacy and security requirements have been identified as crucial to healthcare environments:

1. Each healthcare unit should have the freedom to design its own security policy and to enforce it within its domain [15].
2. Healthcare providers (e.g. General Practitioners) should have the flexibility to arbitrarily define the security of a particular document if so required.
3. Patients should have the right to have control over their own health records, including whether or not to grant access to certain medical practitioners [15].
4. Patients should be able to hide specific items of information contained in their health records from selected medical practitioners.
5. Patients should have the ability to delegate control over their health records to someone else under certain conditions (e.g. mental illness).
6. Managing access control policies should be an easy task, in order to ensure that the system is used and to preserve trust in the system.
7. It is important that legitimate uses of health records are not hindered, e.g. overall system availability service levels, and overriding 'need-to-know' data access requirements in emergencies.

Ensuring each patient's privacy and data security is vital for an Electronic Health Record system. Unlike paper-based models, where an exposure or intrusion is confined to a single document or file, a federated EHR system creates the possibility of a patient's entire medical history being compromised by a single action. However, each of the traditional access control models, reviewed in Section 2, can satisfy only some of the above-listed requirements.

To see the access control weaknesses inherent in these previous models, consider the following scenario:

> Frank prefers to go to two General Practitioners, Tony and Karen. Frank has two sensitive fields in his Electronic Health Record, describing mental illness and sexual issues, respectively. Frank is happy to let Tony have access to his EHR, including the relevant data field within his sexual record, but he wants to hide a data field within his mental illness record from Tony. On the other hand, Frank will allow Karen to access his EHR, including his entire mental illness record, but not the sensitive data field within his sexual record. Apart from these two GPs, Frank won't allow anyone to access the sensitive data fields in his mental or sexual health records. In addition, Frank's father John suffers from Alzheimer's disease, so Frank must manage the access control rights to his father's EHR.

Even this simple and unremarkable scenario creates problems for each of the traditional access control policies, as explained below.

Discretionary Access Control: To use a DAC model we first need to know who owns the Electronic Health Record because DAC assumes that the owner of the data is the one who controls access to it. However, in healthcare an EHR is partially owned by each of the patient, medical practitioner and medical authority [11], immediately creating an issue with respect to ownership. Furthermore, assuming that Frank has ownership of his EHR, he could nominate and grant access to his trusted/preferred medical practitioners (Tony and Karen), but it would be a difficult task for Frank as a non-expert to identify the specific medical data that is needed by each GP to do their job. The 'need-to-know' principle is required here, and in order to have it Frank is required to know the information that is needed for each medical practitioner and then set the access controls accordingly. By granting patients such control over their records, we may hinder the legitimate use of the EHR and, most likely, create another security problem due to the patient's mismanagement of their records. However, delegation of access control can be implemented easily in a DAC model. Since Frank's father owns his EHR, he can delegate access control to his son.

Mandatory Access Control: In a MAC model, Frank won't have any sort of control, because the Electronic Health Record system will be responsible for setting the security labels for users and EHR data objects. Therefore, Frank can't express his access control wishes over his EHR. Also, the 'need-to-know' principle can't be fully achieved here either, even if we apply a security level hierarchy. It's possible that two users might have the same security clearance (e.g. Tony and Karen), but should have different access permissions over a certain data object (e.g. mental health data). In the MAC case, we can't assign more than one security label for the same data object, so providing selective access to data objects is difficult. Moreover, there is no ability to delegate access control because patients have no control over their EHRs.

Role-Based Access Control: In an RBAC model, the 'need-to-know' principle can be satisfied by defining the permissions/operations that are required by a specific medical role, and this process could be done by an appropriate medical expert. However, in this situation Frank won't be able to hide his sensitive medical fields as he won't have any control over the permission assignments. In order to allow Frank to express his wishes, the information security officer must allow Frank to modify the permissions, roles, user-role and role-permission assignments. Frank would need to create three roles in order to satisfy his needs, which would become an unacceptably time-consuming and complicated task for most patients and is likely to lead to a conflict of access control settings. Delegation of roles in RBAC is permitted if the security officer would allow Frank's father to delegate his roles to his son. Generally, RBAC seems a better choice than DAC and MAC, though it is still not an adequate solution for EHR system security.

In summary, it is clear than none of the existing models is adequate on its own, but that each of them has some feature which is essential to an EHR security model. DAC allows patients to control which data can be seen by particular medical practitioners, MAC allows the medical authority to control access to specific kinds of data, and RBAC allows access rights to be associated with certain medical roles.

4 A Combined Access Control Protocol

Although the access control requirements for Electronic Health Records cannot be satisfied by any one access control model alone, we contend that a careful integration of all three existing models is sufficient. Combining existing models, rather then developing an entirely new one for healthcare, allows us to take advantage of the well-understood properties and established implementations for these models.

4.1 Overview of the Combined Protocol

In the combined model access to a particular Electronic Health Record data item is granted only if it satisfies all three policies. The challenge is to determine where and how each of the access control constraints is introduced.

The basis for our combined protocol is shown in Fig. 1. An Electronic Health Record schema is shown where each EHR field has two MAC-based security labels: one is assigned by the patient and the second is assigned by the medical practitioner. These labels are used to express the sensitivity class of the data field. Also a DAC-style Access Control List (ACL) is maintained by the patient, whereby the patient nominates his/her preferred/trusted medical practitioners and sets the security clearance for each of them. This security clearance allows the medical practitioner to access sensitive data that may not be allowed for

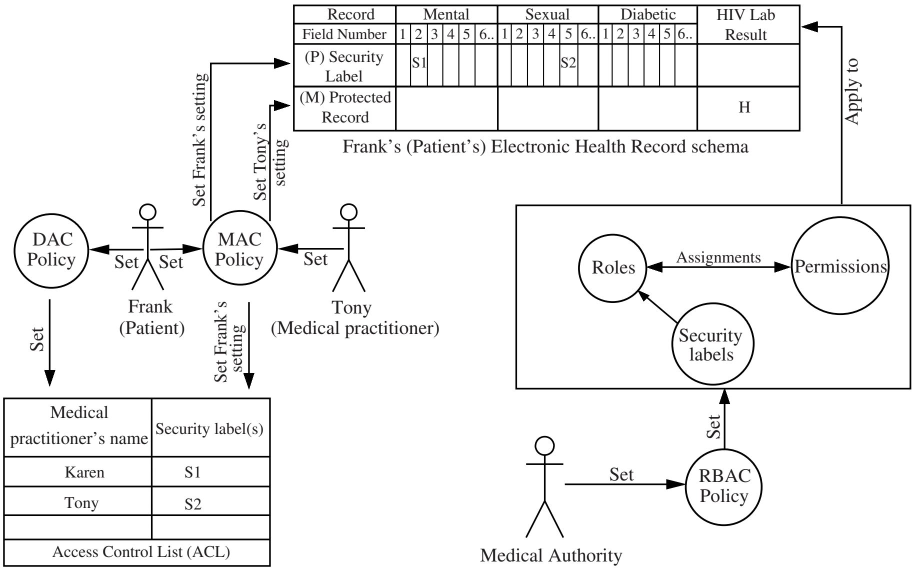


Fig. 1. The logical structure of the combined access control protocol

other medical practitioners. Access to EHR fields is further restricted by overall RBAC-based access control managed by the medical authority.

4.2 Maintenance and Enforcement of Access Control Constraints

Each of the participants in the EHR system (patients, medical practitioners and medical authorities) needs to maintain some aspect of the combined access control policy, and is constrained in what information they can view as a result. In this section we describe the sequence of events needed to do this.

We start with the patients' access control requirements, where the patients want to decide who is authorized to access their Electronic Health Records, to determine what is the sensitive information in their EHRs, and who is authorized to access it. These requirements are satisfied by executing the following steps using the DAC and MAC interfaces in our combined access control policy:

1. The patients nominate the names of specific practitioners who they trust, and this is done through the DAC interface in Fig. 1 to construct the Access Control List (ACL).
2. To categorize data fields as sensitive/protected information, the patient needs to assign security labels to these data fields by using the MAC interface to update the patient's Electronic Health Record schema.
3. To allow specific medical practitioners to gain access to security-classified data fields in the patient's EHR, the patient, via the MAC interface, assigns the same security label of the sensitive data field to the authorized medical practitioners' ACL.

In practice, however, we do not suggest using the "no read up" and "no write down" rules that are introduced in MAC because it would be too complex a task for most patients to keep track of the transitive relationships introduced by a full hierarchy of security levels. Instead patients should just be presented with simple access/no-access settings.

Medical practitioners, as EHR consumers, have certain access control requirements that are important. Medical practitioners need to:

– access all the information that is required to fulfill their medical role in normal scenarios (e.g. a standard consultation with a GP), unless the patient has excluded that practitioner from accessing the particular data field;
– access all the information that is required in emergency cases regardless of the patient's access control settings; and
– hide some medical information from the patient.

Medical practitioners' access control requirements are also satisfied here. The following steps show how these requirements are met:

1. The Medical authority defines roles, permissions and role-permission assignments via the RBAC interface. This process is done by domain experts who know the access requirements for each medical role. Therefore, the 'need-to-know' principle is achieved and medical practitioners' access needs will not be limited unless the patient has set some access control restriction through either the DAC or MAC interface.
2. Since RBAC can incorporate contextual attributes into role assignments, it would be possible for a medical practitioner to have an access role as a GP in a day clinic or as a GP in an emergency department. To allow the GP in an emergency department to access security-classified data records, the RBAC policy would assign a security label to these critical roles to allow them access to secure data. In an emergency case, the DAC constraints are not evaluated, due to the fact that the patient won't be able to know who the attending medical practitioners will be in an emergency.
3. To hide some medical information from the patient, the medical practitioner, through the use of the MAC interface, sets protection labels for these fields which hide the existence of such data in the patient's EHR.

Finally, the medical authority in charge of managing the Electronic Health Record system acts as the 'security officer' in the RBAC interface. It defines roles and permissions, and controls the assignment of permissions to roles in order to associate specific medical roles with the information needed to fulfill them.

4.3 Motivational Example Revisited

In this section we revisit the motivational scenario from Section 3 to see how our access control protocol would satisfy Frank's wishes.

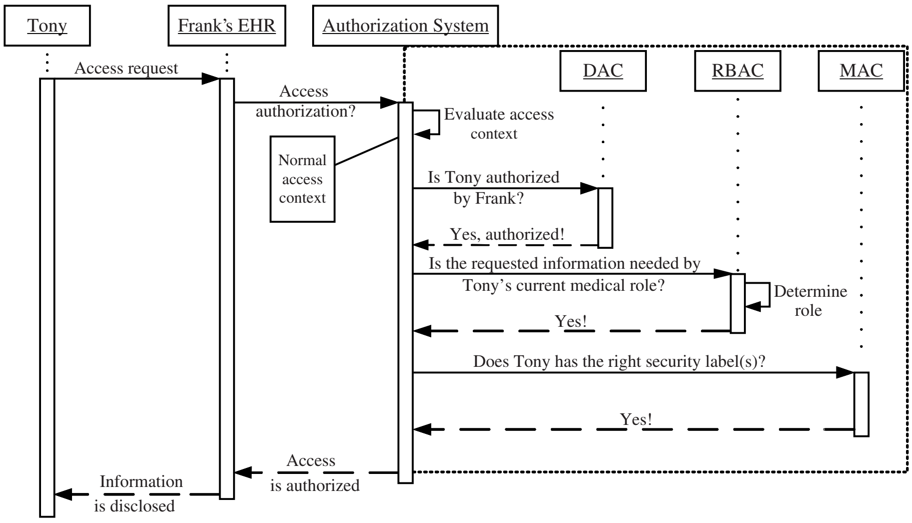


Fig. 2. The authorization evaluation process in the motivational example

1. Frank will classify the mental and sexual data fields of concern as 'sensitive' information by setting security labels S1 and S2 for each record, respectively.
2. He will nominate his preferred GPs Karen and Tony to access his EHR. As Frank is happy to allow Karen to access his sensitive mental data field, he will assign the S1 security label to her, which means that she is authorized to access any sensitive information that has an S1 label, in addition to her authorized access as per her medical role. For the same reason, Frank will assign the S2 security label to Tony which will allow him to access the sensitive sexual data field.
3. When Tony requests access to Frank's Electronic Health Record, to see his sexual history, the following access evaluation occurs (Fig. 2):
   (a) Evaluate access context, 'normal' or 'emergency'. If it's an emergency go to step 3c, otherwise continue.
   (b) DAC policy: Does Frank authorize Tony to access his EHR?
   (c) RBAC policy: Determine Tony's current medical role (e.g. day clinic GP, Emergency doctor) based on current contextual conditions.
   (d) RBAC policy: Is the requested information needed by Tony's current medical role?
   (e) MAC policy: Is Tony cleared to access this sensitive record?
   (f) Tony is granted access to Frank's sexual data only if his access request passes all the steps above.

Also, as Frank needs to take responsibility for his father's Electronic Health Record, the following actions can be performed.

1. Frank's father John needs to delegate control over his EHR to Frank through the use of the DAC interface.

2. Frank can now set the access rights to his father's EHR.

As well as these static assignments, we also need to consider temporary changes to access requirements. For instance, assume that Tony asks to see Frank's mental health record because he thinks that Frank's sexual issue is affected by some mental illness. This means that Frank must give Tony temporary access to his sensitive mental data field.

1. Frank will grant Tony another security label, S1.
2. Tony now has two security labels S1 and S2 from Frank, which means that he is authorized to access both of Frank's sensitive data fields contained in his sexual and mental health records.
3. After the consultation, Frank can revoke this permission by deleting label S1 from Tony's profile.

On the other hand, a medical practitioner may need to change the status of certain fields without involving the patient. For instance, assume that Tony asks Frank to take a blood test which turns out to be positive for HIV. Given Frank's mental state, Tony would prefer to hide the pathology results until Frank's next in-house consultation.

1. Tony assigns a 'hide' flag to the HIV lab result field in Frank's EHR, so that Frank can't see any information contained in that specific field.
2. However, this information can be seen by Frank's authorized medical practitioners, such as the blood bank to which Frank regularly donates.

5 Conclusion and Future Work

Emerging plans for national Electronic Health Record systems raise new concerns about patient privacy and data security, by merging medical records that were previously kept separate and by making them accessible through single access points. None of the three standard access control models, Discretionary Access Control, Mandatory Access Control and Role-Based Access Control, are adequate for an EHR system in isolation. Nevertheless, we have explained how a careful combination of all three access control models can provide the security functionality needed for an EHR system.

At the time of writing we are assessing the security issues associated with a prototype Service Oriented Architecture for healthcare data. Our goal is to determine whether such an 'application-oriented' networking environment can be used to implement the combined access control protocol described above.

Acknowledgements. We wish to thank the anonymous reviewers for their many helpful suggestions. This research was supported in part by Australian Research Council Linkage-Projects grant LP0776344.

References

1. Biba, K.J.: Integrity Considerations for Secure Computer System. Technical report, Mitre Corporation (1977)
2. Chhanabhai, P., Holt, A.: Consumers are Ready to Accept the Transition to Online and Electronic Records if They can be Assured of the Security Measures. Medscape General Medicine 9(1) (2007)
3. Demuynck, L., De Decker, B.: Privacy-Preserving Electronic Health Records. In: Dittmann, J., Katzenbeisser, S., Uhl, A. (eds.) CMS 2005. LNCS, vol. 3677, pp. 150–159. Springer, Heidelberg (2005)
4. Eichelberg, M., et al.: A Survey and Analysis of Electronic Healthcare Record Standards. ACM Computing Surveys 37(4), 277–315 (2005)
5. HealthConnect Business Architecture, version 1.0 (2003)
6. Bell, D.E., LaPadula, L.J.: Secure Computer Systems: Unified Exposition and Multics Interpretation. Technical report, Mitre Corporation (1976)
7. Ferraiolo, D., Kuhn, D., Chandramouli, R.: Role-Based Access Control. Artech House (2003)
8. Finance, B., Medjdoub, S., Pucheral, P.: Privacy of Medical Records: From Law Principles to Practice. In: Computer-Based Medical Systems. Proceedings. 18th IEEE Symposium on, pp. 220–225 (2005)
9. Tracy Gunter, D., Nicolas Terry, P.: The Emergence of National Electronic Health Record Architectures in the United States and Australia: Models, Costs, and Questions. Journal of Medical Internet Research 3, e3 (2005)
10. Hu, V., Ferraiolo, D., Kuhn, D.: Assessment of Access Control Systems. Technical report, National Institute of Standards and Technology (September 2006)
11. Iacovino, L.: Trustworthy Shared Electronic Health Records: Recordkeeping Requirements and HealthConnect. Journal of Law and Medicine 12, 40–60 (2004)
12. Iakovidis, I.: Towards Personal Health Record: Current Situation, Obstacles and Trends in Implementation of Electronic Healthcare Record in Europe. International Journal of Medical Informatics 52(1–3), 105–115 (1998)
13. Motta, G., Furuie, S.S.: A Contextual Role-Based Access Control Authorization Model for Electronic Patient Record. IEEE Transactions on Information Technology in Biomedicine 7(3), 202–207 (2003)
14. Park, J., Sandhu, R.: Towards Usage Control Models: Beyond Traditional Access Control. In: SACMAT 2002. Proceedings of the 7th ACM symposium on Access Control Models and Technologies, pp. 57–64. ACM Press, New York (2002)
15. Ray, P., Wimalasiri, J.: The Need for Technical Solutions for Maintaining the Privacy of EHR. In: EMBS 2006. 28th Annual International Conference of the IEEE, Engineering in Medicine and Biology Society, pp. 4686–4689 (2006)
16. Rishel, W., Handler, T., Edwards, J.: A Clear Definition of the Electronic Health Record. Technical report, Gartner (2005)
17. Sandhu, R., Samarati, P.: Access Control: Principles and Practice. IEEE Communications Magazine 32(9), 40–48 (1994)
18. Simon, R., Zurko, M.: Separation of Duty in Role-Based Environments. In: IEEE Computer Security Foundations Workshop, pp. 183–194 (1997)
19. Wilikens, M., Feriti, S., Sanna, A., Masera, M.: A Context-Related Authorization and Access Control Method Based on RBAC: A Case Study from the Health Care Domain. In: Proceedings of the 7th ACM symposium on Access control models and technologies, pp. 117–124. ACM Press, New York (2002)

Learning Business Process Models: A Case Study

Johny Ghattas, Pnina Soffer, and Mor Peleg

Management Information Systems
University of Haifa, Carmel Mountain 31905, Haifa, Israel
GhattasJohny@gmail.com, SPnina@is.haifa.ac.il,
Peleg.Mor@gmail.com

Abstract. Learning how to improve business processes is an evolutionary process that must be managed as other business processes (BPs) are managed in modern organizations. The proposed model – the learning process model (LPM) – suggests a closed-loop-model approach applied to a generic process model (GPM), which is a formal state-based and goal-based approach to process modeling. LPM strives to establish a learning process by (1) identifying goal and soft-goal states of the initial process model, (2) identifying exceptional states and incomplete state definitions at runtime, and (3) adapting automatically the process model according to the discovered states. Modifications provided by the learning process may be sufficient or may need to be complemented by non-automatic changes, when unacceptable business situations arise. The learning process also aims to adapt the current process model to possible technology, specific domain (e.g., clinical procedures at specific institutions), environmental requirements (e.g., regulations and policies), and process innovations. We demonstrate the application of LPM to a vaccination process.

Keywords: Learning, business process model, generic process model, clinical guidelines, exceptions, process flexibility, process adaptation, goals, soft-goals.

1 Introduction

In a dynamic business environment, business processes (BPs) need to be changed continuously [1-4] without affecting the production of the expected business values. The continuous business environment change, the shortening of required service time to market, the increasing number of inter and intra-organization integrations and the early adoption of new business technologies makes it impossible to fully-represent business processes during their conception time.

In this research, we postulate that it is possible to substitute the practice of fully representing business processes by a flexible adaptive form of designing, implementing, and managing business processes, through a business process learning approach. This approach would allow making a partial (minimal) definition of the business processes that would enable the organization to launch the required services with minimal time to market and seize business opportunities. Once the required business initiative is launched and operating, data regarding the process execution and the extent to which goals and soft goals are attained are collected and analyzed, and deviations between the currently defined process model and the actual business process are detected. This forms the basis for learning and adapting the process based on the day by day process enactment experience.

Such an approach is of particular importance in health-care processes, which may change on the fly in adaptation to specific patients, or change over time as a result of the availability of new knowledge. The clinical guidelines modeling research field provides several approaches for medical guidelines automation, discovery and even adaptation [5-8].

This paper envisions learning in business processes, and constitutes a first step towards the design of automated processes that (1) define and track suitable parameters about the process execution and (2) learn how these processes should improve and update the process model accordingly. We base our approach on the Generic Process Model (GPM) [9, 10] which is a formal process specification, suitable for our purposes due to its explicit representation of goals. The paper illustrates a potential model for the business process learning through a medical immunization case study.

2 Business Process Learning Model – A Proposal

We postulate that a process model needs to relate the process goals to the workflow required to accomplish them. Such a relation is necessary in order to evaluate whether the process execution is attaining its desired outcome (i.e., goals) and the performance of the process execution (i.e., soft-goals). In order for our model to be as formal as possible, we base it on the Generic process Model (GPM).

2.1 Using a Formal BPM – The Generic Process Model (GPM)

The focus of analysis in GPM is a domain, which is a part of the world. We describe the behavior of the domain using concepts from Bunge's ontology [11, 12] and its adaptation to information systems [13, 14]. A domain is represented by a set of state variables, each depicting a relevant property of the domain and its value at a given time. We view a successful process as a sequence of unstable states of the domain, leading to a stable state, which belongs to a set of states that reflect the process goal. An unstable state is a state that must change due to actions within the domain (an internal event) while a stable state is a state that does not change unless forced to by action of the environment (an external event). Internal events are governed by transition laws that define the allowed (or necessary) state transitions (events).

It is possible to define the projection of a process over a sub-domain, where the set of state variables addressed by the law is a subset of domain state-variables. Then, all transitions outside the sub-domain are considered external events, and the sub-domain may be in a stable state while the process is active in other parts of the domain.

The process goal, as addressed by GPM, is the state achieved by the process. However, the goal concept is sometimes used also to describe business objectives. GPM distinguishes process goals from *soft-goals*, which are defined as an order relation on goal states [10]. In other words, soft-goals relate to the desirability of possible states in the goal set (all meeting the condition that terminates the process) according to defined business objectives. These establish a ranking (order) among the goal states. For example, assume the goal of a process is a set of states where some medical treatment has been given to a patient, but a lower level of the patient's blood pressure following this treatment is considered better than a higher one.

Finally, GPM entails criteria for assessing the *validity* of a process, namely, its ability to achieve its goal [10]. It enables the analysis of a process to identify causes for invalidity and suggests appropriate redesign actions to eliminate these causes.

2.2 Business Process Learning – Definition

While a process can reach its goal states through different paths, these paths may attain different soft-goal levels. In addition, while a particular path may improve a specific soft-goal it might simultaneously worsen another. Based upon the GPM process definition, the result of process instance executions may be categorized to two main categories:

(1) Valid process instances which attain some process goal state.
(2) Invalid process instances which do not reach any process goal state and terminate into exceptions (exceptions occur when a process attains an unlawful state during its execution and cannot reach its goal).

The category of valid process instances may be further divided into an undefined number of sub-categories depending on attained levels of soft-goals. It is needless to say that a process path that leads to better soft-goal levels is preferred to others that lead to lower levels of soft-goals. Hence, it is clear that an organization must strive to improve continuously its capabilities to select better process paths in order to attain better soft-goal levels as well as fewer exception occurrences in runtime, namely, fewer instances where the process fails to achieve its goal. In parallel, the organization must strive to adapt (modify) its BP models because of new knowledge generated in the environment. *Business process learning* is the organization's capability to improve path selection through experience acquired from executing the business process, measuring the attained levels of soft goals and the exceptions rate.

2.3 Business Process Learning – A Process-Based Approach

We postulate that a business process learning model can be established analogously to control systems theory, which bases the continuous improvement of the behavior of a controlled system through a closed-loop system-control model [15]. The closed-loop model is based upon a real-time feedback-loop that continuously modifies the system behavior in order to minimize the output error attained during system runtime. A thorough discussion of control systems can be found in [15].

Adapting the model to BPM, at each process enactment, implies that:

(1) Soft-goals are measured and following their scores, the process model is evaluated to identify what specific segments of the state flow caused the improvement or worsening of each soft-goal.
(2) Whenever exceptions occur they are analyzed and learned lessons are used to recommend process model changes
(3) Once soft goal scores and exceptions are identified, the process law may be changed accordingly, so similar state flow will be repeated or avoided in future instances of the process. The updates may require changes in state definitions

(i.e., modification of the set of relevant state variables, addition/modification of predicates defining sets of states, and/or transformation definitions).

A closed-loop BP learning approach leads to the definition of the following learning process:

(1) During process instance execution, a set of state variables that would enable the evaluation of the attained level of soft-goal and the occurrences of exceptions are collected.
(2) Process instance soft-goal levels are evaluated.
(3) Collected process datum of the specific process instance, together with soft-goal levels and exception occurrence indicators are stored.
(4) The current process instance is compared to past experiences and assigned a relative score.
(5) Future executions would use the collected information and score to select in runtime the best known path.

2.4 Learning Process Model (LPM) Assumptions

For any formal conceptual model to be complete and valid, the ontological assumptions of the model need to be made explicit.

The proposed LPM relies on the following assumptions:

(1) The initial process model is valid, as far as we know (i.e., it was set in a way that is meant to attain its goals considering an expected set of possible external events independently of the process learning capability).
(2) Process mining capabilities do not affect process run time nor process performance.
(3) Process soft goals are known a-priori and are measurable through the process mining capabilities.
(4) Learning is based on the gaps between desired goals and actual execution of process instances. The process model is modified by: (a) comparing actual soft goals measurement to historical values of the process soft goals; (b) analyzing the current state flow as compared to past occurrences in order to explain the accomplished levels of soft-goals; (c) drawing required process model changes from exceptional process instances.

2.5 LPM Components

We establish the following postulates:

Postulate 1: In order for the overall business process to be a learning process, it should include three main sub-processes: Acting (A) process, Documenting (D) process, and Learning (L) process, as described below.
Postulate 2: These three processes are (in terms of the GPM) projections of the business process executed by the organization upon the respective sub-process domains (Acting, Documenting and Learning process domains).
Postulate 3: The A, D, and L processes interact through a set of well-defined *commitments*, as explained below. In addition, the external environment can affect these processes.
Postulate 4: The overall BP needs to adapt according to environment inputs.

The respective five model components are hereby described:

Component 1: An acting sub-process – the process that acts in order to accomplish the goals and soft goals of the process.
Component 2: A documentation sub-process – the process that collects the necessary data from the acting process for three main purposes: (1) conditioning next actions in the current acting process instance depending on data collected in previous process instance steps and data collected from past process instances; data may also be collected from external business processes (the fifth component of the model- see below); (2) affecting actions in other process instances (current and future ones); (3) providing learning processes with data collected and processed during the process enactments (i.e., soft-goal measures and exception details).
Component 3: A learning sub-process – the process in charge of adapting the business process model. It analyzes the collected data, produces required changes to support incurred exceptions, models needed changes, and introduces changes to the BPM. Note that each one of these sub-processes has its own goals and soft-goals.
Component 4: Inter-process dependencies: The commitments between the internal business sub-processes (learning, acting, documenting) is the basis for defining a valid overall business process. The acting process commitments are: (1) to provide necessary data for documentation process; (2) to execute according to the business process model that may be changed by the Learning sub-process.

The documenting process commitments are: (1) to collect necessary data for the acting process control/decision points; (2) to provide the acting process with data collected from previous process steps, past process instances, and data collected from external business processes; (3) to collect data for the learning sub-processes.

The learning process commitments are: (1) to provide/execute necessary changes to the business process model. Note that changes (both to D and A processes) have two major sources: needed adaptations following soft-goal assessments and exceptions detected; (2) to provide visibility/traceability of changes, structured process history, reports needed for human intervention.
Component 5: External (Human) processes & commitments: These are processes that affect the L, A, and D processes through inputs that are generated by the external environment**.** External processes are of two kinds: (1) modeling new cases, processes, and introducing innovation, when new medical knowledge is available (e.g., new drug, new available immunization); (2) manual management for handling exceptions. These are scenarios where the acting, learning, or documenting processes fail to continue their executing due to anomalies or unexpected situations. In such cases, automatic learning is not always feasible, and an external entity (e.g., a human) may intervene to correct the situation.

3 LPM Illustration through a Case Study

We use a clinical process based upon the guidelines for immunizations provided by the Institute for Clinical systems improvements (ICSI, [16]) as a case study. We start by presenting a hypothetical local version of the generic algorithm that addresses flu vaccination and is adapted to the workflow and regulation of a particular implementing healthcare institution. Next, we map the local process flow to our learning model and demonstrate how monitoring electronic medical record (EMR) data can be used to follow whether the executed process attains its goals or reaches undesired states, and how we can learn, that is, modify the process model in order to improve the extent to which it attains its goals or avoids undesired states.

3.1 Defining a Local Version of the Immunization Process

Fig. 1 shows a hypothetical local flu immunization process that follows the ICSI clinical algorithm [16]. When the patient is vaccinated for the first time in her life, the vaccine is provided in two portions, which are to be administered in two separate visits that are spaced one month apart.

3.2 Identification of the Flu Vaccine Business Process

We map the algorithm represented above into the components of the GPM process model – goals, soft-goals, states, intermediary states, and laws.

Identification of process goals and sub-goals:

Process goals: **(G1)** "Identify an eligible child"
(G2) "Provide vaccinations to an eligible child"

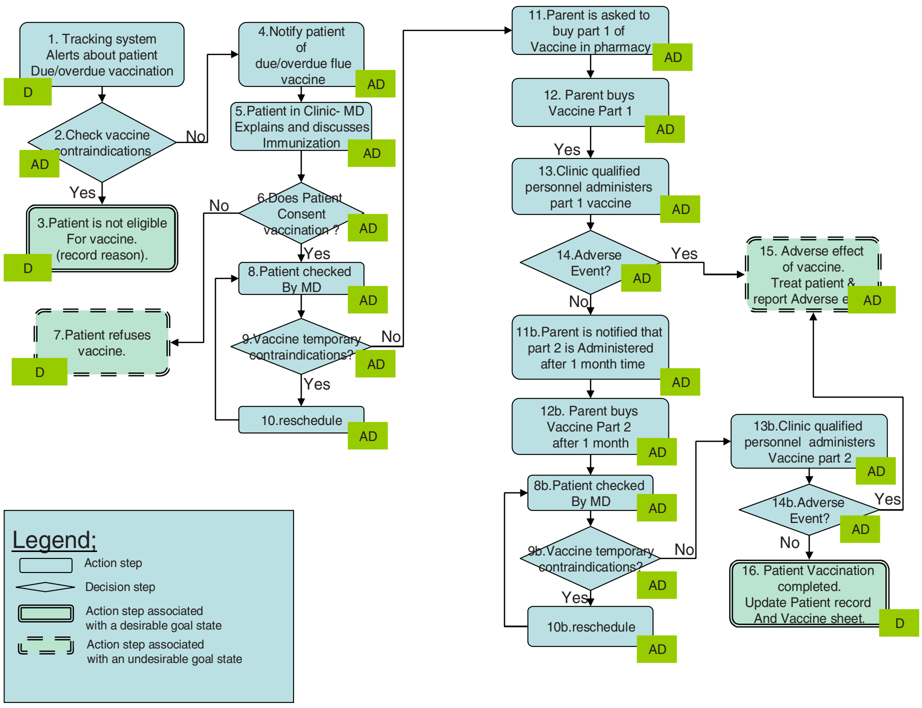

1. Tracking system Alerts about patient Due/overdue vaccination
D
2.Check vaccine contraindications
AD
No
Yes
3.Patient is not eligible For vaccine. (record reason).
D
7.Patient refuses vaccine.
D
4.Notify patient of due/overdue flue vaccine
AD
5.Patient in Clinic- MD Explains and discusses Immunization
AD
6.Does Patient Consent vaccination ?
AD
No
Yes
8.Patient checked By MD
AD
9.Vaccine temporary contraindications?
AD
No
Yes
10.reschedule
AD
11.Parent is asked to buy part 1 of Vaccine in pharmacy
AD
12. Parent buys Vaccine Part 1
AD
Yes
13.Clinic qualified personnel administers part 1 vaccine
AD
14.Adverse Event?
AD
Yes
No
15. Adverse effect of vaccine. Treat patient & report Adverse e
AD
11b.Parent is notified that part 2 is Administered after 1 month time
AD
12b. Parent buys Vaccine Part 2 after 1 month
AD
8b.Patient checked By MD
AD
9b.Vaccine temporary contraindications?
AD
No
Yes
10b.reschedule
AD
13b.Clinic qualified personnel administers Vaccine part 2
AD
14b.Adverse Event?
AD
Yes
No
16. Patient Vaccination completed. Update Patient record And Vaccine sheet.
D
Legend;
Action step
Decision step
Action step associated with a desirable goal state
Action step associated with an undesirable goal state

Fig. 1. Local version of the flu vaccination algorithm (first time vaccination). Each step has been identified with the sub-processes: A-Acting, D-Documenting, L-Learning. We identified soft goals: Undesirable Goal states (steps 7, 15) and Desirable Goal states (steps 3, 16).

Soft-goals: (SG1) "Do our best to make the parent accept vaccinating his child. Parent Refusal to vaccinate is an undesirable outcome". The score of this soft-goal is evaluated at the reached goal state; we consider it having a binary value: {desirable, undesirable}.

Goals and soft-goals are represented by process states in GPM. Although the process flow of Fig. 1 does not have states, only activities, we use the convention that when a process has completed an activity it is in a state named by the activity, and this state remains until execution of a new activity begins. In this way, we associate goal and soft-goal states with outputs of different steps in the algorithm, as summarized in Table 1 and as marked in Fig. 1. Note that since goal states are stable states, the goal state corresponding to G1 is step 16 in the process model and not step 3. Note also that the soft-goal in this case is of a discrete nature (i.e., goal state desirable or undesirable), whereas in other cases it may relate to continuous values.

Table 1. Goal mapping to Flu vaccination algorithm steps (algorithm represented in Fig. 1)

Goal states	Goal state definition	Desirable/Undesirable (soft-goal accomplishment level)	Step Outputs
G1	Identify an eligible child	Desirable	16
		Desirable (or normally expected)	3
G2	"Provide flu vaccination to an eligible child".	Desirable	16
		Undesirable	7
		Undesirable	15

Process states identification

Mapping intermediary process states is done in the same way as process goal states. Each state is identified by a set of variables, as we demonstrate in Table 2 (we present here only state variables that are changed within each state, not the whole set of variables associated with all process states). Note that the process has several states that represent goal states (S2, S6, S11 and S14); process soft-goals are evaluated through the desirability of these goal states (see Table 1).

Our Learning approach is based on collecting data from process instances and evaluating the process execution based on them. In clinical applications, the electronic medical record (EMR) may be used for documenting patient-related process-flows within the clinical system, providing a full tracking of the process state. Obviously, the EMR should include all relevant state variable data of the current and past process states. This has two major outcomes:

(1) The EMR becomes our major data base not only for patient data, but also for process flow information.
(2) We establish important foundations for a formal definition of the data required for process state tracking.

Note that data to be included in the EMR need to reflect not only the current state of the process but also enable the user to have a clear view of the process history. This is a challenge that our approach is capable of solving through the described state mapping- the inherent state variable based state definition. The described example shows how we can assure process tracking by mining the specified state variable data.

Table 2. Business Process states mapping for the vaccination algorithm. State types may be S-stable, U-Unstable or G-Goal. The state description indicates (in parentheses) the steps whose outcome corresponds to these states. Note that the outcome of several steps correspond to the same state. In addition, not all steps necessarily cause a state transition.

ID	**State name (corresponding step outcome)**	**Type**	**State variables update**	**Next state**
S0	Initial state (1).	S	--	S1, S2
S1	Checking Vaccine contraindications (2).	U	Eligible flu V. = Yes.	S3
S2	Contraindications present (3).	G	Eligible flu V. = No.	--
S3	Patient notified-overdue vaccine (4)	S	Flu V. Status= eligible; Notified=Yes.	S3, S4
S4	MD discusses Vaccine with Patient (5)	U	Flue V. MD Check=done.	S5, S6
S5	MD checks Patient (6,8,8b)	U	Flue V. MD checking = Yes.	S7, S8,S13
S6	Patient refuses vaccine (7).	G	Flu V. Status = Refused.	--
S7	Parent asked to buy part 1 of Vaccine (11,12).	S	Flu V. Status= "waiting - part 1"	S7, S9
S8	Temporal contraindications present (9,9b,10,10b)	U	Flu V. Status= "rescheduled-contraindications".	S7, S8, S14
S9	First part vaccine administered (13)	U	Flu V. Status= "1st part administered".	S10, S11
S10	Patient Notified- part 2 within 1 month (11b).	S	Flu V. Status= "waiting for part 2".	S10, S12
S11	Adverse event present reported (14,14b,15)	G	Adverse event = Yes; Adverse event report=<…>.	--
S12	Parent buys part 2 Vaccine (12b)	U	Flu V. Status= "Prepared for part 2".	S5
S13	Part 2 vaccine administered (13b)	U	Flue V.Status= "Part 2-completed".	S11, S14
S14	Flue vaccination completed (16)	G	Flu V. Status= "completed".	--

Process Law specification:
The process law specifies the possible transitions within the process flow. We mapped the possible transitions for each state in Table 2-"Next state" column. Transitions may be triggered by internal events (i.e., part of the process model) or external events (i.e., events that are generated from outside the process domain). According to GPM, internal events are sufficient for triggering a state transition from an unstable state, whereas a transition from a stable state requires an external event. In Table 2, we establish the state type for each state ("Type" column).

3.3 LPM in the Case Study

We can now demonstrate how learning can be performed.

Outcome data assessment: After the first year of executing the local process, the following conclusions were drawn at the implementing institution:

(1) The number of parents that were willing to vaccinate their children is high (state S5 reached).
(2) The number of patients that completed their vaccination was smaller than those who were willing to vaccinate. This was judged by an exception that occurred along different instances: the process did not reach a goal state, but remained stuck in step 12b (state S12).

Identifying possible causes through execution and external data analysis: Further investigation showed that some of the parents who consented to vaccinate their children could not complete their vaccinations due to shortage of the vaccine in the market. This is due to the logistical process in pharmacies, where inventory dependencies were not established between first and second portions of the vaccines. The shortage of vaccines invalidated the whole vaccination process for these patients. That is, the actual process flow was different from the model of the local process flow: step 8b (in Fig. 1) was not always reached.

GPM-based analysis: In GPM terms, state S5 was not reached for the second time, and the process was "stuck" in state S12. Surprisingly, S12 was identified as an unstable state. This implied that one of the following was occurring:

Option (1): S12 is a stable state and an external event is not being delivered, or,
Option (2): the model is missing a state between states S12 and S5 (steps 12b and 8b).

Considering the vaccine purchase within the process domain, option (2) is the valid one for the current case. The vaccine shortage is a newly discovered stable state. Note that we could consider the market as being in the environment of the process domain. Then S12 would be a stable state, awaiting an external event (vaccine purchasing).

Suggesting and evaluating possible modifications to the process model: First, the learning process can modify the modeled local system to reflect the actual flow, turning step 12 and 12b into decision points and adding a new flow into a new step 17 "market shortage" in addition to the normal flow into step 8b. In GPM terms, this results in a new undesired stable state. However, this modification by itself did not solve the problem, as parents did not wish to vaccinate their other children after their bad experiences. Nevertheless, the modification made the problem explicit and understood, so two manual solutions could be proposed:

(1) Modifying the external environment processes in order to eliminate this exception. In our case, the inventory management processes of the clinic's pharmacy were not controlled by the clinical teams and therefore this solution was not feasible. External pharmacies were willing to adapt their processes only if the clinic could give them exclusivity agreements, which contradicted current legislations.
(2) Modifying the internal process flow to eliminate the stable state. To do so, the clinic required parents to purchase both immunization portions before administering the first portion. The second portion would be stored at the clinic until its administration, ensuring that the immunization process will be completed and thus eliminating the undesirable situation of portion 2 shortage. The modified process, including the automatically learned state and manual modification, is presented in Fig. 2. Note that shortage is still possible, but either a patient gets both portions, or none.

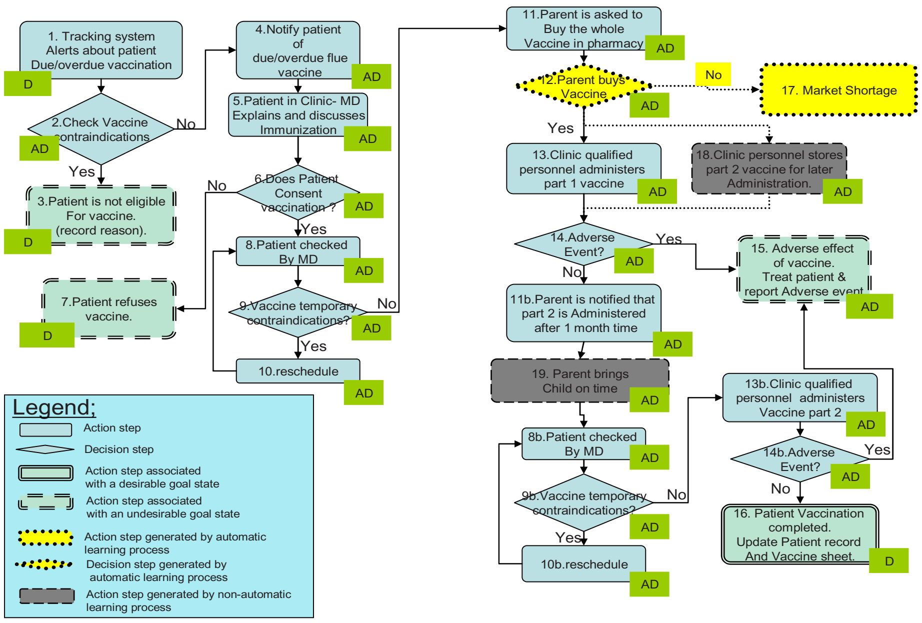

1. Tracking system Alerts about patient Due/overdue vaccination
D
2.Check Vaccine contraindications
AD
No
Yes
3.Patient is not eligible For vaccine. (record reason).
D
4.Notify patient of due/overdue flue vaccine
AD
5.Patient in Clinic- MD Explains and discusses Immunization
AD
6.Does Patient Consent vaccination ?
AD
No
Yes
7.Patient refuses vaccine.
D
8.Patient checked By MD
AD
9.Vaccine temporary contraindications?
AD
No
Yes
10.reschedule
AD
11.Parent is asked to Buy the whole Vaccine in pharmacy
AD
12.Parent buys Vaccine
AD
No
17. Market Shortage
Yes
13.Clinic qualified personnel administers part 1 vaccine
AD
18.Clinic personnel stores part 2 vaccine for later Administration.
AD
14.Adverse Event?
AD
Yes
No
15. Adverse effect of vaccine. Treat patient & report Adverse event
AD
11b.Parent is notified that part 2 is Administered after 1 month time
AD
19. Parent brings Child on time
AD
8b.Patient checked By MD
AD
9b.Vaccine temporary contraindications?
AD
No
Yes
10b.reschedule
AD
13b.Clinic qualified personnel administers Vaccine part 2
AD
14b.Adverse Event?
AD
Yes
No
16. Patient Vaccination completed. Update Patient record And Vaccine sheet.
D
Legend;
Action step
Decision step
Action step associated with a desirable goal state
Action step associated with an undesirable goal state
Action step generated by automatic learning process
Decision step generated by automatic learning process
Action step generated by non-automatic learning process

Fig. 2. Flu vaccine process modified following the learning and external changes. Note the automatic learning process has generated a new process step (step 17- market shortage) and detected a transition to it from step 12 (converted to a decision step). Note also the effect of non-automatic learning - elimination of step 12b (Parent buys part 2 vaccine) and the addition of steps 18 and 19.

4 Discussion

We illustrated how BP learning can be established based upon evaluation of soft-goals and exceptional states in process instances, and proposal of changes (manual, automatic, or semi-automatic) to the process model in order to minimize exceptions and to improve soft-goal accomplishment levels.

Examining the literature, we found several process model adaptation approaches proposed in the general BPM field [2, 4, 17, 18] as well as in medical guidelines modeling [2, 6]. Different process mining methods have been already discussed and applied in the BPM field [1, 4, 17] and in the clinical guidelines research [5, 19]. Comparison of a defined process model to the actual one discovered has been proposed [7, 19], as well as a statistic-based approach to propose ad-hoc changes both at process instance level an at process model levels [2, 3]. However, as far as we know, the examined literature is focused on understanding the actual control flow, without relating it to the process outcomes. Furthermore, it does not provide any methodology for defining the data needed for extending process mining to address both the process and its outcomes**.** Examining the process states definition of Table 2, we can easily identify the minimum EMR data needed to identify the process current state and history at any moment. Moreover, as state transitions cause changes in a subset of the domain state variables, tracking these state variables is sufficient for tracking the process state. Hence, the LPM can provide the data model required for mining a more holistic view of the actual process through the identification of the required state variables and the according documenting process. This is independent of which process mining algorithm is used and whatever process runtime model is used. We consider this to be a considerable advantage of LPM.

Learning is an evolutionary process, whose initial model is presented in this paper. As we execute the process we may face several situations which the learning process must address:

(1) Incomplete process model specification, i.e., unidentified states, incomplete state definitions, missing external events, and potential exceptions that invalidate the process execution.
(2) Causality relationships of selected process paths and achieved soft-goal values.
(3) Progress of new knowledge in the field, novel technology, or process innovations, which require changing the process

We must aim to provide the required changes at a process model proactively. To this end, LPM proposes a learning lifecycle-model based upon the process-goal approach. The contributions that we have made so far include: (a) A formal business process model-based approach that identifies the process execution state and how it attains its goals at any time. (b) A methodology for identifying what data sets need to be collected as part of process mining in order to enable process-learning. (c) A learning approach that introduces changes to the process model based upon identified exceptions, missing external events, and attainment of goals and soft-goals (the data sets described in (b)). The learning demonstrated in this paper relates to exception occurrences, while learning in general should also relate to soft-goal attained as a result of path selection. This will require the application of learning algorithms, which we intend to develop as future research. Note that while some learning may be done automatically, the modification of the business process may require, in no-error tolerant applications (such as health care), some sort of expert confirmation before applying the modification. This does not contradict our approach.

Further work must be done to analyze potential automatic updates to the business process model. Another issue yet to be investigated is how knowledge integration could be accomplished based upon LPM.

References

1. vd Aalst, W.M.P., Guenther, C., Recker, J., Reichert, M.: Using Process Mining to Analyze and Improve Process Flexibility - Position Paper. In: BPMDS 2006. Proceedings 18th International Conference on Advanced Information Systems Engineering Workshops, pp. 168–177 (2006)
2. Reichert, M., Dadam, P.: ADEPT-flex: Supporting dynamic changes of workflows without losing control. JIIS 10(2), 93–129 (1998)
3. Rinderle, S., Reichert, M., Dadam, P.: Correctness criteria for dynamic changes in workflow systems - a survey. Data & knowledge Engineering 50, 9–34 (2004)
4. Weske, M.: Formal foundation and conceptual design of dynamic adaptations in a workflow management system. In: Proceedings of the 34th Annual Hawaii International Conference on System Sciences, p. 7051 (2001)

5. Quaglini, S., et al.: Flexible Guideline-based Patient Careflow Systems. Artificial Intelligence in Medicine 22, 65–80 (2001)
6. Peleg, M., Kantor, R.: Approaches for guideline versioning using GLIF. In: Proc. AMIA Symp., pp. 509–513 (2003)
7. Peleg, M., et al.: Comparing Computer-Interpretable Guideline Models: A Case-Study Approach. J. Am. Med. Inform. Assoc. 10(1), 52–68 (2003)
8. Advani, A., Lo, K., Shahar, Y.: Intention-Based Critiquing of Guideline-Oriented Medical Care. In: Proc. AMIA Annual Symposium, pp. 483–487 (1998)
9. Soffer, P., Wand, Y.: On the Notion of Soft Goals in Business Process Modeling. Business Process Management Journal 11(6), 663–679 (2005)
10. Soffer, P., Wand, Y.: Goal-driven Analysis of Process Model Validity. In: Persson, A., Stirna, J. (eds.) CAiSE 2004. LNCS, vol. 3084, pp. 521–535. Springer, Heidelberg (2004)
11. Bunge, M.: Treatise on Basic Philosophy: Ontology II: A World of Systems. Boston, Reidel, vol. 4 (1979)
12. Bunge, M.: Treatise on Basic Philosophy. In: Ontology 1: The furniture of the world, Boston, Reidel, vol. 3 (1977)
13. Wand, Y., Weber, R.: An Ontological Model of an Information System. IEEE Transactions on Software Engineering 16(11), 1282–1292 (1990)
14. Wand, Y., Weber, R.: On the Ontological Expressiveness of Information Systems Analysis and Design Grammars. Journal of Information Systems 3, 217–237 (1993)
15. Oppenheim, A.V., Willsky, A.S.: Signals & Systems. Prentice Hall, Englewood Cliffs (1997)
16. Institute for Clinical systems Improvements. In: Immunization, 11th edn. (2006), http://www.icsi.org/Immunizations
17. Weber, B., Reichert, M., Rinderle, S., Wild, W.: Towards a framework for the agile mining of business processes. In: Bussler, C.J., Haller, A. (eds.) BPM 2005. LNCS, vol. 3812, Springer, Heidelberg (2006)
18. vd Aalst, W.M.P., et al.: Workflow Mining: A Survey of Issues and Approaches. Data and Knowledge Engineering 47(2), 237–267 (2003)
19. Guth, V., Oberweis, A.: Delta analysis of Petri-net based models for business processes. In: Proceedings of Applied Informatics, pp. 23–32 (1997)

Mining Process Execution and Outcomes – Position Paper

Mor Peleg, Pnina Soffer, and Johny Ghattas

Department of Management Information Systems, University of Haifa, Israel, 31905
{morpeleg, pnina}@mis.hevra.haifa.ac.il

Abstract. Organizational processes in general and patient-care processes in particular, change over time. This may be in response to situations unpredicted by a predefined business process model (or clinical guideline), or as a result of new knowledge which has not yet been incorporated into the model. Process mining techniques enable capturing process changes, evaluating the gaps between the predefined model and the practiced process, and modifying the model accordingly. This position paper motivates the extension of process mining in order to capture not only deviations from the process model, but also the outcomes associated with them (e.g., patient improving or deteriorating). These should be taken into account when modifications to the process are made.

Keywords: Process mining, business process modeling, Clinical guidelines.

1 Introduction

Organizations often try to improve the quality of services provided to their clients by specifying a business process model (BPM) that captures the desired workflow in the organization and how exceptional situations should be handled. The aim is that if these process models would be implemented, the quality of services would increase and costs would be saved. In the healthcare domain, special attention has been given to creating evidence-based clinical guidelines that recommend care processes for patients with defined clinical conditions. Research has been carried out in developing methodologies and tools for specifying guidelines as computer-interpretable algorithms [1], linking them to clinical databases, executing them, and evaluating the impact of these systems [2].

However, organizational processes in general, and patient-care processes in particular, change over time. In real life, we often encounter situations that we neglected to consider in our BPM (e.g., guideline model). These situations could require process paths that were not specified in the BPM. In addition, expanding medical (or in general, organizational) knowledge on new procedures, available treatment options, and evidence for their effectiveness, may push users and organizations into changing their process implementations, often before updating the BPM. Thus, we may find that the actors participating in the business processes (patient-care processes) often do not follow the BPM exactly and may act differently than the model specifies, which may or may not be justified or helpful.

Deviations of the actual performed process from its model have been studied both in the BPM community and in the Medical Informatics community. In the BPM community, process mining has been used to capture process changes, evaluate the gaps between the predefined BPM and the practiced process, and modifying the BPM accordingly [3, 4].

In the Medical Informatics community, critiquing approaches [5-7] have been used to compare the actual processes executed to their specified model. Advani et al. [7] describe a model and algorithm for deriving structured quality indicators and auditing protocols from formalized specifications of guidelines used in decision support systems. This critiquing approach can be used to determine whether the deviations followed the intentions of the original model and thus were justified. Traum-AID [6] is a rule-based system combined with a planner. Its critiquing interface examines actions the physician intends to carry out, identifies errors and calculates their significance, and produces a critique in response to those intentions. In the 1980's, Miller [5]developed several critiquing systems in which the physician inputs medical information describing a patient, a current set of test results, and current actions (e.g., ventilator settings), and a proposed set of new actions. The system assesses appropriate treatment goals, and uses those goals for critiquing.

Similar analysis can be used to offer decision-support to physicians only when they deviated from these intentions. Quaglini et al. [8] developed computerized guideline implementations that allow users not to follow al the actions specified in the model, justify the deviation, and select alternative activities out of a wide range of activities that were not planned by the guideline authors but that are related to the original alternative via hierarchies taken from standard clinical vocabularies. Similarly, many computerized guideline formalisms have tools that allow the user to deviate from the normal sequence of activities [9], as flexibility is often needed when the modeled guideline is to handle emergency situations in patients or when the model is out of date and does not convey the latest medical knowledge. Peleg and Kantor [10] used process mining to automatically analyze differences between two versions of process models that were created due to the expansion of medical knowledge, in order to find differences in particular medical knowledge concepts (e.g., new drug) or in concept relationships (e.g., pathogen is not longer believed to cause a disease).

As we strive to improve our BPMs (patient-care models), we must not only track deviations from the process model, but also the outcomes associated with them (e.g., patient improving or deteriorating) so that these could be taken into account when modifications to the process are made.

2 Background

In order to follow process outcomes and relate them to changes in the process, we need a formal process model that can represent goals and outcomes. We rely on the Generic Process Model (GPM) proposed by Soffer and Wand [11].

2.1 The Generic Process Model (GPM)

GPM is a state-based view of a process including the concept of goals. Briefly, GPM offers a process model which is composed of a quadruple <S, L, I, G>, where S is a set of states representing the domain of the process. Each state in an enacted process holds the values of all the properties (or state variables) of the process domain at a moment in time. The law L specifies possible state transitions as mapping between subsets of states, defined by conditions over values of the domain state variables; I is a subset of unstable states, which are the initial states of the process after a triggering external event has occurred; G is a subset of stable states on which the process terminates, termed the goal of the process.

The process goal as addressed by GPM is a state meeting the conditions that should be achieved by the process. GPM distinguishes process goals from soft-goals, which are defined as an order relation on goal states [12]. In other words, soft-goals relate to the *desirability* of possible states in the goal set (all meeting the condition that terminates the process) according to defined business objectives. For example, the goal of a process may be a state where some treatment has been given to a patient, but a state where the treatment does not incur side effects is considered as "better" than a state where side effects are observed. Finally, GPM entails criteria for assessing the validity of a process, namely, its ability to achieve its goal [11]. It enables the analysis of a process to identify causes for invalidity, and suggests appropriate redesign actions to eliminate these causes.

For operational and representational purposes, GPM's law can be mapped to Petri Nets [13]; states correspond to sets of places of the Petri Net and laws correspond to transitions (including transition guards). Complementing this representation, GPM's clear distinction of goals and soft-goals can form a basis for improving a practiced process, where improvement can be related to attained soft-goal values and to fewer situations where the goals of the process are not met.

3 Research Objectives

Objective 1: Develop a method for establishing process data on which outcome and goal analysis will be based. This includes patient-specific data referred to by the process model (e.g., age), data about activities that were started and completed, and data regarding outcomes, judged by relevant soft-goal attainment.
Objective 2: Develop methods and tools for analyzing process data to identify relationships between patient-specific data, execution paths, process goals, and outcomes.

4 Demonstration of Our Approach

To demonstrate and motivate our approach, we use a process model based on a guideline for treatment of ear infections (acute otitis media, AOM) [14]. Figure 1 shows a Petri Net of the process model adapted from that guideline. Such a Petri Net can be automatically converted from a GLIF algorithm [15].

In order to have a measure of process attainment of soft and hard goals, and also of exceptions, where goals are not met, we need to analyze and mine the execution log of the actual process, or their reflection in electronic medical records (EMRs), to document for each work item (i.e., an activity performed by an actor on a given case). In addition to the existing process mining ability to identify that an activity has been

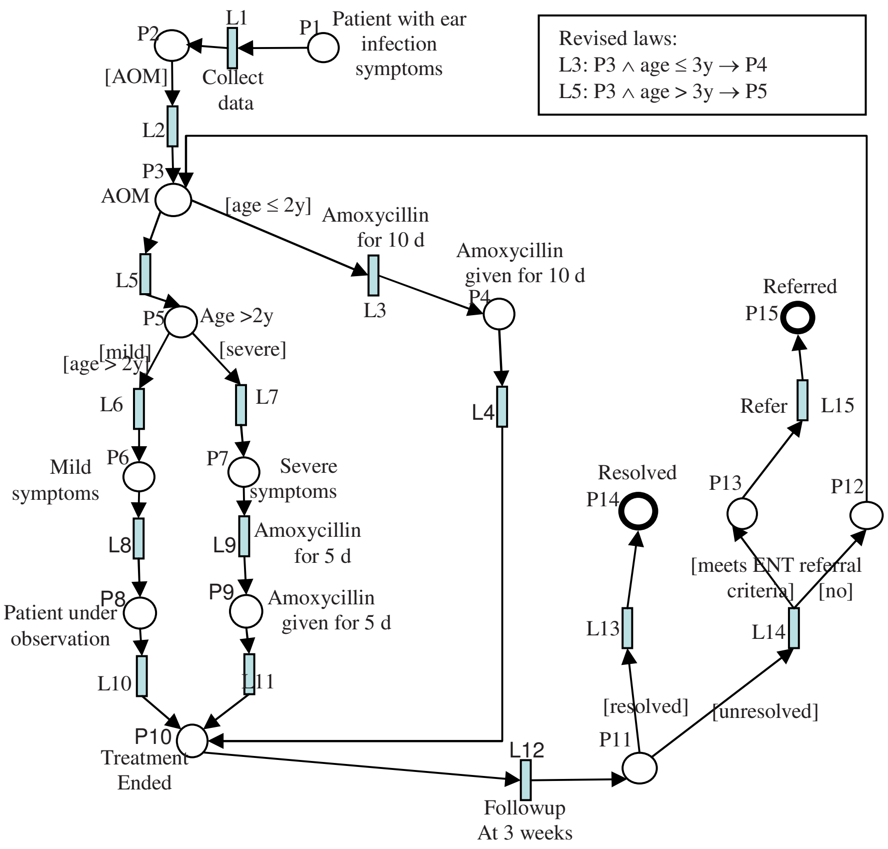


Fig. 1. A Petri Net representation of an ear infection clinical algorithm. Places, corresponding to GPM states, are marked as Pi. Transitions, corresponding to GPM laws, are marked as Li. P14 is the desired goal state, P15 is the undesired goal state, and P1 is the initial state.

performed by an actor at a certain time, we also need to mine changes to state variables (e.g., a patient's temperature or his adverse response to a drug) that were not predicted by the process model, and their timestamp. These data could help us in relating activities (both those that followed the process model and those that represent changes) to outcomes. Table 1 presents a potential EMR of a 2.4 year-old patient reflecting the ear-infection process instance as well as outcomes. As can be seen, the physician first followed the guideline, prescribing Amoxycillin for 5 days, as the patient was over 2 years old. But, when AOM was not resolved and the goal state was not reached, he decided to prescribe a 10-day Amoxycillin, which was not according to the guideline. This time, the goal state was reached. Analysis of EMR data of other patients showed that in many of the 2-3 year old patients, AOM was not resolved after 5-day treatment. These relationships between goals and outcomes could suggest ways to improve the clinical process. For example, change the laws L3 and L5 such that patient under 3 (not under 2) would receive a 10-day treatment (Figure 1).

Table 1. EMR reflecting an ear-infection treatment process and outcomes

Time-Date	State variable	Value	Petri Net place
07-01-01:08:00	AOM	mild	P6
07-01-01:08:10	Medication	5-day Amoxycillin	P9
07-01-21:08:00	AOM	mild	P6
07-01-21:08:10	Medication	10-day Amoxycillin	P4
07-02-12:08:00	AOM	false	P14

5 Future Work

While the above ear infection scenario demonstrates the potential contribution of process execution and outcome analysis, a systematic method for such analysis is still under development. We are currently working on a hypothetical case study, taken from the clinical guideline domain of vaccinations, which examines actual execution or processes (as determined from synthetic EMR records). We will study how we can automatically deduce from the EMR records whether instances of the process have attained the process model's soft and hard goals, and when and how to characterize exceptional situations. We would like to use data about real process execution and outcomes (log files) along with the preconceived process models to test whether our approach could be used to automatically assess attainment of soft and hard goals as well as assess the occurrence of an exception (i.e., the invalidation of the process due to an unexpected event), resulting in processes that do not meet their goals and remain in intermediate states. We would then like to combine outcomes analysis with delta analysis for populations of patient with similar characteristics to suggest a linkage between process changes and process outcomes.

The main challenge we are facing is how to establish a causal relationship between the execution data (or delta analysis) and the obtained outcomes. The outcomes of clinical processes are determined not only by the actions taken, but also by pre-existing patient properties, such as age in the ear infection example. The analysis should take these properties into account as affecting variables, and provide recommendations with respect to specific patient properties.

References

1. Peleg, M., et al.: Comparing Computer-Interpretable Guideline Models: A Case-Study Approach. J. Am. Med. Inform. Assoc. 10(1), 52–68 (2003)
2. Garg, A.X., et al.: Effects of Computerized Clinical Decision Support Systems on Practitioner Performance and Patient Outcomes: A Systematic Review. JAMA 293(10), 1223–1238 (2005)
3. van der Aalst, W.M.P., et al.: Workflow Mining: A Survey of Issues and Approaches. Data and Knowledge Engineering 47(2), 237–267 (2003)
4. Reichert, M., Dadam, P.: ADEPT-flex: supporting dynamic changes of workflows without loosing control. JIIS 10(2), 93–129 (1998)
5. Miller, P.L.: Goal-directed critiquing by computer: ventilator management. Comput. Biomed. Res. 18(5), 422–438 (1985)

6. Gertner, A.S.: Plan Recognition and Evaluation for On-line Critiquing. User Modeling and User-Adapted Interaction 7(2), 107–140 (1997)
7. Advani, A., Goldstein, M., Shahar, Y., Musen, M.A.: Developing Quality Indicators and Auditing Protocols from Formal Guideline Models. In: AMIA Annual Symp., pp. 11–15 (2003)
8. Quaglini, S., et al.: Flexible Guideline-based Patient Careflow Systems. Artificial Intelligence in Medicine 22, 65–80 (2001)
9. Peleg, M.: Guideline and Workflow Models. In: Greenes, R.A. (ed.) Clinical Decision Support - The Road Ahead, Elsevier/Academic Press (2006)
10. Peleg, M., Kantor, R.: Approaches for guideline versioning using GLIF. In: Proc. AMIA Symp., pp. 509–513 (2003)
11. Soffer, P., Wand, Y.: Goal-driven Analysis of Process Model Validity. In: Persson, A., Stirna, J. (eds.) CAiSE 2004. LNCS, vol. 3084, pp. 521–535. Springer, Heidelberg (2004)
12. Soffer, P., Wand, Y.: On the Notion of Soft Goals in Business Process Modeling. Business Process Management Journal 11(6), 663–679 (2005)
13. Peterson, J.L.: Petri Net Theory and the Modeling of Systems. Prentice-Hall, Englewood Cliffs (1981)
14. Institute for Clinical Systems Improvement. Diagnosis and Treatment of Otitis Media in Children (2004)
15. Peleg, M., Tu, S., Mahindroo, A., Altman, R.: Modeling and Analyzing Biomedical. In: Processes using Workflow/Petri Net Models and Tools. MedInfo., pp. 74–78 (2004)

Reference Model Workshop

Introduction to the 10th Reference Modeling Workshop

Reference models are conceptual information models that are developed with the aim of being reused for different but similar proposes. Therefore, they are recommendatory artifacts that can be used to guide a modeler in a conceptual modeling task. Often, reference models claim to represent “best practice” or “common practice” solutions for certain business domains. The domains that are addressed by a reference model can differ extensively.

Due to their claim of representing “best practice” or “common practice” solutions, the main objective of reference models is to realize cost savings in the construction process of purpose-specific models on the one hand. Through reuse of business knowledge stored in the reference models, they accelerate the modeling process. On the other hand, modelers aim at increasing the quality of their models through reuse of reference models, since these claim to contain proven concepts.

The topic of the contributions to the 10th International Workshop on Reference Modeling (RefMod 2007) is current findings on information systems and computer science research dealing with diverse aspects of reference modeling. Academic contributions which provide methodological and contextual recommendations for the development and application of reference models are discussed:

CHRISTIAN JANIESCH and ARMIN STEIN state that the transition of “good practice” models to reusable reference models is problematic due to a lack of standardization. In their Contribution “Adapting Standards to Facilitate the Transition from Situational Model to Reference Model,” they propose an approach that allows for redesigning such “good practice” models using standards in order to make them generally reusable. Since each reference model is aimed at being reused, it should be possible to adapt reference models to the requirements of different user groups. In their contribution “Linking Domain Models and Process Models for Reference Model Configuration,” MARCELLO LA ROSA, FLORIAN GOTTSCHALK, MARLON DUMAS and WIL M. P. VAN DER AALST introduce an approach in order to align reference models with different requirements by configuring them based on a domain model-related questionnaire. A methodology towards “Reference Modeling for Higher Education Budgeting” is put forward by JAN VOM BROCKE, CHRISTIAN BUDDENDICK and ALEXANDER SIMONS. They suggest reusing a modeling language for OLAP reporting in order to support budgeting in higher education. As events are becoming more and more important for companies as an instrument of marketing communication, OLIVER THOMAS, BETTINA HERMES and PETER LOOS introduce a “Reference Process Model for Event Management” in order to provide recommendations for software development and business process management in the field of event management.

The contributions of the workshop have been selected in a rigorous, double-blind peer-review process. We would like to thank all authors, organizers and involved persons that made the workshop and this book section possible. Finally, we thank the organizers of BPM 2007, who provided a professional and competent environment for RefMod 2007.

November 2007

Jörg Becker
Patrick Delfmann

Adapting Standards to Facilitate the Transition from Situational Model to Reference Model

Christian Janiesch and Armin Stein

European Research Center for Information Systems, Leonardo-Campus 3, 48149 Münster, Germany
{janiesch,stein}@ercis.uni-muenster.de

Abstract. Reference Models comprise information on best or common practices for reuse. However, they commonly originate from individual projects. Yet, project models are usually situational and often inconsistent due to a lack of expertise of the employees or the sheer complexity of the specification. Consequently, the transfer of these models into universal reference models is problematic at best. Standards are unified and generally accepted means of conducting or producing something. Thus, they provide a means to transport and interface knowledge of different parties. Since standards tend to be exhaustive, they commonly provide more than is needed for any given situation. We propose to adapt standards to provide a view that is better suited for a task at hand. In this way, standards can help to unify situational models and provide assistance for the transition to reference models.

Keywords: Reference Modeling, Standardization, Model Adaptation.

1 Introduction

Reference models are generic conceptual models that formalize state-of-the-art or best practice knowledge of a certain field [9, 22]. They are of normative nature and can cover different domains such as industry sectors or functional areas. Their universal applicability not only enables researchers and practitioners to use them as guidelines for comparing and validating specific existing processes. They also enable them to derive company specific models. Furthermore, it empowers them to reuse the content.

Numerous approaches have been proposed to construct reference models from scratch and to continuously improve and extend those [e.g., 5]. However, most information stored in reference models does not come from an unbiased source extracted by extensive double blind research, but is of rather practical origin. The results of multiple projects conducted in the real world offer a very detailed insight into the common and best practices of a domain. But these results originating from singular projects comprise situational information, which can distort the underlying idea of universality. This is neither wrong nor objectionable but has to be taken into consideration, when constructing and offering a reference model based on originating from singular projects to the public. In order to limit this *noise* to a minimum, several precautions can be taken.

Standards are agreed-upon specifications for a way of communicating or performing actions [8, 11]. Consistently applied, they produce artifacts, which can be reused by a broader audience than any situational artifact. Hence, it is preferable to utilize standardized methods (often also termed techniques) in terms of notation and procedure as well as standardized expressions for the design of models. Models in general are not created for the use of a single individual, but to be used by a larger amount of persons. Using Standards for creating these models supports this idea. As stated above, reference models are generalized models including best or at least common practices. Therefore, a specific model has to be accepted and agreed upon by multiple subjects to fulfill the requirement of generalization. Consequently, the use of standards for the design of situational models is as important as these models provide potential input for future (revisions of) reference models.

As with reference models, standards also tend to be extensive. In a real world project the standard of knowledge is often considerably lower than desirable so that cutbacks have to be made [e.g., 2]. Effectively, this rules out the utilization of anything more than moderately complex modeling conventions. Adaptive reference modeling is an approach, which provides a means to cope with the complexity of reference models [1]. Similarly, we propose to tackle the problem of extensive and complex standards by adapting standards as proposed by Becker et al. This is meant to provide a better methodological basis for constructing reusable project models.

Combining these two research areas – standardization on the one hand and adaptive reference modeling on the other hand – enables the selection of the most suitable sets of standards to enable the creation of future reference models. This research is not only the basis for a contribution to practitioners, to whom it facilitates the generation of reference models, but also to standardization organizations, who thus are able to better promote their standard.

The paper is structured as follows: In Section 2, we provide a brief introduction to (reference) model adaptation. Additionally, we introduce and cluster standards in information system (IS). We illustrate the clusters with examples in Section 3. Subsequently, we introduce a framework to adapt these standards to construct reference-model-ready situational models. In Section 5 we exemplify our proposal with an ex post project analysis. The paper closes with concluding remarks and an outlook to further research.

2 Related Work

2.1 Adaptive (Reference) Modeling

One of the objectives of reference models is to simplify the creation of company specific models [for an overview cf. e.g. 10]. They support this activity by providing a template from which knowledge about common business processes and organizational structures can be reused. Similarly, other artifacts such as standards or methods can be adapted [cf. 3]. To facilitate this reuse several distinct mechanisms have evolved: configuration, instantiation, specialization, aggregation, and analogy construction [1].

Models can be designed as *configurable* artifacts. They are provided with specific attributes that contain configuration rules. Depending on the values assigned to the condition part of the rule, it can be decided whether the conclusion part of the rule has to be executed. The conclusion part can imply consequences such as the elimination of model elements or the modification of their representation. With this mechanism model variants regarding application-specific characteristics can be created in an automated manner. In the context of this paper, configuration can be used to support the selection as well as tailoring of relevant standards for a certain task.

Configuration rules can be specified for standards chosen from a repository. Each is annotated with an equation such as `context = ISO_9001_compliant_process_documentation`, which leads to the selection of all tagged standards for the context of process documentation following ISO 9001. As stated above, each standard itself can also be adapted by the means of configuration. To do so, terms have to be attached to elements of the standard's meta model, enabling the removal of elements in dependency of the selected application context.

Through *specialization* a specific model is derived from a more general model by adapting, extending and/or partially modifying the more general one. For this purpose, the model is annotated with specialization instructions. Reference models which support specialization usually have a higher level of abstraction than the resulting company specific models. Concerning standards specialization, this is of inverse nature as the specialized standards comprise a lesser amount of items.

Reference models which support *aggregation* are not available as monolithic blocks but rather contain independent parts, so called components or fragments. Through aggregation, a specific artifact is built by assembling these components. Interface descriptions of the components can offer information on their general compatibility and on how to combine them. Aggregation can be applied to use multiple standards concurrently.

Instantiable models are equipped with placeholders. The placeholders are inserted during the construction of the model and annotated with an instantiation domain. When a specific model is created, the placeholders are filled with valid occurrences.

Analogy construction proposes to adapt models to other areas of application by conclusion by analogy. This mechanism is also common with patterns and does only provide vague guidance.

Out of these five mechanisms configuration and instantiation offer the most guidance for model adaptation. Specialization and aggregation can be used with a lesser degree of preparation. Adaptation by analogy construction is usually irreproducible [3]. In the context of the following, we prefer the mechanism of configuration. However, since no comprehensive configurative repository of standards exists yet, we give an example in Section 5 in which standards are aggregated and then specialized.

2.2 Standards in Information Systems

A standard describes a unified and agreed-upon specification for a way of conducting or producing something [8, 11]. It is commonly associated with the methods that are applied. One can distinguish proprietary standards of manufacturers such as the different flash card formats for digital cameras, industry standards, which act as de facto standards such as the ASCII code or the universal serial bus specification (USB), and open standards, which provide open interfaces for all market participants.

In IS, technical standards often form with industry involvement through a series of drafts that are approved by a standardization group. The hurdles for involvement and rules for ratification differ from group to group. Standard-making is not only a technical but rather a socio-ecological process [18]. Examples for prominent IS standardization bodies are the World Wide Web Consortium (W3C), the Organization for the Advancement of Structured Information Standards (OASIS), the United Nations Centre for Trade Facilitation and Electronic Business (UN/CEFACT) or the Object Management Group (OMG). For the purpose of our research, we cluster standards in three groups:

- *notational standards*, which describe methods of documenting things,
- *business semantics standards*, which comprise standardized terms of domain knowledge to do so, and
- *procedural standards*, which describe a way of conducting things.

These clusters emerge from a top-down view, substantiating the business context in three phases, which are going to be described in more detail in the Section 4. What is sometimes otherwise categorized as product or technical standards [17], we perceive as a notational standard as it determines the appearance of the modeled artifact. Naturally, these clusters cannot be without overlaps as our choice of examples shows.

In the following section we will exemplify these three standard groups by the description of the Business Process Modeling Notation 1.0 (BPMN), which is a modeling method for processes [19], the Universal Business Language 2.0 (UBL) [6], which provides a business dictionary for order-to-invoice processes, and the UBL 2.0 Business Processes, which provide interaction patterns for the necessary message exchange. These standards are applied in the example in Section 5. The choice of the standards is not in the focus of this paper but has to be considered in future research. A viable approach may be derived from the procedure the International Organization for Standardization (ISO) uses to specify standards. By using a large group of experts discussing about a certain matter and getting the results critically reviewed by others, a consensus view is generated [13].

3 Exemplification of Standards Clusters

3.1 Notational Standardization with BPMN 1.0

A lot of effort has been made to develop modeling methods, which are capable of retaining knowledge concerning business processes. Different approaches, which emphasize on their different strengths, give the user a wide variety to choose from. Event-driven Process Chains (EPC), Petri Nets or the Unified Modeling Language (UML) Activity Diagrams are common examples. The Business Process Management Initiative (BPMI) Notation Working Group created the Business Process Management Notation (BPMN) as a best-of-breed, based on considerations of methods such as the above. The OMG ratified the approach as a final specification [19]. BPMN is widely seen as de facto standard in business process modeling [e.g. 25]. The distribution in day-to-day business underlines its important role and relevance.

BPMN is intended to enable model developers to create business process models easily and in a standardized way, giving them enough freedom to model real world scenarios. The representational aspect is covered by the business process diagram, which provides a manageable amount of symbols to visualize flow objects, connection objects, swimlanes, and artifacts. Nevertheless, BPMN provides a meta model describing how the elements may or may not be connected with each other. Besides the meta model, three rules have been set up, defining what is allowed as good modeling practice and what is not to preserve BPMN conformity. So, for example, the shapes of the symbols used have to stay the same except modifications of their color or their line style. BPMN is intended to support model users without expertise in business process modeling to understand, use, and discuss the models.

Finally, BPMN allows the direct transition of the models into code like BPEL4WS [16], which is executable by workflow management systems, and to derive respective XML structures for information exchange. Similarly, the generation of WSDL code is possible, enabling the system engineer to develop the underlying business logic in terms of e.g. web services. To support easy and flawless interchange between modeling and/ or meta modeling software, the BPMI demands that either of them comply with the not yet developed model exchange format for the generated models.

Figure 1 shows common participants, activities, and outcomes of a BPMN project. Each of the illustrated elements benefits from the usage of BPMN as a standard for business process modeling.

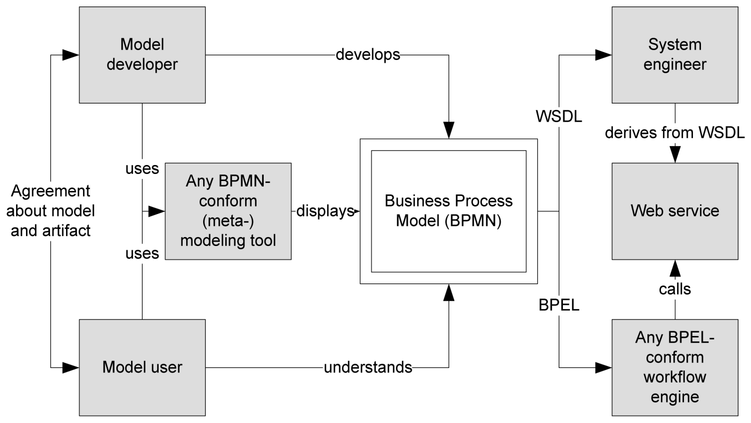

Model developer
develops
System engineer
WSDL
derives from WSDL
uses
Agreement about model and artifact
Any BPMN-conform (meta-) modeling tool
displays
Business Process Model (BPMN)
Web service
uses
calls
BPEL
Model user
understands
Any BPEL-conform workflow engine

Fig. 1. Participants, activities, and outcomes of BPMN development

3.2 Business Semantics Standardization with UBL 2.0

Application domain knowledge or subject matter expertise can be understood as the knowledge of the problem area addressed. Domain knowledge is a necessary condition in order to provide unambiguous business semantics for e.g. conceptual models. An intensively discussed measure to capture domain knowledge is provided by ontologies. Ontologies take the role of an explicit representation of domain knowledge [12]. An important motivation for employing ontologies is the hope to alleviate the subjectivism of different modelers. By means of underpinning and structuring the models with a shared conceptual vocabulary manifested in a domain ontology, the creation of models is facilitated, which can easier be transferred into reference models [cf. similarly 21].

Based on Core Components Technical Specification (CCTS) [7] and XML, UBL is a specification passed by the OASIS to define the exchange of standardized electronic documents for data elements connected with common business processes [6]. As traditional for OASIS, a lot of voluntary contributors like Adobe, SAP or Sun Microsystems work on the development of UBL, making it the only accepted approach to document-centric XML by UN/CEFACT. By the usage of dataset definitions for information and documents like transportation status, purchase order or invoice, companies are enabled to reliably and consistently exchange their data and integrate these into their workflow management systems or applications, independent from existing standards like UN/EDIFACT or RosettaNet.

As such, each of the 31 document types is described within a schema file, which can be instantiated as message. Thereby, core components act as building blocks for semantically correct documents to exchange meaningful information documents. UBL provides a uniform dictionary of terms to be able to unambiguously name every entity of an order-to-invoice process.

Several other standards have been proposed that provide concrete instantiations of CCTS. The Open Applications Group Integration Specification (OAGIS) is another major initiative that aims at providing the uniform language for business documents exchange based on CCTS. SAP's Global Data Types (GDT) also use CCTS for harmonizing the business information of their Enterprise System [23]. In the long term, one of the core benefits of these initiatives is the formation of uniform naming and design rules for business information exchange to provide business semantics, i.e. an ontology of business terms. In addition a comprehensive framework for their customization is also work in progress. The Unified Context Methodology Project (UCM) [24] aims at providing standardized parameters for the description of document variants.

3.3 Procedural Standardization with UBL 2.0 Business Processes

Procedural standardization can take place on two levels. Concerning project management it is necessary, to set a standard for project governance and approval processes. PRINCE2 is a prominent example for this task. But more importantly for the actual content of a project, procedural standards should be followed for the definition of the concrete processes. In this way, it is possible to define tasks and order them in a way that has common ground with other similar endeavors. If, e.g., all modeled variants for IT service management processes follow ITIL to some extend, their integration into one reference model would be eased considerably [20].

Concerning the latter, UBL does not only provide a standard for business semantics but also provides a set of common business processes [6]. 20 data exchange processes in conjunction with the respective actors are illustrated with case diagrams, which are then further detailed via UML Activity Diagrams. Similar to RosettaNet or OAGIS, these processes provide interaction patterns, which act as a procedural standard to unify the sequence of tasks in a process. UBL offers support for sourcing, ordering, fulfillment, billing, payment, transport, certification and reporting processes.

Abiding to these patterns promises a better interfacing with similar order-to-invoice variants of other modeling projects as similar tasks are performed when collaborating with other parties. These are suitable preconditions to nominate UBL as a standard for the design of better interoperable interaction processes.

4 A Framework for Standards Adaptation to Facilitate the Transfer from Model to Reference Model

Becker et al. propose a number of steps to be taken before starting a process modeling project [4]. If followed, these preparations lead to a set of conventions that allow for the design of comparable models which then can be reused more easily. With regard to the transfer of these situational project models to reference models – either as an addition or for the construction of a new reference model – we propose some extensions concerning standards integration.

Becker et al. distinguish several purposes which can be roughly categorized into organizational design, application system design, and communication design [2, 4]. The specific purpose does have implications on the requirements of the models as well as its possible output, e.g. presentation slides as opposed to data exchange interfaces. We propose to implicitly regard the purpose of reference model design as mandatory (*determination of context*). This entails the following (cf. Figure 2 for an overview of the framework):

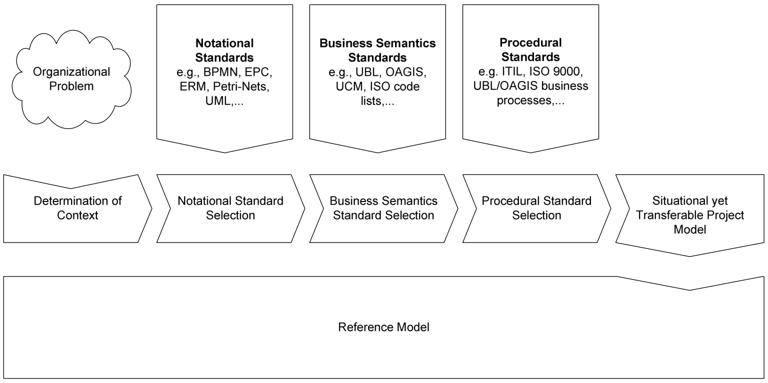

Organizational Problem
Notational Standards e.g., BPMN, EPC, ERM, Petri-Nets, UML,...
Business Semantics Standards e.g., UBL, OAGIS, UCM, ISO code lists,...
Procedural Standards e.g. ITIL, ISO 9000, UBL/OAGIS business processes,...
Determination of Context
Notational Standard Selection
Business Semantics Standard Selection
Procedural Standard Selection
Situational yet Transferable Project Model
Reference Model

Fig. 2. Framework for standards adaptation

Becker et al. propose to select model types, i.e. modeling methods, to decide on modeling conventions as well as to provide for multi-perspective modeling [4]. Regarding the *selection of the notational standard(s)* for modeling it is important not to select methods merely based on the availability of one software environment as then the models are less likely to be reused in other environments. It is sensible to select an open or at least a de facto standard. Any notational standard has to be adapted concerning the purpose of the projects and the capabilities of the project team. Common adaptations comprise the coloring or omission of element types or the modification/ simplification of modeling syntax. Furthermore, if multiple methods are used they has to be clearly related. Messages of UBL should, e.g., be clearly recognizable within BPMN models. When selecting notational standards, one has to keep in mind that the project's success always depends on the availability of a modeling tool. When adapting the standards one should assess the capabilities of the project team conservatively.

Similarly, the definition of modeling conventions determines the existence of element attributes as well as the *definition of business semantics*. Becker et al. only give rough hints on terminology [4]. To construct comparable and, thus, transferable models it is important to utilize domain knowledge to specify business semantics [21]. Here, domain ontologies come into place that allow for the unambiguous definition of terms that are used in the course of modeling. They provide a basis to harmonize, i.e. to semantically relate models. Although it implies a considerable initial effort to do so, the definition of (subsets of) existing code lists and thesauri eases the maintenance as well as the transferability of models. Furthermore, in an international context it might be necessary to maintain multilingual ontologies so that less knowledgeable project members can model technical terms in their native language.

When deciding on the communication channels, a governance process has to be established, which audits the models concerning the transferability towards existing project models or reference models. This is followed by the *adherence to a procedural standard* for the models themselves, especially when modeling processes. When using process patterns, e.g. for data exchange, such as those provided by UBL [6] the integration of variants into one reference model is greatly assisted since clear points of interoperability can be identified.

Thus, we argue that standardized notations, a shared vocabulary, and comparable process structures provide a reasonable basis for the integration of the models. Following these basic steps offers a framework, which allows the creation of models, which are as situational for a project as necessary but inherently offer the possibility to be transferred into a reference model.

Since reference models are meant to be universal and therefore domain-spanning, it is not sensible to utilize domain specific modeling methods with a specialized vocabulary. However, for the focused task of creating a domain reference model the approach using domain-specific modeling methods might be preferable.

5 Exemplary Ex Post Application

Recently, an international corporation in healthcare rolled out a new enterprise system. In the course of the project all current business processes were recorded and analyzed. It was initially planned to utilize integrated modeling software to document every process in English language adherent to standards. However, time, cost and size of the project as well as the complexity of modeling conventions and standards led to the decision that the documentation was continued in German in an easy-to-use software environment to at least maintain a certain set of documentation guidelines. Unfortunately, this meant that process models were mostly only saved as MS PowerPoint data. Cf. Figure 3 for an excerpt of a process variant. All misplaced shapes are intentional in this figure to resemble the original documentation. However, all company-specific technical terms that can relate to the project have been deleted.

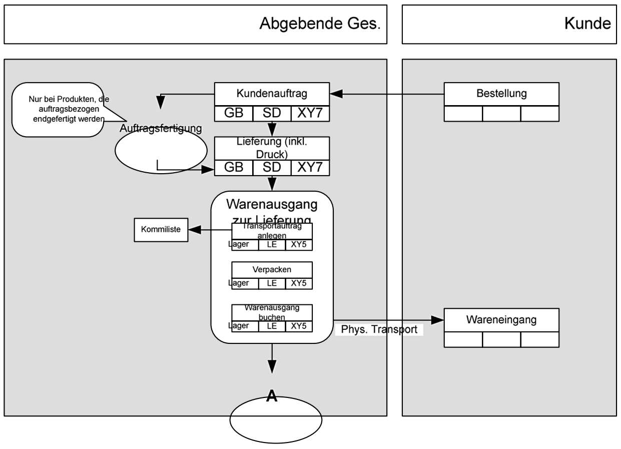


Fig. 3. Process documentation of an international corporation

Due to the shape of the project documentation the compilation of training course material was problematic at best and most material had to be remodeled. If it had been decided in advance to use some parts of standards to facilitate the uniformity of the documentation, the task of creating a reference model for healthcare processes would have been greatly simplified. This reference model would not only have served as documentation and training material but could have been utilized for other purposes.

The following figure contains only a shortened version of the above. A complete model including all of the following suggestion would go beyond the scope of this article. The excerpt is based on BPMN as a notational standard and utilizes some of the procedural as well as business semantics elements of the ordering process of UBL.

Most of the adaptation in this example can only be exemplified implicitly due to length restrictions. It is sensible to assume that due to the limited capabilities of the modelers only a very small subset of the BPMN notation can be used (*specialization*).

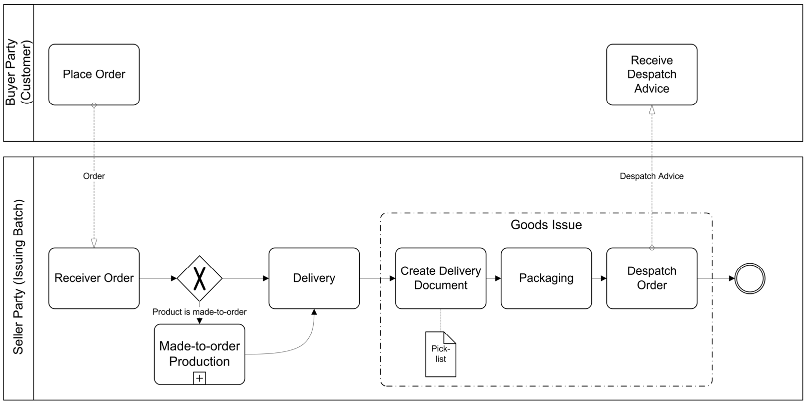


Fig. 4. Excerpt of ex post proposition

This includes simple events (start, intermediate, end), simple tasks (task, subprocess), simple gateways (and, or, xor) as well as only sequence and message flows. Similar considerations can be applied for procedural and business semantics standards. Concerning the latter, it is advisable to support a direct mapping of German and English technical terms with an ontology. Similar to the original informal model, swimlanes have been used to distinguish both parties.

The naming of the parties now mirrors those available in UBL. The interaction of both parties is displayed with a message flow to distinguish the regular sequence flows between tasks. Wherever possible, the tasks from UBL have been integrated and named according to the UBL ordering and fulfillment process to achieve uniform message interfaces (e.g., place order, receive order). Also, the names of the names business documents (order, despatch advice) have been added to the message flow.

The messages can be modeled according to UBL document schemas. The tasks have been modeled in a consistent notation as opposed the arbitrary size and shape of functions in the original documentation. The naming is now in English. Furthermore, a customized code list for healthcare should be added (*aggregation*) to provide further business semantics in order to provide a uniform naming standard that can extend the schema of CCTS and UBL, which is currently limited to the supply chain from sourcing to payment, including the commercial collaborations of international trade.

The process excerpt of this example is one of four variants for the direct order business of the healthcare corporations. The four process variants are not documented in a uniform fashion and, thus, an integration based on the original documentation has never been undertaken. Having modeled the process variants in a standardized manner eases their integration as well as their combination with other models of the reengineering project. Ultimately, this can provide the basis for a comprehensive reference model for healthcare corporations.

6 Conclusion and Future Research

Reference Models comprise information on best or common practices for reuse. However, they commonly originate from individual projects. Their documentation is often situational and does not follow open or at least de facto standards due diverse reasons. Consequently, the transfer of these situational project models into universal reference models is problematic at best.

We identified standards as a means to transport and interface knowledge of different parties. The combination of adaptive modeling mechanisms with notational, business semantics, and procedural standards allows generating modeling methods more sustainable and suitable for a given situation. The careful selection of proper standards enables users to generate situational models with an increased possibility to be understood, exchanged and used by different people or application systems, independent from company or cognition. Therefore, we conclude that standards foster the transition from situational project models to reference model considerably.

This research entails that the careful selection of standards is essential to the transferability of the models. In addition, the adaptation of the standard has to be traceable in order to integrate the model into the reference model. Pending further research, configuration seems to be the most promising adaptation mechanism as it is inherently possible to trace and reproduce the change [3]. Due to the lack of a configurable standards repository, the example has been presented using an aggregation of standards (BPMN, UBL naming, and UBL processes) with subsequent specialization (i.e. using only a subset of the standards). In the long term, instead of applying mechanisms to existing methods, it might also be beneficial to regard methods as social actors and integrate user perspectives already at this level [15].

Furthermore, numerous standardized methods exist, covering different areas of knowledge preservation. Not all of them can be used in the same manner. Depending on the situation, suitable standards have to be selected. The standards discussed in this paper are rather domain unspecific, as such being very general. Nevertheless, they are not the only ones to be taken into consideration. Further research has to include the integration of more standards, covering more aspects. The usefulness of the chosen standards should be supported by flanking explorative empirical studies. Once a standards repository has been generated, the respective parameters have to be explicated, selecting the standards the most suitable for a given problem.

Moreover, reference model providers should explicate their use of the modeling standards and/or conventions to enable third parties to contribute to the model. Projects such as The Open Model Initiative [14] can support the idea by offering recommendations for preconfigured standards packages which include a number of interrelated standards for the creation of better transferable models.

Acknowledgements

This paper was written within the scope of the research project FlexNet (Flexible Information System Architectures for Hybrid Value Networks), funded by the Federal Ministry of Education and Research (BMBF), funding sign 01FD0629. We would like to thank the BMBF and the project executing organization German Aerospace Center (DLR) for their support.

References

1. Becker, J., Delfmann, P., Knackstedt, R.: Adaptive Reference Modeling: Integrating Configurative and Generic Adaptation Techniques for Information Models. In: Becker, J., Delfmann, P. (eds.) Reference Modeling. Efficient Information Systems Design through Reuse of Information Models, pp. 27–58. Physica, Heidelberg (2007)
2. Becker, J., et al.: Perspectives on Process Documentation: A Case Study. In: Chen, C.-S., et al. (eds.) Enterprise Information Systems VII, pp. 167–177. Springer, Dordrecht (2006)
3. Becker, J., Janiesch, C., Pfeiffer, D.: Reuse Mechanisms in Situational Method Engineering. In: Proc. IFIP WG 8.1 Working Conference on Situational Method Engineering (2007)
4. Becker, J., Kugeler, M., Rosemann, M.: Process Management: A Guide for the Design of Business Processes, 2nd edn. Springer, Berlin (to appear, 2007)
5. Becker, J., Schütte, R.: A Reference Model for Retail Enterprises. In: Fettke, P., Loos, P. (eds.) Reference Modeling for Business Systems Analysis, IDEA, Hershey, PA, pp. 182–205 (2007)

6. Bosak, J., McGrath, T., Holman, G.K. (eds.): Universal Business Language v2.0: Committee Specification. OASIS (2006), http://docs.oasis-open.org/ubl/prd3rl-UBL-2.0/UBL-2.0.html
7. Crawford, C. (ed.): Core Components Technical Specification - Part 8 of the ebXML Framework. Version 2.01. UN/CEFACT (2003), http://www.unece.org/cefact/ebxml/CCTS_V2-01_Final.pdf
8. David, P.A., Greenstein, S.: The Economics of Compatibility Standards: An Introduction to Recent Research. The Economics of Innovations and New Technology 1, 3–41 (1990)
9. Fettke, P., Loos, L.: Classification of Reference Models: A Methodology and its Application. Information Systems and e-Business Management 1, 35–53 (2003)
10. Fettke, P., Loos, P.: Perspectives on Reference Modeling. In: Fettke, P., Loos, P. (eds.) Reference Modeling for Business Systems Analysis, IDEA, Hershey, PA, pp. 1–20 (2007)
11. Fomin, V.V., Keil, T., Lyytinen, K.J.: Theorizing about Standardization: Integrating Fragments of Process Theory in Light of Telecommunication Standardization Wars. Sprouts: Working Papers on Information Environments, Systems and Organization 3, 29–60 (2003)
12. Guizzardi, G., Pires, L.F., van Sinderen, M.J.: On the Role of Domain Ontologies in the design of Domain-Specific Visual Modeling Languages. In: Proc. 2nd Workshop on Domain-Specific Visual Languages (2002)
13. International Organization for Standardization: ISO Standards Development Procedures (2007), Available: http://www.iso.org/directives/
14. Koch, S., Strecker, S., Frank, U.: Conceptual Modelling as a New Entry in the Bazaar: The Open Model Approach. In: Proc. 2nd International Conference on Open Source Systems (OSS), pp. 9–20 (2006), http://www.openmodels.org/
15. Lamb, R., Kling, R.: Reconceptualizing Users as Social Actors in Information Systems Research. MIS Quarterly 27, 197–235 (2003)
16. Leymann, F., Roller, D.: Modeling Business Processes with BPEL4WS. Information Systems and e-Business Management 4, 265–284 (2006)
17. Löwer, U.M.: Interorganisational Standards. In: Dissertation, Physica, Heidelberg (2006)
18. Nickerson, J.V., zur Muehlen, M.: The Ecology of Standards Processes: Insights from Internet Standard Making. MIS Quarterly 30, 467–488 (2007)
19. Object Management Group Inc.: Business Process Modeling Notation Specification 1.0 (2006), Available: http://www.omg.org/cgi-bin/apps/doc?dtc/06-02-01.pdf
20. Office of Government Commerce: Programme and Project Management Resources (2007), Available: http://www.ogc.gov.uk/Resource_Toolkit_ppm_resources.asp
21. Pfeiffer, D.: Constructing Comparable Conceptual Models with Domain Specific Languages. In: Proc. 15th European Conference on Information Systems (ECIS), pp. 876–888 (2007)
22. Rosemann, M., van der Aalst, W.M.P.: A Configurable Reference Modelling Language. Information Systems 32, 1–23 (2007)
23. Stuhec, G.: SAP GDTs Based on CCTS (2004), Available: https://www.sdn.sap.com/irj/servlet/prt/portal/docs/library/uuid/b602d790-0201-0010-e3a8-9e4ddfc45d17
24. UN/CEFACT: Unified Context Methodology Project (2007), http://www.untmg.org/
25. Wohed, P., et al.: On the Suitability of BPMN for Business Process Modelling. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102, pp. 161–176. Springer, Heidelberg (2006)

Linking Domain Models and Process Models for Reference Model Configuration

Marcello La Rosa[1], Florian Gottschalk[2], Marlon Dumas[1], and Wil M. P. van der Aalst[2,1]

[1] Queensland University of Technology, GPO Box 2434, Brisbane, QLD 4001, Australia
{m.larosa,m.dumas}@qut.edu.au
[2] Eindhoven University of Technology, PO Box 513, 5600MB Eindhoven, The Netherlands
{f.gottschalk,w.m.p.v.d.aalst}@tue.nl

Abstract. Reference process models capture common practices in a given domain and variations thereof. Such models are intended to be configured in a specific setting, leading to individualized process models. Although the advantages of reference process models are widely accepted, their configuration still requires a high degree of modeling expertise. Thus users not only need to be domain experts, but also need to master the notation in which the reference process model is captured. In this paper we propose a framework for reference process modeling wherein the domain variability is represented separately from the actual process model. Domain variability is captured as a questionnaire that reflects the decisions that need to be made during configuration and their interrelationships. This questionnaire allows subject matter experts to configure the process model without requiring them to understand the process modeling notation. The approach guarantees that the resulting process models are correct according to certain constraints. To demonstrate the applicability of the proposal, we have implemented a questionnaire toolset that guides users through the configuration of reference process models captured in two different notations.

Keywords: reference model, process configuration, variability modeling.

1 Introduction

Business processes like purchasing, recruitment, or customer service processes, are organized in similar ways across companies. To promote the reuse of such processes, Enterprise System vendors provide generic reference process models [4, 16, 17] that can be adapted to the needs of individual companies, thus enabling these companies to leverage proven practices and to avoid designing processes from scratch.

Notations like Configurable Event-driven Process Chains (C-EPC) [13] address the issue of representing and configuring reference process models. Such notations allow users to incorporate multiple process variants into a single configurable process model by means of variation points. By eliminating the undesired variants from the variation points, one can get a configured process model that suits the individual requirements.

Unfortunately, the integration and adaptation of several process variants in such notations often leads to an increase in the model's complexity. Thus, the configuration of a reference process model requires a significant degree of modeling expertise in the particular notation. Moreover, since these notations are designed to capture individual variation points, it is difficult to estimate the impact of high-level configuration decisions on the overall process model, i.e. to determine which variation points are affected by a decision and to what extent. While it is normal to assume that the designers who produce the reference process model itself are familiar with the notation in question, it is less realistic to assume this from stakeholders who have to configure these models (e.g. a logistics or a film production expert).

In this paper we outline a framework that allows users to configure reference process models independently of the process modeling notation employed. We capture the variability of a given domain (e.g. Supply Chain Management) via so-called *domain facts* that form the answers to a set of *questions*. Questions and domain facts are expressed in natural language, thus facilitating the identification and configuration of variants by non-modeling experts. From a given set of facts grouped into questions, we are then able to generate an interactive questionnaire that guides the configuration.

Similarly, we represent this variability at the process level by identifying so-called *process facts*, which relate to the variation points of the configurable process model for the particular domain. We then link domain facts with process facts by means of a *mapping*. In this way, process configuration can be performed by simply answering a set of questions that mask the complexity of the underlying process model. The mapping ensures that the configured process model is semantically correct (e.g. no deadlocks), and consistent with the domain configuration.

The remainder of the paper is structured as follows. In Section 2 we introduce the approaches that are related to and inspired the development of our framework. In Section 3 we present the framework and illustrate it using an application in the area of the Screen Business. We apply the framework to a configurable process modeling notation, namely C-EPC [13] and to a configurable workflow language, namely C-YAWL (an extension of YAWL [1]). In Section 4 we present the approach to link domain and process models and a toolset that supports this approach. Finally, in Section 5 we draw conclusions.

2 Background and Requirements

The separation of process configuration from the context domain has been investigated, among others, by Becker et al. [2,3]. In this approach, adaptation parameters and their possible values are linked to model elements to indicate which sections of the model are relevant or not to a specific application scenario. By assigning values to these parameters, a user can configure a reference model without looking at the process flow. The approach can however be improved by offering guidance to users when assigning values to the adaptation parameters. Moreover, although it is possible to specify local constraints among parameters, no method is provided to check for model-wide consistency that could, e.g., inhibit deadlocks in the process flow or deny parameter settings that are practically not feasible.

To support consistency checking it is possible to use ideas from the CML2 language, which was designed to capture configuration processes for the Linux kernel [12]. It supports the definition of validity constraints based on propositional formulas over so-called symbols. To guide the configuration process, a configuration model in CML2 is composed of questions which lead to a given symbol being given a value.

The use of questions to steer the selection of process alternatives is advocated in [15], where alternatives are depicted as process specializations and the activation of these specializations is linked to conditions expressed as questions. However constraints over questions are not defined and no tool support is offered.

More generally, variability of large software systems has been studied in the field of Software Product Line Engineering (SPLE) [11]. The idea of SPLE is to capture how a collection of available options impact the way a software system is built from a set of components. Parallels can be drawn between SPLE and reference models. A detailed comparison between our proposal and SPLE approaches is given in [7].

In this paper, we follow up on the above approaches and develop an integrated framework for reference process model configuration. The framework has been developed by following a design science approach [10]. By analyzing existing work, we have identified requirements for such an integrated framework. Chief among these requirements is that domain models should be represented separately from process models, in line with the approach of Becker et al. Secondly, domain models should be expressed in terms that are easily understandable by subject matter experts, and that can be directly exploited to guide the configuration process. Following these requirements, we have designed a framework that allows modelers to capture the variability of a given domain by means of a set of ordered questions. The framework provides decision support guidance that helps subject matter experts to answer the question in a way that leads to a valid configuration. An initial outline of this approach is reported in [8]. In this paper, we further extend this initial proposal with concrete mechanisms for linking domain models expressed as questionnaires, with process model elements. We present two such mechanisms in the context of two different modelling languages.

In line with a design science method, we have validated the proposal by means of a software tool implementation which has been tested on comprehensive examples, one of which is shown below.

3 Questionnaire-Based Variability Framework

This section presents our framework. First, we show how to represent *domain configurations*. Then, we introduce *process configurations*, i.e., a way of capturing process variability. The next section shows how to integrate both types of configurations.

3.1 Capturing Domain Variability

We propose to depict the variability of a given domain independently of specific notations or languages, by means of a set of domain facts that form the space of possible answers to a set of questions. A domain fact is a boolean variable representing a feature of the domain, e.g. "Tape shoot", that can be enabled or disabled. Questions group domain facts according to their content, so that all the facts of the same question can be set at once by answering the question. For example, the question "Which shoot media have been used?" allows users to choose the shoot medium between fact "Tape shoot" and "Film shoot". A fact may appear in more than one question: in this case it is set the first time and its value is preserved in the subsequent questions. A fact always has a default value (*true* or *false*), while it can be marked as 'mandatory' if it needs to be set explicitly. For example, fact "Tape shoot" is true by default as the majority of production projects are shot on tape, which is less expensive than film. If a non-mandatory fact is left unset, i.e. if the corresponding question is not answered, the default value can be used instead. This way, each domain fact will always have a value either set explicitly by an answer or by using its default.

To illustrate these concepts, we consider an exemplification of the Post-production process in the area of Screen Business [14]. Figure 1 presents a possible structure of questions/domain facts to capture the variability in this field. All questions and facts are assigned a unique identifier and a description.

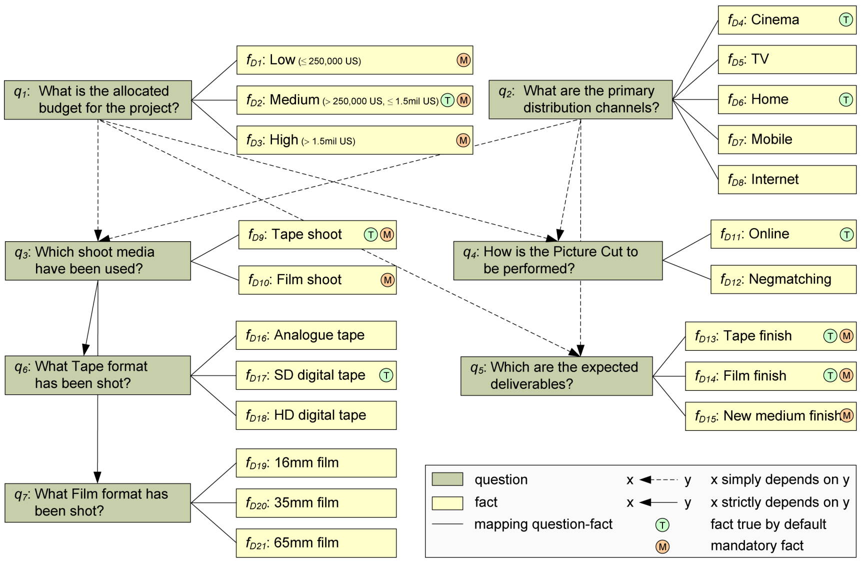


Fig. 1. A possible structure of questions/domain facts for the Post-production example

Post-production aims at the editing and technical completion of a screen business project. Depending on the available budget, a project can be shot on tape, on film, or on both the media. Of the two, film results in a more costly operation due to special treatments for making it visible and permanent. In Figure 1 this choice is modeled by question q_3 and its facts, while the facts of q_6 and q_7 capture the possible sub-formats of tape, resp. film. The picture cut transfers the editing decisions that are taken on a low-resolution format in the Offline, to a high resolution format. The cut can be performed in an editing suite (Online), or on the original negative (Negmatching). This choice is captured by the facts of question q_4 and depends on the type of shoot medium. A project can be finished for delivery on tape, film, new medium, or any combination thereof. This is captured by the facts of q_5. The overall finishing process varies on the basis of the finishing media and may involve further tasks according to the choices made for the picture cut. For example, if the cut is done in Negmatching, the project must be finished at least on film. On the other hand, if the cut is only done Online and a film finish is required, the editing results need to be transferred to film in the so-called Record DFM. Similarly, a Telecine transfer is required to transfer the edited film to tape or file if the cut is done only in Negmatching and the project is to be finished on tape or on a new medium (e.g. DVD).

Interactions like these, which occur among the values of the domain facts, are modeled by a set of *domain constraints* in propositional logic that prune the configuration space. The constraints for the facts of Figure 1 follow:[1]

C1: $f_{D1} \dot{\vee} f_{D2} \dot{\vee} f_{D3}$ **C2:** $f_{D1} \Rightarrow \neg(f_{D10} \vee f_{D14})$ **C3:** $f_{D2} \Rightarrow \neg f_{D10}$
C4: $f_{D4} \vee f_{D5} \vee f_{D6} \vee f_{D7} \vee f_{D8}$ **C5:** $f_{D4} \Rightarrow f_{D14}$ **C6:** $f_{D5} \Rightarrow f_{D13}$
C7: $f_{D6} \Rightarrow (f_{D13} \vee f_{D15})$ **C8:** $(f_{D7} \vee f_{D8}) \Rightarrow f_{D15}$ **C9:** $f_{D9} \vee f_{D10}$
C10: $f_{D11} \vee f_{D12}$ **C11:** $\neg f_{D10} \Rightarrow \neg f_{D12}$ **C12:** $f_{D13} \vee f_{D14} \vee f_{D15}$
C13: $(f_{D16} \dot{\vee} f_{D17} \dot{\vee} f_{D18}) \Leftrightarrow f_{D9}$ **C14:** $\neg(f_{D16} \vee f_{D17} \vee f_{D18}) \Leftrightarrow \neg f_{D9}$ **C15:** $f_{D12} \Rightarrow f_{D14}$
C16: $(f_{D19} \dot{\vee} f_{D20} \dot{\vee} f_{D21}) \Leftrightarrow f_{D10}$ **C17:** $\neg(f_{D19} \vee f_{D20} \vee f_{D21}) \Leftrightarrow \neg f_{D10}$

For example, since it is possible to shoot both tape and film, the facts representing these two features are bound by a logic OR (as per C9). Questions q_1 and q_3 capture the contextual choices for a post-production project, namely the allocated budget and the distribution channel. Their answers can affect the overall project, as shown by the constraints over their facts (C1 to C8). Further on, Negmatching (f_{D12}) cannot be performed if the project has not been shot on film (C11), and since its handcraft style makes it an expensive activity, Negmatching is worthwhile only if it is followed by a film finish (C15). However shooting film (f_{D10}) is only allowed for low/medium budgets (C2, C3), thus limiting the possibility to carry out a Negmatching.

A *domain configuration* is a possible valuation over domain facts that does not violate the constraints.

Order dependencies determine the order in which questions are presented to users. A ‘simple’ dependency (dashed arrow in Figure 1) captures an optional precedence between two questions: e.g. q_3 can be posed after q_1 or q_2. A ‘strict’ dependency (plain arrow) captures a mandatory precedence: e.g. q_6 is posed after q_3 only. This way we can ask the most discriminating questions first, like q_1 and q_2 in Figure 1, so that subsequent questions are (partly) answered by means of the domain constraints. If, e.g., we answer q_3 with “Film shoot” only, the question about the tape formats (q_6) becomes irrelevant. Dependencies can be arbitrary as long as cycles are avoided.

The above concepts form the definition of a Configuration Model (CM) - a first-class model to capture domain variability. The complete definition of CM can be found in [7]. In the following sections we show how CMs can be applied to support the configuration of reference process models.

3.2 Capturing Process Variability

Generally, notations for configurable process models rely on the concept of variation points to capture a point in the process flow where more than one variant exists. To link a CM to any configurable process model, independently of the notation adopted, we identify each process variant with a so-called *process fact*. A process fact is a

[1] $\dot{\vee}$ indicates the exclusive disjunction (XOR), a commutative and associative relation.

boolean variable set to true if the variant it refers to is selected in a given process configuration, and to false otherwise.

Examples of notations for configurable process models are C-EPC [13] and Configurable YAWL (C-YAWL) [5]. C-EPCs extend Event-driven Process Chains (EPCs) [6] with variation points. A variation point can be a configurable connector (extension to the EPC join and split) or a configurable function (extension to the EPC function). A configurable connector can have several variants depending on its type, which can be logical AND, XOR, OR; a configurable function can have three variants: ON if enabled, OFF if disabled and OPT if optionally enabled.

Figure 2 depicts the post-production process in a C-EPC, where variation points are highlighted by a thicker border (trivial events are omitted). The process starts with the preparation of the footage for edit, which depends on the type of shoot medium being used (tape or film). This choice is captured by the configurable OR-join OR_1. Its behavior can be restricted to an AND connector (if both tape and film are used), to branch SEQ_{1a} (for tape only, this results in branch SEQ_{1b} being deleted), or to branch SEQ_{1b} (for film only, branch SEQ_{1a} being deleted). We identify these three variants with three process facts f_{P1}, f_{P2}, f_{P3}, where f_{P1} is true if both tape and film are chosen, f_{P2} is true if only tape is chosen, and f_{P3} is true if only film is chosen. In a C-EPC a configurable OR can also be configured to an OR or XOR connector, but since these variants are not feasible for this process, we do not assign process facts to them.

Once the footage is ready, the project is edited on a low-resolution medium in the *Offline*. The editing decisions are then transferred to a high-resolution medium in the *Online* and/or *Negmatching*, depending on the configuration of the OR-split OR_2 (f_{P4}

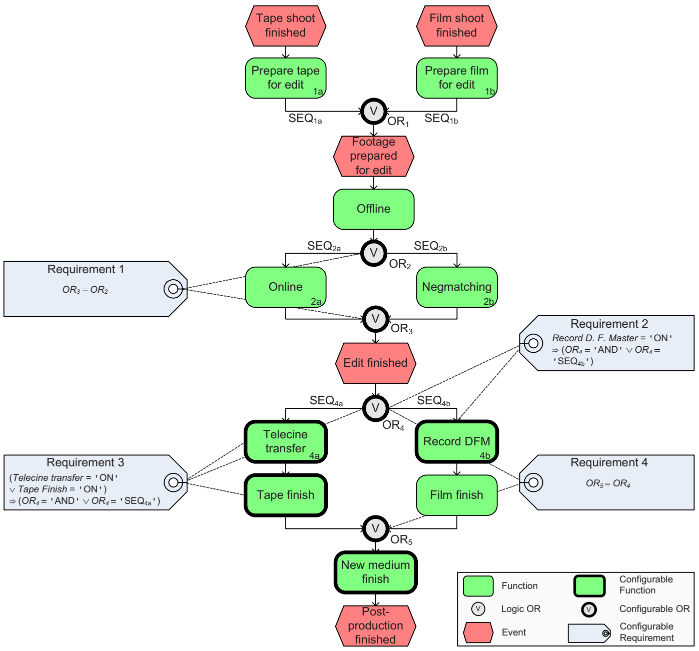


Fig. 2. The post-production reference process model in C-EPC

for AND, f_{P5} for SEQ$_{2a}$, and f_{P6} for SEQ$_{2b}$) and of the OR-join OR_3 ($f_{P7} - f_{P9}$). The possibility to finish the project on any combination of tape, film and new medium is captured by OR_4 ($f_{P10} - f_{P12}$), OR_5 ($f_{P13} - f_{P15}$), and by the configurable functions *Telecine Transfer*, *Record DFM*, *Tape finish* and *New Medium finish*. These functions can be set only to ON or OFF, thus we assign two process facts to each of them. For example, f_{P16} and f_{P17} capture the variants ON and OFF of *Telecine Transfer*.

The same process can also be represented in C-YAWL, an extension of the executable process modeling language YAWL. The YAWL notation is based on conditions and tasks while the logic connectors of AND, XOR and OR are integrated in each task in the form of a join (for the incoming arcs) and a split (for the outgoing arcs). C-YAWL extends YAWL with so-called *ports* as variation points. A task's join has an *input port* for each combination of arcs through which the task can be triggered, whilst a task's split has an *output port* for each combination of subsequent arcs that can be triggered after the task's completion. An input port can be configured as enabled to allow the triggering of the task via this port, as blocked to prevent the triggering, or as hidden to skip the task's execution without blocking the subsequent process. An output port can be enabled to allow the triggering of paths leaving the port, or blocked to prevent their triggering. Like in C-EPC, in C-YAWL we represent each feasible variant of a port with a process fact, thus, e.g., if a port is always enabled we do not assign any process facts to it.

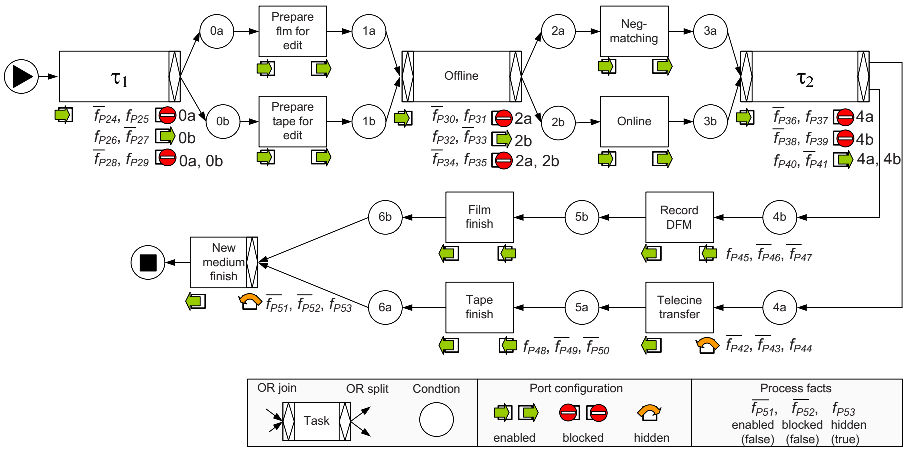

τ_1
$\overline{f_{P24}}, f_{P25}$ 0a
$f_{P26}, \overline{f_{P27}}$ 0b
$\overline{f_{P28}}, f_{P29}$ 0a, 0b
0a
0b
Prepare flm for edit
Prepare tape for edit
1a
1b
Offline
$\overline{f_{P30}}, f_{P31}$ 2a
$f_{P32}, \overline{f_{P33}}$ 2b
$\overline{f_{P34}}, f_{P35}$ 2a, 2b
2a
2b
Neg-matching
Online
3a
3b
τ_2
$\overline{f_{P36}}, f_{P37}$ 4a
$f_{P38}, \overline{f_{P39}}$ 4b
$f_{P40}, \overline{f_{P41}}$ 4a, 4b
4b
Record DFM
$f_{P45}, \overline{f_{P46}}, \overline{f_{P47}}$
5b
Film finish
6b
4a
Telecine transfer
$\overline{f_{P42}}, \overline{f_{P43}}, f_{P44}$
5a
Tape finish
$f_{P48}, \overline{f_{P49}}, \overline{f_{P50}}$
6a
New medium finish
$\overline{f_{P51}}, \overline{f_{P52}}, f_{P53}$
OR join
OR split
Task
Condtion
Port configuration
enabled
blocked
hidden
Process facts
$\overline{f_{P51}}$, $\overline{f_{P52}}$, f_{P53}
enabled (false)
blocked (false)
hidden (true)

Fig. 3. Post-production reference process model in C-YAWL with process facts overlaid

Figure 3 depicts the post-production process in C-YAWL with an example configuration for a project shot on tape, edited Online and finished on film and tape. The first task, τ1, is used to route the process flow according to the shoot media. This task has only one incoming arc from the input condition. Therefore, its join has only one input port which always needs to be enabled, i.e. we do not assign any process fact to its variants. The task's OR-split has three output ports: one to trigger the path to condition 0a (leading to film preparation), one to trigger the path to condition 0b (leading to tape preparation) and one to trigger both paths. In this case we assign a process fact to each variant of these output ports as we want to capture the possibility to choose the shooting media. We thus use two process facts for each port: fP24 for the variant enabled and fP25 for the variant blocked of the first port, fP26, fP27 for the second port's variants, and fP28, fP29 for the third port's variants. In the example the project is shot on tape, so the port to 0b is set to enabled (i.e. fP26 = true), and the other two ports are set to blocked (fP25, fP29 = true).

An OR-join can only have one input port in C-YAWL. In Figure 3 the input port of the OR-join of Offline is configured to enabled as this task is always executed. No process fact is thus assigned to its variants. The project is edited Online. So the output port of Offline that triggers condition *2b* is the only one to be enabled (f_{P32} = true). Similar considerations to the OR-join of *Offline* also hold for the OR-join of task τ_2. As the project is finished on tape and film, the output port of the OR-split of τ_2 that triggers *4a* and *4b* is enabled (f_{P40} = true). We do not need *Telecine transfer* and *New Medium finish*, but we want the process to complete. Therefore, their input ports are hidden. To depict this configuration option for input ports, a third process fact is assigned to each input port (f_{P44}, f_{P53}=true). On the other hand, the tasks *Record DFM* and *Tape Finish* are required, thus their input ports are enabled (f_{P45}, f_{P48} = true).

Notations as C-EPC and C-YAWL provide the ability to define configurable requirements to restrict the variants allowed for a variation point. These requirements need to capture the interdependencies of the domain and to preserve the correctness of the model. Process facts, being an abstraction of process variants, are thus subject to the same requirements. In our framework, however, we need to capture only the requirements for process correctness, as the domain constraints are propagated to process facts by mapping the CM to the configurable process model. Thus our *process constraints* are only a subset of the configurable requirements of C-EPC and C-YAWL. In C-EPC such requirements are annotated to the relevant variation points as labels. In Figure 2 we only report the requirements that correspond to the process constraints. Req. 1 and 4 ensure a synchronized configuration of the OR splits/joins to prevent the process from a deadlock. This corresponds, e.g., to the process constraints: $f_{P4} \Leftrightarrow f_{P7}$, $f_{P5} \Leftrightarrow f_{P8}$, $f_{P6} \Leftrightarrow f_{P9}$. Req. 2 and 3 guarantee that the configurable functions can be performed in any configuration where they are set to ON. In C-YAWL, the configurable requirements are not directly depicted in the model. For example, to prevent the process from deadlocking, it is required that every C-YAWL task with an input port enabled or hidden has at least one output port enabled; or if a task can trigger a condition other than the final one, there needs to be a task with an enabled or hidden input port which can be triggered by this condition.

A *process configuration* is thus a possible valuation over the process facts that does not violate the process constraints, ensuring the configured model is correct. The process configuration complements the domain configuration. The next section shows how both types of configurations can be related.

4 Linking Domain and Process Variability

We link the domain variability captured by a CM with the variability of a configurable process model, by mapping domain facts to process facts. We call $F_D = f_{D1},\ldots,f_{Dn}$ the set of domain facts and $B_D(F_D)$ the boolean function representing the conjunction of the domain constraints, such that B_D holds for every domain configuration. Also, we call $F_P = f_{P1},\ldots,f_{Pn}$ the set of process facts and $B_P(F_P)$ the boolean function for the conjunction of the process constraints, such that B_P holds for every process configuration. A boolean function $B_M(F_D, F_P)$ creates the mapping of domain and process facts, such that each process fact equals a boolean expression over the domain facts.

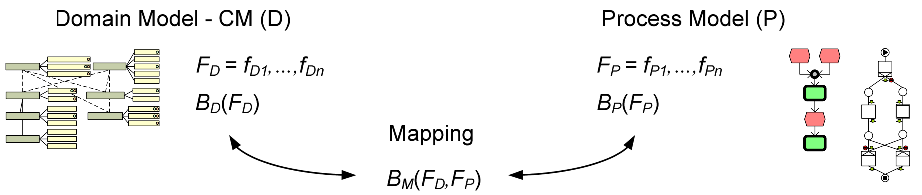


Fig. 4. Mapping configurable domain and configurable process model

For example, the variants of OR_1 in the C-EPC process model of Figure 2 are captured by the process facts f_{P1} (for the variant "tape and film"), f_{P2} (for "tape only") and f_{P3} (for "film only"). At the domain level, this would correspond to answer q_3 (Which shoot media have been used?) with both f_{D9}, f_{D10} = true in the first case, with only f_{D9} = true in the second case, and with only f_{D10} = true in the third case (where f_{D9} is Tape shoot and f_{D10} is Film shoot). Therefore, we map the boolean function $f_{D9} \wedge f_{D10}$ to f_{P1} so that the latter is set to true if and only if f_{D9} and f_{D10} are true. Likewise, we map $f_{D9} \wedge \neg f_{D10}$ to f_{P2} and $\neg f_{D9} \wedge f_{D10}$ to f_{P3}. In this way we can select the proper process variant by checking which of the above expressions holds against a given domain configuration, which is obtained from the answers to the questions of the CM.

The mapping $B_M(F_D, F_P)$ is *valid* if and only if the boolean expressions over the domain facts associated to the process facts of the same variation point are in exclusive disjunction. Given a domain configuration, exactly one variant has to be selected for each variation point. In this way we avoid a domain configuration to lead to zero or more than one variant per variation point, i.e., for example, the domain facts should not require that at the same time a C-EPC function is turned ON and OFF or a port in C-YAWL is blocked and enabled.

An excerpt of the mapping between the CM of Figure 1 and the process models of Figure 2 and 3 is given as follows.

M1: $f_{P1} \Leftrightarrow f_{D9} \wedge f_{D10}$ **M2:** $f_{P2} \Leftrightarrow f_{D9} \wedge \neg f_{D10}$ **M3:** $f_{P3} \Leftrightarrow \neg f_{D9} \wedge f_{D10}$
M4: $f_{P4} \Leftrightarrow f_{D11} \wedge f_{D12}$ **M5:** $f_{P5} \Leftrightarrow f_{D11} \wedge \neg f_{D12}$ **M6:** $f_{P6} \Leftrightarrow \neg f_{D11} \wedge f_{D12}$ [...]
M16: $f_{P16} \Leftrightarrow (\neg f_{D11} \wedge f_{D13}) \vee (\neg f_{D11} \wedge f_{D15})$ **M17:** $f_{P17} \Leftrightarrow \neg((\neg f_{D11} \wedge f_{D13}) \vee (\neg f_{D11} \wedge f_{D15}))$ [...]
M24: $f_{P24} \Leftrightarrow \neg f_{D9} \wedge f_{D10}$ **M25:** $f_{P25} \Leftrightarrow \neg(\neg f_{D9} \wedge f_{D10})$ **M26:** $f_{P26} \Leftrightarrow f_{D9} \wedge \neg f_{D10}$
M27: $f_{P27} \Leftrightarrow \neg(f_{D9} \wedge \neg f_{D10})$ **M28:** $f_{P28} \Leftrightarrow f_{D9} \wedge f_{D10}$ **M29:** $f_{P29} \Leftrightarrow \neg(f_{D9} \wedge f_{D10})$ [...]
M42: $f_{P42} \Leftrightarrow (\neg f_{D11} \wedge f_{D13}) \vee (\neg f_{D11} \wedge f_{D15})$ **M43:** $f_{P43} \Leftrightarrow$ *false* [2]
M44: $f_{P44} \Leftrightarrow \neg((\neg f_{D11} \wedge f_{D13}) \vee (\neg f_{D11} \wedge f_{D15}))$ [...]

[2] Fact f_{P43} can be omitted since it represents a commonality, being always *false*.

Answering a question may affect one or multiple variation points, e.g. the facts of q_1 indirectly affect a number of variation points. Also, configuring a variation point can be determined by answering more than one question, e.g. the process facts that refer to OR_4 in the C-EPC model are affected by q_4 and q_5.

A *valid process configuration* with respect to the domain is given by any domain configuration that leads to a process configuration via a valid mapping, i.e. if the conjunction $B_D \wedge B_P \wedge B_M$ is satisfiable and the mapping is valid. The configuration space is obtained by the intersection of the two configuration spaces (domain and process) via the mapping. By representing the domain variability in a separate model, we can avoid capturing the interdependencies of the domain in the configurable process model, as these interdependencies are represented by the domain constraints in the CM, and propagated to the process model via the valid mapping. Constraints over process facts have thus to deal only with the preservation of the model correctness. As a result, the identification of such constraints becomes simpler.

Once each variation point has been configured with a variant, a set of *actions*, attached to the process fact that captures the selected variant, are performed on the process model to commit a configuration.

In the next section we propose a methodology to constructively define a mapping between a domain and its process model.

4.1 Constructing the Mapping between Domain and Process Model

The first step towards the construction of a mapping is to capture the variability of a given domain by means of a CM. This task should be conducted by the modeller in close collaboration with the domain experts, e.g. the film producers. Similarly, the domain should be represented by a configurable process model in a suitable notation. Process facts should be identified with the variants of the process and process constraints should be defined to preserve the model correctness (these depend on the notation adopted). As a rule of thumb, a fact is meant to represent a variant of the domain or process model. Thus a fact should be considered as such only if it can be freely set before starting the configuration; otherwise it would represent a commonality in the domain or process model, and should be left out.

Once domain facts, process facts and their constraints have been identified, it is possible to define the mapping by means of a two-way impact analysis:

- *from domain to process*: e.g., given a domain fact, we need to estimate what are the implications of setting such a fact to true/false in the process model;
- *from process to domain*: e.g., given a variation point, we need to consider which domain facts are impacted by configuring such a point with a variant.

Although a mapping is valid, the process constraints might restrict the configuration space of the domain so as to deny some domain options, as a result of the application of the mapping. This situation may lead to problems when it comes to communicate such restrictions to the stakeholders, and as such should be avoided. In these cases, either the mapping or the process model should be modified, supposing the domain cannot be changed. On the other hand, the mapping might be so restrictive that no correct process model is allowed at all. In these cases, process facts and constraints can be used to evaluate the feasibility of the domain constraints and of the mapping, and to suggest their revision.

4.2 Tool Support

To establish the practical feasibility of our framework, we have implemented a toolset[3] that provides end-to-end support for reference process model configuration. Each tool is a stand-alone application responsible for a specific task in the configuration process, from the collection of the answers via questionnaires, to the release of a configured process model. Figure 5 provides an overview of the toolset's architecture.

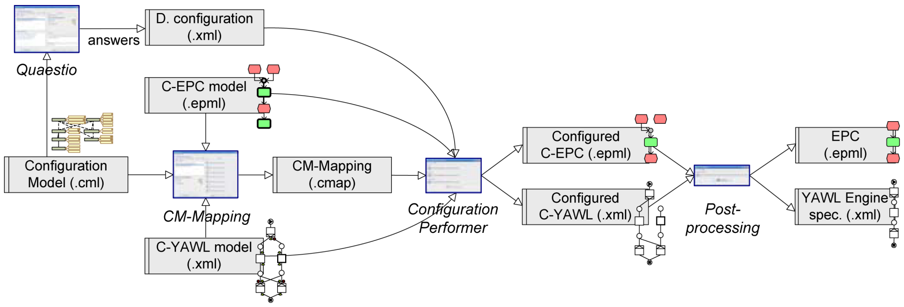


Fig. 5. The software architecture of the tools implemented

Quaestio accepts an XML serialization of a CM as input and generates a questionnaire that guides the configuration interactively by posing only the relevant questions in an order consistent with the order dependencies. The tool also prevents users from entering conflicting answers to subsequent questions, by dynamically checking the domain constraints. Questions can be answered explicitly or by using the default values, and they can be rolled back. Quaestio embodies a SAT solver[4] based on Shared Binary Decision Diagrams (SBDDs) [9], to handle propositional logic formulae. Algorithms based on SBDDs can efficiently deal with systems made up of around one million of possibilities [9]. As a result, Quaestio can scale with CMs yielding around one million domain configurations.

The *CM-Mapping* tool allows designers to define mappings between the domain facts and the process facts of a C-EPC or C-YAWL net, whose serialization is taken as input. This tool uses the SBDD calculator to check the consistency of the domain constraints and of the process constraints, i.e. if they can be satisfied and if each fact can be freely set. It also verifies the validity of the generated mapping, checks for redundancies, and shows possible restrictions of the single configuration spaces (domain and process) that may occur from the application of the mapping.

The *Configuration Performer* takes as input a domain configuration generated by Quaestio, a serialization of a C-EPC or C-YAWL reference process model, and the

[3] Downloadable from `http://sky.fit.qut.edu.au/~dumas/ConfigurationTool.zip`
[4] Downloadable from `http://www-verimag.imag.fr/~raymond/tools/bddc-manual`

corresponding mapping. It outputs an intermediate format to represent the configured net, where each variation point has been marked with one variant. This artefact is then post-processed by a tool that implements a derivation algorithm to generate a syntactically correct EPC or YAWL model.

To achieve the configuration over process facts shown in Figure 3, we use the following domain configuration $\sigma_D = \{\neg f_{D1}, f_{D2}, \neg f_{D3}, \neg f_{D4}, \neg f_{D5}, \neg f_{D6}, f_{D7}, f_{D8}, f_{D9}, \neg f_{D10}, f_{D11}, \neg f_{D12}, \neg f_{D13}, \neg f_{D14}, f_{D15}, \ldots\}$. From the domain configuration we realise that the process configuration of Figure 3 is only feasible for medium/high budgets projects. Figure 6 shows the configured YAWL process obtained by the post-processing tool.

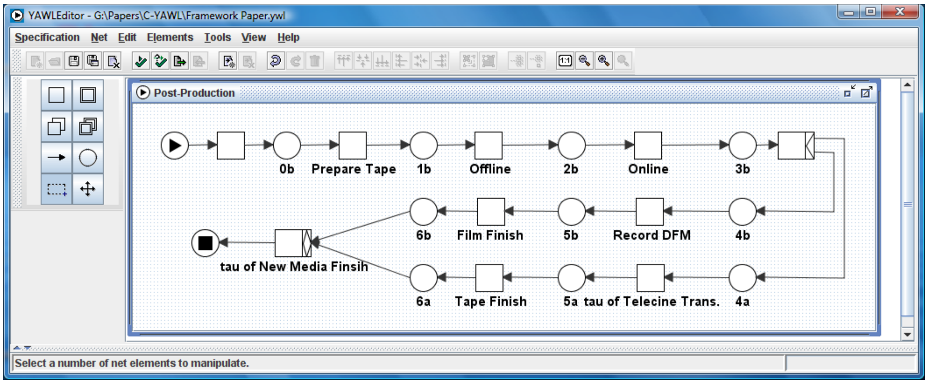

YAWLEditor - G:\Papers\C-YAWL\Framework Paper.ywl
Specification Net Edit Elements Tools View Help
Post-Production
0b
Prepare Tape
1b
Offline
2b
Online
3b
6b
Film Finish
5b
Record DFM
4b
tau of New Media Finsih
6a
Tape Finish
5a
tau of Telecine Trans.
4a
Select a number of net elements to manipulate.

Fig. 6. The YAWL process model derived from the configuration of the model in Figure 3

5 Conclusion and Outlook

This paper presented a framework and a toolset for the configuration of reference process models. The main innovation of the proposal is that the configuration is not driven directly by a (configurable) process model, but rather by a model that captures the variability of the application domain. The domain model is structured in terms of questions, facts and constraints that reflect the decisions that need to be made during configuration. From a domain model, an interactive questionnaire is generated that guides domain experts through the process of configuring the process model without requiring them to understand the process modeling notation.

The domain model is then linked to a configurable process model, so that once all configuration decisions are made, an individualized process model can automatically be generated. The variability of the configurable process model is also captured in terms of facts and constraints. By merging the process constraints with the domain constraints, we obtain a unified set of constraints that is used by the configuration tool to ensure that the model generated from the configuration decisions is always correct.

Capturing reference process models in terms of facts and constraints allows us to abstract away from the modeling notation. We have demonstrated the applicability of the proposal on two configurable process modeling notations: one intended for analysis (C-EPCs) and another intended for implementation (C-YAWL). We have tested the approach on several examples, particularly in the area of film post-production.

In the current framework, process facts and constraints need to be manually extracted from a process model. We are working on applying formal analysis techniques to automate this extraction. Furthermore, we are conducting experiments with screen business experts to empirically evaluate the applicability and impact of the proposed configuration approach on the modeling process.

References

1. van der Aalst, W.M.P., ter Hofstede, A.H.M.: YAWL: Yet Another Workflow Language. Information Systems 30(4), 245–275 (2005)
2. Becker, J., Delfmann, P., Knackstedt, R.: Adaptive Reference Modelling: Integrating Configurative and Generic Adaptation Techniques for Information Models. In: Reference Modeling. Efficient Information System Design through Reuse of Information Models, Berlin, pp. 23–49 (2007)
3. Becker, J., et al.: Configurative Process Modeling – Outlining an Approach to Increased Business Process Model Usability. In: Proceedings of the Information Resources Management Association Conference, New Orleans LA, USA, pp. 615–619 (2004)
4. Curran, T., Keller, G.: SAP R/3 Business Blueprint: Understanding the Business Process Reference Model. Upper Saddle River (1997)
5. Gottschalk, F., et al.: Configurable Workflow Models. BETA Working Paper 222, TU Eindhoven (2007)
6. Keller, G., Nüttgens, M., Scheer, A.W.: Semantische Processmodellierung auf der Grundlage Ereignisgesteuerter Processketten (EPK). Veröffentlichungen des Instituts fur Wirtschaftsinformatik, University of Saarland, Saarbrücken (1992)
7. La Rosa, M., et al.: Variability Modeling for Questionnaire-based System Configuration. Preprint # 7992, Queensland University of Technology (2007), Available at: http://eprints.qut.edu.au/archive/00007992
8. La Rosa, M., et al.: Questionnaire-driven Configuration of Reference Process Models. In: Krogstie, J., Opdahl, A., Sindre, G. (eds.) CAiSE 2007 and WES 2007. LNCS, vol. 4495, Springer, Heidelberg (2007)
9. Minato, S., Ishiura, N., Yajima, S.: Shared Binary Decision Diagram with Attributed Edges for Efficient Boolean function Manipulation. In: Proceedings of the ACM/IEEE Conference on Design Automation (DAC), pp. 52–57 (1990)
10. Owen, C.L.: Design Research: Building the Knowledge Base. Design Studies 19(1), 9–20 (1998)
11. Pohl, K., Böckle, G., van der Linden, F.: Software Product-line Engineering – Foundations, Principles and Techniques. Springer, Berlin (2005)
12. Raymond, E.S.: The CML2 Language (2000), http://catb.org/esr/cml2/cml2-paper.html
13. Rosemann, M., van der Aalst, W.M.P.: A Configurable Reference Modelling Language. Information Systems 32(1), 1–23 (2007)
14. Seidel, S., Rosemann, M., ter Hofstede, A.H.M., Bradford, L.: Developing a Business Process Reference Model for the Screen Business - A Design Science Research Case Study. In: Proceedings of the 17th Australasian Conference on Information Systems (ACIS), Adelaide, Australia (2006)

15. Soffer, P., Golany, B., Dori, D.: ERP modeling: a comprehensive approach. Information Systems 28(6), 673–690 (2003)
16. Stephens, S.: The Supply Chain Council and the Supply Chain Operations Reference Model. Supply Chain Management - An International Journal 1(1), 9–13 (2001)
17. Taylor, C., Probst, C.: Business Process Reference Model Languages: Experiences from BPI Projects. In: Proceedings of INFORMATIK 2003 – Innovative Informatikanwendungen, Band 1, pp. 259–263 (2003)

Reference Modeling for Higher Education Budgeting: Applying the H2 Toolset for Conceptual Modeling of Performance-Based Funding Systems

Jan vom Brocke[1], Christian Buddendick[2], and Alexander Simons[1]

[1] Liechtenstein University,
Fürst-Franz-Josef-Strasse, 9490 Vaduz, Principality of Liechtenstein
{jan.vom.brocke, alexander.simons}@hochschule.li
[2] European Research Center for Information Systems (ERCIS), University of Münster,
Leonardo-Campus 3, 48149 Münster, Germany
christan.buddendick@ercis.de

Abstract. The Higher Education (HE) sector has gained remarkable economic importance worldwide. There is a huge amount of institutions competing in this dynamically evolving market. Emerging concepts like new public management advise to organize HE institutions as autonomous business units that can easily be adapted to market changes. In this context, methods of performance-based funding play an important role in governing the institutions both from an external and internal perspective. However, the right choice of indicators measuring the performance turns out to be a vital factor for the success of these budgeting methods. Wrongly chosen, they may even be misleading. Whereas most studies in this field suggest particular measurement systems, our focus of research is to provide a methodology for the design of individual measurement systems. As part of this method, techniques from reference modeling can be applied in order to reuse typical measures for special situations and then further adopt and extend them. In this paper, we study the modeling language H2 which may serve as an essential part of such a methodology. We conclude with an outlook on further research.

Keywords: Budgeting, H2, Higher Education (HE), Incentive System, Indicator System, Information Model, Performance-Based Funding, Reference Modeling.

1 Introduction

Higher Education (HE) institutions are confronted with a dynamically changing environment [1]. New laws, new competitors and new customers are examples for these changes. Initiatives like the Bologna process in Europe are among the current challenges of HE institutions. In order to deal with these challenges, regulatory reforms have been conducted, especially with regard to the financial management of the assigned funds [2]. Since then, the institutions are entitled to allocate the funds more autonomously. Within this context, they have to apply principles of performance-based funding [3].

In business administration there is a huge body of research on performance-based funding [1,2,3] and supporting information systems. However, experiences of this work show that the crucial part of implementing a budgeting system is scarcely a technical one but essentially lies in the definition of performance measures. Choosing the wrong measures, in particular, may even cause misleading incentives being a threat to the organization. Measures like shareholder value and return on investment are prominent examples that have been discussed in literature.

Against this background, the challenge of implementing a budgeting system in HE lies in finding the right measures for an institution with respect to incentive setting. Loose governance structures and a high degree of decentralization are essential characteristics that need to be taken into account when defining measures. In particular, in the German speaking countries, a remarkable amount of studies has been carried out on these issues [4,5]. However, this work is often mainly focused on the illustration of measurement systems limited to single universities [7], whereas little work has yet been carried out on evaluating these systems towards a more general understanding of performance indicators in HE.

Our work is based on the perception that there is a certain trade-off between the idea of standardization and the idea of individualization when designing a measurement system. The call for individualization is essential, as the budgeting process of an institution is strongly related to its specific context. Take the strategic position of an HE institution, for instance, determining e. g. the importance of cutting-edge research compared to teaching and professional development. As a consequence, instead of suggesting numerous measures, we rather focus on the design of methods enabling institutions to identify their individual set of measures. At the same time, there is a certain need for standardized measurement systems, for efficiency reasons. This leads to the idea of additionally providing pre-designed measures to be reused in similar context situations. For that purpose, methods from reference modeling can be applied.

In this paper, we focus on methodological aspects of specifying measurement systems for performance-based budgeting in the German HE context. The intention is to introduce a method that is suitable to individually identify and negotiate measures to be implemented. According to a design science-oriented approach [6], we first analyze the state-of-the-art in HE budgeting. We then introduce a method for conceptual modeling of according measurement systems and subsequently apply the language in the domain of budgeting in HE institutions. This application serves as a proof of concept and helps to identify fields of further development. We finally conclude with a short summary and give an outline of further research.

2 State-of-the-Art in HE Budgeting

2.1 Introduction to Budgeting in HE

Traditionally, HE institutions in Germany have mainly been state-funded. Additionally, some third-party funds, e. g. research funds by private companies or the European Union, have been provided but in comparison to the governmental funds they have been of minor importance. Therefore, HE institutions have been run like public administrations by applying “Kameralistik” principles of managing and allocating these funds [7]. Nowadays, as the importance of third-party funds rises and a trend towards performance orientation (e. g. "Exzellenzinitiative") can be observed, these management and budgeting principles are overcome. A similar situation occurred in Public Management where new ways of management principles, e. g. public-private partnerships, have evolved [8]. Accordingly, HE institutions have to find new ways to deal with this changing environment and new management issues [9].

The new environment offers HE institutions a broad financial autonomy and correspondingly a shift of decision making competence downwards from the political administration to the HE institutions. This trend towards a decentralization of decision making can also be observed within the HE institutions themselves: Faculties are entitled to manage their financial funds more autonomously [2]. Global budgets and traditional line-item budgets are increasingly being replaced by lump-sum budgeting [4]. Thus, there is a rising demand for new management guidelines in HE. While former approaches aimed at optimizing the organization, for example by means of business process redesign projects [10], the implementation of efficient budgeting principles is one challenge still broadly unsolved in HE context. As budgeting systems in HE have to consider various context-dependent factors, e. g. strategic objectives or geographical issues, existing approaches mainly refer to specific institutions making an individual alignment quite difficult. However, most budgeting systems in HE can be traced back to one general, conceptual framework that may serve as a basis for deriving individual funding principles (cf. Fig. 1).

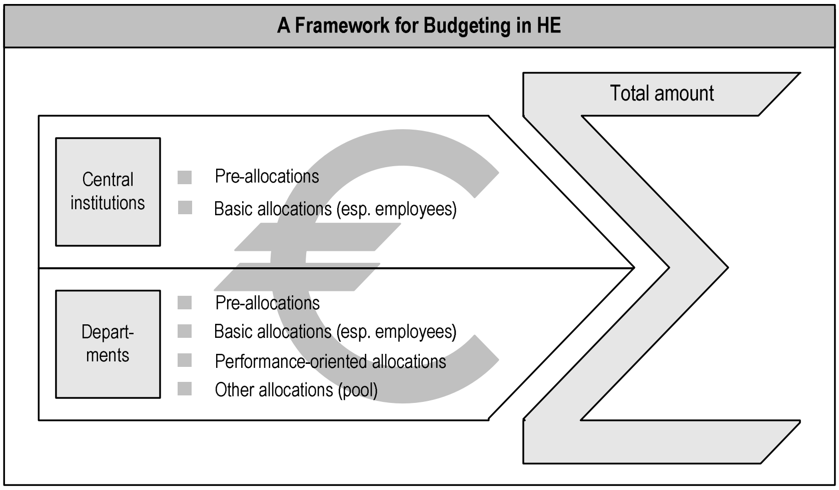

A Framework for Budgeting in HE
Total amount
Central institutions
Pre-allocations
Basic allocations (esp. employees)
Depart-ments
Pre-allocations
Basic allocations (esp. employees)
Performance-oriented allocations
Other allocations (pool)

Fig. 1. Framework for Budgeting in HE

The total amount of available funds is commonly assigned partly to the departments and partly to the central institutions (e. g. libraries or data centers). On both sides, basic allocations and so called "pre-allocations" will be conducted. Pre-allocations are generally not exactly attributable (for example energy costs). Basic allocations predominantly include personnel costs and make up for the biggest part of the entire budget. These two budgeting components are more or less time invariant due to the long term nature of the underlying cost factors. As a third element on the department level, central pool budgets are allocated. They include funds that are available only for a limited time and dedicated to special purposes. Examples for such budgets are funds for the reorganization of a department or the introduction of innovations like new software for computer-based examinations [5].

The residual element of a budgeting system in HE comprises funds to be allocated according to performance-oriented criteria. Up to now, these elements still relate to a small part of the budget but are of increasing importance [11]. They are applied in order to harmonize strategy and goals of the overall central institution with those of the decentralized faculties and departments [4]. However, a theoretically founded analysis of these instruments is still lacking [12]. Due to their growing strategic importance, we will focus on these budgets in the follow-up.

2.2 Performance-Based Funding Instruments in HE

Performance-based budgeting systems offer the potential to establish an incentive orientation within the organization in order to generate motives matching the institution's strategic objectives at decentralized level [4]. To gain a deeper insight into this matching and the underlying principles, theories in the field of psychologies can be applied. Based on the Valence-Instrumentality-Expectancy (VIE) theory by VROOM [13], the relationship between an actor's behavior and motivation can be explained. A modification of this theory that is especially suitable for the analysis of performance-oriented budgeting systems has been developed by PORTER/LAWLER [14]. According to this approach, behavior is mainly based on valence and expectancy of an actor. Both determine the effort an actor is willing to dedicate to a certain task. In combination with his abilities and perception, the performance of fulfilling a task can be observed. This performance determines the intrinsic and extrinsic rewards he will receive and therefore his level of satisfaction. Based on experience gained in the past, he will then adjust his valence and expectancy with regard to the effort necessary for a certain task. These theories propose that only if an actor is able to construct direct relations between rewards and performance, will he adjust his behavior according to the overall goals of an organization [15]. Two instruments that support a reward compatible funding are target agreements and indicator-based methods. These instruments can be distinguished on the basis of four characteristics: time of allocation, specificity, control and integration of objectives [16] (cf. Fig. 2).

	Indicator-Based Methods	Target Agreements
Time of allocation	Ex post	Ex ante
Specificity	Low	High
Control	Limited to design of formula	Consideration of individual cases
Integration of objectives		Process of acknowledgement

Fig. 2. Main Characteristics of Performance-Based Funding Instruments

Applying indicator-based methods, funds are allocated ex post, i. e. depending on the development of defined ratios. Taking these ratios (also called indicators) into account, the amount allocation is specified within a formula. Hence, budgeting is simply based on quantitative outcomes and consequently cannot consider the specific department's needs completely. Nevertheless, the indicators can be weighted differently for each department to ensure a certain degree of fairness in allocation. As a result, indicators enable an HE administration to influence the department's management by specifying an adequate formula.

Within HE context, target agreements contrariwise are usually applied in order to allocate funds ex ante. Therefore, they are especially suitable for start-up and innovation funding for example. In a process of acknowledgement, strategic objectives and ratios of measurement are specified in cooperation with the departments by taking their specific situation into account.

However, the implementation of target agreements is a highly specific, complex and time-consuming task. In contrast, indicator-based allocations conceptually work quite simple (if adequate ratios could be identified). Thus, the two instruments have been outlined to be complementary as one's advantages are the other's disadvantages [4]. Therefore, when designing a performance-oriented budgeting system for a HE institution, both instruments should be implemented at the same time. As (strategic) objectives being specified within a target agreement are usually measured by indicators as well, we will in the following concentrate on HE indicator systems in general.

2.3 Design Issues of Indicator Systems in HE

Any budgeting system can be just as efficient as its underlying indicators. The chosen indicators have to fulfill certain requirements [17]. First, the performance of all departments should be measured by the same set of indicators to ensure fairness in allocation. Furthermore, the number of indicators has to be limited and directly linked to the overall objectives in order to guarantee transparency and clarity in allocation. Another important aspect to be taken into account for reasons of acceptance is that funds being provided to departments reaching a certain performance level for the first time are allocated equally to those that have already reached this level.

As a matter of course, the disregard of these aspects may cause serious problems: The wrong set of indicators can directly result in incentive incompatibilities. Therefore, type and number of indicators have to be chosen with caution. In practice, the length of study is often regarded as an adequate indicator for measuring the quality of education, for example [18]. Within this context, departments could pick up such an indicator as a reason for decreasing their aspiration level of teaching. Thereby they ensure their individual future financial security by collectively acting inefficiently. In order to simultaneously ensure a constant quality level of teaching (by reducing study length at the same time), it has to be measured more accurately, for instance by conducting course evaluations. In conclusion, an institution's management has to choose indicators that complement one another. Furthermore, HE institutions have to arrange these ratios in a clear and transparent manner by especially restricting their number. Only then are departments enabled to understand the underlying budgeting principles. Finally, an HE institution has to decide how to apply the chosen set of indicators.

Within this context, a fix amount of money can either be allocated based on relative indicator values ("relative" as they depend on concurrent departments' values) or based on an absolute reference value (by paying a fix amount of money per realized unit) [19].

These exemplary design issues represent various new management challenges in HE budgeting. In practice, an institution usually meets these challenges by processes of acknowledgement. With this paper we suggest supporting and structuring such processes by means of conceptual modeling, especially reference modeling. Conceptual models provide a promising means to create representation forms that are suitable for discussion between all stakeholders [20]. Reference models, as specific information models, facilitate the design process of a conceptual model by reuse [20, 21, 22]. Therefore, effectiveness and efficiency of the design process are enhanced by reference models, as pre-designed measures can be reused in similar context situations.

In the following, we will present how a conceptual (reference) model for indicator-based funding in HE may be developed. In order to apply conceptual modeling in HE institutions, certain challenges have to be addressed. These comprise the selection and – if necessary – adaptation of a modeling language as well as its application in order to describe domain specific budgeting structures. These topics will be analyzed in the following chapter.

3 Conceptual Modeling in HE Budgeting

3.1 The H2 Method

In the first instance, the design of conceptual models for HE budgeting requires the selection of a suitable modeling language. Established languages for conceptual modeling comprise the Multidimensional Entity Relationship Model (ME/RM), the Application Design for Analytical Processing Technologies (ADAPT) or the Dimensional Fact Model (DFM) for example. As it has been found out that these approaches do not provide all constructs that are relevant for conceptual modeling (for more detailed information cf. [23]), in this section, we will present the H2 language (cf. Figure 3 for an introductorily overview).

H2 especially enables the documentation of ratios and dimensions in a clear and transparent manner. Therefore, it provides the possibility of specifying conceptual models that are especially suitable for HE context as they can serve as an adequate discussion basis for processes of acknowledgement [24].

H2 is based on the MetaMIS approach that has been developed to conceptually support information systems development [23]. H2 models are structured hierarchically and based on defining reference objects that belong to dimension objects. These are used to create and structure the space of analysis. To build individual excerpts out of dimensional hierarchies and in order to create task-specific views, dimension objects can be reduced to dimension scopes which further can be combined to dimension scope groupings. H2 enables the definition of ratios within ratio systems as well as their combination with dimension scope groupings in order to create information objects (which are actually representing facts) [25].

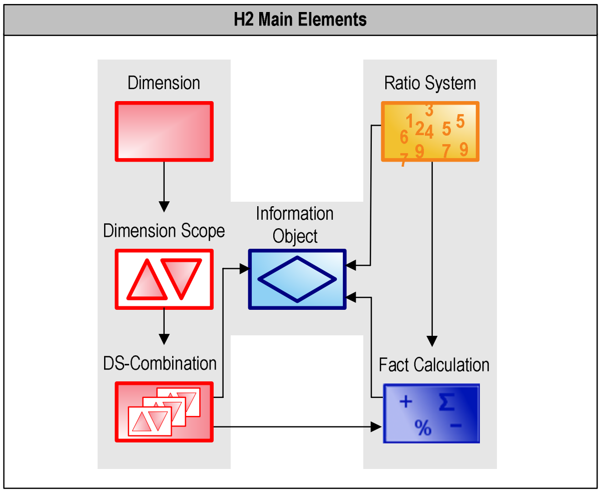


Fig. 3. H2 Main Elements

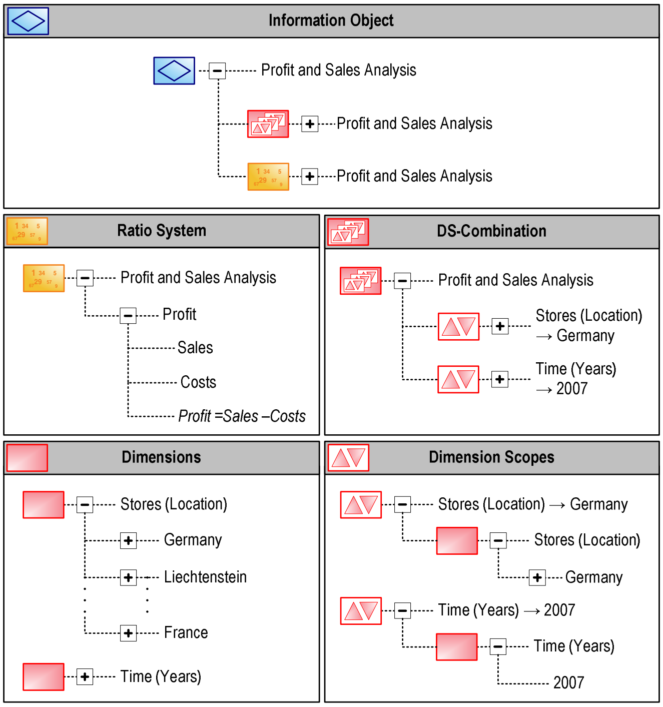


Fig. 4. H2 Example

The fact calculation object is an additional element that allows specifying an algebraic formula in addition to dimension scope groupings and ratio systems [26]. Using fact calculations, H2 enables one to refer to more than one single fact. As a result, fact calculations enable the comparison of a figure over a period of time for example.

An exemplary H2 application is provided in Figure 4 illustrating a time- and location-based sales analysis: A company's stores are arranged in a dimension taking their geographical position into account (visualized by round brackets). As an obligatory dimension, time is modeled. Both dimensions are reduced to dimension scopes (visualized by arrows) which comprise only those reference objects that are relevant for the specific analysis. Subsequent, they are combined in order to enable the analysis of all stores of a specific country (Germany) in a specific year (2007). Finally, the dimension scope grouping is assembled with a ratio system (specifying the sales-formula) to an information object. Fact calculations could be applied in order to compare the development of the sales figures over a certain period of time for example. However, this sample does not show an exemplary application of fact calculations, as in the follow-up they will be used in a manner differing from the one originally intended.

In the next sections, we analyze whether those constructs may serve as an actually suitable basis for conceptual modeling in HE context.

3.2 Adaptation of H2 Modeling Language

As indicators are defined as ratios possessing a special reference to strategic objectives [27], HE indicator systems could conceptually be specified solely based on the introduced H2 ratio system construct. However, for them to serve as an efficient and transparent discussion basis, we suggest using the entire spectrum of H2 modeling elements. As we will present in the following section, H2 provides nearly all constructs that are necessary for specifying indicator-based funding principles in a meaningful manner. However, specific adaptations are needed for this with respect to HE.

In order to support the efficiency of the model design process, case tools can be applied [28]. The H2 Toolset is a meta-modeling tool that provides the possibility of defining (or modifying) languages as well as constructing (or adapting) models that have been specified based on these languages [22, 24]. Thus, we used the toolset's language editor for modifying the language H2 to serve our purposes. In the follow-up, the resulting new language is referred to as H2 for Budgeting (cf. Fig. 5).

In HE practice, indicators are often weighted differently between departments to ensure a fair funding process: An engineering student usually incurs higher costs than an economist for instance. To balance such cost differentials, HE institutions could pay distinct amounts of money per student (considered by different indicator multipliers). H2 provides a possibility to describe such indicator weightings with the fact calculation construct (that can contain an algebraic formula, cf. above). By using fact calculations in such a semantically unusual (but syntactically correct) manner, we attempted to modify the (meta-)modeling language as sparsely as possible. Furthermore, the information object element is replaced with a new budget construct. In fact, it serves the same purpose as an information object, but allows attaching an additional formula as well as to enable the possibility of specifying additional algebraic budgeting principles. Finally, the budget accounting construct is being specified. It implicitly sums up budget elements contained within. Similar to information objects, budgets can contain a sum of fact calculations, ratio systems and dimension scope groupings. Ratio systems and dimension scope groupings, which belong to a budget element, are also associated to a fact calculation element, if it is included in that budget element as well (primarily to ensure clarity).

The other constructs provided by H2 can be applied without further modifications. In the following section, we will show how conceptual models for indicator-based budgeting could be developed on basis of H2 for Budgeting.

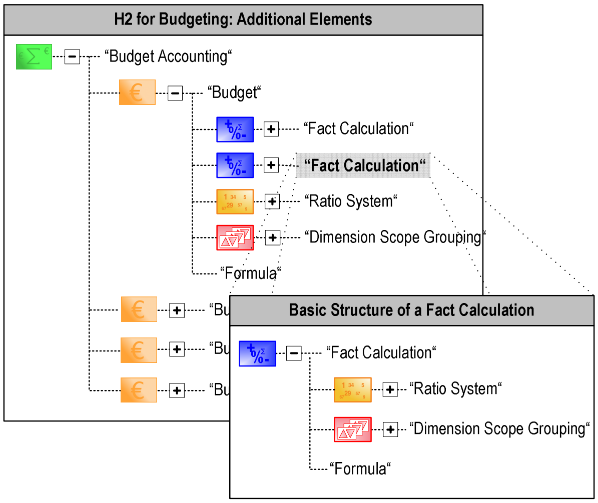
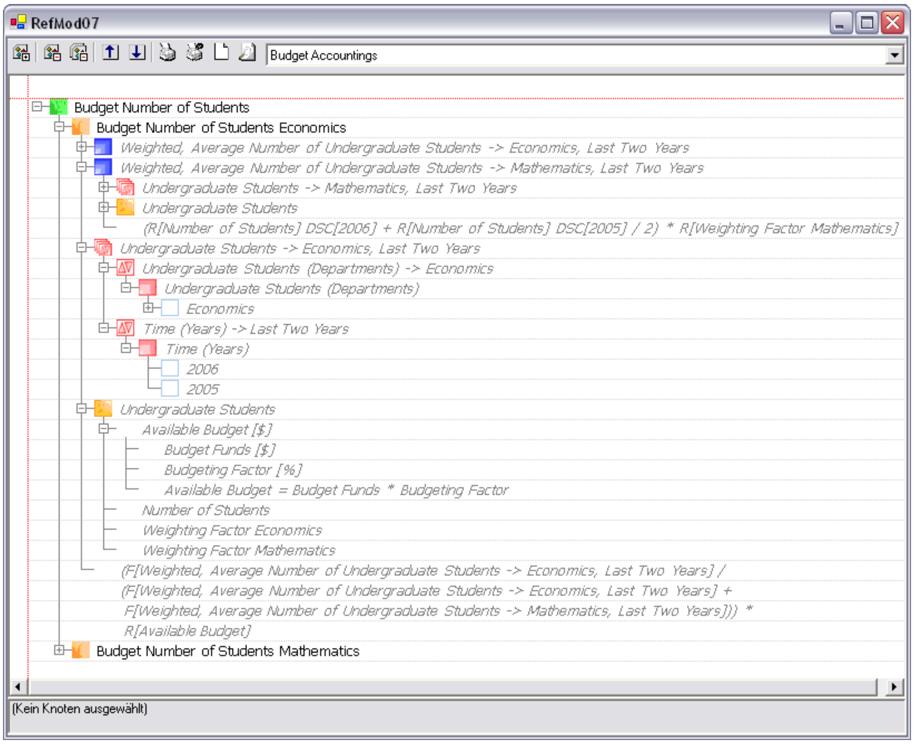


Fig. 5. H2 for Budgeting: Additional Elements

3.3 Example Application of H2 for Budgeting

For the example we assume that the model university consists only of two departments: economics and mathematics. An excerpt of the designed budgeting system is displayed in the following Fig. 6. The university's administration intends to allocate one part of the total amount of funds available in the current financial year (2007) based on the average number of undergraduate students over the last two years. Therefore, two dimensions are specified: The obligatory time dimension including reference objects representing all relevant years and a dimension containing the institution's students (arranged by the two departments). In order to particularly define the space of analysis, these two dimensions are reduced to dimension scopes taking the relevant years (2006, 2005) and students (economics, mathematics) into account. Subsequently, they are assembled to dimension scope combinations that finally specify which students actually have been part of which department in the last two years. Accordingly, an adequate ratio system is defined. It includes basic ratios (e. g. number of students) and a composed ratio representing the available budget as part of the annual budget. To consider different weightings between the departments (for instance, the economic department could be rewarded with twice the amount of money for each student in comparison to the mathematics department), the ratio system contains two weighting factors as well. The weighted budgeting process is finally specified based on two fact calculations referring to the introduced dimension scope


RefMod07

Budget Accountings

```
Budget Number of Students
  Budget Number of Students Economics
    Weighted, Average Number of Undergraduate Students -> Economics, Last Two Years
    Weighted, Average Number of Undergraduate Students -> Mathematics, Last Two Years
      Undergraduate Students -> Mathematics, Last Two Years
      Undergraduate Students
      (R[Number of Students] DSC[2006] + R[Number of Students] DSC[2005] / 2) * R[Weighting Factor Mathematics]
    Undergraduate Students -> Economics, Last Two Years
      Undergraduate Students (Departments) -> Economics
        Undergraduate Students (Departments)
          Economics
      Time (Years) -> Last Two Years
        Time (Years)
          2006
          2005
    Undergraduate Students
      Available Budget [$]
        Budget Funds [$]
        Budgeting Factor [%]
        Available Budget = Budget Funds * Budgeting Factor
      Number of Students
      Weighting Factor Economics
      Weighting Factor Mathematics
    (F[Weighted, Average Number of Undergraduate Students -> Economics, Last Two Years] /
    (F[Weighted, Average Number of Undergraduate Students -> Economics, Last Two Years] +
    F[Weighted, Average Number of Undergraduate Students -> Mathematics, Last Two Years])) *
    R[Available Budget]
  Budget Number of Students Mathematics
```

(Kein Knoten ausgewählt)

Fig. 6. Exemplary H2 for Budgeting Application

combinations. Both calculate the average number of students for each department and multiply them with these two weighting factors. As a result, they specify the weighted, average number of students during the last two years for each department.

To finally specify the resulting budgets, the new budget construct is applied. As funds are allocated relatively in order to enhance competition between the two departments, both fact calculations are included in each budget construct considering the performance of the economics and the mathematics departments.

4 Summary and Outlook

In this paper we analyzed ways of making use of conceptual modeling for performance-based funding in HE by applying H2. Therefore, we first introduced principles of performance-based funding instruments and identified indicator systems as a vital factor for the success of budgeting in HE. On that basis, we derived specific design issues to be taken into account when developing such indicator systems. Within that context, we showed that conceptual models can support the efficiency of the design process essentially as they may serve as a meaningful discussion basis for the institutions' management. Subsequently, we applied the H2 Toolset to specify a language suitable for HE context. Applying this language, called H2 for Budgeting, we presented some exemplary model contents to demonstrate how a conceptual model of a specific budgeting system may be developed.

Further research will focus on the applicability of findings to a more general context. For that purpose, particularly design principles of reference modeling like configuration, instantiation, specialization, aggregation and analogy will play an essential role in order to derive context-specific models [21]. Especially the possibility of applying mechanisms of configurative reference modeling seems to be a promising means for deriving specific models suitable for individual HE contexts [21, 22, 29]. As the H2 Toolset supports this technique [24], the reference model could be parameterized: In Germany's HE, performance-oriented funding depends on geographical issues at first. The federal states ("Bundesländer") usually use indicators such as third-party funds, alumni and number of students or research associates [30]. In most cases, the universities apply these indicators for their internal financial management as well as in order to align objectives of departments and federal state. Hence the state of the university can serve as an essential configuration parameter.

References

1. Hüfner, K.: Governance and Funding of Higher Education in Germany. Higher Education in Europe 28(2), 145–163 (2003)
2. Johnstone, D.B., Arora, A., Experton, W.: The Financing and Management of Higher Education: A Status Report on Worldwide Reforms. State University of New York at Buffalo (1998)
3. Liefner, I.: Funding, Resource Allocation, and Performance in Higher Education Systems. Higher Education 46(4), 469–489 (2003)
4. Ziegele, F.: Financial Autonomy of Higher Education Institutions: The Necessity and Design of an Institutional Framework. für Hochschulentwicklung, C., (ed.) Working Paper No. 17. Gütersloh (1998)
5. Ziegele, F.: Formelgebundene Budgetzuweisung und Zielvereinbarungen. Cordes, J., Roland, F., Westermann, G. (eds.): Hochschulmanagement – Betriebswirtschaftliche Aspekte der Hochschulsteuerung. Wiesbaden (2001) 189-205
6. Becker, J., Niehaves, B.: Epistemological Perspectives on IS Research – A Framework for Analyzing and Systematizing Epistemological Assumptions. Information Systems Journal 17(2), 197–214 (2007)
7. Kreysing, M.: Autonomy, Accountability, and Organizational Complexity in Higher Education: The Goettingen Model of University Reform. Journal of Educational Administration 40(6), 552–560 (2002)
8. Pollitt, C., Bouckaert, G.: Public Management Reform – A Comparative Analysis, 2nd edn. Oxford (2004)
9. Tsichritzis, D.: Reengineering the University. Communications of the ACM 42(6), 93–100 (1999)
10. Allen, D.K., Fifield, N.: Re-engineering Change in Higher Education. Information Research, Vol 4(3) [2007-06-01] (1999), http://informationr.net/ir/4-3/paper56.html
11. Huisman, J.: Higher Education in Germany. In: CHEPS (Center for Higher Education Policy Studies), country report. Enschede (2003)
12. Jaeger, M.: Leistungsorientierte Budgetierung: Analyse der Umsetzung an ausgewählten Universitäten und Fakultäten/Fachbereichen. Hochschul-Informationssystem (ed.), Kurzinformation A1. Hannover (2006)

13. Vroom, V. H.: Work and Motivation. New York, London, Sydney (1964)
14. Porter, L.W., Lawler, E.E.: Managerial Attitudes and Performance. Homewood (1968)
15. Porter, L.W., Lawler, E.E., Hackman, J.R.: Behaviour in Organisations. New York (1975)
16. Ziegele, F.: Das Zusammenwirken von formelgebundener Mittelzuweisung und Zielvereinbarung. In: Das Präsidium der Carl von Ossietzky Universität Oldenburg (ed.): Globalhaushalt in Niedersachsen – Erfahrungen, Beurteilungen, Perspektiven. Hochschulentwicklungsplanung, Oldenburg, vol. 41, pp.175-188 (1999)
17. Ziegele, F.: Indikatorengestützte Mittelvergabe. In: Hanft, A. (ed.): Grundbegriffe des Hochschulmanagements. Bielefeld, pp. 195-201 (2004)
18. Berens, W., Hoffjan, A.: Controlling in der öffentlichen Verwaltung: Grundlagen, Fallstudien, Lösungen. Stuttgart (2004)
19. Körber-Weik, M.: Indikatorsteuerungen – Durch neue Finanzierungsformeln zu besseren Hochschulen und wirksamerer Frauenförderung. In: Roloff, C. (ed.) Reformpotential an Hochschulen. Frauen als Akteurinnen in Hochschulreformprozessen, Berlin, pp. 153–178 (1998)
20. Frank, U.: Conceptual Modelling as the Core of the Information Systems Discipline – Perspectives and Epistemological Challenges. In: Haseman, W.D., Nazareth, D.L., Goodhue, D.L. (eds.) AMCIS 1999. Proceedings of the 5th America's Conference on Information Systems, Milwaukee, pp. 695–697 (1999)
21. vom Brocke, J.: Design Principles for Reference Modeling: Reusing Information Models by Means of Aggregation, Specialization, Instantiation, and Analogy. In: Fettke, P., Loos, P. (eds.): Reference Modeling for Business Systems Analysis, 47–75 (2007)
22. Becker, J., Delfmann, P., Knackstedt, R.: Adaptive Reference Modeling. Integrating Configurative and Generic Adaptation Techniques for Information Models. In: Becker, J., Delfmann, P. (eds.) Reference Modeling. Efficient Information Systems Design Through Reuse of Information Models, Berlin, pp. 23–49 (2007)
23. Becker, J., Janiesch, C., Pfeiffer, D., Seidel, S.: Evolutionary Method Engineering – Towards a Method for the Analysis and Conception of Management Information Systems. In: AMCIS 2006. Proceedings of the 12th America's Conference on Information Systems, Acapulco, pp. 3922–3933 (2006)
24. Delfmann, P., Janiesch, C., Knackstedt, R., Rieke, T., Seidel, S.: Towards Tool Support for Configurative Reference Modeling – Experiences from a Meta Modeling Teaching Case. In: WoMM 2006. Proceedings of the 2nd Workshop on Meta-Modelling and Ontologies, Lecture Notes in Informatics, Karlsruhe, pp. 61–83 (2006)
25. Holten, R.: The MetaMIS Approach for the Specification of Management Views on Business Processes. Department for Information Systems, University of Munster (ed.), Working Paper No. 84. Munster (2001)
26. Holten, R., Dreiling, A.: Specification of Fact Calculations within the MetaMIS Approach. Department for Information Systems, University of Munster (ed.), Working Paper No. 88. Munster (2002)
27. Brüggemeier, M.: Controlling. In: Hanft, A. (ed.) Grundbegriffe des Hochschulmanagements, Bielefeld, pp. 58–67 (2004)
28. Schütte, R.: Guidelines for Reference Modeling. Wiesbaden (1998)
29. van der Aaalst, W.M.P., Dreiling, A., Gottschalk, F., Rosemann, M., Jansen-Vullers, M.H.: Configurable Process Models as a Basis for Reference Modeling. In: BPRM 2005. Paper presented at the First International Workshop on Business Process Reference Models, Nancy (2005)
30. Küpper, H.: Management Mechanism and Financing of Higher Education in Germany. Higher Education Management and Policy 15, 71–89 (2003)

Towards a Reference Process Model for Event Management

Oliver Thomas, Bettina Hermes, and Peter Loos

Institute for Information Systems (IWi)
at the German Research Center for Artificial Intelligence (DFKI), Saarland University,
Stuhlsatzenhausweg 3, Building D3 2, 66123 Saarbrücken (Germany)
{oliver.thomas,bettina.hermes,peter.loos}@iwi.dfki.de
http://www.iwi.dfki.de

Abstract. Events are becoming more and more important for companies as an instrument of marketing communication. Event management is an interdisciplinary task field, addressed in the most diverse fields in practice and in research establishments. Because careful preliminary planning and precise execution are extremely important for events, modeling languages can contribute greatly to the systematic design of event management systems. Accordingly, this article will make recommendations for application system and organization-design in the form of a reference process model for event management.

Keywords: Event marketing, Event management, Business process, Process model, Process modeling, Reference model, Reference modeling.

1 Events as a Trend

In the last years, events have been given more and more attention in research and practice. As a result, a separate, special branch of service geared to events has developed in which event agencies, trade fair constructors, talent agencies and sound and light engineers, etc. are involved in the organization and creation of events. Numerous studies attest a high potential to events as communication instruments and forecast not only quantitative, but also qualitative growth for the event market [1; 9; 12].

Due to the high significance of events in practice, it does not come as a surprise that the scientific world has begun to address the phenomenon of the "event". Noteworthy results have particularly been achieved in marketing and tourism-management [8; 10]. One of the most important insights gained by the research done since the end of the 1980ies is that the management of events must be seen as an interdisciplinary task field requiring effective and efficient cooperation between diverse partners. The strategic preparation, as well as the planning and coordination of the execution of an event require professional handling in order to guarantee the optimal interplay between all participants. Support from modern information and communication systems for this process, summarized here under the term "event management", is a good idea and offers many starting points [16].

Although for several years now, an established approach for the support of a systematic procedure for the analysis, improvement, implementation and control of business processes using information modeling [17; 19; 25] exists, up to now, the design of event management processes, as well as the systematic development of supportive information systems have not occurred. It therefore appears wise to make recommendations for application system and organization-design in the form of a reference process model. The construction of such a model is the topic of this article.

The article is structured in the following manner: Section 2 lays a terminological foundation by differentiating between the terms “event”, “event marketing” and “event management”. A study of the event management process, as well as the model-based development of supportive information systems is the topic of Sections 3 and 4. Following this, in Section 5, the fields “event management” and “reference modeling” are brought together, the requirements for a reference model for event management are defined and the procedure necessary for the creation of such a reference model is determined (construction process). The construction of this reference process model then takes place in Section 6 (construction results). The article closes with a conclusion in Section 7.

2 From Event to Event Management

The every-day and scientific uses of the term “event” do not coincide with each other. Different terms and definitions for “event” have developed in various areas of life and research. In research, this especially leads to communication and comprehension problems. In a first approach, one can understand events as “temporary occurrences, either planned or unplanned” [8, p. 4]. In order to emphasize the difference between planned and unplanned occurrences, the term “special” is added to “event”. A special event is understood to be a “one-time or infrequently occurring event outside a normal program” [8, p. 4]. Often events are classified, in order to better deal with the term. Thus for example, a one-dimensional classification in “hallmark events” (traditional events that take place at a certain location, such as e.g. Mardi Gras in New Orleans) and “mega events” (e.g. the Olympic Games) is possible [8, pp. 3–4].

The activities connected with the planning and control of events are generally summarized under the terms “event marketing” or “event management”. When differentiating between these terms literature on the field argues that event marketing deals with the marketing-theoretical foundations of the phenomenon “event” and in doing so, observes aspects such as visitor motivation and perception or effects on image. Event management on the other hand, emphasizes questions of planning, as well as the quality, personnel and risk management for the event [10, p. 311].

It slowly becomes clear in the search for a definition of the term “event management”, that there is no consensus about the term and the activities connected with it in literature. Often, only the organizational and controlling measures necessary for the ultimate/execution of an event are understood as event management [6; 11]. This however, neglects the strategic alignment of management with its integrative tasks and contradicts the established term “management”, which grants extraordinary decision-making possibilities to those responsible.

In addition, it is important to mention that, as a rule, schemes for the planning and execution of events exhibit two typical characteristics. *First*, they begin with the definition of requirements for an event and end with its conclusion. They are thus limited in time and have a clear start and finish point. And *second*, these ventures are often one-time initiatives in which various internal and external organizations participate. Due to these two characteristics, it is generally said that the processes for the planning and execution of events possess project character. This interpretation of events as projects is based on established definitions of the term "project". Most authors see the time limitation (clearly defined start and finish points), as well as the singularity of an event as distinct project characteristics. These project-characteristics of events are often neglected in the search for a definition in literature.

As a result of these considerations, the following working definition will be used here: *event management* comprises the coordination of all of the tasks and activities necessary for the execution of an event regarding its strategy, planning, implementation and control, based on the principles of event marketing and the methods of project management.

3 Event Management Systems

In addition to general planning activities, it is important to observe aspects regarding information transparency, documentation and controlling possibilities and the exchange and storage of information in order to guarantee comprehensive support for all of the activities and participants in the entire event management process. Proprietary software solutions for word processing, spreadsheets, project management or e-mail-communication do not provide an integrated approach for event management. In addition to the established standard applications, there are application systems geared to special domains, such as for example, gastronomy or ticket systems. These however only provide special functionalities, such as calendars, solutions for the scheduling of rooms, possibilities for storing additional information or solutions for visitor registration [16]. Up to now, no comprehensive IT-support exists for the entire event management process, from the initial idea to its integration in corporate strategy and the conclusion of the event supporting the workflow from strategic planning to event controlling.

The potential of such a software solution lies in the fact that it provides the highest possible information and cost transparency. The increase in efficiency and effectiveness, which would result from the use of such a tool for planning, carrying out and controlling an event can be seen analogue to the use of corresponding systems in supply chain management. Thus, in addition to improved coordination and communication among the participants involved in the process, for example, event agencies and service providers, the customer — respectively the sponsor of the event — also profits from improved transparency. Decisions regarding possible changes can be made more quickly and cost efficiently, because the channels of communication are much shorter which allows information to be exchanged more quickly.

In addition, topics such as controlling or risk management are becoming increasingly interesting for the planning of events. Existing approaches have concentrated on the economic evaluation of an event after its conclusion [4, pp. 2f.]. Often however, it is required that controlling measures can be carried out in all phases of the event management process, in order to guarantee the sustainability of an event. Thus, adequate alternatives for the documentation and provision of appropriate controlling methods are needed and these can only be guaranteed with the appropriate tool-support.

Event management systems, understood here as information systems used for the support of managing events, must function as an intermediary between the business frameworks of event marketing, management and information technology. Because event management systems work on both a business and a technical level, they are — as are generally all information systems — very complex. With the help of a model, we will attempt to create manageable artifacts that make the complexity of these information systems controllable.

4 Modeling Event Management Systems

Information models have established themselves as a medium for bridging the gap between business problems and the realization of an application system. The application possibilities of information models range from software design and the introduction and configuration of standard software to business process reengineering.

Due to the possibility of their reutilization, in many cases the construction of models is connected to the demand to abstract from enterprise-specific characteristics. One must thus differentiate between enterprise-specific information models and reference models. The term "enterprise-specific" characterizes only the individual character of the corresponding model; there is no restriction to legally independent companies connected with it. Thus, due to reasons of linguistic clarity, one must speak of specific models in order to allow for the fact that the specificity of models does not only result from a enterprise-context alone, but rather, for example, also from a project-context. To emphasize this context one can also speak of project-specific models.

In contrast to this, a reference model for the development of specific models constitutes a point of reference, because it represents a class of applications [23]. On the one hand, the possibility of orienting oneself on the technical content of such reference models promises the model-users savings in time and costs, while on the other, the quality of the model to be constructed, and thus the quality of the software based on this model, can be increased by the use of a reference model. The fundamental idea in reference modeling to save process knowledge in models in order to use it at a later point in time, has currently been recognized by event management literature. Thus for example, SCHWANDNER states that "it is almost always better to adopt good ideas from others, follow their tips and then optimize them for your individual needs" [22, p. 27]. Nevertheless, at present there are no reference model-based design recommendations for event management systems. This shortcoming is the result of the following problems:

1. *Lack of process orientation:* Business research in the field of event marketing and event management continues to neglect process management aspects for events: "Less research has been focused on special events operational management" [10, p. 322]. Research has primarily dealt with questions regarding the cultural, social and economic effects of events. A perspective integrating all of the aspects of event management is lacking [18, p. 86].

2. *Lack of standardized forms of representation:* Marketing-oriented research concentrates on explaining interdependencies, which are compiled, as a rule, by way of market research studies. In addition, exemplary descriptions and suggestions for the management of events dominate in literature. The forms of representation used here are hardly standardized and limit the significance of the introduced concepts which makes an application-specific adaptation difficult [15, pp. 219f.]. There are only a few cases where generally accepted methods for example, from project management, were deductively transferred to the field of event management [18].
3. *Lack of models:* It is primarily practice-oriented analyses, dealing with the planning and organization of events, which focus on additional benefits in the form of check lists, tables, forms and road maps [6; 11]. Demonstrative, exemplary representations, customary in the field of information modeling, are quite rare.

The following analyses will attempt to solve these problems by way of reference model-based design recommendations for event management.

5 Requirements for Reference Models in Event Management

5.1 Existing Reference Models in Research and Practice

There are many reference models in literature for many different fields of application – for a current tabular overview cp. [7, p. 46f.]. While early approaches oriented themselves on the representation of aspects from all possible enterprises, the authors of current constructions often assign their reference models to concrete economic branches. Prominent examples of this are the reference model for industrial business processes from SCHEER [20] and the Retail Information Systems from BECKER, SCHÜTTE [3], which both come from the research field.

In practice, reference models can be found by providers of modeling tools and consulting firms. Thus, for example, the IDS SCHEER, Inc. [www.ids-scheer.com] offers diverse reference models. These are reference models for the service sector (financial services, commercial enterprises, local governments, hospitals, mail-order businesses, municipal utility companies and insurance companies), product-oriented manufacturing (plant construction, automobile suppliers, mechanical engineering, the consumer goods industry and the furniture industry) and process-oriented manufacturing (the chemical industry and the paper industry). On the other hand, comprehensive documentation on established ERP systems exists in the form of reference models, such as for example, the SAP R/3 reference model [5]. A reference model assigned to the field of event management is however, unknown to the authors.

5.2 The Necessity of Constructing a Framework

In order to satisfy the claim for reusability in the construction of models, reference models must describe a wide range of company conditions and their interdependencies. They are, in addition, seen from different perspectives, which makes a survey-like graphic representation of reference models very complicated. The data model for the SAP R/3-reference model, for example, contains more than 4000 types of entities and the corresponding reference process model more than 1,000 business processes

[5]. The use of a framework for comprehensive reference models has shown itself to be well proven in research and practice [3; 20]. Reference model frameworks provide a directory, whose domains refer to detailed models of the reference model. The following creation of an event management reference model will therefore be divided up into the design of the framework and the construction of the reference model itself.

In contrast to the creation of detail models, modeling languages are usually not used for the construction of frameworks. Using freely defined graphic symbols model developers can illustrate the wide variety of contextual aspects of a reference model. These can also help to emphasize the trademark character of a reference model framework. Nevertheless, in addition to "established" languages (e.g. ERM, EPC), there are "simple" modeling languages in the language portfolios of the modeling tools from a few providers and these are especially geared to the construction of a framework. In the *ARIS-Toolset* for example, a Y-diagram is used for the function-oriented entrance into complex reference models. The simplicity of these languages refers to the low number of language elements and the constructible relationships between these language elements, as well as the graphic representation of the language elements using elementary geometric structures, such as lines or polygons.

By assigning the parts of a reference model to an index of the framework, the respective elements of the model are grouped according to contextual criteria. The model object upon which the construction of the reference model framework is based is the reference model. Thus, framework and reference model have a macro-micro-relationship. In this spirit, a framework is always on a "higher" aggregation level than the reference model it represents. The disaggregation of macro models can also be pursued "within" a reference model over several aggregation levels. This is especially practical for comprehensive reference models. It, however, assumes that the possibility for disaggregation in the modeling language used is embedded as a supported construction technology.

5.3 Modeling Languages for the Representation of a Reference Process Model

Although the first ideas concerning the reusability of information models date back to more than three decades, up to now, very few modeling languages have been conceived for the creation and use of reference models alone. Two of the few exceptions are the reference process modules from LANG, TAUMANN, BODENDORF [14] and the reference model component diagram from VOM BROCKE [24, pp. 235 ff.]. Most of the research in the field of reference modeling concentrates on an application or domain-specific selection of established languages for information modeling. The spectrum of reasons for the selection of these languages ranges from the basic orientation on paradigms (e.g. object-oriented or non-object-oriented) or modeling methods (e.g. ARIS or UML) to the completely uncritical und unreflected use of these languages. Occasionally, the selected modeling languages are extended.

Since the end of the 1970ies, a multitude of modeling languages has been developed to describe process models [2]. The event-driven process chain (EPC) [13; 21] has especially established itself for the construction of reference process models on a conceptual level [7]. It will be used in the following for the construction of the reference model for event management.

6 Construction of a Reference Model for Event Management

The models discussed in the following were developed at the Institute for Information Systems at DFKI, Saarland University, Saarbruecken (Germany), over a period of six months. They were created with the help of interviews and workshops in cooperation with three large German event agencies, a representative from the marketing department of an automobile company, as well as employees in an internationally active trade fair service provider. In addition to this inductive procedure for extracting knowledge, during which a multitude of actually observed process structures were described, ordered and compared, knowledge was gained from the generally accepted principles and models of the event management "theory" dealt with in business management literature — in compliance with a deductive course of action.

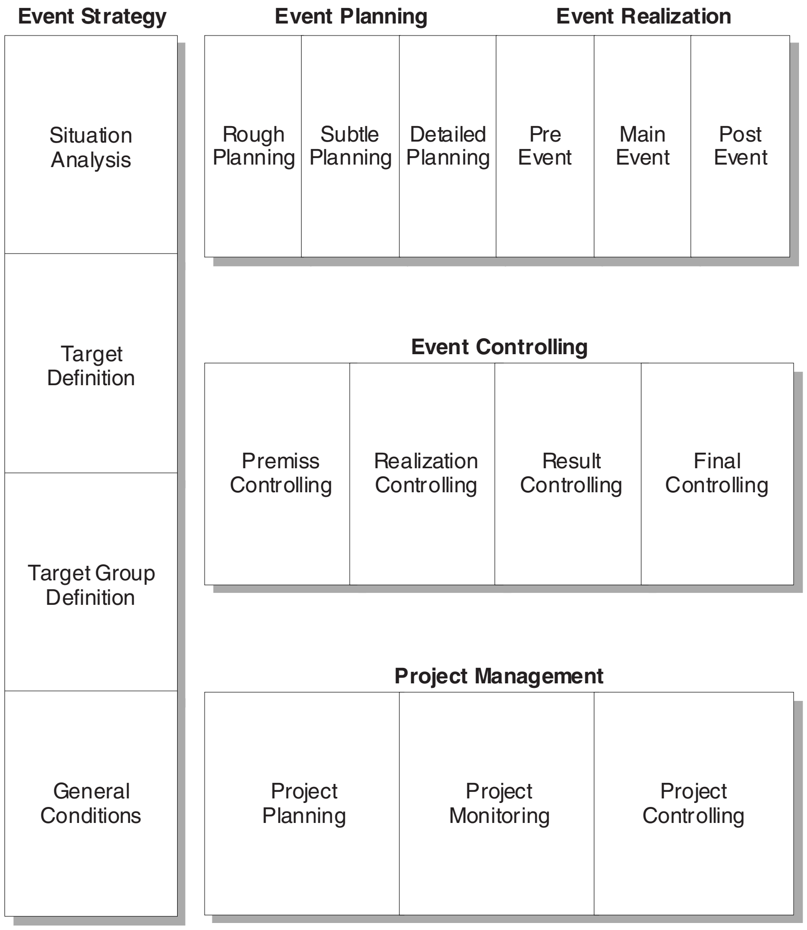


Fig. 1. Event-E – reference model framework for event management

6.1 The Construction of a Reference Model Framework

The framework for event management represented in Figure 1 and named *Event-E* due to its form, structures the activities necessary for the planning and execution of events in a coherent sequence. The framework is divided up into five domains: “Event Strategy”, “Event Planning”, “Event Realization”, “Event Controlling” and “Project Management”. It emphasizes the equality of the sub-processes for the management of events through its design. Each domain is assigned special functions (also called activities). The five levels should not be understood as independent processes. Relationships of interchange und interdependencies exist between all functions. According to the chronological sequence of the event management process, we will start with the domain “Event Strategy”. In this phase, all of the basic problems regarding the event are solved in coordination with the company and marketing strategy. In the planning phase, the chronological and spatial coordination of all of the activities and participants for the event is worked out. The “Event Realization” phase comprises the actual execution of the event at the venue. The “Event Controlling” phase provides the event management team with all of the controlling methods and measures at any possible time. It plays a special role because it takes ongoing functions into account, which support the planning and execution of the event and serve the evolution of the tasks to be achieved. This phase is therefore set in the middle of the framework. The “Project Management” phase forms the knowledge basis for the planning of the entire event management process and thus, forms the foundation for the execution of all types of events along all phases of the event management [18].

With the help of the areas and functions identified by it, the framework makes a recommendation for a procedure for projects where events are planned and/or carried out. Because this procedure can vary in practice, the framework must be adapted to the respective project. The event management framework could then be referred to as a reference model due to its reusability.

6.2 The Construction of Detail-Models

A strategy describes a precisely planned course of action for a project, i.e. it serves as a foundation for further planning. Complete, strategic preparatory work is highly important for the event manager. An EPC-reference model for the event strategy is represented in Figure 2. The two start-events represented in the model illustrate the fact that the event management process can begin within a company or be assigned by a customer to a service provider, e.g. an event agency.

Within the framework of a comprehensive situation analysis, the goals and target groups of the event are defined. In order to carry out an evaluation of the event at a later point in time, the measurability of the goals must be guaranteed. To do so, the goals can be divided up into in strategic and operative goals. Economic goals are also formulated to make financial success measurable. This can comprise increases in sales, increases in market shares or an increase in buying intensity, in addition to the revenues directly relevant for the event. Contact goals can, for example, be operationalized through the number of registrations or participants.

Event goals are connected to a company‘s communication politics via the event-marketing strategy and thus, directly connected with the superordinate corporate strategy. The derivation of an event‘s target structure must be compared with the corporate strategy‘s guidelines. If discrepancies arise, they must be revised.

The narrowing down of the target group is also closely connected with the definition of goals. As a rule, primary and secondary target groups are defined for events (cp. Figure 2). The primary target group is seen as all groups of persons taking part in an event directly. The secondary target group is integrated into the event through media or other forms of communication. Usually, the secondary target group consists of the public not directly taking part in the event. Additional information is collected

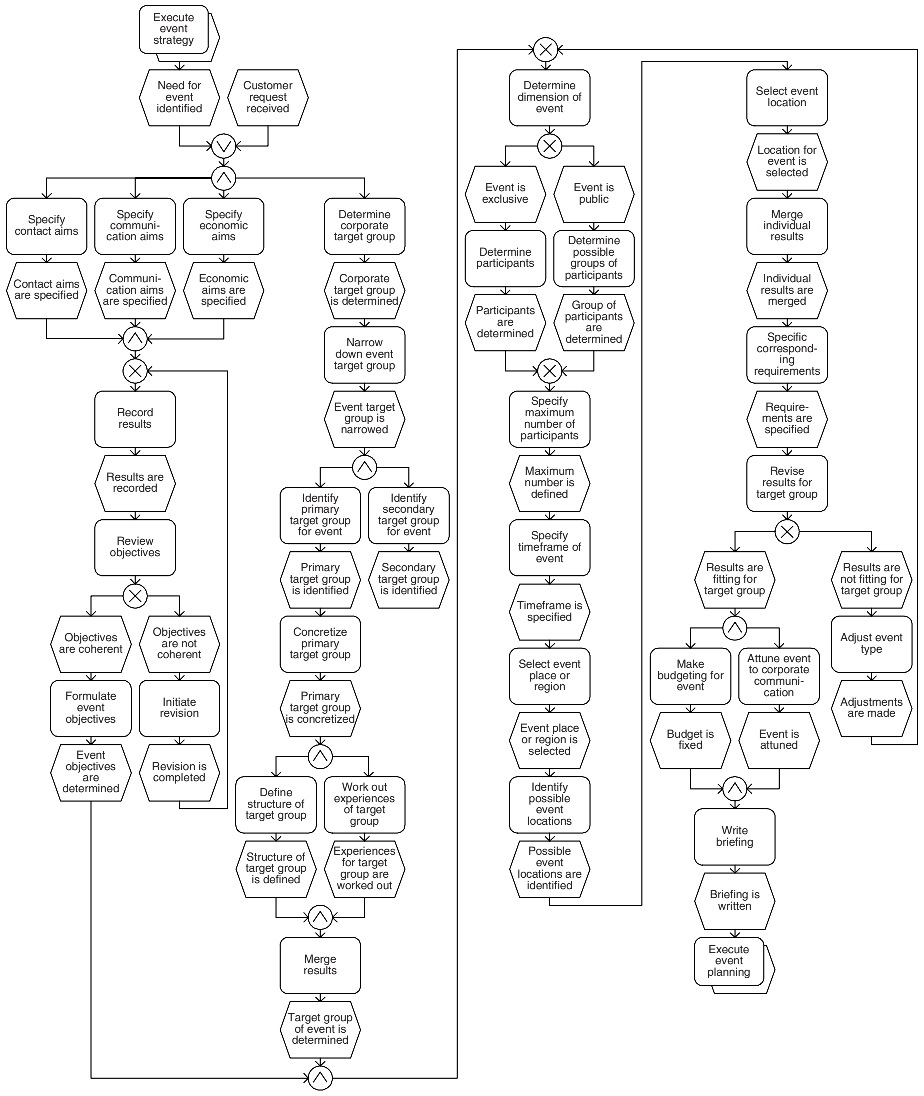

Execute event strategy
Need for event identified
Customer request received
Specify contact aims
Specify communi-cation aims
Specify economic aims
Determine corporate target group
Contact aims are specified
Communi-cation aims are specified
Economic aims are specified
Corporate target group is determined
Narrow down event target group
Event target group is narrowed
Record results
Results are recorded
Review objectives
Objectives are coherent
Objectives are not coherent
Formulate event objectives
Initiate revision
Event objectives are determined
Revision is completed
Identify primary target group for event
Identify secondary target group for event
Primary target group is identified
Secondary target group is identified
Concretize primary target group
Primary target group is concretized
Define structure of target group
Work out experiences of target group
Structure of target group is defined
Experiences for target group are worked out
Merge results
Target group of event is determined
Determine dimension of event
Event is exclusive
Event is public
Determine participants
Determine possible groups of participants
Participants are determined
Group of participants are determined
Specify maximum number of participants
Maximum number is defined
Specify timeframe of event
Timeframe is specified
Select event place or region
Event place or region is selected
Identify possible event locations
Possible event locations are identified
Select event location
Location for event is selected
Merge individual results
Individual results are merged
Specific correspond-ing requirements
Require-ments are specified
Revise results for target group
Results are fitting for target group
Results are not fitting for target group
Make budgeting for event
Attune event to corporate communi-cation
Adjust event type
Budget is fixed
Event is attuned
Adjustments are made
Write briefing
Briefing is written
Execute event planning

Fig. 2. Reference process model for the event strategy

within the function “concretize primary target group”. This information allows one to derive the structure of the target group, as well as experience values about the target group. The definition of the target group-structure extends beyond the registration of age, residence and purchasing power. In fact, more differentiated methods must be consulted, such as for example, lifestyle-groupings or scene marketing. Detailed knowledge about the target group-structure guarantees a high degree of individuality and thus, high contact intensity.

The concretion of the event type and the general conditions for the event follow the definition of the goals and target groups for the event (cp. Figure 2). First, the size of the event is defined. A decision is then made about whether the event is exclusive or open to the public. If a decision is made for an exclusive event, then the number of participants must be determined. This number may tend to vary more strongly for public events than for exclusive events. Therefore, all possible participant groups must be determined for public events. In addition, one must also narrow down the maximum number of participants. All of the following planning, such as the selection of the venue or catering, is oriented on this information. Following this, the exact timeframe for the event must be defined. Here the first dates are set. Events can be held for a day (e.g. a gala or anniversary), several days (e.g. Olympic Games or conferences) or in cycles (e.g. concerts or shows). The location is then selected based on this data. While for example, a concert hall is understood as a venue, the term “location” refers to the geographic area where the event takes place, for example, “the city of Berlin and its surrounding area”.

The individual results regarding the size, timeframe and location of the event are then combined (cp. Figure 2). Requirements for the event are then made based on this data. These requirements then serve, in turn, as a basis for further planning. A comparison of this data with the goals and target groups for the event should secure the consistency of the coming event. If a “non-fit” occurs (e.g. a gym was selected as the venue for an anniversary with senior managers of a company), then the process for the specification of the event-type and the general conditions must be run through again, in order to achieve a match (cp. the loop in Figure 2). In the case of a “fit”, the sub-process is concluded. The results from the event strategy phase are then recorded in a briefing after a final tuning with the superordinate strategic requirements and an initial budget for the event.

7 Conclusion

The topic of this article was the construction of a reference process model for event management. The reference model makes recommendations for the design of process-oriented information systems, which serve to support event management. The merging of the two separately developed fields of research intended here — on the one hand, event marketing resp. event management as a discipline of business economics and on the other, reference modeling as a discipline of information systems research — is new for two reasons: first, up to now, there have been no noteworthy research results on the modeling of event management systems. And second, the construction results in this article are a reaction to the often-criticized lack of reusable domain models in the field of reference modeling.

The construction of the reference model was — as is customary in reference modeling — divided up into the creation of a framework, the Event-E, and the modeling of detail models assigned to the domains of this framework. While the construction of the detail model with the EPC was based on an established domain-independent process modeling language, the motivations for the structure of Event-E showed that thoughts with symbolic character dominate in the construction of the reference model framework. This results in a symbolization of the relationships described by the model for the observer. The train of thought is geared towards the respective subject area and can therefore not be expressed using application domain-independent modeling languages. In addition to their purpose of structuring models, frameworks also serve as trademarks for the reference models assigned to them.

References

1. The George P Johnson Company (ed.): Trends in U.S. Marketing 2001–2002. Detroit (2002)
2. Dumas, M., van der Aalst, W.M.P., ter Hofstede, A.H.M. (eds.): Process-aware Information Systems: Bridging People and Software through Process Technology. Wiley, Chichester (2005)
3. Becker, J., Schütte, R.: Handelsinformationssysteme: Domänenorientierte Einführung in die Wirtschaftsinformatik, 2nd edn. Moderne Industrie, Landsberg/Lech (2004)
4. Clarke, A.: Evaluating Mega-Events: A Critical Review. In: Proceedings of the 3rd DeHaan Tourism Management Conference The Impact and Management of Tourism-Related Events, University of Nottingham (December 14, 2004)
5. Curran, T.A., Keller, G., Ladd, A.: SAP R/3 business blueprint: Understanding the business process reference model. Prentice-Hall, Englewood Cliffs (1998)
6. Erber, S.: Eventmarketing: Erlebnisstrategien für Marken, 3rd edn. Moderne Industrie, Redline Wirtschaft bei Verl (2002)
7. Fettke, P., Loos, P.: Classification of Reference Models – A Methodology and its Application. Information Systems and e-Business Management 1, 35–53 (2003)
8. Getz, D.: Event Management & Event Tourism. In: Cogn. Comm. Corp., New York (1997)
9. Goldblatt, J.J.: A future for Event Management: The analysis of major trends impacting the emerging profession. In: Allen, J., Harris, R., Jago, L.K., Veal, A.J. (eds.) Events beyond 2000. Proceedings of Conference on Event Evaluation, Research and Education, Australian Centre for Event Management, Sydney, pp. 1–9 (2000)
10. Hede, A.-M., Jago, L.K., Deery, M.: Special Event Research 1990–2001: Key Trends and Issues. In: Australian Center for Event Management (ed.) Events & Place Making: Event Research Conference, Sydney. UTS, July 15–16, 2002, pp. 305–338 (2002)
11. Holzbaur, U., et al.: Eventmanagement: Veranstaltungen professionell zum Erfolg führen, 2nd edn. Springer, Heidelberg (2003)
12. Jago, L.K., Shaw, R.N.: Special events: A conceptual and differential framework. Festival Management and Event Tourism 5(1/2), 21–32 (1998)
13. Keller, G., Nüttgens, M., Scheer, A.-W.: Semantische Prozeßmodellierung auf der Grundlage Ereignisgesteuerter Prozeßketten (EPK). In: Scheer, A.-W. (ed.) Veröffentlichungen des Instituts für Wirtschaftsinformatik, Saarbrücken, vol. 89 (1992)
14. Lang, K., Taumann, W., Bodendorf, F.: Business Process Reengineering with reusable Reference Process Building Blocks. In: Scholz-Reiter, B., Stickel, E. (eds.) Business process modelling, pp. 265–290. Springer, Berlin (1996)

15. Larson, M.: Evenemangsmarknadsföringens organisering: Interaktion mellan aktörer på ett politiskt torg. Göteborg: Handelshögskolan, Universitet (2003)
16. Luppold, S.: EDV in der Veranstaltung. In: Haase, F., Mäcken, W. (eds.) Handbuch Event-Management, München: kopaed, pp. 129–143 (2004)
17. Mylopoulos, J.: Information Modeling in the Time of the Revolution. Information Systems 23(3/4), 127–155 (1998)
18. O'Toole, W.: Towards the integration of Event Management Best Practice by the Project Management Process. In: Allen, J., et al. (eds.) Events beyond 2000: Proceedings of Conference on Event Evaluation, Research and Education, July 2000, pp. 86–92. Australian Centre for Event Management, Sydney (2000)
19. Rossi, M., Siau, K.: Information Modeling in the new Millennium. Hershey (2001)
20. Scheer, A.-W.: Business Process Engineering: Reference Models for Industrial Enterprises, 2nd edn. Springer, Berlin (1994)
21. Scheer, A.-W., Thomas, O., Adam, O.: Process Modeling Using Event-driven Process Chains. In: Dumas, M., van der Aalst, W.M.P., ter Hofstede, A.H.M. (eds.) Process-aware Information Systems: Bridging People and Software through Process Technology, pp. 119–145. Wiley, Hoboken (2005)
22. Schwandner, G.: Grundlagen – Projektmanagement und Organisation. In: Haase, F., Mäcken, W. (eds.) Handbuch Event-Management, München, kopaed, pp. 27–45 (2004)
23. Thomas, O.: Understanding the Term Reference Model in Information Systems Research: History, Literature Analysis and Explanation. In: van der Aalst, W.M.P., et al. (eds.) BPM 2005. LNCS, vol. 3649, pp. 484–496. Springer, Berlin (2006)
24. vom Brocke, J.: Referenzmodellierung: Gestaltung und Verteilung von Konstruktionsprozessen. Berlin: Logos (2003)
25. Wand, Y., Weber, R.: Research Commentary: Information Systems and Conceptual Modeling – A Research Agenda. Information Systems Research 13(4), 363–376 (2002)

Semantics Workshop

Introduction to the 2nd Edition of the Workshop "Advances in Semantics for Web Services 2007" (Semantics4ws 2007)

These proceedings contain the papers accepted for the 2nd edition of the workshop "Advances in Semantics for Web services 2007" (semantics4ws 2007). The workshop was held in Brisbane, Australia, on September 24, 2007, in conjunction with the Fifth International Conference on Business Process Management (BPM 2007).

The main topics of the semantics4ws workshop series are related to the applicability of semantic technologies to Web services. Web services have added a new level of functionality to the current Web by taking a first step towards seamless integration of distributed software components using Web standards. Nevertheless, current Web service technologies around SOAP, WSDL and UDDI operate at a syntactic level and, therefore, although they support interoperability (i.e., interoperability between the many diverse application development platforms that exist today) through common standards, they still require human interaction to a large extent. For example, the human programmer has to manually search for appropriate Web services in order to combine them in a useful manner, which limits scalability and greatly curtails the added economic value envisioned with the advent of Web services.

Recent research (to which we refer to as Semantic Web Services - SWS), which draws on a variety of fields such as the Semantic Web, knowledge representation, formal methods, software engineering, process modeling, workflow, and software agents, is gaining momentum, in particular in the context of Web services usage. Research in the mentioned fields can be exploited to automate Web services-related tasks, like discovery, selection, composition, mediation, monitoring, and invocation, thus enabling seamless interoperation between them while keeping human intervention to a minimum. Although several initiatives, like such as OWL-S, WSMO, WSDL-S, or IRS, have emerged in this area aiming at addressing the problem of semantics in Web services, many major challenges still need to be addressed and solved in this field.

In this context, the semantics4ws workshop series aim to provide a forum in which to focus on selected core technical challenges for deployment of SWS, and reach a better understanding of the relationships between commercial Web service standards, current SWS research efforts, and the ultimate requirements for full-scale deployment of these technologies. More specifically, these workshop series aim to tackle the research problems (as well as recent practical experiences) around methods, concepts, models, languages and technology that enable semantics in the context of Web services, as well as discussing recent advances in semantics for Web services. Of particular interest are the architectural, technical, and developmental foundations of SWS, and showing how they combine synergistically to enable service automation on the scale required by today's Internet-connected enterprises.

The semantics4ws workshop series aim to bring together researchers and industry practitioners (e.g., leading modelers, architects, system vendors, open-source projects, developers, and end-users) addressing many of these issues (including recent developments in tools and techniques, and real-world implementations of SWS applications) and to promote and foster a greater understanding of how semantics can assist automation in Web services, thus helping people develop and manage services more efficiently and effectively.

The workshop organizers would like to thank the authors for their high-quality submissions and the members of the Program Committee for their reviewing and review coordination efforts. The organizers would also like to thank Barbara Pernici for her invited talk at the workshop.

November 2007

Steven Battle
John Domingue
David Martin
Dumitru Roman
Amit Sheth

Conference Organization

Programe Chairs

Programe Committee

External Reviewers

SPARQL-Based Set-Matching for Semantic Grid Resource Selection

Said Mirza Pahlevi, Akiyoshi Matono, and Isao Kojima

National Institute of Advanced Industrial Science and Technology (AIST)
Grid Technology Research Center
Tsukuba, Ibaraki 305-8568, Japan

Abstract. Grid is an emerging technology that enables the sharing of a wide variety of resources. However, effective and accurate grid resource matching is difficult because of the dynamic characteristics and heterogeneity of grid resources. Grid resource matching mechanisms that utilize semantic Web technologies have been proposed to deal with this issue, although none support set matching of grid resources based on their semantic description. This paper proposes a novel set-matching algorithm that uses standard RDF query language SPARQL [1] to semantically match a set of grid resources and SPARQL query features to efficiently perform set matching. We evaluated the efficiency and effectiveness of the algorithm by performing a set of experiments and present the results.

1 Introduction

The semantic grid [2] applies semantic Web technologies in the grid environment. For better automated resource discovery and selection, semantic grid applications use semantic Web annotation frameworks to describe resources and services. For example, the MyGrid project [3] has a service ontology that extends the DARPA agent markup language for services (DAML-S) [4] ontology and allows services to be queried and matched by subsumption reasoning over the service descriptions.

Grid resource discovery and selection also include a task known as *set matching* that discovers *a set of resources matching application requirements* [5]. Set matching is especially important in the grid environment because a grid is essentially a distributed, heterogeneous collection of computers that, in principle, can be used as a computing platform; because of the rapid proliferation of the semantic Web/grid and the resource heterogeneity, the ability to semantically match grid resources is crucial.

The following motivational example illustrates the benefits of semantic set matching. Suppose that we have a set of computers with CPU architectures classified according to the CPU ontology shown in Fig. 1. Further suppose that the query is: *find a set of Intel CPU computers with aggregated disk space greater than 8 GB*. A syntactic match-based approach returns an empty result, because the total disk space of n_3 and n_4 with Intel CPU architectures is less than 8 GB. Subsumption reasoning, however, infers that the CPU architecture of n_2 (which

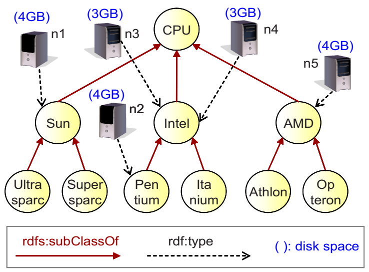


Fig. 1. A CPU architecture ontology and the instances

is a Pentium) is also Intel, so that the result will be $\{n_2, n_3, n_4\}$, with a total disk space equal to 10 GB.

The SPARQL query language [1], which is currently a World Wide Web Consortium (W3C) candidate recommendation, is used for querying RDF data and will become a standard query language for the semantic Web. The demand for standard methods to access RDF data in a grid environment led the Open Grid Forum (OGF) [6] to define specifications for accessing RDF data using the SPARQL query language [7]. Therefore, we argue that the semantic set matching for the grid should use SPARQL as the query language.

Using SPARQL as the set-matching query language provides the following advantages. First, *unrestricted target resource description*[1] *repositories*. Because SPARQL is a "standard" query language and many free SPARQL query engine implementations are available [8], the target repository for set matching can be easily switched if the repository is associated with a SPARQL query engine. Second, *complex set matching*. A requester can use the rich SPARQL constructs (e.g., options and filtering) for advanced set matching. Furthermore, resource descriptions can be retrieved from several sites using the SPARQL "named graph" and "from named" constructs, which allow for set matching to distributed resource descriptions. Third, *semantic set matching*. SPARQL does not support RDF data inference. However, a semantic match can be obtained by applying SPARQL to inferred RDF data/triples constructed using an RDF reasoner.

In this paper, we propose a set-matching mechanism based on SPARQL that enables semantic matching of grid resources. To the best of our knowledge, no studies have been conducted on this issue to date. Set matching in a grid environment using SPARQL is challenging, for the following reasons:

1. SPARQL does not support the aggregation functions necessary for set matching.
2. Because resource description repositories may be scattered over the grid, the set-matching system should be able to select one repository as a search target and to switch from one repository to another.
3. Set matching may include computationally expensive combinatorial searches.

[1] Henceforth, we will use *resource* and *resource description* interchangeably.

To cope with the first and second issues, we have extended SPARQL slightly using a sum function and have adopted a matchmaker-based approach. A service requester uses the extended SPARQL to formulate a set-matching query that is sent to a matchmaker. The matchmaker communicates with the resource repositories using standard SPARQL query language and thus absorbs the non-standard part. We avoided the third issue by proposing a *simple and efficient greedy retrieval algorithm* that makes use of powerful SPARQL constructs. To reduce data transfer cost, the algorithm retrieves as small as possible data from the repository. In constrast, a naive algorithm will retrieve the most of the repository data and perform combinatorial search over the data.

The rest of the paper is organized as follows. Section 2 briefly reviews related works. Section 3 explains the basic system architecture. Section 4 describes the proposed set matching algorithm. Section 5 presents experiment results that clarify the effectiveness and characteristics of the algorithm. Section 6 discuss the algorithm usage guideline and advanced set matching using SPARQL constructs. The final section gives conclusions and outlines future activities.

2 Related Work

The Globus Monitoring and Discovery System (MDS) [9] is the information services component of the Globus Toolkit [10] that provides information about the available resources on the grid and their status. This service collects data from various sources and provides a query/subscription interface to that data. The Semantic MDS (S-MDS) [11] extends the MDS by applying semantic Web technologies to annotate the metadata of grid resources. It describes the grid resources using OWL-S ontology [12] and provides an efficient mechanism to aggregate and maintain the ontology instances. Since S-MDS provides a SPARQL query interface to the ontology data and allows reasoning over the data prior to query processing our set-matching system can work well with S-MDS.

Classad [13,14] is a matchmaking framework to resource management in distributed environment. In this framework, a service advertisement and request are formulated using a semi-structured data model called classified advertisements (classad) which consists of attribute-value pairs. A matchmaker syntactically matches the advertisement and the request based on constraints specified by attributes in the classads. Redline [5] is a grid matchmaking system that reinterprets matching as a constraint problem and exploits constraint-solving technologies to implement matching operations. It provides *set-matching* which is not supported by classad. Classad and Redline, however, are based on text matching rather than semantic matching.

Ontology-based Matchmaker (OMM) [15] is an ontology-based resource selector for the grid. Resources and requests are described by (different) ontologies and they are matched using matching rules. Different from OMM, a semantic matching system proposed in [16] used SPARQL as a requester query language. The system provides query rewriting and result ordering mechanisms that enable a matchmaker to order results based on class subsumption relationships. Both methods provide semantic matching of grid resources but they do not support set-matching.

3 System Architecture

Fig. 2 shows the basic architecture of our set-matching system. Targeted grid resources are described using RDF/OWL and the descriptions are stored in an RDF repository system that is associated with a SPARQL query engine. A matchmaker stands between the repository system and the service requester, which formulates a resource request in SPARQL.

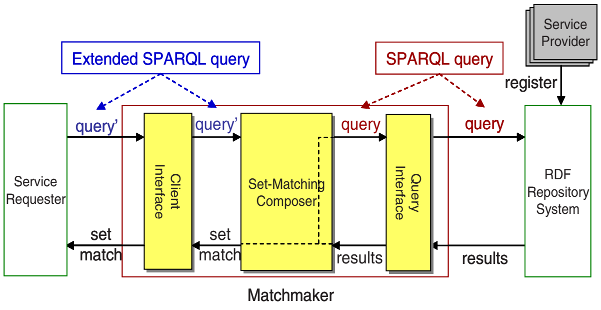


Fig. 2. Basic system architecture

To enable set matching using SPARQL, we slightly extended SPARQL with a sum function specified in a filter clause. For example, the query below retrieves a set of resources in which each resource has an available memory greater than 200 MB and the aggregate memory of the set is greater than 8000 MB.

SELECT $?res$ WHERE{ $?res$ `memory` $?m$. FILTER($?m > 200$).
FILTER(SUM($?m$) > 8000).}

We call a predicate whose values are summed up an *aggregation predicate* (i.e., `memory`); the predicate's variable (i.e., $?m$) the *aggregation variable*; the predicate's aggregation value (i.e., 8000) the *required aggregation value*; and a condition with a sum function (i.e., SUM($?m$) > 8000) an *aggregation condition*. The sum of aggregation predicate values from matched resources is the *resulting aggregation value* of the predicate.

The set-matching composer in the matchmaker accepts an extended SPARQL query from a requester, eliminates any aggregation conditions, and forwards the query to the repository. Upon receiving the query results, the composer determines whether the aggregation conditions are satisfied. If they are satisfied, the composer returns the results to the requester as an answer; if they are not, it caches the results, modifies the query, and resends it to the repository. The details of the retrieval algorithm are given in Section 4.

4 SPARQL-Based Set Matching

4.1 Goal and Outline

Given a SPARQL query containing one or more aggregation conditions, the matchmaker *repeatedly* retrieves a number of resources from the repository while modifying the query. The retrievals are performed until the aggregation conditions are satisfied or until it is known that the conditions cannot be satisfied. The retrievals are based on the following requirements: (1) data volume transferred from the repository to the matchmaker (i.e., the answer-set size) should be as small as possible. That is, it should be *proportional to* the required aggregation values and (2) the number of queries sent to the repository (the retrieval number) should be as small as possible.

To reduce the answer-set size and query number, the matchmaker modifies a requester query as follows:

1. It inserts an ORDER BY DESC clause into the query to place the "most promising" resources at the top of the matched results. The most promising resources are those with relatively large aggregation predicate values.
2. Based on 1), the matchmaker estimates the number of resources that need to be retrieved from the repository to satisfy the aggregation conditions, and sets this estimate as the LIMIT value of the query.
3. It inserts FILTER clauses into the query to eliminate resources that have already been retrieved in previous retrievals.

4.2 Set-Matching Algorithm

For each (possible) aggregation predicate p, the matchmaker retrieves n sample resources from a repository using the following simple SPARQL query:

SELECT ?*value* WHERE {?*res* p ?*value*} ORDER BY DESC(?*value*)
LIMIT n

The ORDER BY DESC clause retrieves resources whose p values are relatively large. An *estimated average of p values* of resources stored in the repository, $p.estAvgValue$, is calculated by taking the average of p values in the samples. This calculation can be done once at the system startup or regularly at specific time intervals.

Algorithm 1 shows the set-matching procedure. It first removes the aggregation predicates from a requester query q and puts them into a list *plist* (line 3). The algorithm then retrieves a set of resources from the repository based on the aggregation predicates (lines 5 – 11).

During retrieval, the algorithm calculates the resulting aggregation value of a current aggregation predicate based on a result accumulation buffer *accres* (line 5), then calculates the remaining aggregation value by subtracting this value from the required aggregation value (line 6). If the remaining value is greater than 0 (i.e., if the aggregation condition is not yet satisfied), the algorithm sets the ORDER BY DESC of q using the aggregation variable of the predicate[2] (line 8)

[2] An ORDER BY DESC clause is inserted if it does not exist.

Algorithm 1. Set Matching

```
   Input: SPARQL query $q$.
   Output: A set of resources.
1  begin
2      $accres \leftarrow \emptyset$;
3      $plist \leftarrow$ aggregation predicates in $q$;
4      foreach $p$ in $plist$ do
5          $ctotal \leftarrow \mathrm{Sum}_p(accres)$;
6          $remainAggValue \leftarrow p.reqAggValue - ctotal$;
7          if $remainAggValue > 0$ then
8              $q$.ORDER_BY_DESC $\leftarrow p.valueVar$;
9              $res \leftarrow \mathrm{Retrieve}(q, p, remainAggValue)$;
10             if $res$ is $\emptyset$ then return unsatisfiable;
11             append $res$ to $accres$;
12     return $accres$;
13 end
```

Algorithm 2. Retrieve

```
   Input: SPARQL query $q$, an aggregation predicate $p$, target aggregation value
          $tarAggValue$
   Output: A set of resources
1  begin
2      $res \leftarrow \emptyset$, $ctotal \leftarrow 0$;
3      while $true$ do
4          $q$.LIMIT $\leftarrow \lceil (tarAggValue - ctotal)/p.estAvgValue \rceil$;
5          $res' \leftarrow$ send $q$ and get the results;
6          if $res'$ is empty then return $\emptyset$;
7          append $res'$ to $res$;
8          $p.estAvgValue \leftarrow \mathrm{Min}_p(res') \times \alpha$;
9          $ctotal \leftarrow \mathrm{Sum}_p(res)$;
10         $q$.FILTER $\leftarrow$ condition to exclude resources in $res'$;
11         if $ctotal \geq tarAggValue$ then return $res$;
12 end
```

and then retrieves additional resources from the repository using the Retrieve function (line 9). If the returned result is empty, the aggregation condition of the predicate cannot be satisfied, and the algorithm returns an "unsatisfied" message (line 10). Results greater than 0 are appended to *accres* (line 11) and are considered at the next retrieval.

Algorithm 2 shows the retrieval procedure used by the set-matching algorithm. This algorithm repeatedly retrieves a set of resources and does the following during each retrieval: 1) Updates the LIMIT value of the query based on the remaining aggregation value at that retrieval (line 4), and 2) inserts a FILTER clause into the query to exclude resources that have been retrieved (line 10).

On receiving query results from the repository, the algorithm inserts the results into a buffer res' (line 5), updates the $estAvgValue$ of the aggregation predicate based on the newly retrieved resource set (line 8), and calculates the total for the aggregation predicate values in res (line 9). If the total value satisfies the target aggregation value ($tarAggValue$), the algorithm returns res (line 11); otherwise it modifies q and resends it to the repository.

The LIMIT value of the query restricts the matched result size and affects the total number of retrievals/queries, so the $estAvgValue$ (line 8) must be well estimated. Because the query results are ranked in descending order according to the aggregation predicate values and the aggregation predicate values of resources to be retrieved in the next turn cannot exceed the minimum predicate value in the current results, the $estAvgValue$ is estimated as $\mathrm{Min}_p(res') \times \alpha$, where $\mathrm{Min}_p(res')$ is the minimum aggregation predicate p value in res' and α $(0 < \alpha \leq 1)$ is a correction factor. α is used to adjust $estAvgValue$ because the average of p values of subsequent results is likely to be less than $\mathrm{Min}_p(res')$.

4.3 An Example

Suppose that a repository contains descriptions of eight computers: $r_1(800, 500)$, $r_2(800, 400)$, $r_3(700, 900)$, $r_4(500, 100)$, $r_5(400, 100)$, $r_6(300, 400)$, $r_7(300, 450)$, and $r_8(350, 900)$, where $r(x, y)$ denotes computer r with x MB available memory and y MB available disk space. Further suppose that a requester query is q_0, as shown in Fig. 3, sample size $n = 2$, $\alpha = 1.0$, and the aggregation predicate `memory` is processed first. Note that, the initial $estAvgValue$ of `memory` and `diskspace` which are calculated from the sample are 800 and 900, respectively.

Initially, because $accres$ is still empty, we get $ctotal = 0$ (line 1.5)[3] and $remainAggValue = 3000$ (line 1.6). The algorithm modifies q_0 into q_1 by inserting "ORDER BY DESC (?m)" (line 1.8) and "LIMIT 4" (line 2.4). Note that LIMIT is 4 because it is estimated that four computers, each with `memory` of 800 MB, will satisfy the aggregation condition. The algorithm then sends q_1 (Fig. 3) to the repository and gets $res' = (r_1, r_2, r_3, r_4)$ as the result (line 2.5).

Next, res' is appended into res (line 2.7) and $estAvgValue$ of the predicate and $ctotal$ is calculated, producing 500 (line 2.8) and 2800 (line 2.9), respectively. Because $ctotal$ is still less than 3000, the algorithm modifies q_1 into q_2 (lines 2.10 and 2.4), sends it to the repository, and gets $res' = (r_5)$ as the result. Now $ctotal$ is greater than 3000 (i.e., 3200), so $res = (r_1, r_2, r_3, r_4, r_5)$ is returned (line 2.11).

Next, the second predicate `diskspace` is processed. At this point, $ctotal$ and $remainAggValue$ with respect to `diskspace` are 2000 and 500, respectively. Because $ctotal$ is less than the required aggregation value of 2500, the algorithm modifies q_2 into q_3, sends it to the repository, and obtains r_8 as the result.

Finally, the algorithm returns $(r_1, r_2, r_3, r_4, r_5, r_8)$ as the answer, with the $resultAggValue$ of `memory` and `diskspace` equal to 3550 and 2900, respectively.

If `diskspace` is processed first, the answer will be $(r_1, r_2, r_3, r_4, r_7, r_8)$ with the $resultAggValue$ of `memory` and `diskspace` equal to 3450 and 3250, respectively.

[3] line $x.y$ denotes line y at algorithm x.

q_0 = SELECT ?*res* WHERE { ?*res* `memory` ?*m*. FILTER(SUM(?*m*) $\geq$ 3000).
?*res* `diskspace` ?*d*. FILTER(SUM(?*d*) $\geq$ 2500)}

q_1 = SELECT ?*res* WHERE { ?*res* `memory` ?*m*. ?*res* `diskspace` ?*d*}
ORDER BY DESC(?*m*) LIMIT 4

q_2 = SELECT ?*res* WHERE { ?*res* `memory` ?*m*. ?*res* `diskspace` ?*d*.
FILTER((?*res* != r_1) && (?*res* != r_2) && (?*res* != r_3) && (?*res* != r_4))}
ORDER BY DESC(?*m*) LIMIT 1

q_3 = SELECT ?*res* WHERE { ?*res* `memory` ?*m*. ?*res* `diskspace` ?*d*.
FILTER((?*res* != r_1) && (?*res* != r_2) && (?*res* != r_3) && (?*res* != r_4)).
FILTER(?*res* != r_5)} ORDER BY DESC(?*d*) LIMIT 1

Fig. 3. Requester and modified queries

5 Performance Evaluation

We use OGSA-DAI-RDF [17], developed by the National Institute of Advanced Industrial Science and Technology (AIST), as an RDF repository system deployed in a Linux node. The repository contains 200 resource descriptions written in OWL. Each resource has two (dynamic) predicates, `memory` and `diskspace`; the values of these predicates are updated every 90 s by assigning a random number ranging from 0 to 1000. The matchmaker and requester are deployed on different nodes, connected to the repository via a LAN. The sample size n is set to 10, and sample resources are retrieved once at the beginning of each experiment.

We use the following parameters: (1) Correction factor α, with values of $\{0.25, 0.5, 0.75, 1\}$. (2) Difference ratio of the two predicates' required aggregation values (dr), with values of $\{0.25, 0.5, 0.75, 1\}$. The required aggregation value of the primary predicate (i.e., a predicate with a bigger value) is set to $m \times 500$, where $m = \{5, 10, 20, 40, 80, 160\}$, and 500 is the average predicate value[4], while the secondary predicate (i.e., a predicate with a smaller value) is to the primary predicate value multiplied by dr. (3) A predicate processing order (first or last).

We performed 192 experiments with different α, dr, required aggregation values, and processing orders. In each experiment, we sent a requester query to the repository 50 times and calculated the average number of queries sent by the algorithm to the repository and the resulting set size. We organized the results into two main topics, described in the next two subsections.

1. Query number and set size with respect to the required aggregation values for various α values (Section 5.1).

[4] We set this value to evaluate the performance of the algorithm for various results/answer-set sizes, i.e., between approximately 2.5% and 80% of the repository size.

2. Query number and set size for $dr < 1$ and different predicate processing orders (Section 5.2).

5.1 Query Number and Set Size for Various α

This section evaluates the query number and set size for different α values and finds an optimum range of *alpha* values that provides the best performance.

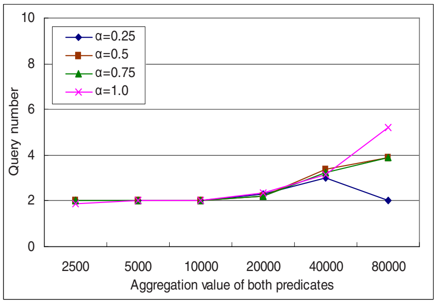


Fig. 4. Query number for $dr = 1$

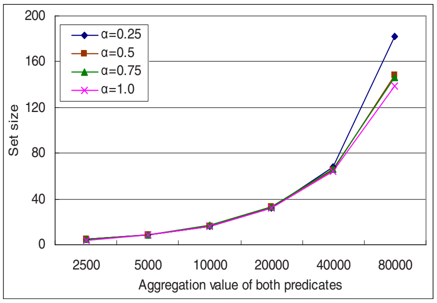


Fig. 5. Set size for $dr = 1$

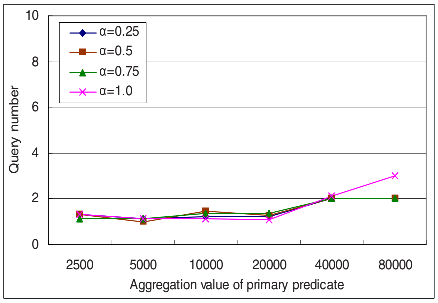


Fig. 6. Query number for $dr = 0.5$

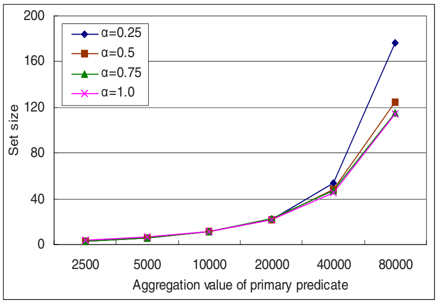


Fig. 7. Set size for $dr = 0.5$

Fig. 4 and Fig. 5 show the query number and set size for $dr = 1$; Fig. 6 and Fig. 7 show the results for $dr = 0.5$[5]. As shown in the figures, doubling the required aggregation value causes a proportional increase in set size, but without any significant increase in the query number. Moreover, for small and medium set sizes (up to approximately 35% of the repository size), the query number generally remains small.

The algorithm performs best for $\alpha = 0.5$ and $\alpha = 0.75$; at these values, the set size increases proportionally to the aggregation value and the query number is, at most, twice the value of the aggregation predicate number.

[5] The results for two other dr values were similar to $dr = 0.5$ and are omitted.

Overestimating the $estAvgValue$ (i.e., setting α to 1.0) results in a relatively low set size increase, but a larger increase in query number. Conversely, underestimating it (i.e., setting α to 0.25) produces a relatively large set size increase, with a relatively small increase in query number. This is because a larger (smaller) $estAvgValue$ produces a smaller (larger) LIMIT value of the query. A small (large) limit value forces the algorithm to send more (less) queries to the repository, because each query retrieves a few (many) resources.

5.2 Query Number and Set Size for $dr < 1$

This Section evaluates the query number and set size when the aggregation values of the two predicates are different. Based on the previous result α is set to 0.75. Fig. 8 and Fig. 9 show the results for the query number and set size, respectively. Each point in the graph represents the ratio of the query number/set size when the primary predicate is processed *last* to query number/set size when it is processed *first*.

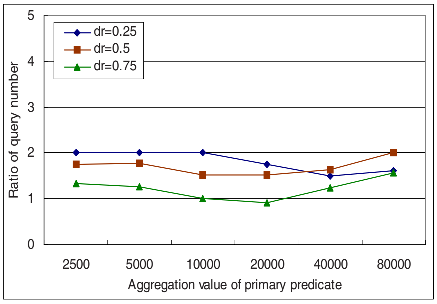


Fig. 8. Query number ratio for $\alpha = 0.75$

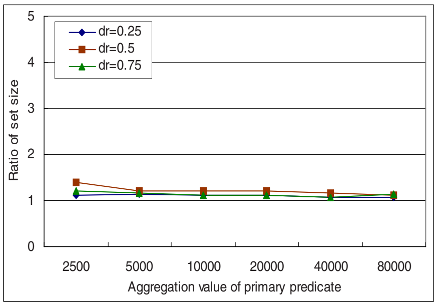


Fig. 9. Set size with for $\alpha = 0.75$

As shown in the figures, processing the primary predicate earlier decreases the query number by up to 50% while keeping the set size mostly unchanged. This is because early processing of the primary predicate makes it likely that the relatively small required aggregation value of the second predicate will be fully satisfied, which, in turn, decreases the number of queries sent to the repository.

In addition, Fig. 8 indicates that, on average, a smaller dr produces a larger decrease in query number, because with a smaller dr, it is more likely that the required aggregation value of the non-primary predicate will be satisfied after the earlier predicate is processed.

6 Discussion

6.1 Set-Matching Guidelines

In general, each (numerical) aggregation predicate p has a different domain value, ranging from 0 or 1 to a specific maximum number. Note that our retrieval algorithm always retrieves matched resources in descending order, according to the aggregation predicate values. Therefore, we can normalize the $p.reqAggValue$ (the x axis in the graphs) as a *maximum required resource number* for p, which is calculated as $p.reqAggValue/(\text{initial } p.estAvgValue)$, where the denominator is the average of p values in the initial n sample resources.

Our results suggest the following guidelines for the set-matching algorithm: (1) Set the value of α between 0.5 and 0.75; (2) If the maximum required resource numbers are very different, the primary predicate should be processed first.

It is important to note that, for enabling semantic set-matching, an RDF repository should store resource descriptions including their entailments.

We also perform experiments for a query with a single predicate and compare the results to those of a query with two predicates. As expected, the results show that the query number and result size increase as the predicate number increases.

6.2 Complex Set-Matching

The value constraint provided by SPARQL enables syntactic match that performs string and arithmetic comparisons. This feature, for example, can be used to restrict resources that can be included in a result-set to those that have a specific minimum memory value or to those organized or owned by a specific VO or user.

SPARQL allows several graph patterns to be used in a query and allows them to refer to same common variables. This enables a set matching that includes a *gang match* [14], which matches a single request with multiple types of resources.

The following example which is based on [14] demonstrates the idea. The purpose is to match a job (resource request) with two types of resources: computers and software package licenses. A job that uses the packages needs to allocate both a set of computers and a license before it can run. Assuming the two resources are described by different RDF graphs and the computer RDF graph is the default graph, the following query requests a set of Intel CPU computers in which each computer has memory greater than 500 MB and the aggregate memory of the set is greater than 8000 MB. In addition the computers must have a license to run “sim_app” application. The license is verified based on the computer addresses. Note that the two graph patterns refer to the same variable $?addr$.

```
SELECT ?comp WHERE {?comp architecture ?arch. ?arch rdf:type Intel.
?comp memory ?m. FILTER(?m > 500). FILTER(SUM(?m) > 8000).
?comp address ?addr. FILTER(?addr =”foo.go.jp“).
GRAPH PackageLicense{
        ?license hasValidHost ?addr. ?license hasApplication ?app.
        FILTER(?app =”sim_app“).}
}
```

7 Conclusions and Future Work

A common characteristic of many grid applications is a need to allocate multiple resources simultaneously. We have proposed such allocation mechanism that is based on semantic Web technologies. The mechanism uses SPARQL to take the advantages of the standard query language and the powerful query constructs. We have evaluated the mechanism by running a set of experiments. The results are promising that the answer-set size is proportional to the required resource number and the query number is, at most, twice the value of the aggregation predicate number.

We are now working on applying the mechanism in a real grid environment by incorporating it with the work done in [11] and [17]. In addition, to achieve better scalability and cope with the distributed nature of the grid we plan to work on a distributed structure of the set-matching system.

References

1. Prud'hommeaux, E., Seaborne, A.: SPARQL query language for RDF (2007)
2. http://www.semanticgrid.org/
3. http://www.mygrid.org.uk/
4. http://www.daml.org/
5. Liu, C., Foster, I.: A Constraint Language Approach to Matchmaking. In: RIDE 2004. Proc. of the 14th International Workshop on Research Issues on Data Engineering: Web Services for E-Commerce and E-Government Applications, pp. 7–14 (2004)
6. http://www.ogf.org/
7. Kojima, I., Said, M.P.: WS-DAI-RDF(S) querying (2006)
8. http://esw.w3.org/topic/SparqlImplementations
9. http://www.globus.org/toolkit/mds/
10. http://www.globus.org/toolkit/
11. Said, M.P., Kojima, I.: Towards Automatic Service Discovery and Monitoring in WS-Resource Framework. In: Proc. of the 1st International Conference on Semantics, Knowledge and Grid, pp. 932–938 (2005)
12. http://www.daml.org/services/owl-s/
13. Raman, R., Livny, M., Solomon, M.: Matchmaking: Distributed resource management for high throughput computing. In: Proc. of the 7th IEEE International Symposium on High Performance Distributed Computing, pp. 140–146 (1998)
14. Raman, R., Livny, M., Solomon, M.: Policy Driven Heterogeneous Resource Co-Allocation with Gangmatching. In: Proc. of the 12th IEEE Int'l Symp. on High Performance Distributed Computing (HPDC-12), pp. 80–89 (2003)
15. Tangmunarunkit, H., et al.: Ontology-based Resource Matching in the Grid—The Grid meets the Semantic Web. In: Proc. of SemPG03, vol. 2870, pp. 706–721 (2003)
16. Said, M.P., Matono, A., Kojima, I.: SPARQL-based OWL-S Service Matchmaking. In: Proc. of 1st International Workshop on Semantic Matchmaking and Resource Retrieval: Issues and Perspectives, pp. 35–50 (2006)
17. OGSA-DAI-RDF, http://wiki.dbgrid.org/index.php?OGSA-DAI-RDF

Calculating the Semantic Conformance of Processes

Harald Meyer

Hasso-Plattner-Institute for IT-Systems-Engineering
Prof.-Dr.-Helmert-Strasse 2-3, 14482 Potsdam, Germany
harald.meyer@hpi.uni-potsdam.de

Abstract. Verifying process properties has been an important research topic in last several years. The idea is to support humans in modeling processes by checking whether their processes are correct according to certain criteria (e.g. always terminate, adhere to a predefined specification). But these approaches were mostly limited to verifying syntactic properties of the process ignoring semantics and functionality of the contained activities.

In this paper we introduce a new property called semantic conformance that ensures that a process has the intended functionality. A process is semantically conformant to a process specification if it fulfills the intended functionality in all situations described in the process specification and if every activity of the process is actually invokable whenever it can be invoked.

1 Motivation

Modeling business processes is an inevitably complex task. It includes identifying the right activities, their ordering, and the data flow between them. Depending on the concrete setting, it can also include modeling the organizational structure of the company or of quality of service properties and constraints. Assisting human modelers is hence crucial. In earlier work[1,2] we identified three such supporting features: filter inappropriate services, suggest partial plan, and verify semantic correctness. In these works we described the first two features in great details but left the exact implementation to verify semantic correctness open. In this paper we will close this gap.

A great deal of research was and still is targeted at verifying process properties like soundness, conformance, and compatibility [3,4,5,6,7,8]. But they only verify syntactical properties of processes (e.g. that they are deadlock-free). They cannot verify whether the process actually implements the intended functionality. Additionally, they can only show whether a process is syntactically correct. But a syntactically correct process can still contain semantic errors (e.g. the precondition for an activity is not fulfilled). This paper therefore introduces a new process property called semantic conformance. A process is semantically conformant to a process specification if it provides the intended functionality and the preconditions of all contained activities are satisfied whenever they are enabled.

The idea is that the specification exists prior to modeling the process. It is either created by the process modeler himself, given by the requirements of future users of the process or defined by industry standards. The process modeler then models the process trying to adhere to the specification. When he is finished he can check using semantic conformance whether he modeled it correctly.

The calculation of semantic soundness of processes is based on earlier work on algorithms to calculate the preconditions and effects of service compositions [9]. Processes are modeled as so-called semantic workflow nets, a special form of Petri nets extended by semantic annotations. While the approach presented here is independent of services and service oriented architectures, it can easily be mapped to such a setting: service compositions are the processes consisting of special activities called service invocations. In [9] we actually motivated the foundations using SOA: the ability to automatically calculate precondition and effect of a service composition enables us to easily publish such a service composition as a service.

The next section refreshes some fundamentals on Petri nets. The main contribution of this paper, the definition of semantic conformance and the approach to calculate it, is presented in section 3. The paper closes with a look on related work and the summary.

2 Semantic Workflow Nets

In this section, we introduce Petri nets and workflow nets but also the advanced concepts from [9]. Using petri nets to model (business) processes or workflows is a common approach [3] and serves as the foundation for many process verification techniques. A Petri net is a directed bipartite graph. It contains two sets of vertices: places and transitions. The directed edges connect either a place with a transition or a transition with a place.

Definition 1. *A Petri net is a triple $n = (P, T, f)$ with:*

- *P: set of places*
- *T: set of transitions*
- *$f \subseteq (P \times T) \cup (T \times P)$: the flow relation*

The sets of places and transitions are disjoint : $P \cap T = \emptyset$

The transitions of a Petri net are active components. They represent the activities or service invocations. Places contain tokens to represent the current state of the net. The input and output places of a transition t are denoted with $\cdot t$ and $t\cdot$. Each place can host multiple token. The assignment of tokens to the places of a Petri net is called the marking:

Definition 2. *A marking is function $m : P \rightarrow N$ that assigns to each place the number of tokens in this place.*

State changes of the Petri net result in different markings. State changes can occur when an *enabled* transition *fires*:

Definition 3. *A transition t is enabled if all its input places $\cdot t$ contain at least one token.*

If a transition t fires one token is removed from each input place and one token is added to each output place.

Petri nets are a very generic concept to describe processes. Workflow nets restrict the notation of Petri nets to a subset sufficient for modeling the control flow of workflows:

Definition 4. *A Petri net $n = (P, T, f)$ is a workflow net iff:*

- *The net has one input place i (no incoming transitions)*
- *The net has one output place o (not outgoing transitions)*
- *Every vertex $v \in P \cap T$ is on a path from the input place to the output pace.*

In the following we will only use workflow nets. Another important concept are logical expressions. In this paper we will use a fairly simple notion of logical expressions: a logical expression is a collection of facts and negated facts. We will use them to express the semantics of the activities of a process. As the activities of a process are represented by the transitions of the workflow net, we add a labeling function that attaches the semantics to each transition:

Definition 5. *A semantic workflow net is a 4-tuple $sn = (P, T, f, l_s)$ is a Petri net $n = (P, T, f)$ with a function $l_s : T \rightarrow INV$ that maps each transition to an activity instance.*

An activity specification $s = (\mathcal{I}, \mathcal{O}, pre, eff)$ is a tuple with

- *$\mathcal{I}$: List of input parameters consisting of variables $\in V$*
- *$\mathcal{O}$: List of output parameters consisting of variables $\in V$*
- *pre: The precondition of the activity is a logical expression and must be satisfied in order to invoke the activity.*
- *eff: The effect of the activity is a logical expression. It describes the changes to the current state resulting from the invocation of the activity.*

An activity instance $i = (s, z)$ is a pair consisting of an activity specification s and a variable assignment $z : \mathcal{V} \rightarrow \mathcal{T}_{ground}$ that assigns every variable a ground term. Variables $v \in V$ are all the elements from $\mathcal{I}$ and $\mathcal{O}$ plus the variables in pre and eff.

Activity instances, the atomic elements of a process, are instances of activity specifications containing the actual semantics. As the last step in this section, we will describe briefly the algorithms to calculate preconditions and effects of processes modeled as semantic workflow nets. The idea is to assign a logical state to each marking of the workflow net. But before the algorithms can be described, we need to have a look at what actually happens if an activity is invoked:

Definition 6. *An activity instance $i = (s, z)$ with $s = (\mathcal{I}, \mathcal{O}, pre, eff)$ is* ***invokable*** *in logical state a if $a \models pre$.* ***Invoking activity*** *s with variable assignment z in logical state a leads to a state transition. This is defined by the state transition function $\gamma(a, i) = a \bigcup eff \setminus (\{x|\neg x \in eff\} \bigcup \{\neg x|if x \in eff\})$. γ is a partial function only defined if $a \models pre$.*

How invocation is performed formally, is out of scope of this paper. But approaches exist to base it on transaction logics [10]. Other approaches to express this are based on the notion of abstract states [11]. Calculating the effect of a process can then be achieved by simulating the invocation of all contained activities. To do this, we will use the reachability graph of the semantic workflow net. The reachability graph is the state transition graph of the markings of a semantic workflow net:

Definition 7. *Given a semantic workflow net $n = (P, T, f, l_s)$ and an initial marking m_1 the reachability graph is a directed, labeled graph $rg = (V, E, l_V, l_E)$. The vertices V represent the possible markings. The labeling function $l_V : V \to M$ assigns to each vertex the according marking. The edges E represent the transitions of the Petri net. The labeling function $l_E : E \to T$ assigns a transition to every edge.*

Given the final marking of a workflow net, we can then traverse the reachability graph backwards until we reach the initial marking adding up all the effects of contained activities. While doing so, we must take splits and joins into account. This can be achieved with a recursive algorithm that calculates the logical state for a given marking based on the logical states of its preceding markings. If the marking is the final one, this logical state is the process' effect:

Definition 8. *Given a semantic workflow net $n = (P, T, f, l_s)$ and its reachability graph $rg = (V, E, l_V, l_E)$, the logical state s for a marking m is given as $s_m = \bigvee \gamma(s_{m'}, i')$ with $i' = l_s(l_E(e))$ and $e = (m', m) \in E$ where $s_{m'}$ is the logical state of m'. The initial marking has the empty logical state s_0.*

The effect of a process is the logical state assigned to its final marking.

This algorithm is able to deal with processes containing sequences, and-split and -join, and xor-splits and -joins. Cyclic nets are not supported as the algorithm would not terminate. Additionally it still needs to be investigated what a loop actually means in terms of preconditions and effects.

Calculating the precondition works quite similar: instead of starting with the final marking, one starts with the initial marking and instead of adding up the effects one add up the preconditions. While these changes seem obvious, there is another, more subtle one. If the precondition of a activity is satisfied by preceding activities, this precondition should not be added to the precondition of the whole process. It is an internal precondition that is imposed and satisfied by the process itself. Hence, when calculating the precondition for a marking, already satisfied preconditions are not added. This is achieved by calculating the already known information (using the previous effect calculation algorithm) and subtract it from the precondition to be added:

Definition 9. *Given a semantic workflow net $n = (P, T, f, l_s)$ and its reachability graph $rg = (V, E, l_V, l_E)$, the precondition pre_m for a marking m is given by $pre_m = \bigvee pre_{m'} \cup (pre_{i'} - s_m)$ for all pairs $pre_{m'}$ and i' with $i' = l_s(l_E(e))$ where $e = (m, m') \in E$ and $pre_{m'}$ is the precondition of m'. The final marking has the empty precondition pre_{final}.*

The precondition of a process, is the precondition pre_{m_1} of its initial marking.

3 Semantic Conformance

In the previous section, we introduced the means to calculate the precondition and effect of a process. In this section, we apply them to verify the semantic conformance of a process to the previously defined process specification. The process specification describes the intended functionality under certain assumptions. Formally the process specification is just an activity specification describing the intended functionality of the whole process. Recapturing what a activity specification is, it consists of input parameters, output parameters, precondition and effect. The input parameters and the precondition define the situations in which the process should work and the output parameters and the effect define what changes the process may yield in these situations.

A process is semantically conformant to a process specification if the precondition of the process is always fulfilled in the precondition of the specification, if the effect of the process fulfills the effect of the specification, and if each activity in the process is actually invokable when it can be invoked. Formally:

Definition 10. *A process* $n{=}(P, T, f, l_s)$ *with a reachability graph* $rg{=}(V, E, l_V, l_E)$ *is* semantically conform *to a process specification* $R = (I, O, pre, eff)$ *if:*

1. $pre_R \models pre_n$ *with* pre_n *the precondition of the process,*
2. $eff_n \models eff_R$ *with* eff_n *the effect of the process, and*
3. $(pre_R \cup s_m) \models pre_i$ *for all activity instances* i *and according markings* m *with* $\exists(m, m') \in E$ *and* $l_E(l_s((m, m'))) = i$.

Checking the first two properties is rather straightforward using the algorithms from the previous section to calculate pre_n and eff_n. The third property is more complicated. For all activities in the process we need to determine the markings in which they are invokable ($l_E(l_s((m, m'))) = i$), calculate the logical state of the the marking (s_m) and check whether this logical state together with the precondition of the process specification entails the precondition of the activity ($(pre_R \cup s_m) \models pre_i$).

One might question whether the last property is actually necessary. We are already checking whether the process has any preconditions that are not satisfied in the process specification. Depending on the expressiveness of the used logical formalism, checking the third property can be unnecessary. This is for example the case if the logical formalism does not support negation. Then every time property one is violated, property three is violated, too. The same is true for the other direction. The only use of the third property without negation is that it gives detailed hints where the problem lies.

With more expressive logical formalisms, the third property becomes crucial. Then the first property is only a necessary (but not sufficient) criterion for the third property. If the precondition of the process is not satisfied by the process specification we know that the precondition of at least one activity instance is not satisfied. But it can be the case that the precondition of one activity instance is not satisfied while still the precondition of the overall process is satisfied. How this can happen, will be illustrated in the next section.

3.1 Example

Let us now look at an example illustrating how semantic conformance can be checked. We will use an example process to demonstrate subsequently violations of each property. The example we will use is a very simple order shipment process. After the order is received, the receipt and the actual goods are shipped separately and finally the order is closed. Shipping of the goods can be performed by one of two different shippers. Let us first specify the specification for this process. The precondition of the specification is $shipper_1 \vee shipper_2$ meaning that the customer must have specified which shipper to use. The effect of the specification is $receipt\_sent \wedge shipped$ meaning that the receipt should be send and the goods should be on their way to the customer.

Table 1. Preconditions and effects of the activities

Transition	**Precondition**	**Effect**
order		*ordered*
send_receipt	*ordered*	*receipt_sent*
shipping1	$ordered \wedge shipper_1$	*shipped*
shipping2	$ordered \wedge shipper_2$	*shipped*
close	$receipt\_sent \wedge shipped$	*order_closed*

Now let us look at a first example. In Table 1 the preconditions and effects of each activity are specified. Figure 1 illustrates our first try at modeling a correct process. But checking whether it fulfills the process specification's effect will actually show us that it is not sending the receipt.

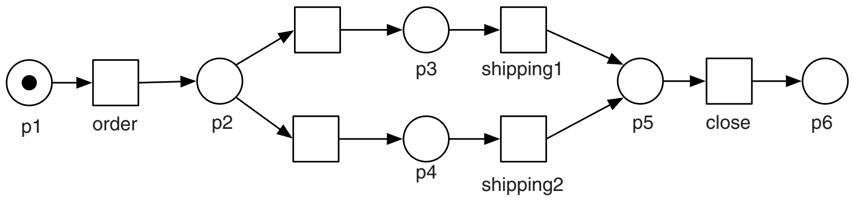


Fig. 1. First version of the modeled process

We start with the final marking in which the only token is in the place after *order* and traverse the reachability graph backwards until we reach the initial marking (the one depicted). The reachability graph in this case resembles the process: it contains one marking for each place in the process. And two markings are connected if the according places are connected via a transition. Hence we know that a marking s_{m_i} is the marking in which the token is in place p_i. When calculating the effect we start with $\gamma(s_{m_5}, close)$ and continue until we reach s_{m_1}: $\gamma(s_{m_5}) = order\_closed \wedge (\gamma(s_{m_3}, shipping_1) \vee$

$\gamma(s_{m_4}, shipping_2)) = (order\_closed \wedge shipped \wedge \gamma(s_{m_1}, order)) \vee (order\_closed \wedge shipped \wedge \gamma(s_{m_1}, order)) = order\_closed \wedge shipped \wedge ordered$. If we check this against the effect of the process specification we see that we do not achieve *receipt_sent*. Hence the process is not complete. We will actually achieve the same result when checking for the precondition. Because the order may only be closed if the goods have been shipped and the receipt has been send.

In Figure 2 an updated version of the process is displayed. It now contains the necessary sending of the receipt as well as a new activity: package gift. The idea is that we want to give each customer for a limited time a gift (to apologize for the delayed receipt sending). The new error we introduced is apparent: we only package the gift after we already shipped the goods. This will not work. To check this formally we need to know the precondition of packaging a gift. It is $\neg shipped$.

Calculating the precondition and checking it against the precondition of the specification will not help us here. If negation as failure is used, the precondition of the process is $shipper_1 \vee shipper_2$ [1] Therefore we need to check individually for each activity whether its precondition is satisfied. As we already know that the problem is packaging the gift, we will only check the precondition for this activity. Packaging the gift has exactly one preceding marking, namely the one where there are tokens in p_6 and p_7. We need to calculate the logical state for this marking and check whether it entails the precondition of packaging the gift. The logical state is $ordered \wedge receipt\_sent \wedge shipped$. And it is $ordered \wedge receipt\_sent \wedge shipped \not\models \neg shipped$. Hence, this activity is not invokable. to correct this process we need to move packaging the gift before shipping the goods.

With the, rather artificial, example in this section we have seen how using semantic conformance we can identify three different errors: unachieved effects, unsatisfied process preconditions, and not invokable activities.

3.2 Complexity

Analyzing the complexity includes analyzing the complexity of checking each property. Complexity largely depends on the expressiveness of the used logical formalism and the complexity of the reasoning tasks. Each property includes the same reasoning tasks:

- Checking for entailment: This is used at the end of precondition and effect calculation to check properties (1) and (2) and done once for each activity instance to check property (3).
- Updating the knowledge base: This is performed at each marking to update the calculated precondition and effect.

Assuming constant efforts for both reasoning tasks [2] checking all three properties has similar complexity. We also assume that intermediate results are stored.

[1] With classical negation it would be $(shipper_1 \vee shipper_2) \wedge \neg shipped$ but the precondition of the specification would also include $shipper_1 \vee shipper_2$.

[2] In reality, this is seldom the case. But as we will soon see, the complexity of the reasoning tasks matters hardly.

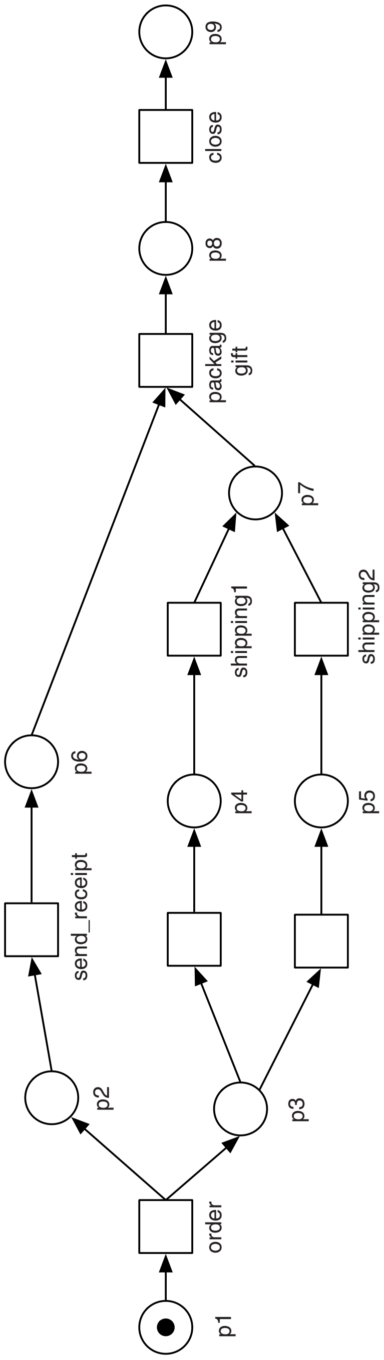


Fig. 2. Second version of the modeled process

Hence in case of a split and join, preceding markings are only visited once. Then calculating the effect of a process requires traversing the reachability graph exactly once. Calculating the precondition requires traversing the reachability graph twice (once for adding up the preconditions and once for calculating the effects to remove the from the precondition). Hence both precondition and effect calculation are linear in size of the reachability graph. This means that, in the worst case, they are exponential to the size of the process[3]. Checking the precondition of each contained activity instance to check the third property is actually mostly the same as calculating the precondition of the whole process. The only difference is that we need to perform additional reasoning tasks for each activity. But this does not change the complexity. Hence, complexity is still exponential to the size of the process.

4 Related Work

A lot of work exist regarding the verification of certain process properties. Based on the notion of soundness of Petri nets [3]. A Petri net is sound if it will terminate eventually leaving only a token in the final place and the net does not contain any dead tasks. Several approaches to relax this criterion were invented later on including relaxed soundness [4], weak soundness [6], and lazy soundness [7]. All have in common that they only verify syntactic properties of the process.

Additionally, all these soundness properties are properties of the process itself and not according to a specification or description. Our approach has much more in common with notions of conformance and consistency from choreography research [5,6,8]. In these approaches it is tested whether the concrete implementation matches the behavior specified in the interface. The interface can be the partner description of a choreography. Again these approaches only work on the syntactical level. They check whether the right action is performed at the right time. But to be the right action it has to have the same syntactical structure (e.g. same name) as in the specification. This approach is actually quite promising if you want to check whether to protocol (the sent and received messages) is adhered to. But for our use case it does not help much, because the process specification needs to model the whole process to capture all functionality. Our approach allows for structurally vastly different implementations yet achieving the same goal. But it cannot check whether the communication protocol of the process is correct.

Matteo Baldoni et al. [12] use a quite different approach to a reach a similar goal. They model the interaction protocol of services using logical programming to perform reasoning to select and compose them. This approach could most probably be used as well to check semantic conformance. While expressing the interaction protocol using logical programming makes Petri nets unnecessary leading therefore to a more coherent implementation, we still favor the Petri net approach. It allows us to take up the tremendous amount of work on expressing

[3] Size of a process $n = (P, T, f, l_s)$ is $|f|$.

processes as Petri nets and verifying certain properties. Hence we can use the related work presented earlier to check soundness and the adherence to a predefined communication protocol.

5 Summary

In this paper, we presented the novel property of semantic conformance. It enable a process designer to check whether a process is not only syntactically correct but also achieves the the intended functionality. A process is conformant to a request template if the precondition of the process if entailed in the precondition of the specification, the effect of the process entails the effect of the specification, and each activity is invokable whenever its invocation is possible.

In the future we want to extend the presented approach in two directions. On the one hand it should allow for checking more complex processes and on the other hand more errors should be detected. To achieve the first goal we want to support cyclic processes in the future. Our current idea to realize this is by defining a fix point semantic for loops. Errors that are currently not detected are for example errors in parallelism. If for example two activities depending on each other or who are in conflict invoked in parallel, the result is undetermined. Finally, we are also working on embedding our approach to check semantic conformance into a larger framework checking not only functional but non-functional properties on a semantic level as well.

References

1. Schaffner, J., Meyer, H., Tosun, C.: A semi-automated orchestration tool for service-based business processes. In: Proceedings of the 2nd International Workshop on Engineering Service-Oriented Applications: Design and Composition, pp. 54–65 (2006)
2. Schaffner, J., Meyer, H., Weske, M.: A formal model for mixed initiative service composition. In: SCC 2007. Proceedings of The IEEE International Conference on Services Computing (2007)
3. van der Aalst, W.: The application of petri nets to workflow management. The Journal of Circuits, Systems and Computers 8(1), 21–66 (1998)
4. Dehnert, J., Rittgen, P.: Relaxed soundness of business processes. In: Falsafi, B., VijayKumar, T.N. (eds.) PACS 2000. LNCS, vol. 2008, pp. 157–170. Springer, Heidelberg (2001)
5. Basten, T., van der Aalst, W.: Inheritance of behavior. JLAP 47, 47–145 (2001)
6. Martens, A.: On Compatibility of Web Services. Petri Net Newsletter 65, 12–20 (2003)
7. Puhlmann, F., Weske, M.: Investigations on soundness regarding lazy activities. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102, pp. 81–96. Springer, Heidelberg (2006)
8. Decker, G., Weske, M.: Behavioral consistency for b2b process integration. In: CAISE. Proceedings of the 19th International Conference on Advanced Information Systems Engineering (2007)

9. Meyer, H.: On the semantics of service compositions. In: Marchiori, M., Pan, J.Z., de Sainte Marie, C. (eds.) RR 2007. LNCS, vol. 4524, Springer, Heidelberg (2007)
10. Kifer, M., Lara, R., Polleres, A., Zhao, C., Keller, U., Lausen, H., Fensel, D.: A logical framework for web service discovery. In: ISWC 2004. Workshop on Semantic Web Services: Preparing to Meet the World of Business Applications, CEUR Workshop Proceedings. Hiroshima, Japan, vol. 119 (2004)
11. Keller, U., Lausen, H., Stollberg, M.: On the semantics of functional descriptions of web services. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, Springer, Heidelberg (2006)
12. Baldoni, M., Baroglioa, C., Martellia, A., Pattia, V.: Reasoning about interaction protocols for customizing web service selection and composition. Journal of Logic and Algebraic Programming 70, 53–73 (2007)

Towards a Formal Framework for Reuse in Business Process Modeling

Ivan Markovic and Alessandro Costa Pereira

SAP Research
Karlsruhe, Germany
{ivan.markovic, alessandro.costa.pereira}@sap.com

Abstract. Business process models are created by business users with an objective to capture business requirements, enable a better understanding of business processes, facilitate communication between business analysts and IT experts, identify process improvement options and serve as a basis for derivation of executable business processes. Designing a new process model is a highly complex, time consuming and error prone task. In order to address this problem, we propose an approach to business process modeling through reuse of existing business process artifacts - process fragments. In addition, we provide a rich formalism for business process description based on π-calculus and ontologies as a basis of the approach. The formalism integrates different workflow perspectives and thus exposes the complete process model description to expressive querying and reasoning.

1 Introduction

In the modern world, businesses constantly strive to reinvent and differentiate themselves under continuous pressures of regulatory and technological change. The most prominent issue is the lack of automation when trying to incorporate new business requirements into existing information systems - adding new components and adapting existing ones as priorities and perspectives change. This brings companies to a state of everlasting search for new technologies that help them reduce time to market of new or improved products and services.

We want to address this problem and facilitate the design of flexible and agile business systems by structuring the business knowledge into reusable business knowledge components. These components can be seen as a complement to traditional software engineering components. Software component engineering focuses on the back end in the process engineering chain. Software components have evolved from concepts such as common subroutines and general purpose application packages to reusable data services objects based on standards such as CORBA[1] and Web services based on XML[2] and WSDL[3] standards. The main

[1] http://www.corba.org/
[2] http://www.w3.org/XML/
[3] http://www.w3.org/TR/wsdl

concern here is program code and interfaces for communicating data. In contrast to this, the economic impact is far larger at the front end of the process engineering chain - on reusing business knowledge to configure and innovate business processes, products and services.

The sources of business knowledge that describe the processes of an organization are diverse and scattered in IT-supported processes, business documents, presentations and the heads of business people. Business knowledge is independent of the technology that implements it. It is embedded in processes supported by diverse technologies, depending on the organization's legacy and infrastructure. This knowledge can only be accessed and reused if it is extracted and stored as a single piece of information inside a knowledge repository. Ontologies provide appropriate means to formally describe the concepts and relations in a particular domain of discourse. "An ontology is an explicit specification of a conceptualization" [1]. Making the business knowledge explicit in terms of an ontology would enable systems to manipulate meanings rather than program code. Following this idea we create a formal model for describing business processes, which integrates different workflow perspectives and allows for expressive querying and reasoning on business process models.

In this work, we illustrate the benefits of using our formal model in the first phase of the process engineering chain, namely business process modeling. Designing a new process model is a highly complex, time consuming and error prone task. To overcome this problem, we present a framework for supporting business users in the modeling task by reusing existing business process artifacts during modeling. This facilitates the task of modeling business processes in two ways: i) it improves the quality of the models through reuse of established and optimized artifacts; ii) it reduces the process modeling time by avoiding modeling the same business process or part of it multiple times. Furthermore, we introduce a novel approach to process model design through reuse of existing business process artifacts - process fragments. With process fragment, we refer to a self-contained, coherent building block of a process model with a clear business meaning. Process fragments decompose overall business functionality into manageable, meaningful business units. Therefore, we think of process fragments as reusable modeling entities that can be reused in modeling different business scenarios. Process fragments reflect best practices on how to model a specific business functionality, which make their reuse an important step towards more effective business process modeling. The fragments can be reused without modifications (as is), or modified by the user for their adjustment to the context of reuse.

The remainder of the paper is organized as follows. In Section 2, we describe the addressed problem in more detail. To make the paper self-contained, Section 4 gives a short introduction to the formal foundations used in this work. Section 3 provides a requirements analysis for creating a reuse framework. The ontology we built for describing business processes, BPO is presented in Section 5. Section 7 discusses some related work. We conclude and give an outlook on future research in Section 8.

2 Usage Scenarios

In order to illustrate different types of queries that the user could perform in searching for process fragments, we give an overview of some usage scenarios for our reuse framework:

- *Scenario 1: Goal-based process fragment discovery*
  This scenario covers the goal-based matchmaking where the user formulates his request (goal) consisting of a set of criteria using a query template. The user can specify constraints both on static (function, domain, role, resource) as well as behavioral aspects of the process fragment description. This request is then matched to the process fragment descriptions from the repository and a ranked list of fragments matching the specified goal is presented to the user. The described scenario enables the business users to create underspecified process models and let the framework suggest a refinement of these models for them. We see this as a significant improvement comparing to current modeling techniques where users have to model everything in detail.
- *Scenario 2: Process model autocompletion*
  We distinguish between two cases of autocompletion for this scenario:
  - *Case 1:* Here we assume that the user has already modeled a part of the process and wants to find process fragments that can complete what he started to model. The modeled part of the process is matched against the process fragment descriptions from the repository and the user can select the most suitable fragment from the resulting list to include it in his design.
  - *Case 2:* After the user selects process fragments that refine the underspecified parts of their model (Scenario 1), the system can suggest how to connect them to form the complete process model. The input and output documents of process fragments and their context information (business domain, business function) can be used as criteria for suggesting their inter-connections.
- *Scenario 3: Process fragment substitution*
  This scenario supports the replacement of a selected process fragment with the one that e.g. corresponds to the new process redesign goals. The newly discovered fragment needs to be equivalent to the old one in terms of behavior so that the remaining process parts can stay unchanged. This functionality enables the user to easily find relevant process fragment candidates for substitution.

3 Requirements Analysis

In this section we provide a list of requirements that a framework designed for reuse of process fragments should fulfill.

- *Req. 1: Rich process description* The process model needs to be formally described to enable automatic matchmaking of user requests (goals) against process descriptions. In order to support the user to expressively search the process repository, we need a rich process description. We distinguish two main aspects of a process description: dynamic and static aspect. Within the dynamic aspect we want to capture the behavior of the process, i.e. process control flow. This will give the user a possibility to impose behavioral constraints on the process fragments he wants to retrieve. Within the static aspect of the process description we want to describe other workflow perspectives, e.g. organizational and informational. We want to describe processes in terms of their input/output data, business function, business domain, organizational roles which perform certain process parts, etc. In this way, the user will be able to specify this type of information in his request and use it to express additional constraints in his query. Note that in this paper, we address and focus on solving this requirement.

– *Req. 2: Intuitive user request specification* The user must be provided with a user-friendly query interface for specifying his requests. The user must be able to query for processes both on the static and dynamic aspects of their description.
– *Req. 3: Query language* There needs to be a query language with expressive power that is sufficient to formally describe the user requests. Note that the user request can be in the form of a query template, but it can also be a part of a process for autocompletion or a process fragment for substitution.
– *Req. 4: Querying mechanism* There needs to be a mechanism that will perform expressive matchmaking of user requests against process descriptions. The algorithm should take a user request as an input and return a ranked list of process fragments that match the request.
– *Req. 5: Flexibility* The reuse framework must provide support for relaxation and refinement of user queries. In the case that we get too less results matching the query, we can relax the query, i.e. incrementally abstract elements of the query, e.g. by using subsumption hierarchy. Similarly, if we get too many results, the user needs to be provided with a possibility to refine his request.
– *Req. 6: Ranking* To increase the usability of process fragment matching results, the resulting list of fragments should be ordered w.r.t. the level of match. Similarity measures need to be defined for process fragments to support the ranking of results based on multiple criteria.
– *Req. 7: Computational efficiency* A lesser requirement, yet the underlying matchmaking mechanism must be computationally tractable and efficient to provide the required design-time user support.

4 Foundations

In order to solve the first of the aforementioned requirements, we have selected the π-calculus for capturing the dynamic aspect of business processes and proposed an ontology stack for describing the static aspect of processes. This section gives a brief overview of the formalisms used in the work.

4.1 π-Calculus

The mathematical foundation which we use to describe the behavior of business processes was created by Milner, Parrow and Walker, published in [2]. The π-calculus a is formal language for specifying mobile processes, which uses communication channels (names) interaction. Basically, the π-calculus consists of **processes** and **names**. Here we summarize the π-calculus notation used in this work:

$$P ::= M \mid P|P \mid \nu z P \mid !P \tag{1}$$

$$M ::= 0 \mid \pi.P \mid M + M \tag{2}$$

$$\pi ::= \overline{x}\langle y\rangle \mid x(z) \mid \tau \mid [x = y]\pi \tag{3}$$

Equation 1 is the definition of a **process** and it defines the following: $P|P$ is a composition where processes P and P run in parallel – **concurrent** execution. $\nu z P$ represents a **restriction**, which ensures that the name z is fresh. $!P$ is the notation for a **replication**, where multiple instances of P run in parallel. The replication operator also satisfies the equation: $!P = P \mid !P$.

Equation 2 gives the summations behind M: 0 is the **inaction**, a process that can do nothing. $M + M$ is the **exclusive choice** between M and M. The π is a prefix.

Equation 3, finally, defines the **prefix** π: $\overline{x}\langle y\rangle$ is the **output** prefix, which sends the name y over the name x and then continues as P. On the other hand, $x(z)$ is the **input** prefix receiving any name over x, and then continues as P with z replaced by the received name. τ is an unobservable **internal action** of the process. The last symbol, the **match** prefix $[x = y]\pi.P$ behaves as $\pi.P$, if x is equal to y.

The syntax of π-calculus is used as a basis for creating the grammar of the Business Process Definition Ontology explained in Section 5.

4.2 WSMO and OWL-S

The Web Services Modeling Ontology (WSMO) [3] provides a conceptual framework and a formal language for semantically describing all relevant aspects of Web services in order to facilitate the automation of discovering, combining and invoking electronic services over the Web. WSMO identifies four top level elements as the main concepts which have to be characterized: ontologies, goals, web services and mediators.

The Web Service Modeling Language (WSML) [4] provides a formal syntax and semantics for WSMO. WSML is based on different logical formalisms, namely, Description Logics, First-Order Logic and Logic Programming, which are useful for modeling of Semantic Web services (SWS). WSML Grammar can be obtained from [4].

OWL-S [5] is an OWL-based upper ontology for describing the properties and capabilities of Web services, which enables automatic Web service discovery, invocation, composition and interoperation.

WSMO and OWL-S are two major initiatives that try to achieve the same goal: both aim to provide appropriate description means that enable effective exploitation of semantic annotations w.r.t. discovery, composition, execution and interoperability of Web services. However, WSMO provides several advantages when compared to OWL-S: its conceptual model has a better separation of the requester and provider point of view, it provides better language layering and it describes user requirements in a more natural fashion [6].

In this work, we use WSML as a representation language for the ontologies that capture static aspects of the process model description, which is discussed in the following section of the paper.

5 Business Process Ontology (BPO)

A business process ontology, in our approach, should do more than just represent workflow patterns [7], as it must also represent other workflow perspectives (functional, informational, organizational). In [8], it is envisioned that the third generation of BPM products will be marked by ontology-based business process management. We follow this idea and create a framework of ontologies capable to represent multiple aspects of a process model. We aim for a process model representation rich enough to enable automatic process verification, simulation, discovery, composition and execution. For this reason, we propose the Business Process Ontology (BPO), which captures dynamic and static aspects of a process model description. We discuss the ontology in more detail in the following two subsections.

5.1 Representing the Dynamic Aspect of Business Processes

For representing the dynamic aspect (behavioral semantics) of a process model, we use process algebra, the π-calculus. The π-calculus was introduced in Section 4 with its syntax and constructors. The language is quite simple and suitable to constitute the foundation for the BPO.

We follow the argumentation in [9], [10] and [11] to select the π-calculus as theory for describing the dynamics of modern business process management systems. First, the mobility theory fulfills arising requirements of BPM [9]; in addition, it provides the powerful framework based on simple mathematics [10]. Furthermore, [12] introduces the formal semantics for all workflow patterns from [7] based on the π-calculus. A crucial benefit of using π-calculus are the facilities given by the bisimulation theorem, e.g., it helps finding equivalent processes for substitution.

After creating the π-calculus ontology, we had to show that our ontologized π is able to express the semantics of all workflow patterns. Therefore, all representations from [12] were carefully modeled in the WSML language referencing BPO concepts.

The concepts in BPO are visualized in Figure 1, using WSMO Studio[4]. BPO starts by representing hierarchically the π-calculus language. The ontology kernel

[4] http://www.wsmostudio.org/

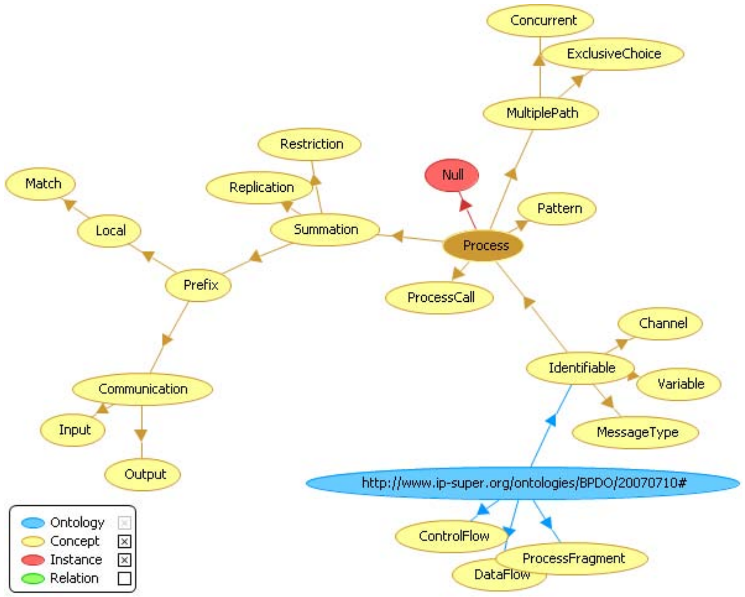


Fig. 1. BPO – Concepts

grows with concepts for differentiating data and control flow. Further on, we see that channels between tasks can be specified. At this point, an ontologized business process can model inter/intra processes interaction and dataflow.

The leaf concepts are instantiated for describing the process sequence. All process parts are *Identifiable*, which means that they should have a name. Ontologized π-calculus (Subsection 4.1) has the *Process* as top level concept. A process can have a *hasDefinition* attribute (defining a π-process), and/or it can have a *hasNext* attribute, representing a sequence.

The concept Process has four sub-concepts: *Process Call*, used to make a call (i.e. transferring the execution) to another process, passing some variable names; *Multiple Path* containing the attribute *subdivide* (listing process bifurcation); Pattern, which can have annotations and is used for describing well-known business processes without refined task definitions; and *Summation*. Multiple Path is subdivided again into the π-calculus elements *Exclusive Choice* and *Concurrent*.

A Summation has the sub-concepts explained directly by the π-calculus syntax: *Replication*, *Restriction* and *Prefix*. Prefix is further specialized into Communication, concept which groups *Input* and *Output* channels; and *Local*, the unobservable action. *Match* is the last sub-concept of Prefix.

There is the necessity to semantically annotate the type of communication channel. The possible types are *Data Flow* and *Control Flow*, which results in introducing these two concepts in our ontology. The annotation is done by having a multiple inheritance from the concept Input or Output and Data Flow or Control Flow.

Moreover, *Channels* can annotate elements derived from the concept Communication. This annotation contains information about *Message Type* and the *Protocol* (both attributes of Channel).

5.2 Representing the Static Aspect of Business Processes

In order to integrate behavioral with other workflow perspectives, BPO imports concepts from several other ontologies, thus forming the ontology framework for rich process description shown in Figure 2.

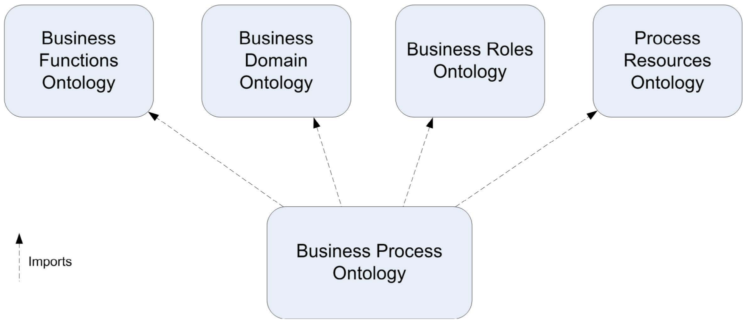

Business Functions Ontology
Business Domain Ontology
Business Roles Ontology
Process Resources Ontology
Imports
Business Process Ontology

Fig. 2. Ontology Framework

Imported ontologies have the task to describe business and industry specific concepts used to create semantic annotations for concrete process definitions. For describing the functional perspective, we have designed the *Business Functions Ontology* (BFO). This ontology provides a structural breakdown of the organization's business functions. It does so by splitting the domain in two dimensions, namely horizontal and vertical. Horizontal dimension describes concepts such as CRM, SRM, PLM, SCM, etc. The vertical dimension describes concepts such as: procurement, manufacturing, warehousing, order fulfillment, etc. Concepts from this ontology classify process models by their functionality, independent of the business domain. BFO concepts are also used to define what a process intends to achieve from a business point of view, i.e. they are used to capture the Business Goal of a process fragment. *Business Domain Ontology* complements BFO and describes the domain inside the organization where the process is used. Examples of business domain concepts are: product area, client area, localization area, etc. Business Domain together with Business Function define the context of a process model. *Business Roles Ontology* includes concepts representing roles in the organization e.g. Manager, Engineer, Clerk, Secretary, etc. *Process Resources Ontology* describes the resources (documents, systems, machines) which are required to operate the activities in processes. Due to the lack of writing space, explaining the designed ontologies in more detail is out of scope for this paper.

The semantic annotation of Processes is done by five defined relations (shown in Figure 3): *hasBusinessGoal*, *hasBusinessFunction*, *hasBusinessDomain*, *hasBusinessRole* and *hasProcessResource*. These relations group pairwise a Process or a ProcessFragment and respectively Business Goal, Business Function, Business Domain, Business Role and Process Resource. These semantic annotations can be used as query arguments for finding process fragments in our scenarios, see Section 2.

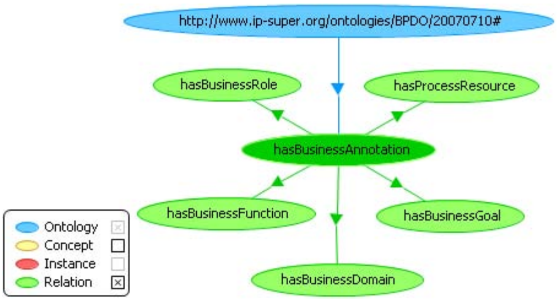


Fig. 3. BPO – Business Annotations

6 Example

To illustrate the use of this ontology for business process description, we give an example process fragment which performs customer order processing. The example fragment is composed of simple tasks (receiving a message, checking an order, sending a message), an exclusive choice for sending the correct message back, and the merge for synchronization. It is depicted in Figure 4 using the BPMN[5] notation.

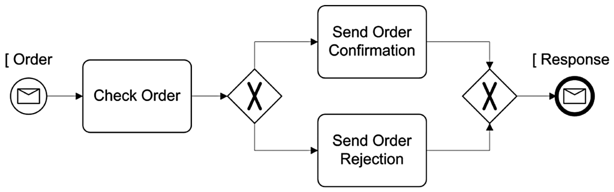


Fig. 4. Customer Order Processing – Process Fragment

The next step is the translation of a visual diagram into our ontology. The Figure 5 represents the process fragment translated to BPO. Each element has either a *hasDefinition* or *hasNext* attribute, if it continues the execution. If a task should go to sleep, it references the process instance *bpo#Null*.

[5] http://www.bpmn.org/

```
wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"
namespace { _"http://www.ip-super.org/business/process/example1v2#",
     wsmostudio _"http://www.wsmostudio.org#",
     bco _"http://www.ip-super.org/ontologies/BCO/20070626#",
     bpdo _"http://www.ip-super.org/ontologies/BPDO/20070710#",
     dc _"http://purl.org/dc/elements/1.1#" }
ontology _"http://www.ip-super.org/business/process/example1v2#"
     nonFunctionalProperties
          dc#contributor hasValue {"Alessandro Costa Pereira",
                                    "Ivan Markovic"}
     endNonFunctionalProperties
     importsOntology
          { _"http://www.ip-super.org/ontologies/BPDO/20070710#",
            _"http://www.ip-super.org/ontologies/BCO/20070626#"}
// This Process represents a simple sequence, for Ordering
instance CustomerOrder memberOf bpdo#Process
     bpdo#hasName hasValue "Customer Order"
     bpdo#hasDefinition hasValue data_1
//Receive the data
instance data_1 memberOf {bpdo#Input, bpdo#DataFlow}
     bpdo#hasName hasValue "info"
     bpdo#hasNext hasValue checkOrder
//Process Something - Unobservable for the User
instance checkOrder memberOf bpdo#Local
     bpdo#hasName hasValue "Process Request"
     bpdo#hasNext hasValue decide_1
instance decide_1 memberOf bpdo#ExclusiveChoice
     bpdo#hasName hasValue "split"
     bpdo#subdivide hasValue {sendConfirmation, sendRejection}
//Send Confirmation: case 1
instance sendConfirmation memberOf {bpdo#Output, bpdo#DataFlow}
     bpdo#hasName hasValue "answer"
     bpdo#asName hasValue "confirmation"
     bpdo#hasNext hasValue bpdo#Null
//Send Rejection: case 2
instance sendRejection memberOf {bpdo#Output, bpdo#DataFlow}
     ... (similar to sendConfirmation)
//Merge after Exclusive Choice
instance mergeMessage memberOf bpdo#Process
     bpdo#hasName hasValue "Merge Message"
     bpdo#hasDefinition hasValue receiveInfo
```

```
//Receive info, and process it further calling it orderAnswer:
//It can be either Confirmation or Rejection
instance receiveInfo memberOf {bpdo#Input, bpdo#DataFlow}
     bpdo#hasName hasValue "answer"
     bpdo#asName hasValue "orderAnswer"
     bpdo#hasNext hasValue sendResponse
//Send Response
instance sendResponse memberOf {bpdo#Output, bpdo#DataFlow}
     bpdo#hasName hasValue "orderAnswer"
     bpdo#hasNext hasValue bpdo#Null
//Creating Business Goal, Function and Domains
//These will be used to annotate processes
//(Processes with definition)
instance CustomerFeasible memberOf bco#BusinessFunction
     bco#hasName hasValue "DetermineCustomerOrderFeasibility"
     bco#hasDescription hasValue "For proceeding, the
                Customer should be feasible to make an Order"
instance DataAcquisition memberOf bco#BusinessFunction
     bco#hasName hasValue
                "Information Acquisition"
     bco#hasDescription hasValue "Receiving data from Customer"
instance Customer memberOf bco#BusinessDomain
     bco#hasName hasValue "CustomerDomain"
     bco#hasDescription hasValue
                "Enterprise deals with Customers"

//Fragment Definition
instance fragment_1 memberOf bpdo#ProcessFragment
     bpdo#constituent hasValue {data_1, checkOrder}
     //after checkOrder, hasNext has to be redefined

//Relation Instances:
//They connect together Processes and (Goal, Function or Domain)
relationInstance
bpdo#hasBusinessFunction(CustomerOrder, CustomerFeasible)

relationInstance
bpdo#hasBusinessDomain(CustomerOrder, Customer)

relationInstance
bpdo#hasBusinessFunction(fragment_1, DataAcquisition)
```

Fig. 5. WSML Instance

The shown example is a process fragment, which can be composed with a respective task responsible for sending the message *"Order"*. Also, some parallel task should continue the execution, when receiving the message *"Response"*.

The process is semantically annotated using the *relationInstance* association between the ontologies in Figure 2). Annotations are the key for enabling querying of processes or process fragments based on the static part of their description (Req. 4).

The exemplified process is annotated for doing the work in the *customer* domain, having business functionality: *determine customer order feasibility*.

7 Related Work

Much of the related work focuses on reuse of software components for business process modeling. In [13], an approach to model Web services composition for business process model implementation is presented. The user is supported in modeling by filtering out the services that can be used in the next modeling step based on matching the pre- and postconditions. Main difference to our approach is that we abstract from the technical representation and focus on reusing modeling artifacts. In addition, our underlying formalism is more expressive as it captures also the behavioral aspect of the process description.

The approach in [14] presents matchmaking of Web services based on π-calculus and description logics. The ontology is used for modeling input/output data exchanged between service operations. In our case, we capture additional business knowledge in the ontology. Furthermore, we reuse modeling entities which enables the design of truly flexible and agile systems, as argued in Section [1].

Another line of research investigates formal foundations for Business Process Management (BPM). The paper [9] revealed the strengths of the π-calculus for modeling workflows as well as service choreographies making it a good candidate to formally ground the dynamic nature of modern BPM. Further work [15] by the author focuses on exploring process verification. We extend this approach by integrating other workflow perspectives in the formalized process description, thus opening the doors for other types of reasoning on process models.

The approach presented in [16] discusses a process component model for process knowledge reuse. Here, the process component model is characterized only using static information (domain, function, performance, life-cycle). Therefore, it is not possible to represent and reason on behavioral aspects of the process description.

Finally, in [17], a set of ontologies for Semantic Business Process Management [18] is proposed. This work gives a rather high-level overview of the ontologies in question based on the ARIS [19] methodology. In addition, the need for a reuse framework in business process modeling is not addressed in this work.

So far, we have not seen other approaches addressing the problem of reuse in business process modeling using rich formal models.

8 Conclusion

In this paper we presented an expressive formalism for describing business process models to support reuse in business process modeling. The formal model captures different workflow perspectives and can be used for various querying and reasoning purposes (process model reuse, verification, simulation, execution).

We are currently working on the design of the matchmaking algorithm that can evaluate rich queries coming from the business expert against the repository of process fragments. As part of our future work, we plan to address the remaining requirements presented in Section [3].

Acknowledgments

This work has been partially supported by the European Union within the FP6 IST project SUPER[6] (FP6-026850).

References

1. Gruber, T.R.: Towards principles for the design of ontologies used for knowledge sharing. In: Guarino, N., Poli, R. (eds.) Formal Ontology in Conceptual Analysis and Knowledge Representation, The Netherlands, Kluwer Academic Publishers, Deventer (1993)

[6] www.ip-super.org

2. Milner, R., Parrow, J., Walker, D.: A calculus of mobile processes, parts I and II. Technical Report -86, University of Edinburgh (June 1989)
3. ESSI WSMO working group: Web Service Modeling Ontology (WSMO) – Final Draft (October 2006), `http://www.wsmo.org/TR/d2`
4. ESSI WSML working group: Web Service Modeling Language (WSML) – Final Draft (October 2006), `http://www.wsmo.org/TR/d16/d16.1/`
5. Martin, et al.: OWL-S: Semantic Markup for Web Services (2004), `http://www.w3.org/Submission/OWL-S/`
6. Lara, et al.: A conceptual comparison between wsmo and owl-s (2005), `http://www.wsmo.org/TR/d4/d4.1/v0.1/d4.1v0.1_20050412.pdf`
7. Aalst, W.M.P.V.D., et al.: Workflow patterns. Distrib. Parallel Databases 14(1), 5–51 (2003)
8. Jenz, D.E.: Ontology-based business process management. In: Strategic White Paper – Draft, Erlensee, Germany, Jenz & Partner GmbH (2003)
9. Puhlmann, F.: Why do we actually need the pi-calculus for business process management? In: Abramowicz, W., Mayr, H.C. (eds.) BIS. LNI., GI, vol. 85, pp. 77–89 (2006)
10. Smith, H., Fingar, P.: Business Process Management - The Third Wave. Meghan-Kiffer Press (2002)
11. Dong, Y., Shen-sheng, Z.: Approach for modeling using pi-calculus. Journal of Zhejiang University Science 4(6), 643–650 (2003)
12. Puhlmann, F., Weske, M.: Using the pi-calculus for formalizing workflow patterns. In: Business Process Management, pp. 153–168 (2005)
13. Schaffner, J., Meyer, H., Weske, M.: A formal model for mixed initiative service composition. In: SCC 2007. IEEE International Conference on Services Computing, Salt Lake City, USA (2007)
14. Agarwal, S., Studer, R.: Automatic matchmaking of web services. In: ICWS 2006. Proceedings of the IEEE International Conference on Web Services, Washington, DC, pp. 45–54. IEEE Computer Society, USA (2006)
15. Puhlmann, F., Weske, M.: Investigations on soundness regarding lazy activities. In: Business Process Management, pp. 145–160 (2006)
16. Mou, Y., Cao, J., sheng Zhang, S.: A process component model for enterprise business knowledge reuse. In: IEEE SCC, pp. 409–412. IEEE Computer Society Press, Los Alamitos (2004)
17. Hepp, M., Roman, D.: An ontology framework for semantic business process management. In: Proceedings of Wirtschaftsinformatik 2007, Karlsruhe, Germany (March 2007)
18. Hepp, M., Leymann, F., Domingue, J., Wahler, A., Fensel, D.: Semantic business process management: A vision towards using semantic web services for business process management
19. Scheer, A.W.: ARIS - Vom Geschäftsprozess zum Anwendungssystem. Springer, Heidelberg (1998)

A Vocabulary and Execution Model for Declarative Service Orchestration

Stijn Goedertier and Jan Vanthienen

Department of Decision Sciences & Information Management,
Katholieke Universiteit Leuven, Belgium
stijn.goedertier,jan.vanthienen@econ.kuleuven.be

Abstract. In this paper we introduce the EM-BrA^{2}CE Framework: a vocabulary and execution model for dynamic service orchestration that allows to combine many business rule types, independently of the used methods for knowledge representation and reasoning. The vocabulary is described in terms of the Semantics for Business Vocabulary and Rules (SBVR) and the execution model is presented as a colored Petri net (CP-Net).

1 Introduction

Declarative service orchestrations must be described by truly declarative process models that capture information about the business concerns that govern business processes leaving as much freedom as is permissible at execution time for determining a valid and suitable execution plan. In the literature a number of languages for declarative process modeling can be identified with very distinct knowledge representation backgrounds. These languages only model one aspect of the many business concerns that exist in reality. What is needed are meaningful ways to combine several kinds of expressions, called business rule types, independently of the used methods for knowledge representation and reasoning. In this paper we introduce the EM-BrA^{2}CE (Enterprise Modeling using Business Rules, Agents, Activities, Concepts and Events) Framework [1] that is intended to be used as a foundation in integrating and developing existing and new forms of declarative business process modeling.

The remainder of this paper is structured as follows. In section 2 we introduce the EM-BrA^{2}CE Vocabulary as an extension to the SBVR and provide a declarative view on state. In section 3 we introduce the EM-BrA^{2}CE Execution Model as a life cycle of state transitions and define sixteen business rule types that can constrain these state transitions. Finally, in section 4 we relate the framework to the relevant literature.

2 A Declarative View on State

Declarative process models should be on the one hand comprehensible so that they can be understood by business people and on the other hand formal so that they can be enforced by information systems. The Semantics of Business Vocabulary and Business Rules (SBVR) [2] is a language for business modeling that has such property. On the one hand, the SBVR provides a number of conceptual vocabularies for modeling a business domain as a vocabulary and a set of rules. On the other hand, the SBVR has a vocabulary to describe the semantic structure and meaning of expressions in terms of semantic formulations. This combination of linguistics and formal logic provides the fundamentals for developing a natural language parser that allows to express the meaning of rules that have a textual notation [3].

The current SBVR specification [2] does not have a built-in vocabulary for expressing process-related concepts such as agent, activity, event or deontic assignment. In [1] we define an SBVR vocabulary for expressing process-related concepts, called the EM-BrA^{2}CE Vocabulary. In natural language, but also in the literature, service is an overloaded term. The vocabulary makes a distinction between **service provider**, **service capability** and **service**. A service provider (or agent) performs the (coordination) work, e.g. a web service or an employee, [4]. A service capability is the ability to perform activities of a particular activity type. A service is the activity of delivering actual physical and informational value to a customer [5].

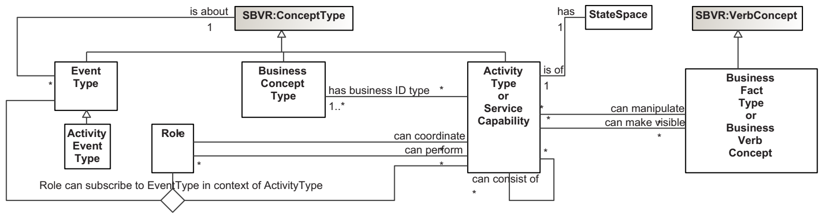
In the EM-BRA^{2}CE Framework, each service capability (or activity type) can be **modeled** by describing its **state space** and the set of business rules that constrain movements in this state space. As displayed in Fig. 1(a), the state space of a service capability is described by facts about:

– the roles that are involved in the service capability
– the sub-service capabilities out of which the service capability can consist
– the business fact types whose instances can be manipulated or made visible by agents that coordinate or perform the service capability
– the event types that agents of a particular role can perceive in the context of the service capability

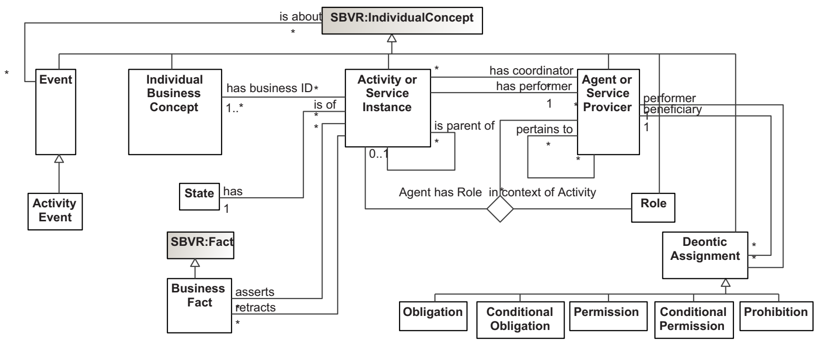
As displayed in Fig. 1(b), the state of a service is described by facts about:

– the agents and the roles that the agents have in the context of the service
– the sub-services out of which the service consists
– the business facts that hold at that particular time
– the events that have occurred within the context of the service
– the deontic assignments (i.e. the obligations and permissions) that apply

Unlike many ontologies for business modeling [6,7], a distinction is made between activities and events. Activities are performed by agents and have a particular duration whereas events occur instantaneously and represent a state change in the world. The distinction between activity and event allows for reactive behavior. At each point during execution the history of a business process instance might be inspected through the use of an event query language. When an external event is added to the current state of an activity, that activity enters a new state. In this new state, the activity can undergo an additional transition as a reaction to the external event. Because this second transition is also recorded as an activity event, the system keeps track of its own state, reflecting the external (composite) events that have been reacted upon. The latter prevents the system from reacting twice to the same event.



(a) type level



(b) instance level

Fig. 1. A MOF/UML representation of the EM-BrA^{2}CE Vocabulary

3 An Execution Model and Business Rule Types

The possible movements within a service capability's state space, can be described by twelve generic activity life cycle state transitions that are enumerated in Table 1. Because these transitions are generic, they provide a means of defining an **execution model**. In [1] we formally specify the EM-BrA^{2}CE Execution Model in terms of colored Petri nets. When an activity state transition occurs, a business process instance enters a new state and the transition is recorded as an activity event.

Such a dynamic system could be interpreted as non-monotonic. However, each state must be considered as a logical system in its own right. Each time the facts of a service state are changed, this creates a new and different logic system with a monotonic semantics. Monotonicity is a desired property. The contrary, having derived facts that become inconsistent trough the manipulation of facts, would not be in keeping with the intent of the framework to provide a unifying execution semantics within which several knowledge representation paradigms

Table 1. The state transitions in the EM-BrA^{2}CE Execution Model [1]

aspect	**state transition**	**meaning**	**activity event type**
coordination	create	the creation of the activity	created
	schedule	setting the due date of the activity	scheduled
	assign	assigning a worker to the activity	assigned
	revoke	revoking an assignment	revoked
actual work	start	the start of the activity	started
	addFact	the addition of facts	factAdded
	updateFact	the updating of facts	factUpdated
	removeFact	the removal of facts	factRemoved
	complete	the completion of the activity	completed
exceptions	skip	the incomplete termination of the activity	skipped
	abort	the termination with compensation	aborted
	redo	the redoing with preliminary roll-back	redone

Table 2. Relating transition types to business rule types

aspect	**business rule type**	**related work**	**create**	**schedule**	**assign**	**revoke**	**start**	**addFact**	**removeFact**	**updateFact**	**complete**	**skip**	**abort**	**redo**
control flow	**Temporal deontic rule**	[8],[9]		x			x							x
	Activity precondition	[4]					x							x
	Activity postcondition	[4],[10]									x			
	Dynamic integrity	[6]					x	x	x	x				x
	Activity cardinality	[11]	x				x					x		x
	Serial activity constraint	[12]		x			x							x
	Activity order	[12],[11]		x			x							x
	Activity exclusion	[12],[11]	x				x							x
	Activity inclusion	[12],[11]									x			
	Reaction rule	[8]	x				x							x
data	**Static integrity**	[6]						x	x	x				
	Derivation rule	[6]						x		x				
organization	**Activity authorization**	[13]			x									
	Activity allocation rule				x									
	Visibility constraint	[13]												
	Event subscription	[13]												

can be used independently of one another. The reason for this is that non-monotonicity requires belief revision. In a setting of complex logical formulae of different kinds of knowledge representation paradigms, belief revision would be a complex operation.

Each service orchestration can be modeled by describing its state space and the set of business rules that constrain the possible transitions in this state space.

In [1] a total of sixteen business rule types are identified. They refer to one of the three aspects of business process modeling that are generally considered: the control-flow, the data and the organizational aspect. Prior to the occurrence of a given state transition, particular business rule types must be evaluated, as indicated in Table 2.

4 Evaluation and Related Work

In the literature, languages such as the case handling paradigm [10], OWL-S [14], ContractLog [8], the constraint specification framework of Sadiq et al. [12], the Web Service Modeling Ontology (WSMO) [4], the ConDec language [11] and the PENELOPE language [9] can be categorized as declarative languages. However, these languages only model one aspect of the many business concerns that exist in reality. For instance, the ConDec language and the PENELOPE language only allow to express business rules about sequence and timing constraints, i.e. the control-flow perspective. Web Service Orchestration standards such as OWL-S and WSMO [4], on the other hand, include the organizational and data model aspects, but do not provide a temporal logic to express temporal relationships between concepts such as activities or events. Moreover, these languages make use of very different knowledge representation paradigms. For instance, the ConDec language is expressed in Linear Temporal Logic (LTL) whereas the PENELOPE language is expressed in terms of the Event Calculus. These heterogenous knowledge representation paradigms raise the question how it will be possible to reason about such heterogeneously expressed knowledge. Finally, these languages do not have an explicit execution model or have an execution model that explicitly assumes either human or machine-mediated service enactment. The WSMO, for instance, has a specific execution model (WSMX) that is focused on web service mediated service orchestration. The case handling paradigm, for instance, assumes humans to perform atomic tasks but has an orchestration engine to perform the orchestration (coordination) work.

The EM-BrA^{2}CE Framework has a declarative view on process state, stays independent of the knowledge representation paradigm, covers all process modeling aspects, has a formal execution model, [1] and makes abstraction of the differences between humans and machines. We have used the colored Petri net model, supplemented with state space descriptions and business rules, to generate simulation event logs. In [15] we show how these business rules can in turn be learned from these event logs supplemented with noise.

5 Conclusion

In the literature there are many languages that express business concerns regarding service orchestration. In this paper we have introduced a unifying framework with a vocabulary and a formal execution model in which several business rule types for declarative process modeling can be modularly incorporated. The business rule types identified by the framework allow to consider a broad range of control flow, data and organizational modeling aspects.

References

1. Goedertier, S., Vanthienen, J.: EM-BrA^{2}CE v0.2: A Vocabulary and Execution Model for Declarative Process Models. Fetew research report, K.U.Leuven (2007)
2. Object Management Group: Semantics of Business Vocabulary and Business Rules (SBVR) – Interim Specification. OMG Document – dtc/06-03-02 (2006)
3. Unisys: Unisys rules modeler. [10-11-2005] (2005), http://www.unisys.com
4. Roman, D., et al.: Web service modeling ontology. Applied Ontology 1(1), 77–106 (2005)
5. Preist, C.: A conceptual architecture for semantic web services. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 395–409. Springer, Heidelberg (2004)
6. Wagner, G.: The agent-object-relationship metamodel: towards a unified view of state and behavior. Inf. Syst. 28(5), 475–504 (2003)
7. Guizzardi, G., Wagner, G.: Ontologies and Business Systems Analysis. In: Rosemann, M., Green, P. (eds.) Some Applications of a Unified Foundational Ontology in Business Modeling, pp. 345–367. IDEA Publisher (2005)
8. Paschke, A., Bichler, M., Dietrich, J.: Contractlog: An approach to rule based monitoring and execution of service level agreements. In: Adi, A., Stoutenburg, S., Tabet, S. (eds.) RuleML 2005. LNCS, vol. 3791, pp. 209–217. Springer, Heidelberg (2005)
9. Goedertier, S., Vanthienen, J.: Designing compliant business processes with obligations and permissions. [16], 5–14
10. van der Aalst, W.M.P., Weske, M., Grünbauer, D.: Case handling: a new paradigm for business process support. Data Knowl. Eng. 53(2), 129–162 (2005)
11. Pesic, M., van der Aalst, W.M.P.: A declarative approach for flexible business processes management. [16], 169–180
12. Sadiq, S.W., Orlowska, M.E., Sadiq, W.: Specification and validation of process constraints for flexible workflows. Inf. Syst. 30(5), 349–378 (2005)
13. Strembeck, M., Neumann, G.: An integrated approach to engineer and enforce context constraints in rbac environments. ACM Trans. Inf. Syst. Secur. 7(3), 392–427 (2004)
14. The OWL Services Coalition: OWL-S 1.2 Pre-Release (2006), Available from http://www.ai.sri.com/daml/services/owl-s/1.2/
15. Goedertier, S., Martens, D., Baesens, B., Haesen, R., Vanthienen, J.: A new approach for discovering business process models from event logs. FETEW Research Report KBI_0716, K.U.Leuven (2007)
16. Eder, J., Dustdar, S. (eds.): BPM Workshops 2006. LNCS, vol. 4103. Springer, Heidelberg (2006)

Towards Dynamic Matching of Business-Level Protocols in Adaptive Service Compositions

Alan Colman, Linh Duy Pham, Jun Han, and Jean-Guy Schneider

Faculty of ICT, Swinburne University of Technology
PO Box 218, Hawthorn, 3122, Australia
{acolman, lpham, jhan, jschneider}@ict.swin.edu.au

Abstract. In a service composition, it is necessary to ensure that the behaviour of a constituent service is consistent with the requirements of the composition. In an *adaptive* service composition those behavioural requirements may be continually changing. This paper shows how the behavioural requirements in abstract service definitions (roles) can be dynamically and incrementally defined using constraints. These constraints are then used to generate finite state automata, which are used to check the compatibility of candidate services that have their behaviour expressed in static interface descriptions such as OWL-S.

1 Introduction

In addition to the development of services, Service-Oriented Computing (SOC) application development is a process consisting of service discovery, evaluation and composition. To support the composition of services, a variety of standards such as WSDL and SOAP have helped resolve the heterogeneity in implementation platforms. However, standardisations in the syntactic interface description of services (e.g. WSDL) alone are not sufficient to ensure the correct interoperation of services.

Previous work in component-based software engineering suggests that there are four levels of component interface specification: *syntactic*, *behavioural*, *synchronisation* and *QoS (Quality of Service)* [2]. In Web services, the need for semantically rich descriptions of services has resulted in a number of initiatives such as OWL-S and WSMO. In this paper we focus on *behavioural* interoperability, in particular the sequence of exchanged messages (protocols) between services. Of the above initiatives OWL-S provides explicit semantics for specifying the behaviour of services in its process model. OWL-S specifies its interaction in terms of definitions for *atomic*, *simple* and *composite processes*. WSMO, on the other hand, specifies the behavioural protocol of a service by using Abstract State Machines as the underlying formalism to represent the service's orchestration and choreography. WSMO also provides the specification of mediators to solve the mismatches at the data, communication protocol, and process levels. In contrast, rather than stipulating the existence of a new type of component in the Web services infrastructure, OWL-S provides to Web services and their clients the information that is needed to find existing mediators that can reconcile their mismatches [1].

The limitation of behavioural descriptions in OWL-S and WSMO is that they are *static* descriptions defined on the interface of a service. This is fine for individual service instances if all we are attempting to do is to match two services. To achieve interoperation of two static services at a behavioural level, the services need to share a behavioural ontology, and some mechanism needs to be provided to check the compatibility of the descriptions so that the service can interoperate. For example, recent work [4, 7, 8] addresses the problem of interoperation and matching between services that provide static behavioural descriptions.

As well as describing behaviour of the interface of a service, other approaches describe the behaviour of service *compositions*, either as orchestrations (BPEL) or choreographies (WS-CDL). However, like the above rich interface descriptions, these compositional descriptions are not designed to be dynamically generated, or incrementally altered, at runtime. The Role-Oriented Adaptive Design (ROAD) framework [3], on the other hand, does support the dynamic composition of services. In this paper, we show how the dynamic behavioural descriptions that are contained within a ROAD composite can be matched to the static behavioural descriptions presented by service interfaces in the form of OWL-S.

If we are to create dynamic compositions of services, the composition itself may have changing behavioural requirements. Consider a Library that buys books from a range of vendors. To do this the Library uses a broking service composite (Book Broker composition) that has relationships with a range of vendors. These vendors provide Web services that have various business-level rules for quoting, ordering and payment. Clearly, there are many business level protocols that govern the interactions between the buyer and the vendors. Some of these involve the defining acceptable sequences of interactions or constraints on ordering of interactions; for example, terms of payment (payment before delivery, or delivery before payment), conditions of order cancellation, non-delivery of goods, etc. As these services have been developed by different organisations, there may be mismatches between the constraints on the sequence of interactions between services (i.e. *"protocol mismatch"*).

For example, one book vendor may require payment before it delivers an order of books, while another vendor may be prepared to deliver books on receipt of an order and expect payment on invoice. The Book Broker composition has to mediate a range of interaction requirements from both the buyers and the vendors that use its service. As the Book Broker service cannot know in advance all the possible protocols that potential vendors and buyers might present, it therefore needs to be able to adapt or dynamically generate the interfaces it presents in order to interact with a range of vendors and buyer services whose interaction requirements have not been foreseen during the design phase. The protocol descriptions in these dynamically generated interfaces must be consistent with the protocols of the constituent services that are being dynamically added to the composition.

2 Dynamic Protocol Specification and Aggregation in ROAD

Our approach to creating adaptive service compositions is to use the ROAD (Role-Oriented Adaptive Design) framework [3]. In ROAD, service compositions are role-based interaction structures, where *roles are dynamic abstract service definitions* that (among other things) define the expected behaviour of a service that is bound to that role. Services interact with each other *via* their roles. To show how protocols are represented in ROAD, we will illustrate a service composite consisting of two roles: Buyer, and Vendor. As shown in Fig. 1, different services can play a role at different times, i.e. Amazon and Barnes & Noble services are candidate players of the Vendor role.

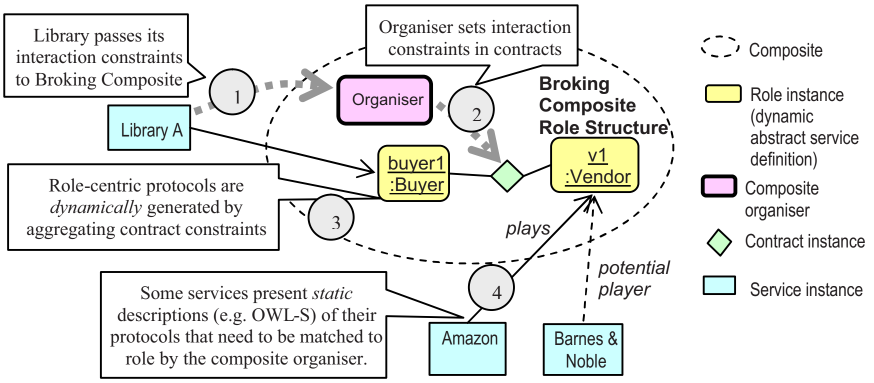

Library passes its interaction constraints to Broking Composite
Organiser sets interaction constraints in contracts
Composite
Role instance (dynamic abstract service definition)
Composite organiser
Contract instance
Service instance
1
2
3
4
Organiser
Broking Composite Role Structure
Library A
buyer1 :Buyer
v1 :Vendor
Role-centric protocols are dynamically generated by aggregating contract constraints
plays
potential player
Some services present static descriptions (e.g. OWL-S) of their protocols that need to be matched to role by the composite organiser.
Amazon
Barnes & Noble

Fig. 1. Protocols in a ROAD service composite

A ROAD *contract* is a rich connector between two roles. More than just a binding, it stores the interaction constraints and provides a mechanism to intercept the messages exchanged between two roles during run-time in order to verify the interaction. The monitoring ability of ROAD contracts is discussed elsewhere [9]. Of particular relevance to the discussion in this paper, is the ability of ROAD contracts to define *protocol clauses* that describe permissible sequences of transactions that can occur between roles.

An *organiser* provides an overall management over roles and contracts within its composition. The organiser adapts the composite by creating (and destroying) roles, and creating (and revoking) contracts between these roles and the binding between roles and services. The organiser also provides a management interface that allows the non-functional requirements (i.e. behavioural or QoS requirements) to be set. For example, in Fig. 1 (step 1) the library passes its interaction constraints to the composite organiser. In order to specify the protocols, ROAD uses a *temporal constraint* language Interaction Rule Specification (IRS) [5]. IRS is used to dynamically specify the temporal constraints of the protocols between two roles. These constraints are stored in the contract between those roles. Listing 1 illustrates a (somewhat simplified) constraint specification for a Buyer-Vendor contract protocol, and shows how constraints can be incrementally modified at run-time by removing unwanted constraints and adding new constraints; unlike other approaches such as BPEL where the entire composition script has to be reworked. Our approach also provides an automatic consistency checking to ensure no violation has occurred during the changes. The reader is referred to [9] where we describe how IRS constraints can be incrementally added or deleted from ROAD contracts.

```
contract protocol BuyerVendor{
   Vendor.order precedes Buyer.orderConfirmation globally;
   Buyer.orderConfirmation precedes Buyer.receiveDelivery globally;
   // ~~Buyer.receiveDelivery leads to Broker.receivePayment globally;~~  // deleted
   Broker.receivePayment precedes Buyer.receivePaymentAck globally;
  Buyer.receivePaymentAck precedes Buyer.receiveDelivery globally;  // newly added constraint
}
```

Listing 1. Modified temporal constraints in Buyer-Broker contract in IRS notation

In the context of maintaining valid protocol descriptions in its composite, the organiser is responsible for writing protocol constraints into the contracts it creates between the roles (Step 2). These constraints are converted into finite state automata (FSA) within the contract so that interaction can be checked at runtime, as discussed in [5, 9]. (For multiple contracts in a composite, the organiser also maintains a model of any dependency constraints between these contract protocols.) Where a role is bound to *more* than one contract (not shown in Fig 1.), these FSAs are aggregated into a single protocol description for the role (a 'role-centric protocol' as in step 3).

Now that we have a dynamically constructed a behavioural specification in a role's abstract service description, the problem remains how to match concrete services with this specification (step 4 in Fig. 1). For the purposes of this discussion, we will assume that candidate services provide a description of their behaviour in OWL-S; however our general approach is not limited to any particular behavioural description. In order to match the service with its behavioural requirements as defined in the role, we need a common formalism that enables reasoning about temporal constraints. We use FSAs for this purpose because, as described in the next section, FSAs allow us to identify different types of mismatch. To convert the claimed behaviour of a service as expressed in its OWL-S description (or other similar descriptions) to an FSA, we utilise the Mindswap API [6] to parse the OWL-S files. Each atomic process is converted to a simple FSA. Following a method described by others in [7], these elementary atomic process FSAs are then composed, taking account of the control constructs of which they are a part (e.g. sequence, choice, repeat-while, repeat-until, split, or split-join). We now have FSA representations of the protocols of both the role and any candidate services that can be compared for matching.

3 Towards Resolving Protocol Mismatches

As described above, the behavioural protocols of the role (i.e. the role-centric protocol) and the service's OWL-S files are translated into FSAs. In order to check for the compatibility of a service against a role, we extend an approach described in [5, 10], whereby the service provider can be said to *satisfy* the behavioural requirements of a role when the intersection of the role's FSA and the complement of the service provider's FSA is empty. While the above approach can find an *exact* match between the behavioural requirements, as defined in a role's abstract service description and a concrete service's OWL-S description, in many cases it may be difficult to find a service that matches the composition's requirements exactly[1]. We therefore extend this approach by matching FSAs based on the following categories:

- **fully compatible:** For every path that leads to final states in the role's FSA, there is an identical path in the service's FSA. This is not symmetrical in the sense that the service's FSA might also support other paths that do not exist in the role's FSA. However from the role's perspective, these extra paths are of no interest.
- **exactly compatible:** This is the special case of the fully compatible case where every path that leads to final states in the role's and service's FSAs are identical.
- **partially compatible:** The role's FSA has some paths that lead to final states which are identical to those in the service's FSA. Not all the paths that lead to final states in the role's FSA are supported by the service's FSA.
- **incompatible:** The role's FSA and the service's FSA do not have common paths that lead to final states.

The above categories are illustrated in Fig. 2.

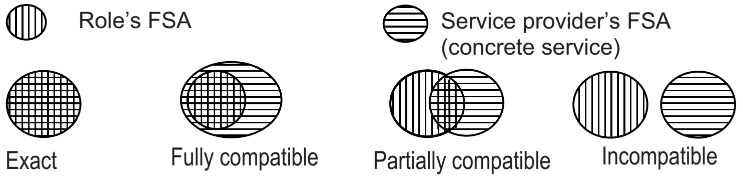


Fig. 2. Categories of protocol matching between services

To carry out the matching process, we make use of an FSA intersection algorithm. Firstly, the intersection of the role FSA and the service's FSA is found. In the case that FSA_intersect does not have any path that leads to final states (i.e. the FSA_intersect is *empty*), it corresponds to the 'Incompatible' case and we do not proceed any further. If the FSA_intersect has some paths that lead to final states (i.e. the FSA_intersect is *not empty*), these paths are evaluated to see if they are exactly the same as all the paths in role's FSA. The evaluation is done by computing the intersection of the role's FSA with the complement of FSA_intersect. If the resultant FSA is *empty*, it is concluded that the service FSA matches the role's requirement ('Exact' or 'Fully compatible' case). If the resultant FSA is *not empty,* it corresponds to the 'Partially compatible' case.

In the case of *full* or *exact* compatibility, the service can fulfil the role's behavioural protocol. The service will then be bound to the role as its player. In the case of 'Incompatible', the service cannot fulfil any of the role's interaction, so this service is of no interest to the composition.

[1] There exists other work [4, 8] that address the matching and ranking the compatibility at the structural level, in this paper we are concerned only at the compatibility categorisation at the behavioural level.

In the case of *partial* compatibility, the service can partially satisfy (*satifice*) the role's behavioural protocol, i.e. there is at least one possible conversation that can take place among the role and the service. If the composition is to use this service, it needs to have an adaptation mechanism to ensure that all the conversations between the role and the service are supported by the service, i.e. only the conversations that follow the common paths are allowed. For the remaining interaction paths required by the role, the composite organiser will search for other services that can satisfy those paths. This becomes the problem of functional service composition rather than just protocol matching which we plan to address in future work.

In conclusion, it is necessary in a service composition to ensure that the behaviour of a constituent service is consistent with the requirements of the composition. In *adaptive* compositions, those behavioural requirements may be continually changing. In the context of the ROAD framework, this paper shows how the behavioural requirements in abstract service definitions (roles) can be dynamically and incrementally defined using IRS constraints. These constraints are then used to generate FSAs (finite state automata). These FSAs are then used to automatically check the compatibility of candidate services that have their behaviour expressed in static interface descriptions such as OWL-S. Further work needs to be done to address the problem of mismatches between composite behavioural requirements and the actual behaviour of services.

References

1. Ankolekar, A., Martin, D., McGuinness, D., McIlraith, S., Paolucci, M., Parsia, B.: OWL-S' relationship to selected other technologies. [Online]: http://www.w3.org/Submission/2004/SUBM-OWL-S-related-20041122/
2. Beugnard, A., et al.: Making components contract aware. IEEE Computer 32, 38–45 (1999)
3. Colman, A., Han, J.: Using role-based coordination to achieve software adaptability. Science of Computer Programming 64, 223–245 (2007)
4. Jaeger, M.C., et al.: Ranked Matching for Service Descriptions using OWL-S. In: KiVS 2005. Kommunikation in verteilten Systemen, pp. 91–102. Springer, Heidelberg (2005)
5. Jin, Y., Han, J.: Consistency and interoperability checking for component interaction rules. In: Proc. of the 12th Asia-Pacific Software Engineering Conference, pp. 595–602 (2005)
6. Mindswap: Mindswap OWL-S API. [Online]: http://www.mindswap.org/2004/owl-s/api/
7. Mokhtar, S.B., Georgantas, N., Issarny, V.: COCOA: Conversation-based service composition for pervasive computing environments. In: ICPS 2006. Proc. of International Conference on Pervasive Services, France, pp. 29–38 (2006)
8. Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.: Semantic matching of web services capabilities. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, pp. 333–347. Springer, Heidelberg (2002)
9. Pham, L.D., Colman, A., Schneider, J.-G.: Dynamic protocol aggregation and adaptation for service oriented computing. In: ASWEC 2007. Proc. of the 18th Australian Software Engineering Conference, pp. 39–48. IEEE Computer Society, Australia (2007)
10. Yu, J., et al.: Pattern based property specification and verification for service composition. In: Aberer, K., et al. (eds.) WISE 2006. LNCS, vol. 4255, pp. 156–168. Springer, Heidelberg (2006)

Retrieving Substitute Services Using Semantic Annotations: A Foodshop Case Study

F. Calore, D. Lombardi, E. Mussi, P. Plebani, and B. Pernici

Politecnico di Milano, Italy
barbara.pernici@polimi.it

Abstract. A case study discussing the development of semantically annotated web services with SAWSDL is presented in the paper. The annotations are used to identify substitute services in case of failures, in the context of food shops. An ontology for food has been developed, a set of annotated services, and an algorithm to evaluate service substitutability based on semantic information has been applied.

1 Introduction

Semantic annotations for web services are being proposed to facilitate service retrieval and composition. Several approaches to semantic web services are being proposed in the literature, such as the WSMO approach using a logic-based approach to characterize services to enable their composition with a goal-based approach [1], Meteor-S, were services are selected on the basis of their semantic characteristics described with WSDL-S [2] and QoS propoperties, and more recently the SAWSDL language, proposed by W3C as an evolution of WSDL-S.

While semantic annotations are being proposed in the literature and research work focus on semantically based service composition, there is limited work in the literature focusing on the design of semantically annotated web services. A first approach proposed by W3C is mainly a usage guide to SAWDL language [3].

The main goal of this paper is to discuss the steps involved in the creation and usage of semantically annotated services in a case study. The goal of the annotation is to support service retrieval and substitutability in highly variable usage environment. The case study being presented in the domain of food items. This work has been developed within the European WS-Diamond (Web Services - DIAgnosability, Monitoring and Diagnosis) project, which has the goal of supporting service repair in case of failure with monitoring and diagnostic techniques, and planning repair according to a set of available repair actions [4].

In Section 2, the case study and the development of a food ontology to be used for web services annotation, based on the European classification of food are presented. The semantic annotations are used to support service retrieval for substitutability in Section 3.

2 A Food Ontology for Web Services

The reference scenario is a FoodShop Company that sells and delivers food using Web service technology. The scenario is taken from the WS-Diamond Project [4]. The company has an online Shop that customers use to select and order food. The Shop does not have a physical counterpart, as it stores and delivers food using either Warehouses or Suppliers. Warehouses are responsible for stocking unperishable goods and physically delivering items to customers. In case of perishable items, that cannot be stocked, or of out-of-stock items, the FoodShop Company must interact with a Supplier.

The Shop, Warehouses, and Suppliers are Web services. They are described by means of WSDL interfaces and electronic interactions among them are carried out exchanging SOAP messages.
The runtime system used by the FoodShop Company to process orders relies on the PAWS framework [5], which enables composing Web services by using a model-driven approach. With PAWS, the process designer has the capability to define the abstract structure of the process while it is the framework itself that selects the most suitable Web services to be used during the orchestration of the composition. The PAWS framework selects Web services according to functional criteria and semantic criteria. Functional criteria ensure that selected Web services match the requirements from a functional point of view, that is they provide all the necessary operations to enact the orchestration of the composition, while semantic criteria ensure that selected Web services match the semantic requirements generated by the customers of the FoodShop Company, to provide the food items that customers want to purchase.

A domain-specific ontology has been designed to support the realization of the Foodshop Company application. We decided to design a new ontology instead of using an existing one since existing food ontologies would not fit with the needs expressed by our scenario. For example, the food ontology used in W3C examples [1] is not suited to describe food from a shopping point of view, since it is better tailored for the description of prepared food, such as menus in restaurants. Another food ontology which has been made available to support ontology alignment evaluations [2], focuses on agro-food characteristics, such as plants and diseases, rather than on food as shopping item.

The ontology designed for our application has been derived from a document, created by the Food Safety Authority of Ireland [6], based on the European Union classification of food. It describes twenty-one categories in which goods are classified, with a very large number of examples. The ontology has been created using the Protégé ontology editor [3], tailoring it to the case study [4].

The twenty-one classes have been subdivided in subclasses to increase the level of detail and, since it was needed to separate perishable from not perish-

[1] http://www.daml.org/ontologies/76
[2] http://oaei.ontologymatching.org/2006/food/
[3] http://protege.stanford.edu/
[4] The food ontology developed in the case study is available at http://web.tiscali.it/lchkl/ontology/FoodOntology.owl

able food, leaves containing both of them have been subdivided according to this principle. A datatype property named "isPerishable", with Boolean range, has been defined and associated with classes to distinguish the ones with perishable individuals from the others. This restriction and the disjoint was among siblings classes entails that the ontology uses OWL-DL [4] sublanguage. The ontology population with individuals has been done using the examples inside Food Safety Authority of Ireland document as main reference, with an appropriate selection of additional individuals from information on food producers and big food resellers websites, reaching this way approximately a thousand instances. Figure 1 reports an extract of our FoodShop ontology.

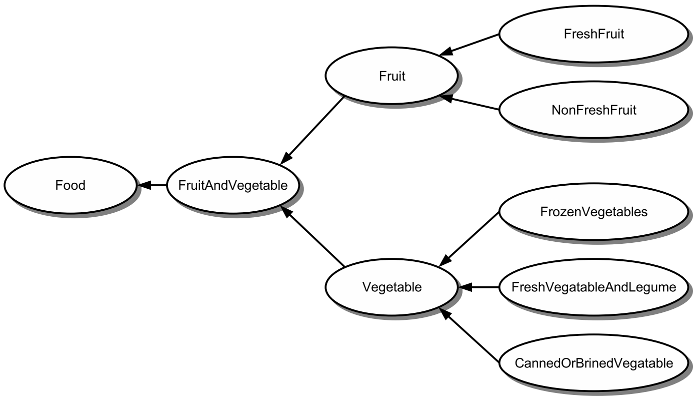


Fig. 1. FruitAndVegetable ontology class

3 Service Retrieval for Substitutability

SAWSDL proposes an approach to enhance the semantic of a WSDL description by annotation. According to this specification, the elements in the WSDL description can be associated to the concepts included in the ontology. Annotation can affect all the elements of a WSDL description, i.e., part, message, operation, or even some of them. An example of annotated message follows:

```
<wsdl:message name="reserveRequest">
    <wsdl:part name="itemList" type="xsd:string"
               wssem:modelReference="Ontology5#FrozenMeat"/>
    <wsdl:part name="customerInfo" type="xsd:string"/>
    <wsdl:part name="PID" type="xsd:int"/>
</wsdl:message>
```

This annotation of the document has been bound to the "itemList" part of message "reserveRequest". This choice is motivated because of what "itemList" represents, that is a string used by the message to hold food items while the remaining parts, and therefore the message itself from a global point of view, do not closely match with any of Foods semantic concepts.

Semantic annotations have been exploited during Web service retrieval. We assume that a user request is specified as a SAWSDL description where the desired operations are listed and annotated as well as their input and output parameters.

The Web service retrieval algorithm is based on a similarity distance computation. The higher a published Web service is similar to the requested one, the better the published Web service fulfill the user request.

Similarity among Web services is computed both comparing names and annotations at all levels in the SAWSDL description: service level, operation level and parameter level. On the one hand, name comparison relies on the assumption that all the names are included in a reference ontology. Such an ontology can be, for instance, Wordnet. On the other hand, annotation comparison relies on the same ontology adopted for annotating the Web service description.

Name similarity. Given two names their similarity is returned by the function $simName(name_a, name_b)$. In detail:

- $simName = 1$ if $name_a = name_b$.
- $simName = \frac{1}{(lengthpath(name_a, name_b)+1)}$ if $name_a$ and $name_b$ are connected by a subsumption path and *lenghtpath* return how many hops constitutes such a path.
- simName = 0 if $name_a$ and $name_b$ are not connected or there are connected by a “opposite-to” relationship.

Annotation similarity. The similarity evaluation between two annotations depends on the nature of the annotations that could be terms or properties. More precisely, the $simAnn(ann_a, ann_b)$ is defined as follows:

- $simAnn = simName$ if ann_a and ann_b are both terms.
- $simAnn = \frac{max(simAnn(ann_a, term_{b,i}))}{2} \forall i \in cod(ann_b)$ if ann_a is a term and ann_b is a restriction on a property.
- $simAnn = \frac{max(simAnn(term_{a,i}, ann_b))}{2} \forall i \in cod(ann_a)$ if ann_a is a restriction on property and ann_b is a term
- in case both ann_a and ann_b are restriction on property the similarity takes into account the relationship among the restrictions:
  - if the restriction are equivalent $simAnn = 1$
  - if the properties have not any relation then $simAnn = 0$
  - if the properties have some relations:

$$SimAnn = \frac{SimProp(ann_a, ann_b)}{2} + \frac{SimName(dom(ann_a), dom(ann_b))}{2} \quad (1)$$

$$SimProp(ann_a, ann_b) = \frac{1}{level(ann_a, ann_b) + 1}$$

SimWS. Given these two functions, i.e., *simName* and *simAnn*, the similarity among Web services is obtained as the average of the similarity among operations:

$$SimWS(s_a, s_b) = \Sigma_{i=1,N} \frac{max(SimOp(op_{a,i}, op_{b_j}))}{n} \quad (2)$$

SimOp returns the operation similarity which takes into account both the similarity among the operation names (*simOpName*) and the similarity among the input and output parameters (*simOpPar*)

$$SimOp(op_a, op_b) = \frac{SimOpName(op_a.name, op_b.name)}{2} + \frac{SimOpPar(op_a.par, op_b.par)}{2} \tag{3}$$

if the both requested and published operations are annotated then *simOpName* = *simAnn*. Otherwise the *simName* are used to compare the names adopted to identify the operations.

About the parameters similarity the same importance has given to both the inputs and output parameters.

$$SimPar(par_a, par_b) = \frac{SimParIn(par_a.in, par_b.in)}{2} + \frac{SimParOut(par_a.out, par_b.out)}{2} \tag{4}$$

The similarity is obtained comparing the names or the annotation associated to the parameter name using the *nameSim* and *annSim* introduced above.

To test the effectiveness of semantic annotations, similar services have been annotated with links to different semantic concepts inside the food ontology. Then, with a semantic search tool [7], some searches have been done, to see if the results provided by the tool were coherent with the annotations. In particular, the test concerned Web services annotated with classes related with father/child relationships and with completely disjoint classes. The search was guided by means of a similarity threshold to filter out results, and for the tests in this paper its value has been set to 0.1, a low value, in order to better observe how the tool works with disjoint classes.

Obtained results were good. In case of two Web services annotated with classes related with a father/child relationship, the tool retrieves both services, assigning them two different scores, the higher to the searched one and the lower to its child (or the father in a second inverted test). On the contrary, for services annotated with disjoint classes only one service has been retrieved by the tool, in accordance to the disjointness.

In the case study, one of the major constraints about semantic annotations for WSDL is related to multiple annotations. There is actually a way to represent a single entity within a WSDL file using different semantic concepts but neither WSDL-S nor SAWSDL declare any relation between different URIs composing a multiple annotation, even if they all have to be considered admissible. It is hence impossible, for this project, to semantically define a Web service as a multi-item store.

4 Concluding Remarks and Future Work

In this work an ontology was designed describing a specific domain, namely food, and it has been used to annotate WSDL documents in compliance with SAWSDL rules to dynamically retrieve Web services.

Open issues remain concerning the design of ontologies for web service annotation, which emerged during the case study development. While in the present case study the adopted ontology focuses on the semantics of contents of exchanged data, there is no systematic criteria for deciding which is the most appropriate ontology. Some experiments using an ontology for operations and another one for messages has lead to imprecise results, in particular if a number of support operations, e.g., to handle exceptions, is provided in a service. In fact, in this case the similarity of support operations might prevail on the semantics of the business part of the service, showing as similar services with mostly different semantics, but similar exception handling operations.

An interesting issue concerns the comparison of services with annotations at different levels. Are the services semantically equivalent if their interface is the same, but one has semantic annotations on messages, while the other has the same annotations on the operations?

Acknowledgements

This work has been partially supported by the European Commission under FP6 project WS-Diamond.

References

1. WSMO Working Group: (Web Service Modeling Ontology), http://www.wsmo.org
2. Cardoso, J., Sheth, A.P.: Semantic e-workflow composition. J. Intell. Inf. Syst. 21(3), 191–225 (2003)
3. Akkiraju, R., Sapkota, B.: Semantic annotations for WSDL and XML Schema - Usage guide (January 2007)
4. Console, L.: WS-Diamond: WS-DIAMOND: an approach to Web Services, DIAgnosability, MONitoring and Diagnosis. In: International e-Challenges Conference, The Hague (October 2007)
5. Ardagna, D., Comuzzi, M., Mussi, E., Plebani, P., Pernici, B.: PAWS: a framework for processes with adaptive web services (accepted for publication on IEEE Software, Special Issue on Realizing Service-Centric Software Systems)
6. Food Safety Authority of Ireland: (Guidance Note on the EU Classification of Food)
7. Prazzoli, P.: Algoritmo di web service retrieval basato su WSDL-S Politecnico di Milano thesis (2006)

A Need for Business Assessment of Semantic Web Services' Applications in Enterprises

Witold Abramowicz, Agata Filipowska, Monika Kaczmarek, and Tomasz Kaczmarek

Poznan University of Economics, Department of Information Systems
{w.abramowicz,a.filipowska,m.kaczmarek,
t.kaczmarek}@kie.ae.poznan.pl

Abstract. Enterprises today face ever increasing pressure to innovate, deliver more value to their customers, decrease costs and shorten product time-to-market etc. The solution to these problems may be application of the SOA paradigm along with the SWS technology. Most of the on-going projects on SOA and SWS focus mainly on the technical aspects. However, the need appears to assess and quantify the business aspects of application of the SWS-based solutions. This position paper elaborates shortly on the motivation and requirements that need to be met in order to supplement the existing SWS technology stack with a business perspective.

1 Introduction

Enterprises today face ever increasing pressure to innovate, deliver more value to their customers, decrease costs and shorten product time-to-market, etc. Their IT models need to support the new requirements of the business environment. Fixed and hard-coded IT applications do not allow adapting quickly to changes, as tight coupling leads to monolithic and brittle distributed applications. Even minor changes in one part of the system may lead to its serious malfunctioning. Moreover, small changes applied in one application often require parallel changes in partners' applications.

The sound solution seems to be the Service Oriented Architecture paradigm focusing on business functions and requirements rather than on the technical layer. SOA encapsulates business functions and makes them available to be used within and between companies. In consequence, it provides the uniform means to offer, discover, interact with and use business capabilities to produce desired effects. Combination of the SOA paradigm with Web services (WS) technology and especially with Semantic Web Services (SWS) has a lot to offer and may bring benefits to various companies what has been confirmed by the current interest of many of them in SOA and SWS-based solutions. Especially appealing is an idea of (semi)automatic dynamic composition of SWSs to implement business processes and in this way ensure the adaptability and flexibility of businesses. However, while IT experts and in some cases business analysts may understand the benefits that may follow the SOA and (S)WSs adoption, the business people need to have some tangible results and metrics that will allow them to assess the real profits, for example the ROI from the implementation of this approach. Moreover, there is a lack of assistance and methodology that would guide the companies through the process of implementation of SOA and SWS-based solutions and give them guidelines how such an investment should be evaluated later on and whether it is profitable at all. Moreover, the additional burdens and challenges that the SWS technology puts on the companies need to be considered.

Whereas much attention lately in various projects was paid to the SWSs and various SWS-based interactions inter alia their composition (e.g. DIP [1], ASG [2], InfraWebs [3], SUPER [4]]), surprisingly little effort was spent on the profitability and business side of these aspects. The aim of our research is to bridge this gap and enhance the SWS technology stack with business perspective.

2 Motivation and Main Goals of the Research

In order to enhance the SWS technology stack with business perspective, the following scientific objectives have to be addressed:

1. Development and validation of the methodology allowing assessment of costs and benefits, in terms of business value, of using SOA solutions and SWS in companies. Enterprises need to know in terms of business value how much they will gain or how much they will loose not introducing certain solutions. The risks connected with the application of the techniques also need to be considered. Moreover, the following questions should be posed: what is the financial benefit of the applications of SOA/semantics-based solutions? what are the risks of following ontological approaches (what requires learning and maintenance)?
   This should be possible if the following dimensions will be taken into account: costs of traditional solution (for a baseline), initial costs (of implementing composition-based solution), maintenance costs, service utilization costs, predicted incomes etc. As the non-functional parameters will be used, the methodology will also allow for quantification of e.g. reliability, security etc.
2. Development and validation of the method of profitability driven dynamic process composition. When performing dynamic service composition not only functional aspects but also business rules and requirements concerning the non-functional properties of business processes should be taken into account. Therefore, the description of the designed process as well as the information on SWSs should be enriched with additional data that would allow performing business analysis. This encompasses:
   - Enhancement of current techniques for the (semi)automatic business process composition using services,
   - Development of methods and tools supporting enterprises in choosing the most suitable service chain to perform business process, with consideration of technical performance and cost-effectiveness (business perspective) of the composition,
   - Research activities into service composition with regard to the effects/benefits and the costs of business processes' adaptability,
   - Evaluation of composition in terms of technical performance and its scalability.

3. Development and adaptation of techniques to perform data mining on service execution data, business analysis based on user feedback, SLAs as well as quality of result quantification etc.
4. The concept of business re-design and replacement of various process fragments, although being very active research area, still lacks maturity. In order for the SWS composition to fulfill its promise on the adaptiveness to the changing conditions of the environment, the more sophisticated algorithms of replacing services need to be taken into account. Since the business environment is very dynamic, the business processes need to be adapted to its changes. However, their quality should not deteriorate in any aspect. Questions related to the replacement of one service with another, often asked by business analysts (however, almost never by IT people), are as follows: Will such a replacement improve my process? How will it affect the overall non-functional characteristic of my process? Are there any additional costs attached to the replacement?
5. Enhancement of current approaches to business process/SWSs description as well as SLA contracts, etc.
6. Adaptation of existing reasoners to the specific needs of business process composition.
7. Enhancement of SWS foundation by learning from their usage in business process composition.

3 Conclusions and Future Work

The business perspective of adoption of SOA and SWS in companies is not fully investigated. The usage of SOA, WS and SWS as well as their compositions, to fulfill certain tasks/business processes in companies, will not gain its momentum unless the business perspective of all aspects is taken into account. Investigating the above mentioned issues may result in establishing of a project that will bring closer the business and IT worlds and allow assessing the SOA and SWS composition in terms of business value. It may lead to the wider adoption and implementation of SOA concepts in companies.

References

[1] http://dip.semanticweb.org
[2] http://asg-platform.org/
[3] http://www.infrawebs.org
[4] http://www.ip-super.org

Author Index