

CONFERENCES IN RESEARCH AND PRACTICE IN
INFORMATION TECHNOLOGY

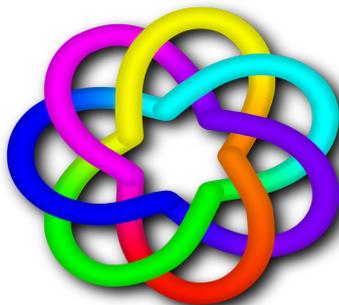
VOLUME 110

CONCEPTUAL MODELLING 2010

AUSTRALIAN COMPUTER SCIENCE COMMUNICATIONS, VOLUME 32, NUMBER 9



AUSTRALIAN
COMPUTER
SOCIETY



 **CORE**
Computing Research & Education

CONCEPTUAL MODELLING 2010

Proceedings of the Seventh Asia-Pacific Conference on
Conceptual Modelling (APCCM 2010),
Brisbane, Australia,
January 2010

Sebastian Link and Aditya Ghose, Eds.

Volume 110 in the Conferences in Research and Practice in Information Technology Series.
Published by the Australian Computer Society Inc.



Published in association with the ACM Digital Library.

Conceptual Modelling 2010. Proceedings of the Seventh Asia-Pacific Conference on Conceptual Modelling (APCCM 2010), Brisbane, Australia, January 2010

Conferences in Research and Practice in Information Technology, Volume 110.

Copyright ©2010, Australian Computer Society. Reproduction for academic, not-for-profit purposes permitted provided the copyright text at the foot of the first page of each paper is included.

Editors:

Sebastian Link

School of Information Management
Victoria University of Wellington
PO Box 600
Wellington
New Zealand
Email: sebastian.link@vuw.ac.nz

Aditya Ghose

School of Computer Science and Software Engineering
University of Wollongong
Wollongong, NSW 2522
Australia
Email: aditya@uow.edu.au

Series Editors:

Vladimir Estivill-Castro, Griffith University, Queensland
Simeon J. Simoff, University of Western Sydney, NSW

crpit@scm.uws.edu.au

Publisher: Australian Computer Society Inc.
PO Box Q534, QVB Post Office
Sydney 1230
New South Wales
Australia.

Conferences in Research and Practice in Information Technology, Volume 110.
ISSN 1445-1336.
ISBN 978-1-920682-92-7.

Printed, December 2009 by UWS Press, Locked Bag 1797, South Penrith DC, NSW 1797, Australia

Cover Design by Matthew Brecknell, QUT
CD Production by FATS Digital, 318 Montague Road, West End QLD 4101, <http://www.fats.com.au/>

The *Conferences in Research and Practice in Information Technology* series aims to disseminate the results of peer-reviewed research in all areas of Information Technology. Further details can be found at <http://crpit.com/>.

Table of Contents

Proceedings of the Seventh Asia-Pacific Conference on Conceptual Modelling (APCCM 2010), Brisbane, Australia, January 2010

Preface	vii
Programme Committee	viii
Organising Committee	ix
Welcome from the Organising Committee	x
CORE - Computing Research & Education	xi
ACSW Conferences and the Australian Computer Science Communications	xii
ACSW and APCCM 2010 Sponsors	xiv

Keynote

A conceptually rich model of business process compliance	3
<i>Guido Governatori, Antonino Rotolo</i>	

Invited Paper

Ontology consolidation in bioinformatics	15
<i>Sven Hartmann, Henning Koehler, Jing Wang</i>	

Contributed Papers

Conceptual modelling in 3D Virtual Worlds for Process Communication	25
<i>Ross A. Brown</i>	
Towards a comprehensive requirements architecture for privacy-aware social recommender systems ..	33
<i>Shan Chen, Mary-Anne Williams</i>	
A conceptual modeling approach for web service composition supporting service re-configuration	43
<i>Georg Grossmann, Michael Schrefl, Markus Stumptner</i>	
The e-decisional community: an integrated knowledge sharing platform	53
<i>Leonardo Mancilla-Amaya, Cesar Sanin, Edward Szczerbicki</i>	
Hetero-homogeneous hierarchies in data warehouses	61
<i>Bernd Neumayr, Michael Schrefl, Bernhard Thalheim</i>	
A researcher expertise search system using ontology-based data mining	71
<i>Ravikarn Punnarut, Gridaphat Sriharee</i>	
Business modeling for service descriptions: a meta model and a UML profile	79
<i>Gregor Schreithauer, Guido Wirtz</i>	

Metric of intrinsic information content for measuring semantic similarity in an ontology	89
<i>Md. Hanif Seddiqui, Masaki Aono</i>	
Author Index	97

Preface

This volume contains the proceedings of the *7th Asia-Pacific Conference on Conceptual Modelling (APCCM 2010)*, held at the Queensland University of Technology in Brisbane, Australia from January 18 to 21, 2010 as part of the Australasian Computer Science Week (ACSW 2010).

The APCCM series focuses on disseminating the results of innovative research in conceptual modelling and related areas, and provides an annual forum for experts from all areas of computer science and information systems with a common interest in the subject. The scope of APCCM 2009 includes areas such as:

- Business, enterprise, process and services modelling;
- Concepts, concept theories and ontologies;
- Conceptual modelling and user participation;
- Conceptual modelling for decision support and expert systems; digital libraries; e-business, e-commerce and e-banking systems; health care systems, knowledge management systems; mobile information systems; user interfaces; and Web-based systems;
- Conceptual modelling of semi-structured data and XML;
- Conceptual modelling of spatial, temporal and biological data;
- Conceptual modelling quality;
- Conceptual models in management science;
- Design patterns and object-oriented design;
- Evolution and change in conceptual models;
- Implementations of information systems;
- Information and schema integration;
- Information customisation and user profiles;
- Information recognition and information modelling;
- Information retrieval, analysis, visualisation and prediction;
- Information systems design methodologies;
- Knowledge discovery, knowledge representation and knowledge management;
- Methods for developing, validating and communicating conceptual models;
- Philosophical, mathematical and linguistic foundations of conceptual models;
- Reuse, reverse engineering and reengineering;
- Semantic Web; and
- Software engineering and tools for information systems development.

The program committee has selected the contributed papers from 29 submissions. All submitted papers have been refereed by at least two international experts, and have been discussed thoroughly. The eight papers judged best by the program committee members have been accepted and are included in this volume.

The program committee invited Dr. Guido Governatori, to present the APCCM 2010 keynote on *A conceptually rich model of business process compliance*. Dr. Governatori is the Associate Education Director of the NICTA Queensland Research Laboratory, Australia. The committee also invited Professor Sven Hartmann from the Clausthal University of Technology, Clausthal-Zellerfeld, Germany to give an invited talk. Professor Hartmann's talk was entitled *Ontology consolidation in bioinformatics*.

The program committee selected the article *Hetero-Homogeneous Hierarchies in Data Warehouses* by Bernd Neumayr, Michael Schreff, and Bernhard Thalheim for the APCCM 2010 Best Paper Award. This includes a cash prize in the amount of NZ\$500, sponsored by the School of Information Management, The Victoria University of Wellington, New Zealand. The same article was selected for the APCCM 2010 Best Student Paper Award. This award includes a cash prize of AU\$500, sponsored by CORE, Australia. Warmest congratulations to Bernd Neumayr and his co-authors Michael Schreff and Bernhard Thalheim.

We wish to thank all authors who submitted papers and all the conference participants for the fruitful discussions. We are grateful to the members of the program committee and the additional reviewers for their timely expertise in carefully reviewing the papers. We like to acknowledge the excellent work of the APCCM 2010 Publicity Chair Dr Markus Kirchberg who also maintained the conference web site and, in particular, the MuCoMS conference management system (<http://www.mucoms.org/>). Finally, we wish to express our appreciation to the local organisers at the Queensland University of Technology for the wonderful days in Brisbane.

Sebastian Link

Victoria University of Wellington, New Zealand

Aditya Ghose

University of Wollongong, Australia

APCCM 2009 Programme Chairs

January 2009

Conference Organisation

Program Committee Chairs

Aditya K. Ghose, University of Wollongong, Australia
Sebastian Link, Victoria University of Wellington, New Zealand

Publicity Chair

Markus Kirchberg, Institute for Infocomm Research, A*STAR, Singapore

Program Committee Members

Boualem Benatallah, University of New South Wales, Australia
Denise de Vries, Flinders University, Australia
Gillian Dobbie, University of Auckland, New Zealand
Angela Eck Soong Goh, Nanyang Technological University, Singapore
Tiong-Thye Goh, Victoria University of Wellington, New Zealand
Sven Hartmann, Clausthal University of Technology, Germany
Brian Henderson-Sellers, University of Technology, Sydney, Australia
Annika Hinze, University of Waikato, New Zealand
Markus Kirchberg, Institute for Infocomm Research, A*STAR, Singapore
Yasushi Kiyoki, Keio University, Japan
Aneesh Krishna, Curtin University of Technology, Australia
Chiang Lee, National Cheng-Kung University, Taiwan
Jixue Liu, University of South Australia, Australia
Pavle Mogin, Victoria University of Wellington, New Zealand
Shamkant B. Navathe, Georgia Institute Of Technology, USA
Martin Nečaský, Charles University, Czech Republic
Sudha Ram, University of Arizona, USA
Michael Rosemann, Queensland University of Technology, Australia
Motoshi Saeki, Tokyo Institute of Technology, Japan
Klaus-Dieter Schewe, Information Science Research Centre, New Zealand
Il-Yeol Song, Drexel University, USA
Nigel Stanger, University of Otago, New Zealand
Markus Stumptner, University of South Australia, Australia
Ernest Teniente, Universitat Politècnica de Catalunya, Spain
Bernhard Thalheim, Christian-Albrechts-University Kiel, Germany

Additional Reviewers

Hui Ma, Victoria University of Wellington, New Zealand
Andrea Schweer, The University of Waikato, New Zealand
Rajesh Thiagarajan, University of South Australia, Australia
Jing Wang, Massey University, New Zealand

Organising Committee

Co-Chairs

Dr. Wayne Kelly
Prof. Mark Looi

Budget and Facilities

Mr. Malcolm Corney

Catering and Booklet

Dr. Diane Corney

Sponsorship and Web

Dr. Tony Sahama

Senior Advisors

Prof. Colin Fidge
Prof. Kerry Raymond

Finance and Travel

Ms. Therese Currell
Ms. Carol Richter

Registration

Mr. Matt Williams

DVD and Signage

Mr. Matthew Brecknell

Satchels and T-shirts

Ms. Donna Teague

Welcome from the Organising Committee

On behalf of the Australasian Computer Science Week 2010 (ACSW2010) Organising Committee, we welcome you to this year's event hosted by the Queensland University of Technology (QUT). Striving to be a "University for the Real World" our research and teaching has an applied emphasis. QUT is one of the largest producers of IT graduates in Australia with strong linkages with industry. Our courses and research span an extremely wide range of information technology, everything from traditional computer science, software engineering and information systems, to games and interactive entertainment.

We welcome delegates from over 21 countries, including Australia, New Zealand, USA, Finland, Italy, Japan, China, Brazil, Canada, Germany, Pakistan, Sweden, Austria, Bangladesh, Ireland, Norway, South Africa, Taiwan and Thailand. We trust you will enjoy both the experience of the ACSW 2010 event and also get to explore some of our beautiful city of Brisbane. At Brisbane's heart, beautifully restored sandstone buildings provide a delightful backdrop to the city's glass towers. The inner city clusters around the loops of the Brisbane River, connected to leafy, open-skied suburban communities by riverside bikeways. QUT's Garden's Point campus, the venue for ACSW 2010, is on the fringe of the city's botanical gardens and connected by the Goodwill Bridge to the Southbank tourist precinct.

ACSW2009 consists of the following conferences:

- Australasian Computer Science Conference (ACSC) (Chaired by Bernard Mans and Mark Reynolds)
- Australasian Computing Education Conference (ACE) (Chaired by Tony Clear and John Hamer)
- Australasian Database Conference (ADC) (ADC) (Chaired by Heng Tao Shen and Athman Bouguet-taya)
- Australasian Information Security Conference (AISC) (Chaired by Colin Boyd and Willy Susilo)
- Australasian User Interface Conference (AUIC) (Chaired by Christof Lutteroth and Paul Calder)
- Australasian Symposium on Parallel and Distributed Computing (AusPDC) (Chaired by Jinjun Chen and Rajiv Ranjan)
- Australasian Workshop on Health Informatics and Knowledge Management (HIKM) (Chaired by Anthony Maeder and David Hansen)
- Computing: The Australasian Theory Symposium (CATS) (Chaired by Taso Viglas and Alex Potanin)
- Asia-Pacific Conference on Conceptual Modelling (APCCM) (Chaired by Sebastian Link and Aditya Ghose)
- Australasian Computing Doctoral Consortium (ACDC) (Chaired by David Pearce and Rachel Cardell-Oliver).

The nature of ACSW requires the co-operation of numerous people. We would like to thank all those who have worked to ensure the success of ACSW2010 including the Organising Committee, the Conference Chairs and Programme Committees, our sponsors, the keynote speakers and the delegates. Special thanks to Justin Zobel from CORE and Alex Potanin (co-chair of ACSW2009) for his extensive advice and assistance. If ACSW2010 is run even half as well as ACSW2009 in Wellington then we will have done well.

Dr Wayne Kelly and Professor Mark Looi

Queensland University of Technology

ACSW2010 Co-Chairs

January, 2010

CORE - Computing Research & Education

CORE welcomes all delegates to ACSW2010 in Brisbane. CORE, the peak body representing academic computer science in Australia and New Zealand, is responsible for the annual ACSW series of meetings, which are a unique opportunity for our community to network and to discuss research and topics of mutual interest. The original component conferences ACSC, ADC, and CATS, which formed the basis of ACSWin the mid 1990s now share the week with seven other events, which build on the diversity of the Australasian computing community.

In 2010, we have again chosen to feature a small number of plenary speakers from across the discipline: Andy Cockburn, Alon Halevy, and Stephen Kisely. I thank them for their contributions to ACSW2010. I also thank the keynote speakers invited to some of the individual conferences. The efforts of the conference chairs and their program committees have led to strong programs in all the conferences again, thanks. And thanks are particularly due to Wayne Kelly and his colleagues for organising what promises to be a strong event.

In Australia, 2009 saw, for the first time in some years, an increase in the number of students choosing to study IT, and a welcome if small number of new academic appointments. Also welcome is the news that university and research funding is set to rise from 2011-12. However, it continues to be the case that per-place funding for computer science students has fallen relative to that of other physical and mathematical sciences, and, while bodies such as the Australian Council of Deans of ICT seek ways to increase student interest in the area, more is needed to ensure the growth of our discipline.

During 2009, CORE continued to work on journal and conference rankings. A key aim is now to maintain the rankings, which are widely used overseas as well as in Australia. Management of the rankings is a challenging process that needs to balance competing special interests as well as addressing the interests of the community as a whole. ACSW2010 includes a forum on rankings to discuss this process. Also in 2009 CORE proposed a standard for the undergraduate Computer Science curriculum, with the intention that it be used for accreditation of degrees in computer science.

CORE's existence is due to the support of the member departments in Australia and New Zealand, and I thank them for their ongoing contributions, in commitment and in financial support. Finally, I am grateful to all those who gave their time to CORE in 2009; in particular, I thank Gill Dobbie, Jenny Edwards, Alan Fekete, Tom Gedeon, Leon Sterling, and the members of the executive and of the curriculum and ranking committees.

Justin Zobel

President, CORE
January, 2010

ACSW Conferences and the Australian Computer Science Communications

The Australasian Computer Science Week of conferences has been running in some form continuously since 1978. This makes it one of the longest running conferences in computer science. The proceedings of the week have been published as the *Australian Computer Science Communications* since 1979 (with the 1978 proceedings often referred to as *Volume 0*). Thus the sequence number of the Australasian Computer Science Conference is always one greater than the volume of the Communications. Below is a list of the conferences, their locations and hosts.

2011. Volume 33. Host and Venue - Curtin University of Technology, Perth, WA.

2010. Volume 32. Host and Venue - Queensland University of Technology, Brisbane, QLD.

2009. Volume 31. Host and Venue - Victoria University, Wellington, New Zealand.

2008. Volume 30. Host and Venue - University of Wollongong, NSW.

2007. Volume 29. Host and Venue - University of Ballarat, VIC. First running of HDKM.

2006. Volume 28. Host and Venue - University of Tasmania, TAS.

2005. Volume 27. Host - University of Newcastle, NSW. APBC held separately from 2005.

2004. Volume 26. Host and Venue - University of Otago, Dunedin, New Zealand. First running of APCCM.

2003. Volume 25. Hosts - Flinders University, University of Adelaide and University of South Australia. Venue - Adelaide Convention Centre, Adelaide, SA. First running of APBC. Incorporation of ACE. ACSAC held separately from 2003.

2002. Volume 24. Host and Venue - Monash University, Melbourne, VIC.

2001. Volume 23. Hosts - Bond University and Griffith University (Gold Coast). Venue - Gold Coast, QLD.

2000. Volume 22. Hosts - Australian National University and University of Canberra. Venue - ANU, Canberra, ACT. First running of AUC.

1999. Volume 21. Host and Venue - University of Auckland, New Zealand.

1998. Volume 20. Hosts - University of Western Australia, Murdoch University, Edith Cowan University and Curtin University. Venue - Perth, WA.

1997. Volume 19. Hosts - Macquarie University and University of Technology, Sydney. Venue - Sydney, NSW. ADC held with DASFAA (rather than ACSW) in 1997.

1996. Volume 18. Host - University of Melbourne and RMIT University. Venue - Melbourne, Australia. CATS joins ACSW.

1995. Volume 17. Hosts - Flinders University, University of Adelaide and University of South Australia. Venue - Glenelg, SA.

1994. Volume 16. Host and Venue - University of Canterbury, Christchurch, New Zealand. CATS run for the first time separately in Sydney.

1993. Volume 15. Hosts - Griffith University and Queensland University of Technology. Venue - Nathan, QLD.

1992. Volume 14. Host and Venue - University of Tasmania, TAS. (ADC held separately at La Trobe University).

1991. Volume 13. Host and Venue - University of New South Wales, NSW.

1990. Volume 12. Host and Venue - Monash University, Melbourne, VIC. Joined by Database and Information Systems Conference which in 1992 became ADC (which stayed with ACSW) and ACIS (which now operates independently).

1989. Volume 11. Host and Venue - University of Wollongong, NSW.

1988. Volume 10. Host and Venue - University of Queensland, QLD.

1987. Volume 9. Host and Venue - Deakin University, VIC.

1986. Volume 8. Host and Venue - Australian National University, Canberra, ACT.

1985. Volume 7. Hosts - University of Melbourne and Monash University. Venue - Melbourne, VIC.

1984. Volume 6. Host and Venue - University of Adelaide, SA.

1983. Volume 5. Host and Venue - University of Sydney, NSW.

1982. Volume 4. Host and Venue - University of Western Australia, WA.

1981. Volume 3. Host and Venue - University of Queensland, QLD.

1980. Volume 2. Host and Venue - Australian National University, Canberra, ACT.

1979. Volume 1. Host and Venue - University of Tasmania, TAS.

1978. Volume 0. Host and Venue - University of New South Wales, NSW.

Conference Acronyms

ACDC	Australasian Computing Doctoral Consortium
ACE	Australasian Computer Education Conference
ACSC	Australasian Computer Science Conference
ACSW	Australasian Computer Science Week
ADC	Australasian Database Conference
AISC	Australasian Information Security Conference
AUIC	Australasian User Interface Conference
APCCM	Asia-Pacific Conference on Conceptual Modelling
AusPDC	Australasian Symposium on Parallel and Distributed Computing (replaces AusGrid)
CATS	Computing: Australasian Theory Symposium
HIKM	Australasian Workshop on Health Informatics and Knowledge Management

Note that various name changes have occurred, which have been indicated in the Conference Acronyms sections in respective CRPIT volumes.

ACSW and APCCM 2010 Sponsors

We wish to thank the following sponsors for their contribution towards this conference.



CORE - Computing Research and Education,
www.core.edu.au



CEED,
www.corptech.com.au



Queensland University of Technology,
www.qut.edu.au



CSIRO ICT Centre,
www.csiro.au/org/ict.html



AUSTRALIAN
COMPUTER
SOCIETY

Australian Computer Society,
www.acs.org.au



SAP Research,
www.sap.com/about/company/research



School of Information Management,
The Victoria University of Wellington
www.sim.vuw.ac.nz

KEYNOTE

A Conceptually Rich Model of Business Process Compliance

Guido Governatori *

Antonino Rotolo **

* NICTA, Australia PO Box 6020, St Lucia, Queensland 4067, Australia

** CIRSIFID and Law Faculty, University of Bologna, Via Galliera 3, 40122, Bologna, Italy

Email: guido.governatori@nicta.com.au, anotnino.rotolo@unibo.it

Abstract

In this paper we extend the preliminary work developed elsewhere and investigate how to characterise many aspects of the compliance problem in business process modeling. We first define a formal and conceptually rich language able to represent, and reason about, chains of reparational obligations of various types. Second, we devise a mechanism for normalising a system of legal norms. Third, we specify a suitable language for business process modeling able to automate and optimise business procedures and to embed normative constraints. Fourth, we develop an algorithm for compliance checking and discuss some computational issues regarding the possibility of checking compliance runtime or of enforcing it at design time.

Keywords: Business Process, Regulatory Compliance, Obligations

1 Introduction

1.1 Background

Business processes specify the activities a business does to achieve its business objectives. Businesses are typically regulated. The requirements businesses have to comply with may stem from legislation and regulatory bodies, standards and codes of practice, and also business partner contracts. *Business Process Compliance* is the relationship between two sets of specifications: the specification for the processes/procedures adopted by a business to achieve its goal, and the specifications corresponding to the regulations relevant for a business. Essentially, compliance is aimed at ensuring that business processes, operations and practise are in accordance with a prescribed and/or agreed set of norms. Compliance requirements may stem from legislation and regulatory bodies (e.g., Sarbanes-Oxley, Basel II, HIPAA), standards and codes of practice (e.g., SCOR, ISO9000) and also business partner contracts. To align the two sets of specification a fundamental requirements is to be able to represent formally the two sets of specifications. Accordingly, any attempt to provide a formal framework to investigate whether processes comply with relevant regulation must provide a conceptually sound formalisation of the norms and the obligations process are subject to.

In this paper we extend the preliminary work developed in (Governatori & Rotolo 2008a) and further investigate how to model compliance in business processes. In (Governatori & Rotolo 2008a) we proposed (1) a method to align the language specifying the activities of a business

process and the conditions set up by the norms relevant for the process and (2) an efficient algorithm to determine whether a process is compliant. The method was based on (semantic) annotations, where the annotations are written in the formal language chosen to represent the normative specifications. The idea was that business processes are annotated and the annotations provide the conditions a process has to comply with. Annotations can be at different levels; for example we can annotate a full process or a single task in a process. In addition we can have different types of annotation. Annotations can range from the full set of rules (norms) specific to a process or a single task to simple semantic annotation corresponding to one effect of a particular task (e.g., after the successful execution of task *A* in a process *B* the value of the environment variable *C* is *D*).

Checking compliance amounts to a relatively affordable operation when we have to see whether processes are compliant with respect to simple normative systems. But things are tremendously harder when we deal with processes to be tested against complex, large and articulated systems of norms such as bodies of legal provisions.

In (Governatori & Rotolo 2008a) we suggested a first solution to overcome the difficulties arising when the violation of a legal obligation activates other obligations able to compensate for this violation. In particular, we proposed to use the logic originally developed in (Governatori & Rotolo 2006). This paper adopts the same methodology, but it generalises (Governatori & Rotolo 2008a)'s approach to reparational obligations and it addresses other two significant sources of complexities. As we will see, the proposed framework will be helpful to address and analyse some still unsolved research issues in business process modeling. The following subsection summarises the complexities which we will discuss in this paper.

1.2 Motivation

A *first source of complexities* was already (partially) addressed in (Governatori & Rotolo 2008a). It resides in the fact that legal norms regulate processes by usually specifying actions to be taken in case of breaches of some of the norms, actions which can vary from (pecuniary) penalties to the termination of an interaction itself. These constructions, i.e., obligations in force after some other obligations have been violated, are known in the deontic literature as *contrary-to-duty obligations* (CTDs) or *reparational obligations* (because they are meant to 'repair' or 'compensate' violations of primary obligations (Carmo & Jones 2002)). Thus a CTD is a conditional obligation arising in response to a violation, where a violation is signalled by an unfulfilled obligation. These constructions identify situations that are not ideal for the interaction but still acceptable. The ability to deal with violations and the reparational obligations generated from them is an essential requirement for processes where, due to the nature of the environment where they are deployed, some failures can occur, but it does not necessarily mean that the

whole interaction has to fail. However, the main problem with these constructions is that they can give rise to very complex rule dependencies, because we can have that the violation of a single rule can activate other (reparational) rules, which in turn, in case of their violation, refer to other rules, and so forth.

A *second source of complexities* depends on the fact that processes may be regulated by different types of obligations (see Section 2.1). In (Governatori et al. 2007) we proposed a classification of obligations, according to which different obligations may require distinct compliance conditions. We may have obligations requiring (1) to be always fulfilled during the execution of the entire process or of some subpaths of it, (2) that a certain condition must occur at least once before the execution of a certain task *A* of the process and such that the obligations may, or may not, persist after *A* if they are not complied with, (3) that something be done in a single task. Clearly, the peculiarities of these types of obligation make things more complex when we deal with the compliance of a process with respect to chains of reparational obligations. For example, if the primary obligation is persistent and states to pay before task *A*, and the secondary (reparational) obligation is to pay a fine in the task *B* successive to *A*, the process is compliant not only when we pay before *A*, but also when we do not meet this deadline, pay later and pay the fine at *B*. If the secondary obligation rather requires to be always fulfilled during the execution of all tasks successive to *A*, compliance conditions will change. In addition, other types of obligation can be considered: for instance, we may have provisions stating that some *A* is obligatory and which are fulfilled even if *A* was obtained before the provision was in force, whereas other provisions state that *A* is obligatory but they are complied with only when *A* holds after they are in force.

A *third source of complexities* depends on whether our aim is to check compliance at *runtime* or to model and enforce it at *design time*. Indeed, ensuring automated detection of compliance of business processes with normative documents is a complex problem, involving a number of alternatives.

Currently there are two main approaches towards achieving runtime compliance¹: *auditing* and *monitoring*. Both methods detect compliance by means of *retrospective reporting*. *Auditing* is conducted for “after-the-fact” detection. Much of the existing software solutions for compliance follow 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 specialise in certain class of checks, for example the widely supported checks that relate to Segregation of Duty violations in role management and user provisioning systems. This approach resides in the space of “after-the-fact” detection, as compliance is checked at the end of the process execution. *Monitoring* is based on the same principle, but is performed by checking compliance step by step (after each task is executed, for example).

The main limit of monitoring and auditing is that they do not offer any guarantee to correctly fix the processes when they are not compliant. Such methods are basically able to answer the question of whether a given process fulfilled all regulations applying to it. If the process is not compliant we would rather need a method which provides a means of visualizing the impact of compliance controls on process models, assist in compliance checking and analysis and also feedback for subsequent (re)design of the process models.

Hence, a sustainable approach for achieving compliance should fundamentally have a *preventative* focus. As we will see, we describe an approach that provides the

capability to capture compliance requirements through a generic requirements modelling framework, and subsequently facilitate the propagation of these requirements into business process models and enterprise applications, thus achieving *compliance by design*.²

However, we will see that, when all types of obligations mentioned above are used to regulate business processes, the enforcement of compliance at design time can rise serious computational problems.

1.3 Layout of the Paper

Ensuring automated detection and/or enforcement of compliance requires to address the following related research tasks:

- (1) Define a formal language able to represent, and reason about, chains of reparational obligations of the types recalled above; we address this in Section 2.2;
- (2) Devise a mechanism for normalising a system of legal norms, namely, identify formal loopholes, deadlocks and inconsistencies in it, and to make hidden conditions explicit; without this task, we do not have any guarantee that a given process is compliant, because we do not know if all relevant norms have been considered; this task will be addressed in Section 2.3;
- (3) Specify a suitable language for business process modeling (in our case based on annotations) able to automate and optimise business procedures and to embed normative constraints; this task will be addressed in Section 3;
- (4) Develop a (possibly efficient) algorithm for compliance checking; the procedure will also require to develop a reasoning mechanism able to establish what chains of reparational obligations are active for each step in the process; this task will be addressed in Section 4.

A further section (Section 5) will be devoted to discussing the problem of computing runtime or design time compliance when different types of obligations regulate a given process. A summary and related work will conclude the paper.

2 Normative Constraints

2.1 Violations and Types of Obligations

Achievement, Maintenance and Punctual Obligations
We can distinguish *achievement obligations* from *maintenance obligations* (Governatori et al. 2007).

For an *achievement obligation*, a certain condition must occur at least once before the deadline:

Example 1 *Customers must pay before the delivery of the good, after receiving the invoice*

In this example, the deadline (before the delivery of the good) refers to an obligation triggered by receipt of the invoice: such an obligation is persistent. After that the customer is obliged to pay. The obligation terminates only when it is complied with. Note that, in this example, the obligation itself obviously persists after the deadline, until it is achieved. But we may have cases where achievement obligations do not persist after the deadline:

Example 2 *Once the submissions to APCCM are made available to APCCM PC members, the reviewers must send their reports before the notifications are delivered to the authors*

Indeed, the obligation to deliver a review does not persist after the deadline, since after the review result has been notified to the authors, the paper has been accepted or rejected on the basis of the other reports delivered in time.

¹For a comprehensive exposition of compliance for business process models, see (Governatori & Sadiq 2009, Sadiq & Governatori 2009).

²See also (Lu et al. 2007) for the compliance by design methodology.

In general, a deadline signals that a violation of the obligation has occurred. This may trigger an explicit sanction (see below).

For *maintenance obligations*, a certain condition must obtain during all instants before the deadline:

Example 3 *After opening a bank account, customers must keep a positive balance until bank charges are taken out.*

By definition, maintenance obligations do not persist after the deadline. In Example 3, the deadline only signals that the obligation is terminated. A violation occurs when the obliged state does not obtain at some point before the deadline.

Finally, we may have *punctual obligations*, which only apply to single tasks or instants:

Example 4 *When banks proceed with any wire transfer, they must transmit a message, via SWIFT, to the receiving bank requesting that the payment is made according to the instructions given.*

Many norms can be associated with an explicit sanction. Consider, for instance, the obligations in Examples 1, 2, and 3:

Example 5 *Customers must pay before the delivery of the good, after receiving the invoice. Otherwise, an additional fine must be paid.*

Example 6 *After opening a bank account, customers must keep a positive balance until bank charges are taken out. Otherwise, their account is blocked.*

Example 7 *Once the submissions to APCCM are made available to APCCM PC members, the reviewers must send their reports before the notifications are delivered to the authors. Otherwise, they will be blacklisted for inclusions in future APCCM PCs.*

An explicit sanction is often implemented through a separate obligation, which is triggered by a detected violation. In this setting, legislators may need further deadlines to enforce the sanctions, leading to a chain of obligations. For instance, the payment of a fine mentioned in Example 5 could be due before the execution of a subsequent task.

Preemptive or Non-preemptive Obligations We can also distinguish *preemptive obligations* from *non-preemptive obligations*.

Consider again the obligation in Example 1 and suppose that the price to be paid by a customer is 200\$. Suppose now that this customer, by mistake, had transferred an amount of 200\$ to the bank account of the seller before the delivery of the invoice. In this case, the early transfer may count as a payment and the customer could claim that her obligation to pay the seller is already fulfilled. This is an example of *preemptive* obligation.

Non-preemptive obligations do not work as above. Consider this example:

Example 8 *Executors and administrators of a decedent's estate will be required to give notice to each beneficiary named in the Will within 60 days after the date X of an order admitting a will to probate has been signed.*

If an executor gives a notice to the beneficiaries before X, she will not comply with the above obligation and will have to resend the notification after that. Note that, in general, the distinction between preemptive and non-preemptive obligations applies only to achievement obligations, while it does not make sense with the maintenance and punctual ones.

Violations The expression of violation conditions and the reparation obligations is an important requirement for designing subsequent processes to minimise or deal with such violations and also to determine the compliance

of a process with the relevant norms. The violation expression consists of the primary obligation, its violation conditions, an obligation generated upon the violation condition occurs, and this can recursively be iterated, until the final condition is reached. We introduce the non-boolean connective \otimes , whose interpretation is such that $OA \otimes OB$ is read as “*OB* is the reparation of the violation of *OA*”. In other words, the interpretation of $OA \otimes OB$, is that *A* is obligatory, but if the obligation *OA* is not fulfilled (i.e., when $\neg A$ is the case), then the obligation *OB* is activated and becomes in force until it is satisfied or violated. In the latter case a new obligation may be activated, followed by others in chain, as appropriate.

However, the violation condition of an obligation varies depending on whether it is an achievement or a maintenance obligation, or a preemptive or a non-preemptive one. In the next section, we will extend the approach of (Governatori & Rotolo 2008a, 2006) to cover these cases.

2.2 Process Compliance Language (PCL)

A conceptually sound formalisation of norms (for assessing the compliance of a process) should take into account all the aspects mentioned in Section 2.1. Thus, we now provide here a formal account of the ideas presented above. Our formalism, called Process Compliance Language (PCL), is a combination of an efficient non-monotonic formalism (Defeasible Logic (Antoniou et al. 2001)) and a deontic logic of violations (Governatori & Rotolo 2006). The current version of PCL significantly extends the logic of (Governatori & Rotolo 2008a) by working on all types of obligations discussed in Section 2.1. Hence, this particular combination allows us to represent exceptions as well as the ability to capture violations and any types of obligations resulting from the violations; in addition our framework has good computational properties: the extension of a theory (i.e., the set of conclusions/normative positions following from a set of facts) can be computed in time linear to the size of the theory.

The ability to handle violation is very important for compliance of agents' processes. Often businesses operate in dynamic and somehow unpredictable environments. As a consequence in some cases, maybe due to external circumstances, it is not possible to operate in the way specified by the norms, but the norms prescribe how to recover from the resulting violations. In other cases, the prescribed behaviours are subject to exceptions. Finally, in other cases, one might not have a complete description of the environment. Accordingly the process has to operate based on the available input (this is typically the case of the *due diligence* prescription), but if more information were available, then the task to be performed could be a different one.

PCL is sound in this respect given the combinations of the deontic component (able to represent the fundamental normative positions and chains of violations/reparations) and the defeasible component that takes care of the issue about partial information and possibly conflicting prescriptions.

PCL formal language consists of the following set of atomic symbols: a numerable set of propositional letters p, q, r, \dots , intended to represent the state variables and the tasks of a process. Formulas of the logic are constructed using the negation \neg , the non-boolean connective \otimes (for the Contrary-To-Duty (CTD) operator), and the deontic operators O_y^x (for obligation), where y can be empty. The deontic operators have subscripts and superscripts to specify whether they denote achievement, maintenance or punctual obligations, or whether they are or not preemptive. Table 1 summarises all possible combinations. The formulas of PCL will be constructed in two steps according to the following formation rules:

- every propositional letter is a literal;

Obligation operators	Intuitive reading
$O_{pr}^{a,\pi}$	achievement, persistent, preemptive
$O_{n-pr}^{a,\pi}$	achievement, persistent, non-preemptive
$O_{pr}^{a,\tau}$	achievement, non-persistent, preemptive
$O_{n-pr}^{a,\tau}$	achievement, non-persistent, non-preemptive
O^m	maintenance
O^p	punctual

Table 1: Types of obligations

- the negation of a literal is a literal;
- if X is a deontic operator and l is a literal then Xl and $\neg Xl$ are deontic literals.

After we have defined the notions of literal and deontic literal we can use the following set of formation rules to introduce \otimes -expressions, i.e., the formulas used to encode chains of obligations and violations.

- every deontic literal is an \otimes -expression;
- if Xl_1, \dots, Xl_n are deontic literals, then $Xl_1 \otimes \dots \otimes Xl_n$ is an \otimes -expression.

The connective \otimes permits combining primary and CTD obligations into unique regulations. The meaning of an expression like $O_{pr}^{a,\pi}A \otimes O^pB \otimes O^mC$ is that the primary provision is an achievement, persistent, preemptive obligation to do A , but if A is not done, then we have a punctual obligation to do B . If B fails to be realised, then we obtain a maintenance obligation to do C . Thus B is the reparation of the violation of the obligation $O_{pr}^{a,\pi}A$. Similarly C is the reparation of the obligation O^pB , which is in force when the violation of A occurs.

Each norm is represented by a rule in PCL, where a rule is an expression $r : A_1, \dots, A_n \Rightarrow C$, where r is the name/id of the norm, A_1, \dots, A_n , the *antecedent* of the rule, is the set of the premises of the rule (alternatively it can be understood as the conjunction of all the literals in it) and C is the conclusion of the rule. Each A_i is either a literal or a deontic literal and C is an \otimes -expression.

The meaning of a rule is that the normative position (obligation, permission, prohibition) represented by the conclusion of the rule is in force when all the premises of the rule hold. PCL is also equipped with another type of rules, called defeaters (marked with arrow \rightsquigarrow) and a superiority relation (a binary relation) over the rule set.

In Defeasible Logic, the superiority relation (\prec) determines the relative strength of two rules, and it is used when rules have potentially conflicting conclusions. For example, given the rules $r_1 : a \Rightarrow O^m b \otimes O_{n-pr}^{a,\pi} c$ and $r_2 : d \Rightarrow \neg O_{pr}^{a,\pi} c$, $r_1 \prec r_2$ means that rule r_1 prevails over rule r_2 in situations where both fire and they are in conflict.

Defeaters play in Defeasible Logic a peculiar role, as they cannot lead to any conclusion but are used to defeat some rules by producing evidence to the contrary. In this sense, defeaters are suitable to model the termination of the persistence of obligations (Governatori et al. 2005, Governatori & Rotolo 2008b). Consider Example 5 (we assume that the reparational obligation is a punctual one):

$$\begin{aligned} \text{inv}_{init} \quad \text{invoice} &\Rightarrow O_{pr}^{a,\pi} \text{pay} \otimes O^p \text{pay.fine} \\ \text{inv}_{term} \quad \text{pay} &\rightsquigarrow \neg O_{pr}^{a,\pi} \text{pay} \end{aligned}$$

Here, compliance is the only condition that terminates the obligation to pay: the obligation in fact persists beyond the deadline.

Example 6 is modeled as follows:

$$\begin{aligned} \text{pos}_{init} \quad \text{open_account} &\Rightarrow O^m \text{positive} \otimes O^p \text{blocked} \\ \text{pos}_{term} \quad \text{bank_charges} &\rightsquigarrow \neg O^m \text{positive} \end{aligned}$$

On account of the nature of maintenance obligations, the termination of the primary obligation occurs only when bank charges are taken out.

Finally, the termination of the primary obligation in Example 7 is captured as follows:

$$\begin{aligned} \text{rev}_{init} \quad \text{papers_available} &\Rightarrow O_{n-pr}^{a,\tau} \text{review} \otimes O_{pr}^{a,\pi} \text{blacklist} \\ \text{rev}_{term_1} \quad \text{notification} &\rightsquigarrow \neg O_{n-pr}^{a,\tau} \text{review} \\ \text{rev}_{term_2} \quad \text{review} &\rightsquigarrow \neg O_{n-pr}^{a,\tau} \text{review} \end{aligned}$$

Note that we have two termination rules: one stating that the obligation to send the review no longer holds when the reports are notified to the authors, another establishing that such an obligation is terminated if it is complied with.

2.3 Normal Forms

We introduce transformations of an PCL representation of a normative system to produce a normal form of the same (NPCL). A normal form is a representation of a normative system based on an PCL specification containing all conditions that can be generated/derived from the given PCL specification. The purpose of a normal form is to “clean up” the PCL representation of a normative system, that is to identify formal loopholes, deadlocks and inconsistencies in it, and to make hidden conditions explicit.

In the rest of this section we introduce the procedures to generate normal forms. First (Section 2.3.1) we describe a mechanism, based on (Governatori & Rotolo 2006), to derive new conditions by merging together existing normative clauses. In particular we link an obligation and the obligations triggered in response to violations of the obligation. Then, in Section 2.3.2, we examine the problem of redundancies, and we give a condition to identify and remove redundancies from the formal normative specification. Section 2.3.3 discusses how to solve possible conflicts between deontic provisions.

2.3.1 Merging Norms

One of the features of the logic of violations is to take two rules, or norms, and merge them into a new clause.

Consider a norm like (Γ and Δ are sets of premises)

$$\Gamma \Rightarrow O^m A.$$

Given an obligation like this, if we have that the violation of $O^m A$ is part of the premises of another norm, for example,

$$\Delta, \neg A \Rightarrow O^p C,$$

then the latter must be a good candidate as reparational obligation of the former. This idea is formalised as follows:

$$\frac{\Gamma \Rightarrow O^m A \quad \Delta, \neg A \Rightarrow O^p C}{\Gamma, \Delta \Rightarrow O^m A \otimes O^p C}$$

This reads as follows: given two policies such that one is a conditional obligation ($\Gamma \Rightarrow O^m A$) and the antecedent of second contains the negation of the propositional content of the consequent of the first ($\Delta, \neg A \Rightarrow O^p C$), then the latter is a reparational obligation of the former. Their reciprocal interplay makes them two related norms so that they cannot be viewed anymore as independent obligations. Therefore we can combine them to obtain an expression (i.e., $\Gamma, \Delta \Rightarrow O^m A \otimes O^p C$) that exhibits the *explicit reparational obligation* of the second norm with respect to the first. Notice that the subject of the primary obligation and the subject of its reparation can be different, even if very often they are the same.

Let X, Y, Z be deontic operators. The following is the general rule for merging norms based on (Governatori & Rotolo 2006, Governatori 2005):

$$\frac{\Gamma \Rightarrow Xa \otimes (\otimes_{i=1}^n Yb_i) \otimes Zc \quad \Delta, \neg b_1, \dots, \neg b_n \Rightarrow Zd}{\Gamma, \Delta \Rightarrow Xa \otimes (\otimes_{i=1}^n Yb_i) \otimes Zd} \quad (1)$$

2.3.2 Removing Redundancies

Given the structure of the inference mechanism it is possible to combine rules in slightly different ways, and in some cases the meaning of the rules resulting from such operations is already covered by other rules. In other cases the rules resulting from the merging operation are generalisations of the rules used to produce them, consequently, the original rules are no longer needed in the specifications. To deal with this issue we introduce the notion of subsumption between rules. Intuitively a rule subsumes a second rule when the behaviour of the second rule is implied by the first rule. We first introduce the idea with the help of an example and then we show how to give a formal definition of the notion of subsumption appropriate for PCL.

Let us consider the rules

$$r : Invoice \Rightarrow O_{pr}^{a,\pi} PayWithin7Days \otimes O^p PayWithInterest$$

$$r' : Invoice, \neg PayWithin7Days \Rightarrow O_{n-pr}^{a,\pi} PayWithInterest.$$

The first rule says that after the seller sends the invoice the buyer has the achievement, persistent and preemptive obligation to pay within one week, otherwise immediately after the violation the buyer has to pay the principal plus the interest. Thus we have the primary obligation $O_{pr}^{a,\pi} PayWithin7Days$, whose violation is repaired by the secondary obligation $O^p PayWithInterest$. According to the second rule, given the same set of circumstances $Invoice$ and $\neg PayWithin7Days$ we have the achievement, persistent and non-preemptive obligation $O_{n-pr}^{a,\pi} PayWithInterest$. However,

- the primary obligation of r' obtains when we have a violation of the primary obligation of r ;
- after the obligation $O_{pr}^{a,\pi} PayWithin7Days$ is violated, complying with the secondary obligation $O^p PayWithInterest$ of r entails complying with the primary obligation $O_{n-pr}^{a,\pi} PayWithInterest$ of r' (but not vice versa);
- hence, r is more general than r' , and so the latter can be discarded from the formal representation of the specifications.

The intuitions we have just exemplified is captured by the following definitions.

Definition 1 Let X, Y be two deontic operators in $\{O_{pr}^{a,\pi}, O_{n-pr}^{a,\pi}, O_{pr}^{a,\tau}, O_{n-pr}^{a,\tau}, O^m, O^p\}$. Then, $Y \sqsubseteq X$ iff

- if $Y = O_{pr}^{a,\pi}$, then $X \in \{O_{pr}^{a,\pi}, O_{n-pr}^{a,\pi}, O_{pr}^{a,\tau}, O_{n-pr}^{a,\tau}, O^m, O^p\}$;
- if $Y = O_{n-pr}^{a,\pi}$, then $X \in \{O_{n-pr}^{a,\pi}, O_{n-pr}^{a,\tau}, O^m, O^p\}$;
- if $Y = O_{pr}^{a,\tau}$, then $X \in \{O_{pr}^{a,\pi}, O_{n-pr}^{a,\pi}, O_{pr}^{a,\tau}, O_{n-pr}^{a,\tau}, O^m, O^p\}$;
- if $Y = O_{n-pr}^{a,\tau}$, then $X \in \{O_{n-pr}^{a,\pi}, O_{n-pr}^{a,\tau}, O^m, O^p\}$;
- if $Y = O^m$, then $X = O^m$;
- if $Y = O^p$, then $X \in \{O^p, O^m\}$.

Definition 2 Let Xa be a deontic literal and Y any deontic operator. If $X = \neg Y$, X is a negative operator; if $X = Y$, it is a positive operator.

Definition 3 Let $A = \otimes_{i=1}^m Xa_i$ and $B = \otimes_{i=1}^n Yb_i$ be two \otimes -expressions. Then, A deontically includes B iff $m = n$, and for each Xa_i, Yb_i

- $a_i = b_i$, and
- if X and Y are positive operators, then $Y \sqsubseteq X$.

Definition 4 Let $r_1 : \Gamma \Rightarrow A \otimes B \otimes C$ and $r_2 : \Delta \Rightarrow D$ be two rules, where $A = \otimes_{i=1}^m Xa_i$, $B = \otimes_{i=1}^n Yb_i$ and $C = \otimes_{i=1}^p Zc_i$. Then r_1 subsumes r_2 iff

1. $\Gamma = \Delta$ and A deontically includes D ; or
2. $\Gamma \cup \{\neg a_1, \dots, \neg a_m\} = \Delta$ and B deontically includes D ;
or

3. $\Gamma \cup \{\neg b_1, \dots, \neg b_n\} = \Delta$ and $A \otimes \otimes_{i=0}^{k \leq p} Zc_i$ deontically includes D .

The intuition behind subsumption is that the normative content of r_2 is fully included in r_1 . Thus r_2 does not add anything new to the system and it can be safely discarded. To show this, some auxiliary notions should be used. In fact, Definition 3 (together with Definitions 1 and 2) establishes when the compliance conditions for an \otimes -expression cover the compliance conditions of another \otimes -expression³. For the sake of simplicity, take, for example, the single obligation $B = O_{n-pr}^{a,\tau} b$. Then, if another obligation A is equal to B , compliance conditions for both are trivially the same. If A is either $O_{n-pr}^{a,\pi} b$, $O^m b$, or $O^p b$, A deontically includes B , because, if both are in force, the compliance of A implies the compliance of B . However, if A is a preemptive achievement obligation, we have no guarantee that its compliance supports the compliance of B : indeed, b could have been obtained before A and B were in force, which is sufficient for fulfilling only A .

2.3.3 Solving Conflicts

Conflicts often arise in normative systems. Since PCL is based on Defeasible Logic, our framework allows us to solve them. In this regard, however, we have to determine whether we have genuine conflicts between \otimes -expressions or whether such \otimes -expressions admit states where all can be complied with.

Suppose, for example, that $A = O^p a \otimes O^m b$ and $B = O_{pr}^{a,\pi} \neg a \otimes O^m \neg b$ are both in force. The secondary obligations of A and B are clearly in contradiction but their primary obligations do not necessarily lead to a joint non-compliance. Indeed, if it is now forbidden to pay, and it is obligatory to pay by tomorrow, I can comply with both obligations by simply paying tomorrow.

Therefore, we have first to identify what \otimes -expressions do conflict with one another. First of all, let us define when two single obligations are in conflict:

Definition 5 Let l, Xl , and Y be a literal, a deontic literal, and a positive operator, respectively. The complement $\sim l$ is $\neg p$ if $l = p$, and p if $l = \neg p$. The complement $\sim Xl$ is defined as follows:

- If $Xl = Yl$, $\sim Xl = \{Zp | Z \text{ is positive, } p = \sim l, \text{ either } Z \sqsubseteq Y \text{ or } Y \sqsubseteq Z\} \cup \{\neg Zq | Z = Y, q = l\}$;
- If $Xl = \neg Yp$, $\sim Xl = \{Zq | Z \text{ is positive, } Z = Y, q = l\}$.

The following definition states under what conditions two \otimes -expressions are in conflict.

Definition 6 Let $A = \otimes_{i=1}^m Xa_i$ be an \otimes -expression. Then, $\sim A = \{B = \otimes_{i=1}^n Yb_i | m = n, \forall Xa_i, Yb_i : Xa_i = \sim Yb_i\}$.

Given a theory consisting of a set of rules R , a set S of facts (literals and deontic literals), and a superiority relation, we can use the inference mechanism of Defeasible Logic to compute, in time linear to the size of the theory, the set of its conclusions. This implies to solve genuine conflicts by resorting to the superiority relation over the rules. Once we have defined when two \otimes -expressions are in conflict (Definition 6), we can simply use the same reasoning mechanism described in (Governatori 2005).

2.3.4 Normalisation Process

We now describe how to use the machinery presented in Section 2.3.1 and Section 2.3.2 to obtain PCL normal forms. The PCL normal form of a normative system provides a logical representation of normative specifications in a format that can be used to check the compliance of a process. This consists of the following steps:

³Notice that, in Definition 3, we do not care about the types of obligation when the deontic literals to be compared are negative. In fact, an expression like $\neg Ob$ is always complied with.

1. Starting from a formal representation of the explicit clauses of a set of normative specifications we generate all the implicit conditions that can be derived from the normative system by applying the merging mechanism of PCL.
2. We can clean the resulting representation by throwing away all redundant rules according to the notion of subsumption.
3. Finally we detect and solve normative conflicts.

In general the process at step 2 must be done several times in the appropriate order as described above. The normal form of a set of rules in PCL is the fixed-point of the above constructions. A normative system contains only finitely many rules and each rule has finitely many elements. In addition it is possible to show that the operation on which the construction is defined is monotonic (Governatori & Rotolo 2006), thus according to standard set theory results the fixed-point exists and it is unique.

3 Process Modelling

A business process model (BPM) describes the tasks to be executed (and the order in which they are executed) to fulfill some objectives of a business. BPMs aim to automate and optimise business procedures and are typically given in graphical languages. A language for BPM usually has two main elements: tasks and connectors. Tasks correspond to activities to be performed by actors (either human or artificial) and connectors describe the relationships between tasks: a minimal set of connectors consists of sequence (a task is performed after another task), parallel –AND-split and AND-join– (tasks are to be executed in parallel), and choice –(X)OR-split and (X)OR-join– (at least (most) one task in a set of task must be executed).

3.1 Execution Semantics

The basic execution semantics of the control flow aspect of a business process model is defined using token-passing mechanisms, as in Petri Nets. The definitions used here extend the execution semantics for process models given by (Vanhatalo et al. 2007) with semantic annotations in the form of effects and their meaning.

A process model is seen as a graph with nodes of various types –a single start and end node, task nodes, XOR split/join nodes, and parallel split/join nodes– and directed edges (expressing sequentiality in execution). The number of incoming (outgoing) edges are restricted as follows: start node 0 (1), end node 1 (0), task node 1 (1), split node 1 (>1), and join node >1 (1). The location of all tokens, referred to as a *marking*, manifests the state of a process execution. An execution of the process starts with a token on the outgoing edge of the start node and no other tokens in the process, and ends with one token on the incoming edge of the end node and no tokens elsewhere. Task nodes are executed when a token on the incoming link is consumed and a token on the outgoing link is produced. The execution of an XOR (Parallel) split node consumes the token on its incoming edge and produces a token on one (all) of its outgoing edges, whereas an XOR (Parallel) join node consumes a token on one (all) of its incoming edges and produces a token on its outgoing edge.

3.2 Annotation of Processes

A process model is then extended with a set of annotations, where the annotations describe (i) the artifacts or effects of executing the tasks and (ii) the rules describing the obligations (and other normative positions) relevant for the process.

As for the semantic annotations, the vocabulary is presented as a set of predicates P . There is a set of process variables (x and y in Table 2), over which logical statements can be made, in the form of literals involving these

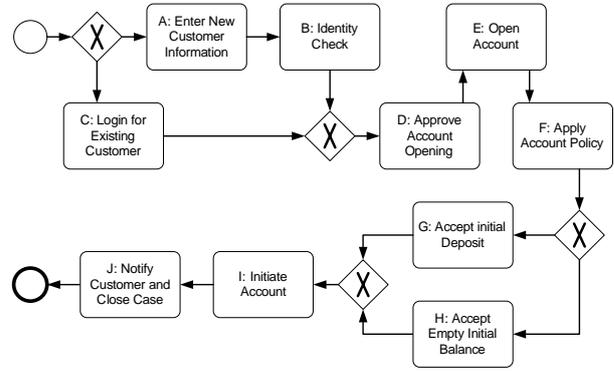


Figure 1: Example account opening process in private banking

variables. The task nodes can be annotated using *effects* which are conjunctions of literals using the process variables. The meaning is that, if executed, a task changes the state of the world according to its effect: every literal mentioned by the effect is true in the resulting world; if a literal l was true before, and is not contradicted by the effect, then it is still true (i.e., the world does not change of its own accord). We further assume that effects in parallel tasks do not contradict each other.

The obligations for this example are motivated by the following scenario: A new legislative framework has recently been put in place in Australia for anti-money laundering. The first phase of reforms for the *Anti-Money Laundering and Counter-Terrorism Financing Act 2006* (AML/CTF), covers the financial sector including banks, credit unions, building societies and trustees and extends to casinos, wagering service providers and bullion dealers. The act namely AML/CTF imposes a number of obligations, which include: customer due diligence (identification, verification of identity and ongoing monitoring of transactions); reporting (suspicious matters, threshold transactions and international funds transfer instructions); and record keeping. Table 2 shows the semantic effect annotations of the process activities.

Task	Semantic Annotation
A	$newCustomer(x)$
B	$checkIdentity(x)$
C	$checkIdentity(x), recordIdentity(x)$
D	$accountApproved(x)$
E	$owner(x, y), account(y)$
F	$accountType(y, type)$
G	$positiveBalance(y)$
H	$\neg positiveBalance(y)$
I	$accountActive(y)$
J	$notify(x, y)$

Table 2: Annotations for the process in Fig 1.

Here we give the norms governing this particular class of processes.

- All new customers must be scanned against provided databases for identity checks.

$$r_1 : newCustomer(x) \Rightarrow O_{pr}^{a,\tau} checkIdentity(x)$$

The meaning of the predicate $newCustomer(x)$ is that the input data with $Id = x$ is a new customer, for which we have the obligation to check the provided data against provided databases $checkIdentity(x)$. The obligation resulting from this rule is a non-persistent obligation, i.e., as soon as a check has been performed, the obligation is no longer in force. In addition the obligation is preemptive, this means that if for some reasons the check was already previously performed there is no need to perform it again, e.g., if

the customer were an existing customer of a company recently acquired.

- Retain history of identity checks performed.

$$r_2 : checkIdentity(x) \Rightarrow O^m recordIdentity(x)$$

This rule establishes that there is a permanent obligation to keep record of the identity corresponding to the (new) customer identified by x . In addition this obligation is not fulfilled by the achievement of the activity (for example, by storing it in a database). We have a violation of the condition, if for example, the record x is deleted from the database.

- Accounts must maintain a positive balance, unless approved by a bank manager, or for VIP customers.

$$r_3 : account(y) \Rightarrow O^m positiveBalance(y) \otimes O_{n-pr}^{a,\pi} approveManager(y)$$

The primary obligation is that each account has to maintain a positive balance *positiveBalance*; if this condition is violated (for any reason the account is not positive), then we still are in an acceptable situation if a bank manager approve the account not to be positive. In this case the obligation of approving persists until a manager approves the situation; after the approval the obligation is no longer in force.

$$r_4 : account(x), accountType(x, VIP) \Rightarrow P\text{-}positiveBalance(x)$$

This rule creates an exception to rule r_3 . Accounts of type VIP are allowed to have a non positive balance and no approval is required for this type of accounts (this is achieved by imposing that rule r_4 is stronger than rule r_3 , $r_4 \prec r_3$).

4 Compliance Checking

Our aim in the compliance checking is to figure out (a) which obligations will definitely appear when executing the process, and (b) which of those obligations may not be fulfilled. In a way, PCL constraint expressions for a normative system define a behavioural and state space which can be used to analyse how well different behaviour execution paths of a process comply with the PCL constraints. Our aim is to use this analysis as a basis for deciding whether execution paths of a process are compliant with the PCL and thus with the normative system modelled by the PCL specifications. To this end we use the following procedure:

1. We traverse the graph describing the process and we identify the sets of effects (sets of literals) for all the tasks (nodes) in the process according to the execution semantics outlined in Section 3.1.
2. For each task we use the set of effects for that particular task to determine the normative positions (obligations, permissions, prohibitions) triggered by the task. This means that effects of a task are used as a set of facts, and we compute the conclusions of the defeasible theory resulting from the effects and the PCL rules annotating the process (see Sections 2 and 3.2). In the same way we accumulate effects, we also accumulate (undischarged) obligations from one task in the process to the task following it in the process.
3. For each task we compare the effects of the tasks and the obligations accumulated up to the task. If an obligation is fulfilled by a task, we discharge the obligation, if it is violated we signal this violation. Finally if an obligation is not fulfilled nor violated, we keep the obligation in the stack of obligations and propagate the obligation to the successive tasks.

Here, we assume that the obligations derived from a task should be fulfilled in the remaining of the process. Variations of this schema are possible: for example, one could stipulate that the obligations derived from a task should be fulfilled by the tasks immediately after the task. In another approach one could use a schema where for each task one has both preconditions and effects. Then the obligations derived from the preconditions must be fulfilled by the current task (i.e., the obligations must be fulfilled by the effects of the task), and the obligations derived from the effects are as in our basic schema.

4.1 From Tasks to Obligations

The second step to check process compliance is to determine the obligations derived by the effects of a task. Given a set of rules R and a set of literals S (plain literals and deontic literals), we can use the inference mechanism of defeasible logic (Section 2) to compute the set of conclusions (obligations) in force given the set of literals. These are the obligations an agent has to obey to in the situation described by the set of literals. However, the situation could already be sub-ideal, i.e., such that some of the obligations prescribed by the rules are already violated. Thus, given a set of literals describing a state-of-affairs one has to compute not only the current obligations, but also what repair chains are in force given the set.

Consider a scenario where we have the rules $A \Rightarrow OB$ and $\neg B \Rightarrow OC$, and the effects are A and $\neg B$. The normal form of the rules is $A \Rightarrow OB \otimes OC$ and $\neg B \Rightarrow OC$. The only obligation in force for this scenario is OC . Since we have a violation of the first rule ($A \Rightarrow OB$ and $\neg B$), then we know that it is not possible to have an ideal situation here. Hence, computing only the current obligation does not tell us the state of the corresponding process. What we have to do is to identify the chain for the ideal situation for the task at hand. To deal with issue we have to identify the *active* repair chains.

Some notational conventions. Given a rule r , $A(r)$ denotes the set of premises of the rules, and $C(r)$ the conclusion. For any set of rules R , $R[C]$ denotes the subset of R of rules whose conclusion is C . If $C = p_1 \otimes \dots \otimes p_n \otimes q$ is a *repair chain*, we use $\pi_i(C)$ to denote the i -th element of the chain.

Definition 7 A *repair chain* C is *active* given a set of literals S , if

1. $\exists r \in R[C] : \forall a_r \in A(r), a_r \in S$ and
2. $\forall s \in R[D]$ such that $\pi_1(C) \in D$, either
 1. $\exists a_s \in A(s) : \sim a_s \notin S$, or
 2. $\exists i \pi_i(D) = \sim \pi_1(C)$ and $\exists k, k < i, \sim \pi_k(D) \notin S$, or
 3. $\exists t \in R[E] : \pi_j(E) = \pi_1(C), \forall a_t \in A(t), a_t \in S, \forall m, m < j, \sim \pi_m(E) \in S$ and $t \prec s$.

To illustrate the above definition we examine the following example.

Example 9 Consider the rules

$$\begin{aligned} r_1 : A_1 &\Rightarrow OB \otimes OC, \\ r_2 : A_2 &\Rightarrow O\text{-}B \otimes OD, \\ r_3 : A_3 &\Rightarrow OE \otimes O\text{-}B. \end{aligned}$$

The situation S is described by A_1 and A_3 . In this scenario the active chains are $OB \otimes OC$ and $OE \otimes O\text{-}B$. The chain $OB \otimes OC$ is active since A_1 is in S and r_2 cannot be used to activate the chain $O\text{-}B \otimes OD$. For r_3 and the resulting chain $OE \otimes O\text{-}B$, we do not have the violation of the primary obligation OE of the rule (i.e., $\neg E$ is not in S), so the obligation $O\text{-}B$ is not entailed by r_3 .

4.2 Checking Compliance

A repair chain is in force if there are a rule of which the repair chain is the consequent and a set of facts (effects of a task in a process) including the rule antecedents.

In addition we assume that, once in force, a reparation chain remains as such unless we can determine that it has been violated or the obligations corresponding to it have all been obeyed to (these are two cases when we can discharge an obligation or reparation chain). This means that it is not possible to have two instances at the same time of the same reparation chain. Accordingly, a reparation chain in force is uniquely determined by the combination of the task T when the chain has been derived and the rule R from which the chain has been obtained.

Definition 8 Given a sequence of sets of literals S_1, \dots, S_n (corresponding to a sequence of tasks in a process model) a chain C is terminated by S_n if $\exists S_i$ such that C is active at S_i , $\exists s \in R[B]$ such that $B = A \otimes C$, and $\exists r \in R[E]$, $\pi_1(E) = \sim \pi_1(C)$, $A(r) \subseteq S_n$ and $s \neq r$.

A terminated chain means that we have reached a deadline. First of all to terminate an active chain the chain must have been active. This means that there is a rule from which the chain has been derived (rule s above), and then we have reached a termination condition (encoded by rule r). The termination condition must be triggered ($A(r) \subseteq S_n$), and the terminating rule should not be weaker than the rule from which the chain has been derived. Given a task/set of literals S we will use *Terminated* to refer to the set of rules terminated by S .

The procedure for compliance checking has two steps. In the first step, given a set of literals S , corresponding to the effects of a task T in a process model, we identify the set (*Current*) of active chains for the process. The set of the active chains includes the new active chains triggered by the task, as well as the chains carried over from previous tasks. Then, in the second phase, the algorithm *CheckCompliance* scans all elements of *Current* against the set of literals S , and determines the state of each reparation chain ($C = A_1 \otimes A_2$, where A_2 can be \perp , making $A_1 \otimes A_2$ equivalent to A_2 , see (Governatori & Rotolo 2006)) in *Current* and *Terminated*. *CheckCompliance* operates as follows:⁴

```

for  $C \in \text{Current}$ 
  if  $A_1 = O^p B$ , then
    if  $B \in S$ , then
      remove( $[T, R, A_1 \otimes A_2], \text{Current}$ )
    if  $[T, R, B_1 \otimes B_2 \otimes A_1 \otimes A_2, B_2] \in \text{Violated}$ , then
      add( $[T, R, B_1 \otimes B_2 \otimes A_1 \otimes A_2, B_2], \text{Compensated}$ )
    else
      remove( $[T, R, A_1 \otimes A_2], \text{Current}$ )
      add( $[T, R, A_1 \otimes A_2, B], \text{Violated}$ )
      add( $[T, R, A_2], \text{Current}$ )
  if  $A_1 = O^{a,x} B$ ,  $x \in \{\pi, \tau\}$ , then
    if  $B \in S$ , then
      remove( $[T, R, A_1 \otimes A_2], \text{Current}$ )
      remove( $[T, R, A_1 \otimes A_2], \text{Unfulfilled}$ )
      if  $[T, R, B_1 \otimes B_2 \otimes A_1 \otimes A_2, B_2] \in \text{Violated}$ , then
        add( $[T, R, B_1 \otimes B_2 \otimes A_1 \otimes A_2, B_2], \text{Compensated}$ )
      else
        add( $[T, R, A_1 \otimes A_2], \text{Unfulfilled}$ )
    if  $A_1 = O^m B$ , then
      if  $B \notin S$  or  $\neg B \in S$ , then
        add( $[T, R, A_1 \otimes A_2, B], \text{Violated}$ )
        add( $[T, R, A_2], \text{Current}$ )
for  $C \in \text{Terminated}$ 
  if  $[T, R, A_1 \otimes A_2] \in \text{Unfulfilled}$ , then
    add( $[T, R, A_2], \text{Current}$ )
    add( $[T, R, A_1 \otimes A_2, B], \text{Violated}$ )

```

⁴In the presentation of the algorithm we do not differentiate between preemptive and non-preemptive achievement obligations. A simple solution to handle both at the same time is that of labelling each effect with the task the effect is associated with. Accordingly to fulfil a preemptive obligation $O_{p^x}^a$ in force from a task A we have to have p^B , where B is the identified of the task the effect p is associated with, such that either task B follows task A (and this is the case also for non-preemptive achievement obligations), or task B precedes task A and there is no task C between B and A such that $\neg p^C$. We overload the inclusion operator \in in the algorithm below with such functionalities.

if $A_1 = O^{a,\tau}$, then
remove($T, R, A_1 \otimes A_2, \text{Current}$)

Let us examine the *CheckCompliance* algorithm. Remember the algorithm scans all active reparation chains one by one, and then for each of them reports on the status of it. For each chain in *Current* (the set of all active chains), it looks for the first element of the chain and it determines the content of the obligation (so if the first element is $O^x B$, the content of the obligation is B). Then it checks whether the obligation has been fulfilled (B is in the set of effects), or violated ($\neg B$ is in the set of effects), or simply we cannot say anything about it (none of B and $\neg B$ is in the set of effects). In the first case, for punctual and achievement obligations we can discharge the obligation and we remove the chain from the set of active chains (similarly if the obligation was carried over from a previous task, i.e., it was in the set *Unfulfilled*); for maintenance obligation we have to keep the obligation among the current obligations. In case of a violation, we add the information about it in the system. Notice that we can use current to detect a violation only for punctual and maintenance obligations. To record that we have a violation we insert a tuple with the identifier of the chain and what violation we have in the set *Violated*. In addition, we know that violations can be compensated, thus if the chain has a second element we remove the violated element from the chain and put the rest of the chain in the set of active chains. Here we take the stance that a violation of a maintenance obligation does not discharge it, thus we do not remove the chain from the set of active chains; however, this is the case for a punctual obligation. In case we do not have information about whether B or $\neg B$ we do not have the information to assert that a maintenance obligation has been fulfilled, thus it has been violated. However, for achievement obligations the set of effects does not tell us if the obligation has been fulfilled or violated, so we propagate the obligation to the successive tasks by putting the chain in the set *Unfulfilled*. The algorithm also checks whether a chain/obligation was previously violated but it was then compensated. In the final part of the algorithm we consider the terminated chains, i.e., the chains for which we have reached a deadline. If the obligation has not been fulfilled we signal a violation as we have already described above. The important point to notice here is that in case of a persistent achievement obligation, the chain is kept in the set of current chains, but this is not the case for a non-persistent achievement obligation.

To check the compliance of business process against a set of normative specifications we can use the algorithm *CheckCompliance*. The algorithm given a set of literals a set of rules determines (1) the rules that have been fired (deriving thus the obligations in force for the situation described by the set of literals) (2) which obligations have been fulfilled, violated, or we do not have enough information to assert their normative state.

At run time an instance of a business process defined by a business process model is a sequence of (sets) of tasks. For each element of the sequence we generate the set of effects and then we use the *CheckCompliance* to determine the state of the obligations. Thus determining whether a business process is compliant is to determine whether the business process can be executed without end in a state where there are unrepaired violations.

The conditions below relate the state of a process based as reported by the *CheckCompliance* algorithm and the semantics for PCL expressions. In particular, a process is compliant if the situation at the end of the process is at least sub-ideal (it is possible to have violations but these have been compensated for). Similarly a process is fully compliant if it results in an ideal situation (i.e., there are no violations).

Definition 9

- An execution path is compliant iff for all $[T, R, A] \in$

Current, $A = OB \otimes C$, for every $[T, R, A, B] \in Violated$, $[T, R, A, B] \in Compensated$ and $Unfulfilled = \emptyset$.

- An execution path is fully compliant iff for all $[T, R, A] \in Current$, $A = OB \otimes C$, $Violated = \emptyset$ and $Unfulfilled = \emptyset$.

Accordingly, an execution path is not compliant if the set of unfulfilled obligations (*Unfulfilled*) is not empty. Consider, for example the rule

$$r_3 : account(y) \Rightarrow O^m positiveBalance(y) \otimes O_{n-pr}^{a,\pi} approveManager(y)$$

relative to the process of Figure 1 with the annotation as in Table 2. After task *E* we have, among others, the effect $account(y)$. This means that after task *E* we have the chain

$$[E, r_4, O^m positiveBalance(y) \otimes O_{n-pr}^{a,\pi} approveManager(y)]$$

in *Current* for task *F*. After task *F*, the above entry for the chain obtained from rule r_4 is moved to the set *Unfulfilled*. Suppose now that tasks *G* and *H* do not have any annotation attached to them. In this case at the end of the process we still have the active chain, but the resulting situation is not ideal: the antecedent of the rule is a subset of the set of effects, but we do not have the first element of the chain as one of the effects. Thus, the process is not compliant.

It is possible to give two definitions of compliance for business processes.

Definition 10 A business process is absolutely compliant if every execution path is compliant. A business process is weakly compliant if at least one execution path is compliant.

5 Compliance at Runtime and Design Time: A Balance and Open Problems

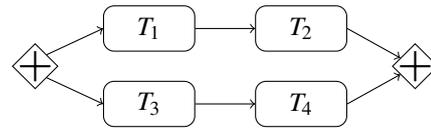
The algorithm to check compliance determines, given an execution path of a business process, whether the execution path is compliant or not. The algorithm has linear time complexity. As we have seen *CheckCompliance* work on an execution path thus the same algorithm can be used to check compliance at design time and at run time. Thus checking compliance at run time (also known as conformance) can be performed in linear time. However, at run time the algorithm returns the obligations in force for every task with no look ahead features. Despite the linear time complexity of *CheckCompliance* the problem of checking compliance of a business process at design time is in general computationally hard no matter the notion of compliance one chooses (absolute or weak). This depends on the number of possible execution paths for a process.

The first source of complexity is given by (X)OR-split/(X)OR-joins. The semantics for such a construct is that for each (X)OR-split at least one of the paths is executed. Thus the number of execution paths to examine for multiple (X)OR-split/joins exhibits a tree-like growth.

As we have seen the algorithm takes the set of effects that hold after a task and it uses them to determine the active chains and the obligations in force for the successive task. Given the persistence of some types of obligations, some obligations carry over from one task to the (immediately) successive task(s).

The propagation of persistent obligation from task to task is (computationally) trivial for a linear sequence of tasks, and so is the propagation of effects. Again the propagation of effects over AND-splits and AND-joins has linear time complexity. All one has to do is to traverse the single paths, cumulate the effects, and then union the effects when one reaches the AND-join. For the accumulation of effects we assume, as is typically the case, that effects persist from one task to the successive one, and if the new task introduces an effect contradicting an effect

obtained in a previous task, this corresponds to an update, i.e., we remove the old one and we add the new one. Based on this assumption we have that for every task the set of effects cumulated for the task is consistent, and so is every path. Hence, it has been argued that parallel paths should not introduce inconsistency (inconsistent parallel paths correspond to an incorrect design of a process). It is possible to adopt such a solution for weak compliance. We check the compliance of each parallel sub-process in an AND-split/AND-join if all of them are compliant we are guaranteed that there exists at least one execution path combining all the tasks to be executed in parallel is compliant. However, such a strategy is far from being satisfactory for absolute compliance apart for the case where we only have pre-emptive⁵, non-terminating achievement obligations, and the triggering of the obligation either happened before the AND-split or the obligation does not admit exceptions. For the other cases the compliance of all individual parallel executions does not guarantee that the process is absolutely compliant. Consider for example a process with the following sub-process



where T_1 is annotated with a , T_2 with b , T_3 with c , and T_4 with d . The rules are $r_1 : a \Rightarrow O^m c$ and $r_2 : b \Rightarrow O^m d$. It is immediate to verify that both execution paths T_1, T_2 and T_3, T_4 are compliant. At the same time it is easy to see that some sequential combinations of the four tasks are compliant, e.g., T_1, T_3, T_2, T_4 is not compliant since b is not an effect of T_3 and thus we have a violation of the maintenance obligation obtained from the effect of T_1 and rule r_1 . At the same time the execution path $T_1, \{T_2, T_3\}, T_4$ is compliant⁶. Similar examples can be given for punctual and non-preemptive achievement obligations.

Accordingly, a process designer is left with the dilemma of choosing between weak compliance at design time with the consequence that the execution of the process might not be compliant (the designer has no control on what execution path will be performed at run time), or to face absolute compliance with a combinatorial explosion of the number of possible execution paths to be verified for compliance. Another alternative is to spend more time on the design of the process and to give additional structure to the process to introduce further synchronisation for parallel tasks. However, most of the current languages for modelling business processes have very limited capabilities to model true synchronisation. A possible solution is to extend business process language with temporal constraint language to introduce truly synchronisation constructs (see for example (Lu et al. 2006)).

6 Summary and Related Work

This paper discusses a means of visualizing the impact of compliance controls on business process and of assisting in compliance checking, analysis and feedback for subsequent (re)design of the processes. The procedure is based in principle on efficient algorithms and is able to deal with repair chains of deontic statements of various types.

A number of works have been devoted to compliance in control modelling. (Goedertier & Vanthienen 2006) presents the logical language PENELOPE, that provides the ability to verify temporal constraints arising from compliance requirements on effected business processes.

⁵The meaning of a pre-emptive obligation is that something must happen in a process, but it does not matter where and when it happens in the process.

⁶Remember we have defined an execution path as a sequence of sets of tasks. The intended meaning is that each element of the sequence is a granule of time (identified with at least one task), and we group together truly concurrent tasks. To simplify the notation we drop the set theoretic notation for singletons.

(Küster et al. 2007) develops a method to check compliance between object lifecycles that provide reference models for data artifacts e.g. insurance claims and business process models. (Giblin et al. 2006) provides temporal rule patterns for regulatory policies, although the objective of this work is to facilitate event monitoring rather than the usage of the patterns for support of design time activities. Furthermore, (Agrawal et al. 2006) presented an architecture for supporting Sarbanes-Oxley Internal Controls, which include functions such as workflow modelling, active enforcement, workflow auditing, as well as anomaly detection. (Farrell et al. 2005) studies the performance of business contract based on their formal representation. (Desai et al. 2008) seeks to provide support for assessing the correctness of business contracts represented formally through a set of commitments. The reasoning is based on value of various states of commitment as perceived by cooperative agents. (Ghose & Koliadis 2007) consider an approach where the tasks of a business process model, written in BPMN, are annotated with the effects of the tasks, and a technique to propagate and cumulate the effects from a task to a successive contiguous one is proposed. The technique is designed to take into account possible conflicts between the effects of tasks and to determine the degree of compliance of a BPMN specification. Also, there have been recently some efforts towards support for process modelling against compliance requirements. (zur Muehlen & Rosemann 2005) provides a method for integrating risks in business processes. The proposed technique for “risk-aware” business process models is developed for EPCs (Event-driven Process Chains) using an extended notation. (Sadiq et al. 2007) proposes an approach based on control tags to visualize internal controls on process models. (Liu et al. 2007) takes a similar approach of annotating and checking process models against compliance rules, although the visual rule language (BPSL) does not directly address the deontic notions providing compliance requirements.

As far as we are aware of the issue of studying compliance with a rich ontology of obligations has been neglected. Indeed, the majority of work ignore the normative aspects related to the issue, and those that address the normative aspects are limited to pre-emptive achievement obligations. The only exception is (Governatori et al. 2008) where, an algorithm for approximate weak compliance is presented, based on some heuristics.

Acknowledgements

Many people contributed to the ideas and results presented in this paper. In particular special thanks are due to Shazia Sadiq, Zoran Milošević, Arthur ter Hofstede, Marcello La Rosa, Ingo Weber, Jörg Hoffmann, Aditya Ghose and Peter Baumgartner, for the discussions we had with them about various issues of business process compliance.

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy, the Australian Research Council through the ICT Centre of Excellence program and the Queensland Government.

References

- Agrawal, R., Johnson, C. M., Kiernan, J. & Leymann, F. (2006), Taming compliance with Sarbanes-Oxley internal controls using database technology, in ‘Proc. ICDE 2006’.
- Antoniou, G., Billington, D., Governatori, G. & Maher, M. J. (2001), ‘Representation results for defeasible logic’, *ACM Transactions on Computational Logic* 2(2), 255–287.
- Carmo, J. & Jones, A. (2002), Deontic logic and contrary to duties, in D. Gabbay & F. Guenther, eds, ‘Handbook of Philosophical Logic, 2nd Edition’, Kluwer.
- Desai, N., Narendra, N. C. & Singh, M. P. (2008), Checking correctness of business contracts via commitments, in ‘Proc. AAMAS 2008’.
- Farrell, A. D. H., Sergot, M. J., Sallé, M. & Bartolini, C. (2005), ‘Using the event calculus for tracking the normative state of contracts’, *International Journal of Cooperative Information Systems* 14.
- Ghose, A. & Koliadis, G. (2007), Auditing business process compliance, in ‘Service Oriented Computing, ISOC 2007’, LNCS, Springer, pp. 169–180.
- Giblin, C., Müller, S. & Pfützmann, B. (2006), From regulatory policies to event monitoring rules: Towards model driven compliance automation, Technical report, IBM Zurich Research Lab.
- Goedertier, S. & Vanthienen, J. (2006), Designing compliant business processes with obligations and permissions, in ‘Business Process Management (BPM) Workshops’.
- Governatori, G. (2005), ‘Representing business contracts in RuleML’, *International Journal of Cooperative Information Systems* 14(2-3), 181–216.
- Governatori, G., Hoffmann, J., Sadiq, S. W. & Weber, I. (2008), Detecting regulatory compliance for business process models through semantic annotations, in ‘Business Process Management Workshops’, pp. 5–17.
- Governatori, G., Hulstijn, J., Riveret, R. & Rotolo, A. (2007), Characterising deadlines in temporal modal defeasible logic, in ‘Proc. Australian AI 2007’.
- Governatori, G. & Rotolo, A. (2006), ‘Logic of violations: A Gentzen system for reasoning with contrary-to-duty obligations’, *Australasian Journal of Logic* 4, 193–215.
- Governatori, G. & Rotolo, A. (2008a), An algorithm for business process compliance, in G. Sartor, ed., ‘Jurix 2008’, IOS Press, pp. 186–191.
- Governatori, G. & Rotolo, A. (2008b), Changing legal systems: Abrogation and annulment. part ii: Temporalised defeasible logic, in ‘Proceedings of Normative Multi Agent Systems (NorMAS 2008)’.
- Governatori, G., Rotolo, A. & Sartor, G. (2005), Temporalised normative positions in defeasible logic, in ‘10th International Conference on Artificial Intelligence and Law (ICAIL05)’, pp. 25–34.
- Governatori, G. & Sadiq, S. (2009), The journey to business process compliance, in H. Cardoso & W. van der Aalst, eds, ‘Handbook of Research on BPM’, IGI Global, pp. 426–454.
- Küster, J. M., Ryndina, K. & Gall, H. (2007), Generation of business process models for object life cycle compliance, in ‘Proc. BPM 2007’.
- Liu, Y., Müller, S. & Xu, K. (2007), ‘A static compliance-checking framework for business process models’, *IBM Systems Journal* 46(2), 335–362.
- Lu, R., Sadiq, S., Padmanabhan, V. & Governatori, G. (2006), Using a temporal constraint network for business process execution, in ‘Seventeenth Australasian Database Conference (ADC2006)’, pp. 157–166.
- Lu, R., Sadiq, S. W. & Governatori, G. (2007), Compliance aware business process design, in ‘Business Process Management Workshops’, pp. 120–131.
- Sadiq, S. & Governatori, G. (2009), A methodological framework for aligning business processes and regulatory compliance, in J. van Brocke & M. Rosemann, eds, ‘Handbook of Business Process Management’, Springer.
- Sadiq, S., Governatori, G. & Naimiri, K. (2007), Modelling of control objectives for business process compliance, in ‘Proc. BPM 2007’.
- Vanhatalo, J., Völzer, H. & Leymann, F. (2007), Faster and More Focused Control-Flow Analysis for Business Process Models through SESE Decomposition, in ‘Proc. ICSOC 2007’.
- zur Muehlen, M. & Rosemann, M. (2005), Integrating risks in business process models, in ‘Proc. ACIS 2005’.

INVITED PAPER

Ontology Consolidation in Bioinformatics

Sven Hartmann¹Henning Köhler²Jing Wang³

¹ Clausthal University of Technology, Germany
Email: sven.hartmann@tu-clausthal.de

² The University of Queensland, Brisbane, Australia
Email: henning@itee.uq.edu.au

³ Massey University, Palmerston North, New Zealand
Email: j.w.wang@massey.ac.nz

Abstract

Ontologies enjoy increasing popularity among bioinformatics researchers who seek assistance in coping with the rapid upgrowth of biological data to be handled and related information to be considered. While communities of committed ontology pioneers drive the development and optimisation of ontologies for a wide range of relevant domains, the ample adoption of ontologies in bioinformatics research is still decelerated by technological, managerial and communication barriers. This paper contains a brief review of current trends, challenges and perspectives in ontology research and practice in bioinformatics.

Keywords: biomedical ontology, ontology uptake

1 Introduction

The advent of high-throughput technology in analytical research laboratories calls for high-throughput tool support that enables bioinformatics researchers to keep pace with processing and analysing ever-increasing amounts of new biological data, and to draw new insights from them. Ontologies guide this process as knowledge bases that comprise existing knowledge about some subject of interest in a systematic and unequivocal way. This helps the bioinformatics community to create a common understanding of the subject and supports bioinformatics researchers in performing a variety of tasks.

In bioinformatics the notion ‘ontology’ is used to refer to a variety of modelling artifacts, ranging from controlled vocabularies, thesauri and taxonomies to conceptual schemas for particular kinds of biological data. Common to them is the aim to provide a set of concepts that can serve as abstractions for biological entities that are of interest for some application. Typically concepts have a name, a definition and a list of synonyms. Concept names are frequently called terms. The distinction between concepts and terms is not always explicit. Often multiple synonymous terms (including abbreviations and acronyms) are used to refer to the same concept. For convenience, concepts are often equipped with a unique id (or accession number) that can be used for identification and for cross-referencing. Definitions can be textual descriptions or by listing properties that characterise the concept. For an example, see Fig. 1.

Copyright ©2010, Australian Computer Society, Inc. This paper appeared at the Seventh Asia-Pacific Conference on Conceptual Modelling (APCCM 2010), Brisbane, Australia, January 2010. Conferences in Research and Practice in Information Technology (CRPIT), Vol. 110, Sebastian Link and Aditya K. Ghose, Ed. Reproduction for academic, not-for profit purposes permitted provided this text is included.

Concepts are often organised in hierarchies, as known from taxonomies. Subsumption is most frequent, but meronymy is common, too. More advanced ontologies capture other kinds of relationships, too, that serve as abstractions for further associations between biological entities. For a survey of relationships used by ontologies in bioinformatics we refer to (Smith & al. 2005), and for a brief historical outline to (Bodenreider & Stevens 2006).

AccessionNo	GO:0000001
Name	mitochondrion inheritance
Definition	The distribution of mitochondria, including the mitochondrial genome, into daughter cells after mitosis or meiosis, mediated by interactions between mitochondria and the cytoskeleton.
Synonyms	exact: mitochondrial inheritance

Figure 1: An example of a concept from the Gene Ontology (biological process).

2 Ontologies for Bioinformatics Researchers

Primary sources for bioinformatics researchers are the National Center for Biomedical Ontology (NCBO) and the European Bioinformatics Institute (EBI). As of November 2009 the NCBO BioPortal¹ lists 179 ontologies with a total of 1,438,792 concepts, while the Ontology Lookup Service (OLS)² hosted by EBI lists 74 ontologies with a total of 923,606 terms. Both include the about 60 ontologies of the Open Biomedical Ontologies (OBO) foundry³. The OBO foundry is a special interest group bringing together ontology providers from the life science disciplines. Major objectives are to establish common design principles for and to foster inter-operability between ontologies (Smith & al. 2007).

Ontologies used in bioinformatics differ with respect to their scope, size and granularity. Some ontologies are devoted to anatomy and physiology of specific organisms, some provide the means for describing biomedical resources and experiments in analytical research labs. Other ontologies are devoted to related areas like health care and medical applications, e.g., disease ontologies. In addition to open ontologies there are proprietary ontologies. Many of them are just used by a single research group, project or tool. Some are publicly available, but may be subject to intellectual property restrictions.

In the next section we will briefly outline a few established OBO ontologies for illustration.

¹ bioportal.bioontology.org/

² www.ebi.ac.uk/ontology-lookup/

³ www.obofoundry.org/

2.1 Examples of Ontologies in Use

Gene Ontology (GO) is probably the most popular ontology in bioinformatics. Its origin dates back to 1998 when several model-organism database projects (initially yeast, fruit fly and mouse) noticed that a common vocabulary will improve inter-operability across databases and simplify data integration (Ashburner & al. 2000). Gene Ontology assembles concepts that serve for classifying gene products according to what they do, where they act and how they perform these activities. It actually comprises three separate ontologies, one for molecular functions, one for cellular components, and one for biological processes. In addition to subsumption (is-a) and meronymy (part-of) it captures a third kind of relationships (regulates) for describing interactions between biological processes and other biological processes, molecular functions or biological qualities.

Most biological databases used by bioinformatics researchers for their investigations contain data at the molecular level. *Sequence Ontology (SO)* provides concepts for describing the features and attributes of biological sequences (Eilbeck & al. 2005). This includes biological, biomaterial and experimental features. SO captures further kinds of spatial relationships, relationships for locating features on sequences.

Molecular-level data studied in research labs is increasingly linked to concepts reflecting the global structure and anatomy of organisms. Suppose we want to know which genes influence the development of organs and tissue, or which genetic mutations cause deviations from the standard phenotype. Then it helps to find out which genes are expressed at which development stages and in which parts of an organism. The *Foundational Model of Anatomy Ontology (FMA)* is a comprehensive ontology of human anatomy, with more than 75,000 concepts for describing elements of human morphology, anatomic structures and the organisation of the human body (Rosse & Mejino 2003). It uses about 170 different kinds of relationships, e.g. to describe spatial associations between anatomic structures like organs and tissue.

Plant Ontology (PO) assembles concepts that serve for classifying gene products according to morphological and anatomical criteria (Jaiswal & al. 2005). It comprises two ontologies, one for plant structure (anatomy and morphology) and one for growth and development (growth stages in a plant's life cycle and developmental stages of plant structures). Besides subsumption (is-a) and meronymy (part-of) PO also captures a kind of temporal relationships (develops-from) for describing associations between derived structures and progenitor structures.

2.2 Ontology Formats

As Gene Ontology is the longest established ontology in there, most of the OBO ontologies are represented in the native format of GO, also known as the OBO flat file format (OBOF). The OBO repository also contains a translation of FMA into OBOF. This translation, however, omits all kinds of relationships other than is-a, part-of and has-part, as they do not yet conform to the guidelines set by the OBO foundry for relationship specification (Smith & al. 2005). The primary format of FMA is frame-based and can be manipulated with Protégé which is probably the leading general-purpose ontology editor available today.

As a matter of fact, tool support is crucial for the uptake of ontologies by application developers and bioinformatics researchers. A range of tools is available for processing OBO flat files, e.g., ontology browsers like AmiGO (Carbon & al. 2009) and ontology editors like DAG-Edit and OBO-Edit (Day-

Richter & al. 2007). These tools were designed particularly for use in bioinformatics, having the needs of bioinformatics researchers in mind. Today there is a clear tendency to use the Web Ontology Language (OWL) for representing ontologies. It enables the use of a variety of tools developed for a wider user group, provides better support for capturing advanced ontology elements like axioms and constraints, and unlocks new application areas such as ontology reasoning and the semantic web, cf. (Golbreich et al. 2008).

In the last few years many of the OBO ontologies have been translated into OWL and are now available in both formats. The NCBO has recently agreed on a mapping between OBOF and OWL (Moreira & al. 2009) and released a conversion tool that can be embedded into OBO-Edit and Protégé for import/export (Moreira & Musen 2007). OBO Explorer (Aitken et al. 2008) is a recent graphical ontology editor based on Protégé and the NCBO conversion tool that allows native access to both formats. As OBOF does not have a formal grammar, the intended semantics of concepts and relationship types had to be mapped. On that occasion, a number of modelling sins were observed that called for further clarification, cf. (Aranguren & al. 2007), some of them are still to be addressed. For a discussion of naming conventions in OBO and OWL see (Schober & al. 2009). Even worse, many current OBO concepts and relationships lack information essential for formal definition in OWL. Other common formats that are in use for exchanging ontologies are plain XML and RDF/S.

2.3 Access to Ontologies

Though most established ontologies are equipped with dedicated web sites, bioinformatics researchers often prefer web portals with an integrated web interface for browsing and querying all ontologies of interest. Examples of such web portal are the NCBO BioPortal (Noy & al. 2009) and the Ontology Lookup Service (OLS) at EBI (Cote & al. 2008). They can be used to search for particular terms in any of the participating ontologies, to retrieve descriptions and additional meta-data stored with the concepts, and to navigate through the ontology along relationships. Both use a relational DBMS as backend to store the concepts and relationships of the participating ontologies. To keep the database up-to-date the ontology providers are polled on a regular basis, updated files are downloaded and scanned for changes.

A major advantage of web portals like OLS and BioPortal is that they greatly reduce the time needed to familiarise oneself with a query interface and to actually execute queries against multiple ontologies. This allows users to consider a larger set of ontologies at no extra costs. Query results are typically presented in a unified format, thus simplifying further processing. While standard tasks like browsing ontologies and detecting ontology-enriched databases, there is currently only limited support for other routine tasks. More usable web applications would enable bioinformatics researchers to make better use of online information. This includes the implementation of ideas from social software (e.g. community feedback, community-based ontology evaluation, collaborative maintenance of ontology mappings), software customisation, service and grid computing, cf. (Li & al. 2007). Both, OLS and BioPortal can be accessed by client applications as web services. They come with standard WSDL descriptions, thus making it easy to invoke them in scientific workflows.

For most routine tasks like data analysis bioinformatics researchers access ontologies through web-based or desktop tools that are integrated with one or more ontologies. Often such tools come along with a

backend database that stores the ontology, or they can be connected to a user-defined database. For a recent survey of 68 ontology-based analysis tools, see (Huang et al. 2009). Many of them still rely on Gene Ontology only, but considerable efforts are underway to better integrate other ontologies, too. For examples, cf. (Khatri & al. 2007, Sherman & al. 2007, Draghici et al. 2006, Kirov & al. 2005).

2.4 Storing Ontologies in Local Databases

If ontologies are frequently used it is worth considering to store them locally. Potential benefits are faster response times to queries, and the option to flexibly use an ontology with available software tools, to keep ontologies together with the actual annotations of biological data, to store several versions of an ontology, to extend ontologies by own concepts, descriptions or other useful meta-data. On the downside, however, local storage increases IT support costs and requires the implementation of a suitable update strategy.

Many providers of OBO ontologies offer users to download their ontology in OBOF or OWL, but also as an SQL dump file. This includes GO, PO and SO. For details see (Harris & al. 2005). OLS offers SQL export of their entire internal term database.⁴ FMA supports SQL export, too. Its authors, however, recommend to use the SQL dump with Protégé. For a detailed discussion of how ontologies are kept in relational databases, we refer to (Keet 2006). Alternatively, if ontologies are offered in XML format for download then XML-enhanced database servers like DB2, Oracle or Tamino can be used to store the ontology, too.

A promising approach to enable the use of multiple ontologies is Chado (Mungall & Emmert 2007) which was developed by the Generic Model Organism Database (GMOD) project⁵. It offers a large relational database schema that is centred around ontologies. It predefines generic tables for flexible storage of ontology concepts and relationships, and can be extended by tables for storing biological data linked to them. For example, there is a special table collection for storing sequence features. Biological databases that conform to this schema can easily inter-operate with one another, and can be used with software from the GMOD toolkit, e.g., the sequence viewer GBrowse (Donlin 2009). For some major ontologies like GO, SO and PO, Perl scripts are available for translating them into the Chado format.

3 Use of Ontologies

In analytical research labs a good deal of time is spent with acquiring, storing, retrieving and analysing biological data obtained by running experiments or by querying biological databases. Bioinformatics researchers can use of ontologies for all these tasks. For a recent survey of such activities we refer to (Bodenreider 2008, Rubin et al. 2007).

3.1 Annotating Data

Terms (or the ids of the corresponding concepts) can be used to annotate biological data in a consistent manner. Semantic annotations are assertions that link biological entities in the application domain to concepts in the ontology. Annotations are useful for a variety of routine tasks in analytical research labs, such as querying databases or analysing data sets. Often annotations are created on the basis of the data

contained in experimental reports, primarily those published in the literature and available from digital libraries for the life sciences, such as PubMed⁶. The annotations may be stored directly in the biological database, or in a separate annotation database together with the ontology. A recent example of annotating organism-specific data is given in (Beck & al. 2009). For a brief summary of where annotations come from and what they mean, see (Hill et al. 2008).

A range of model-organism and multi-species databases have their data enriched by annotations. This includes all major biological databases, such as UniProt or Ensembl. To assure the correctness of annotations, publicly available biological databases often have expert curators in place for this task. The annotation of biological data using ontologies is a time-consuming exercise as it is performed manually for the most part. Empirical studies show that manual curation is far too slow (Baumgartner & al. 2007). Some approaches have been proposed for automated annotation, cf. (Doms & Schroeder 2005, Vinayagam & al. 2006, Couto & al. 2006, Daraselia & al. 2007). So far none of the underlying annotation algorithms works accurately enough to fully compensate human curators. This is partly due to the lack of benchmarks and reliable training data. High-throughput data analysis, however, will eventually require automated annotation to cope with the amount of data to be processed. For the related task of biological text mining, we refer to (Chapman & Cohen 2009). This can help to automatically detect mentions of relevant concepts or interesting biological entities in scientific publications.

3.2 Querying and Analysing Data

Once the data in a biological database is annotated the annotations can be used for queries. For example one can ask for all biological entities linked to a particular concept, or for all concepts a biological entity is linked to, or for biological entities that are linked to the same concepts. Using the appropriate terms a bioinformatics researcher can search, for example, for all genes that are involved in a particular biological process or are expressed in a particular plant structure. Search criteria may be combined, filters may be set, and aggregation functions like counting may be used to form more involved queries. Search results can be ranked by the extend to which they match non-boolean search criteria. Queries can also be combined with similarity search tools like BLAST. The best BLAST hit method (Jones et al. 2005), for example, finds the concepts that the BLAST top hit of a search sequence is linked to. A popular web-based query tool is AmiGO (Carbon & al. 2009).

Experiments undertaken in analytical research labs produce data sets to be analysed. Studying the annotations available for them may offer new insights into the potential biological meaning of an experiment. If an experiment generates a list of genes as output, one may want to identify those concepts that are most characteristic for the genes in the list. For that, over-representation of concepts is determined in the context of some background superset, such as all the genes on a microarray chip, or all the genes in a genome. Such an analysis takes into account annotations that are inherited by transitivity of certain kinds relationships (is-a, part-of), cf. (Grossmann et al. 2007). Generally speaking, a concept may help to differentiate the genes in the list from the rest if the observed over-representation is statistically significant. Similarly one may look for characteristic relationships between the genes in such lists and other biological

⁴www.ebi.ac.uk/ontology-lookup/implementationOverview.do

⁵www.gmod.org

⁶www.ncbi.nlm.nih.gov/pubmed/

entities. Ontology-based data analysis can help to validate hypotheses about the domain of study and make implicit knowledge explicit. In the literature a wide range of analysis tools has been suggested to support bioinformatics researchers in answering such questions. We refer the reader to (Huang et al. 2009, Khatri & Draghici 2005) for a detailed discussion of the scope and capabilities of ontology-based analysis tools, including the statistical models, ontologies, and data mining algorithms deployed.

More recently, scientific workflows have emerged in analytical research labs to streamline and automate data analysis, cf. (Cure & Jablonski 2007, Ram et al. 2008). Popular tools for designing and executing such workflows are Kepler and Taverna (Hull & al. 2006).

3.3 Exchanging and Integrating Data

An important aspect of developing and using ontologies is that concepts need to be identified, named, and enhanced by descriptions. In life sciences there are often several terms in use that refer to the same or similar concepts. Descriptions should help to decide whether terms refer to the same, related or different concepts. As bioinformatics researchers exchange annotated data and retrieve annotated data from different biological databases it is advantageous to use terms consistently for annotations or at least to keep track of terminological associations like synonymy, abbreviations or acronyms. Commonly agreed ontologies help achieving that goal. Currently a large portion of biological data in publicly available databases is annotated with free-text fields that are not yet associated with ontology concepts (Shah & al. 2009). Empirical studies demonstrate benefits of ontology-based over free-text search (Moskovitch & al. 2007). Automated mappings of free-text terms to concepts have studied in (Dai & al. 2008).

The thorough analysis of new experimental data requires bioinformatics researchers to consult multiple data sources such as local databases with outcomes of earlier experiments and studies, and publicly available databases with reference data for comparisons. Relevant biological data is spread across and disseminated through an ever-increasing number of databases. Ontologies provide common languages to overcome semantic heterogeneity among independently designed and maintained biological databases, enable data exchange between different research groups, foster inter-operability between diverse databases, and detect distributed data that is inter-related by associations between the corresponding biological entities. Ontologies can guide data transformations for data warehouses, but also query transformations for mediation-based integration, cf. (Hernandez & Kambhampati 2004, Bodenreider 2008). OntoFusion (Perez-Rey & al. 2006) is an example of a mediation tool that exploits Gene Ontology. For a recent case study on ontology-driven data integration, see (Smedley & al. 2008).

When data from diverse sources is integrated data inconsistencies are likely to occur. These inconsistencies can be due to inaccurate and missing data, or due to wrong decisions made during integration. Data analysis based on inconsistent biological databases may lead to uninteresting and wrong results. Ontologies can help to discover such inconsistencies in advance (Chen et al. 2007).

3.4 Representing and Sharing Knowledge

Ontologies offer a conceptualisation of a scientific domain in a formal and unambiguous way. Their formal specification makes the available knowledge about the domain explicit in a form that is accessible to both

humans and machines. Ontologies enable researchers to describe the entities in the domain consistently. This forms the basis for a shared and commonly accepted understanding of that domain. Ontologies are consistent but not necessarily complete. So they are challenged each time when new information about the underlying domain becomes available, and updated if necessary. Due to the rapid growth of information this happens frequently in bioinformatics.

4 Development of Ontologies

4.1 Ontology Design

Most of the established ontologies in bioinformatics were originally designed by hand. Since then, however, a range of methodologies and tools have emerged that aim to support ontology designers in developing and maintaining ontologies. In the literature a few design methodologies have been proposed, cf. (Cristani & Cuel 2005, Kamel et al. 2007, Darlington & Culley 2008) but none of them is widely known nor used. In practice new ontologies are usually created in an iterative refinement process, similar to other modelling artifacts like database schemas. For an interesting discussion of differences between ontology and database schema design, we refer to (Noy & Klein 2004). The generic design principles (Smith & al. 2007) of the OBO foundry provide arguably the most influential guidance for the design of ontologies in bioinformatics, even though not all of the principles are well-justified, e.g. the call for a unique root element.

Recurring steps in the design process are the creation of new concepts (including names, description, and synonyms) and the specification of relationships to other concepts (including the placement of the concept in is-a and other hierarchies). Design decisions need to be checked against already existing parts of the ontology, e.g., name clashes should be avoided and descriptions should be consistent with specified relationships. Ontologies can be built bottom-up with the most specific concepts first, or top-down starting with the most general ones, or both approaches may be mixed. Some ontologies were built in a modular fashion that allows the cooperation of several designers or the integration of already existing ontologies or parts thereof, cf. (Thomas & al. 2006, Pathak et al. 2009). Design patterns were exploited in (Aranguren & al. 2008) for designing the *Cell Cycle Ontology (CCO)*. Some ontologies were built from database schemas of databases to be annotated, cf. (Lubyte & Tessaris 2009, Zhao & Chang 2007).

4.2 Ontology Evolution and Maintenance

High-throughput analysis of biological data results in new observations and insights that contribute to our understanding of the respective application domain. Ontologies need to be updated to keep pace with the new knowledge. Typical update operations are the addition of new concepts and relationships to meet new requirements or to reflect new insights, the modification of exiting concepts and relationships if they need to be clarified or adapted to stay consistent with new insights, or the deletion of outdated concepts and relationships that are no longer needed or in line with our understanding. For example, one might want to add a new intermediate concept into the is-a hierarchy to support a new abstraction level. To implement such an update one needs to name and describe the new concept and to decide where to insert it.

Updates in ontologies may be motivated by the desire to coordinate the development of ontologies as promoted by the OBO foundry. When ontologies

cover related domains it is likely that they possess common or similar concepts. There are plenty of reasons to make such implicit connections explicit. This can be done, for example, by unifying concepts or by defining mappings between matching concepts. Discovering such connections is by far not trivial and often hampered by semantic heterogeneity or different levels of granularity across ontologies and the scientific communities using them. There are tools available that compute mappings automatically, cf. (Ghazvinian et al. 2009, Jean-Marya et al. 2009) but in most cases human expertise is needed.

Once ontologies are in use the evolution process needs to be adequately documented and managed. Updates of ontologies are likely to affect exiting annotations and exiting mappings to alternative formats or to other ontologies. Updates need to be propagated effectively and potential problems need to be resolved, e.g. in case of modifications and deletions that may impair previously correct annotations. This results in a trade-off between robustness and up-to-dateness. While new knowledge should be incorporated as soon as possible to be ready-to-use, existing tools and applications require a certain level of stability to be useful and constrain maintenance costs. Due to the dynamic nature of the represented knowledge, many ontology providers release separate versions of their ontology at regular intervals. Updates are accumulated in a new release of the ontology while previously released versions stay unchanged. For a systematic study ontology evolution, including examples and consequences for annotations and ontology mappings, we refer to (Hartung et al. 2008).

Most established ontologies in bioinformatics have seen continuous change since their creation. In Gene Ontology, for example, the number of concepts has reduplicated since 2002. OnEX (Hartung et al. 2009) is a simple web-based tool for exploring the evolutionary aspect of major ontologies in bioinformatics. A more generic approach towards monitoring ontology evolution is presented in (Park et al. 2008). Traditionally, many OBO ontologies had a strong focus on concepts while relationships were often neglected. Considerable efforts are underway to overcome this situation, cf. (Smith & al. 2007). Following the recommendation of the OBO foundry, logical definitions are provided for relationships, and relationships are defined at both concept- and instance-level when appropriate. We also refer to (Eilbeck & Mungall 2009) for the particular example of Sequence Ontology.

The maintenance of established ontologies is in most cases done by dedicated teams of ontology curators who retrieve update proposals from the literature and from the scientific community, reach consensus in case of conflicting proposals, verify and approve updates, and generate new releases of the respective ontologies. This is time-consuming and requires highly specialised experts, however, many ontology providers consider this necessary due assure a reasonable level of quality and scientific rigour. Automated extension has also been suggested and first experiences recorded, e.g. for extending Gene Ontology using text mining and ontology matching techniques to extract new terms, concepts and relationships (Pesquita et al. 2009). For a generic approach towards automated ontology integration, alignment and extension, see (Novacek et al. 2009).

5 Trends, Challenges and Directions

5.1 User Involvement

Despite the recent growth in the number, size and coverage of ontologies in bioinformatics they are still

far from unfolding their full potential in bioinformatics research. This is mainly due to communication barriers between different communities and research groups. Still today many bioinformatics researchers come in touch with ontologies only as part of analysis tools. However, there is an increasing interest and awareness of ontologies among researchers. Yet only few of them know how precisely ontologies can accelerate their work. As web portals and tools for querying annotation databases mature their users will become increasingly curious about the scope and granularity of the underlying ontologies.

Most bioinformatics researchers have their focus on a small, individualised portion of the huge multi-faceted life science domain, however, in their sub-domain they are interested in highly specialised knowledge. Browsing the integrated term databases underlying web portals like OLS or the NCBO BioPortal can be confounding due to their sheer complexity. On the other hand, selecting particular ontologies that better meet individual needs is not an easy exercise either. Often there is not enough information available to make an informed choice. Even when ontology providers offer detailed documentation and online tutorials this will only be perceived useful later on. For efficient decision-making most users will prefer more condensed information.

As argued in (Tan & Lambrix 2009) ontology selection relies on evaluation and comparison of candidate ontologies. Ontologies can be evaluated against a catalogue of human-made criteria (Lozano-Tello & Gomez-Perez 2004) or exploit statistics about the ontology (Gangemi et al. 2006) and its usage (Maiga & Williams 2008, Porzel & Malaka 2004). None of these approaches is perfect. Community-based assessment, collaborative filtering and recommender services as known from social networks and Web 2.0 applications can help to tackle that problem. Social functionality like community feedback and peer-review as recently added to the BioPortal is likely to improve the uptake of ontologies and ontology-based tools. Ontology pioneers can act as multipliers in these user communities.

When using ontologies bioinformatics researchers will find it helpful if they are tailored to their particular needs. It might be desirable to have user- or application-specific ontologies at hand that cover just the sub-domain of interest and are sufficiently fine-grained. An application of such personalised ontologies for information extraction is discussed in (Tao & Embley 2007). Specialised ontologies may be extracted and further evolved from existing ontologies that continue to serve as reference ontologies. Users are domain experts and will play an active role in the generation of such specialised ontologies. Since bioinformatics researchers have only limited experience with ontology evolution and maintenance a new generation of ontology editors will be required to keep the learning curve flat. Eventually this may lead to the creation of a collection of coupled ontologies centred around a master ontology that need to be harmonised on a regular basis.

The development of specialised ontologies may again be a collaborative effort of research groups or small communities with a common interest in certain sub-domains. Some first experience with collaborative ontology evolution using an extension of the ontology editor Protégé have been reported in (Tudorache et al. 2008), and using a wiki-based tool in (Hoehndorf & al. 2009). To foster reuse of specialised ontologies it would be helpful to set up a dedicated ontology registry that allows other researchers to discover existing ontologies that best meet their needs.

5.2 Enhancing Ontologies for Reasoning

Ontologies as formal specifications of knowledge are amenable to automated reasoning. Reasoning tools can help in developing and maintaining ontologies, but also in making inferences from existing knowledge, and checking hypotheses against ontologies. Ontology design often focusses on concepts, but relationships and axioms deserve more attention as they represent valuable knowledge, too. The popularity of OWL is partly due its expressiveness that allows one to capture more of the semantics of the underlying domain than other ontology formats like OBOF. Thus OWL promotes the creation of more precise and comprehensive artifacts. In particular, it provides powerful means for specifying axioms, even if we restrict ourselves to the OWL-Lite or OWL-DL fragments of OWL that guarantee decidability in reasoning. These fragments are also well-supported by reasoning tools such as FaCT++, Pellet and Racer that all can be integrated with Protégé. Potential applications of reasoning in ontology design were presented in (Lutz & al. 2006), and requirements for reasoning in bioinformatics were discussed in (Keet et al. 2007).

Typical examples include existence, disjointness, covering, cardinality and universal constraints, transitivity and symmetry. (Aranguren & al. 2007) claims that biologists would not widely use axioms as that would require more knowledge than is usually available. We rather believe that axioms can be beneficial in answering questions that go beyond simple instance or relationship checking. Clearly it will take considerable efforts to add information that is currently missing, e.g. by reinspecting annotations or even rerunning experiments. Relationships and axioms can impose necessary and/or sufficient conditions on entities to be linked to certain concepts. For example, there has been a long discussion about the meaning of the part-of relationships in Gene Ontology and others. Axioms can reflect that entities linked to a particular concept must occur in a certain relationship, or that occurrence is optional. Reasoning helps ontology designers to understand the consequences of design decisions. Careful reconsideration of existing ontologies helps to detect inconsistencies, modelling errors and gaps. Automated reasoning can guide this process and provide informed feedback for ontology repair. Reasoning helps bioinformatics researchers to think about what new observations mean in the context of what is already known. Reasoning with adequately enhanced ontologies offers better support for objectives like computing derived relationships, generating new hypotheses, consistency checking, discovering equivalence or similarity among concepts, re-formalising ill-specified domains, reusing ontologies, or extracting and evolving specialised ontologies.

5.3 Automation

Bioinformatics researchers will allocate increasing portions of their time to ontology engineering rather than data engineering. For that, however, they need to be released from tedious tasks like data processing or experimentation. The combination of high-throughput experimental devices and analysis tools is a first step to automatise many routine tasks that results in a very large number of experiments that can be executed simultaneously. Data analysis workflows can be executed in flexible ontology-based workflow tools that integrate experimental data with other data resources to validate hypotheses. So far, however, the design of experiments and workflows is still done by human researchers. Experience tells that these tasks are tedious, too, in particular when they need to be reiterated several times.

A major step towards lab automation are robot scientists as presented in (King & al. 2009). The prototype system has been used to refine hypotheses about yeast and test them by conducting the respective experiments. The operation is fully automated except for the periodic replacement of lab consumables. The system uses an in-house developed ontology that has been evolved from the *Ontology of Scientific Experiments (EXPO)* (Soldatova et al. 2006). EXPO captures generic knowledge about experiments and is available in OWL, thus enabling automated reasoning. (Epple & Scherf 2009) reports on an ontology-based expert system that implements a strategy for building hypotheses from inspected data. Automated hypothesis generation may guide bioinformatics researchers to experiments that will result in potentially interesting observations. In the near future we will see more examples of hypothesis-led investigations. For that, however, ontologies are required that capture the concepts and principles of human knowledge discovery.

6 Discussion

Ontologies in bioinformatics are still ‘work-in-progress’. Over the last few years ontologies have gradually transformed the way bioinformatics researchers approach experiments, analyse biological data and generate new knowledge. Active involvement of users enabling them to adopt ontologies for their own needs, better reasoning support through high-quality ontologies, and ontology-based automation will promote the future uptake of ontologies in analytical research labs.

Acknowledgements

We are grateful to the organisers of the Asia-Pacific Conference on Conceptual Modelling (APCCM) for inviting us to write this survey paper.

References

- Aitken, S., Chen, Y. & Bard, J. (2008), ‘OBO Explorer: an editor for open biomedical ontologies in OWL’, *Bioinformatics* **24**(3), 443–444.
- Aranguren, M. E. & al. (2007), ‘Understanding and using the meaning of statements in a bio-ontology: recasting the Gene Ontology in OWL’, *BMC Bioinformatics* **8**(57), 1–13.
- Aranguren, M. E. & al. (2008), ‘Ontology design patterns for bio-ontologies: a case study on the cell cycle ontology’, *BMC Bioinformatics* **9**(Suppl5), S1.
- Ashburner, M. & al. (2000), ‘Gene ontology: tool for the unification of biology’, *Nature Genet* **25**, 25–29.
- Baumgartner, W. A. & al. (2007), ‘Manual curation is not sufficient for annotation of genomic databases’, *Bioinformatics* **23**, i41–i48.
- Beck, T. & al. (2009), ‘Practical application of ontologies to annotate and analyse large scale raw mouse phenotype data’, *BMC Bioinformatics* **10**(Suppl5), S2.
- Bodenreider, O. (2008), Biomedical ontologies in action: Role in knowledge management, data integration and decision support, in ‘IMIA Yearbook Medical Informatics’, pp. 67–79.
- Bodenreider, O. & Stevens, R. (2006), ‘Bio-ontologies: current trends and future directions’, *Briefings in Bioinformatics* **7**(3), 256–274.

- Carbon, S. & al. (2009), 'AmiGO: online access to ontology and annotation data', *Bioinformatics* **25**(2), 288–289.
- Chapman, W. W. & Cohen, K. B. (2009), 'Current issues in biomedical text mining and natural language processing', *J Biomedical Informatics* **42**(5), 757–759.
- Chen, Q., Chen, Y.-P. P. & Zhang, C. (2007), 'Detecting inconsistency in biological molecular databases using ontologies', *Data Min Knowl Disc* **15**, 275–296.
- Cote, R. G. & al. (2008), 'The ontology lookup service: more data and better tools for controlled vocabulary queries', *Nucleic Acids Research* **36**, 372–376.
- Couto, F. M. & al. (2006), 'GOAnnotator: linking protein GO annotations to evidence text', *J Biomed Discov Collab* **1**, 19.
- Cristani, M. & Cuel, R. (2005), 'A survey on ontology creation methodologies', *Int J Semantic Web Inf Syst.* **1**, 49–69.
- Cure, O. & Jablonski, S. (2007), 'Ontology-based data integration in data logistics workflows', in 'Advances in Conceptual Modeling', Vol. 4802 of *LNCS*, pp. 34–43.
- Dai, M. & al. (2008), 'An efficient solution for mapping free text to ontology terms', in 'AMIA Summit on Translational Bioinformatics'.
- Daraselia, N. & al. (2007), 'Automatic extraction of gene ontology annotation and its correlation with clusters in protein networks', *BMC Bioinformatics* **8**, 243.
- Darlington, M. J. & Culley, S. J. (2008), 'Investigating ontology development for engineering design support', *Advanced Engineering Informatics* **22**(1), 112–134.
- Day-Richter, J. & al. (2007), 'OBO-Edit - an ontology editor for biologists', *Bioinformatics* **23**(16), 2198–2200.
- Doms, A. & Schroeder, M. (2005), 'GoPubMed: exploring pubmed with the gene ontology', *Nucleic Acids Research* **33**, W783–W786.
- Donlin, M. J. (2009), 'Using the generic genome browser (GBrowse)', *Curr Protoc Bioinform* **9**.
- Draghici, S., Sellamuthu, S. & Khatri, P. (2006), 'Babel tower revisited: a universal resource for cross-referencing across annotation databases', *Bioinformatics* **22**, 2934–2939.
- Eilbeck, K. & al. (2005), 'The Sequence Ontology: a tool for the unification of genome annotations', *Genome Biology* **6**(R44), 1–12.
- Eilbeck, K. & Mungall, C. J. (2009), 'Evolution of the Sequence Ontology terms and relationships', in 'Nature Precedings'.
- Epple, A. & Scherf, M. (2009), 'Bibliosphere hypothesis generation in regulatory network analysis', in 'Bioinformatics for Systems Biology', pp. 401–412.
- Gangemi, A., Catenacci, C., Ciaramita, M. & Lehmann, J. (2006), 'Modelling ontology evaluation and validation', in 'European Semantic Web Conference'.
- Ghazvinian, A., Noy, N. F. & Musen, M. A. (2009), 'Creating mappings for ontologies in biomedicine', in 'AMIA Annual Symposium'.
- Golbreich, C., Horridge, M., Horrocks, I., Motik, B. & Shearer, R. (2008), 'OBO and OWL: Leveraging semantic web technologies for the life sciences', in 'Semantic Web', Vol. 4825 of *LNCS*, pp. 169–182.
- Grossmann, S., Bauer, S., Robinson, P. N. & Vingron, M. (2007), 'Improved detection of overrepresentation of gene-ontology annotations with parentchild analysis', *Bioinformatics* **23**(22), 3024–3031.
- Harris, M. A. & al. (2005), 'The gene ontology (GO) database and informatics resource', *Nucleic Acids Research* **32**, D258–D261.
- Hartung, M., Kirsten, T., Gross, A. & Rahm, E. (2009), 'OnEX: Exploring changes in life science ontologies', *BMC Bioinformatics* **10**, 250.
- Hartung, M., Kirsten, T. & Rahm, E. (2008), 'Analyzing the evolution of life science ontologies and mappings', in 'DILS', Vol. 5109 of *LNBI*, pp. 11–27.
- Hernandez, T. & Kambhampati, S. (2004), 'Integration of biological sources: Current systems and challenges ahead', *SIGMOD Record* **33**(3), 51–60.
- Hill, D. P., Smith, B., McAndrews-Hill, M. S. & Blake, J. A. (2008), 'Gene Ontology annotations: what they mean and where they come from', *BMC Bioinformatics* **9**(Suppl5), S2.
- Hoehndorf, R. & al. (2009), 'BOWiki: an ontology-based wiki for annotation of data and integration of knowledge in biology', *BMC Bioinformatics* **10**(Suppl5), S5.
- Huang, D. W., Sherman, B. T. & Lempicki, R. A. (2009), 'Bioinformatics enrichment tools: paths toward the comprehensive functional analysis of large gene lists', *Nucleic Acids Research* **37**, 1–13.
- Hull, D. & al. (2006), 'Taverna: a tool for building and running workflows of services', *Nucleic Acids Research* **34**, W729W732.
- Jaiswal, P. & al. (2005), 'Plant ontology (PO): a controlled vocabulary of plant structures and growth stages', *Comp Funct Genom* **6**, 388–397.
- Jean-Marya, Y. R., Shironoshitaa, E. P. & Kabuka, M. R. (2009), 'Ontology matching with semantic verification', *Web Semantics: Science, Services and Agents on the World Wide Web* **7**(3), 235–251.
- Jones, C. E., Baumann, U. & Brown, A. L. (2005), 'Automated methods of predicting the function of biological sequences using GO and BLAST', *BMC Bioinformatics* **6**, 272.
- Kamel, M. N., Lee, A. Y. & Powers, E. C. (2007), 'A methodology for developing ontologies using the Ontology Web Language (OWL)', in 'ICEIS', pp. 261–268.
- Keet, C. M. (2006), 'Using and improving bio-ontologies stored in relational databases', in 'SBI-OLBD'.
- Keet, C. M., Roos, M. & Marshall, M. S. (2007), 'A survey of requirements for automated reasoning services for bio-ontologies in OWL', in 'OWL: Experiences and Directions'.
- Khatri, P. & al. (2007), 'Onto-Tools: new additions and improvements', *Nucleic Acids Research* **35**, W206–W211.

- Khatri, P. & Draghici, S. (2005), 'Ontological analysis of gene expression data: current tools, limitations, and open problems', *Bioinformatics* **21**, 3587–3595.
- King, R. D. & al. (2009), 'The automation of science', *Science* **324**, 85–89.
- Kirov, S. A. & al. (2005), 'GeneKeyDB: a lightweight, gene-centric, relational database to support data mining environments', *BMC Bioinformatics* **6**, 72.
- Li, K.-C. & al. (2007), BioPortal: A portal for deployment of bioinformatics applications on cluster and grid environments, in 'High Performance Computing for Computational Science', Vol. 4395 of *LNCS*, pp. 566–578.
- Lozano-Tello, A. & Gomez-Perez, A. (2004), 'Ontometric: A method to choose the appropriate ontology', *J Database Management* **15**(2), 1–18.
- Lubyte, L. & Tessaris, S. (2009), Automatic extraction of ontologies wrapping relational data sources, in 'DEXA', Vol. 5690 of *LNCS*, pp. 128–142.
- Lutz, C. & al. (2006), Reasoning support for ontology design, in 'Experiences and Directions'.
- Maiga, G. & Williams, D. (2008), 'A user centered approach for evaluating biomedical data integration ontologies', *European J Scientific Research* **24**(1), 55–68.
- Moreira, D. A. & al. (2009), The NCBO OBOF to OWL mapping, in 'Nature Precedings', pp. 1–6.
- Moreira, D. A. & Musen, M. A. (2007), 'OBO to OWL: A Protege OWL tab to read/save OBO ontologies', *Bioinformatics* **23**, 1868–1870.
- Moskovitch, R. & al. (2007), 'A comparative evaluation of full-text, concept-based, and context-sensitive search', *J Am Med Inform Assoc* **14**(2), 164–174.
- Mungall, C. J. & Emmert, D. B. (2007), 'A Chado case study: an ontology-based modular schema for representing genome-associated biological information', *Bioinformatics* **23**, i337–i346.
- Novacek, V., Laera, L., Handschuh, S. & Davis, B. (2009), 'Infrastructure for dynamic knowledge integration - automated biomedical ontology extension using textual resources', *J Biomedical Informatics* **41**(5), 816–828.
- Noy, N. F. & al. (2009), 'BioPortal: Ontologies and integrated data resources at the click of a mouse', *Nucleic Acids Research* **37**, W170–W173.
- Noy, N. F. & Klein, M. (2004), 'Ontology evolution: Not the same as schema evolution', *Knowledge and Information Systems* **6**(4), 428–440.
- Park, J. C., Kim, T. & Park, J. (2008), 'Monitoring the evolutionary aspect of the gene ontology to enhance predictability and usability', *BMC Bioinformatics* **9**(Suppl3), S7.
- Pathak, J., Johnson, T. M. & Chute, C. G. (2009), 'Survey of modular ontology techniques and their applications in the biomedical domain', *J Integrated Computer-Aided Engineering* **16**(3), 225–242.
- Perez-Rey, D. & al. (2006), 'ONTOFUSION: ontology-based integration of genomic and clinical databases', *Comput Biol Med* **36**(7-8), 712–730.
- Pesquita, C., Grego, T. & Couto, F. (2009), Identifying gene ontology areas for automated enrichment, in 'Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living', Vol. 5518 of *LNCS*, pp. 934–941.
- Porzel, R. & Malaka, R. (2004), A task-based approach for ontology evaluation, in 'ECAI Workshop Ontology Learning and Population'.
- Ram, S., Zhang, K. & Wei, W. (2008), Linking biological databases semantically for knowledge discovery, in 'Advances in Conceptual Modeling', Vol. 5232 of *LNCS*, pp. 22–32.
- Rosse, C. & Mejino, J. V. L. (2003), 'A reference ontology for biomedical informatics: the Foundational Model of Anatomy', *J Biomed Inform* **36**, 478–500.
- Rubin, D. L., Shah, N. H. & Noy, N. F. (2007), 'Biomedical ontologies: a functional perspective', *Briefings in Bioinformatics* **7**, 75–90.
- Schober, D. & al. (2009), 'Survey-based naming conventions for use in OBO foundry ontology development', *BMC Bioinformatics* **10**(125), 1–9.
- Shah, N. H. & al. (2009), 'Ontology-driven indexing of public datasets for translational bioinformatics', *BMC Bioinformatics* **10**(Suppl2), S1.
- Sherman, B. T. & al. (2007), 'DAVID knowledgebase: a gene-centered database integrating heterogeneous gene annotation resources to facilitate high-throughput gene functional analysis', *BMC Bioinformatics* **8**, 426.
- Smedley, D. & al. (2008), 'Solutions for data integration in functional genomics: a critical assessment and case study', *Briefings in Bioinformatics* **9**(6), 532–544.
- Smith, B. & al. (2005), 'Relations in biomedical ontologies', *Genome Biology* **6**(R46), 1–15.
- Smith, B. & al. (2007), 'The OBO foundry: coordinated evolution of ontologies to support biomedical data integration', *Nature Biotechnol* **25**, 1251–1255.
- Soldatova, L. N., Clare, A., Sparkes, A. & King, R. D. (2006), 'An ontology for a robot scientist', *Bioinformatics* **22**(14), e464–e471.
- Tan, H. & Lambrix, P. (2009), Selecting an ontology for biomedical text mining, in 'Workshop on BioNLP', pp. 55–62.
- Tao, C. & Embley, D. (2007), Seed-based generation of personalized bio-ontologies for information extraction, in 'Advances in Conceptual Modeling', Vol. 4802 of *LNCS*, pp. 74–78.
- Thomas, C. J. & al. (2006), Modular ontology design using canonical building blocks in the biochemistry domain, in 'Formal Ontology in Information Systems', pp. 115–127.
- Tudorache, T., Noy, N. F., Tu, S. & Musen, M. A. (2008), Supporting collaborative ontology development in protege, in 'The Semantic Web', Vol. 5318 of *LNCS*, pp. 17–32.
- Vinayagam, A. & al. (2006), 'GOPET: a tool for automated predictions of Gene Ontology terms', *BMC Bioinformatics* **7**(161), 1–7.
- Zhao, S. & Chang, E. (2007), Mediating databases and the semantic web: a methodology for building domain ontology from databases and existing ontologies, in 'Semantic Web & Web Services'.

CONTRIBUTED PAPERS

Conceptual Modelling in 3D Virtual Worlds for Process Communication

Ross A. Brown

School of Information Technology
Queensland University of Technology
PO Box 2434, Brisbane 4001, Queensland

r.brown@qut.edu.au

Abstract

Traditionally, conceptual modelling of business processes involves the use of visual grammars for the representation of, amongst other things, activities, choices and events. These grammars, while very useful for experts, are difficult to understand by naive stakeholders. Annotations of such process models have been developed to assist in understanding aspects of these grammars via map-based approaches, and further work has looked at forms of 3D conceptual models. However, no one has sought to embed the conceptual models into a fully featured 3D world, using the spatial annotations to explicate the underlying model clearly. In this paper, we present an approach to conceptual process model visualisation that enhances a 3D virtual world with annotations representing process constructs, facilitating insight into the developed model. We then present a prototype implementation of a 3D Virtual BPMN Editor that embeds BPMN process models into a 3D world. We show how this gives extra support for tasks performed by the conceptual modeller, providing better process model communication to stakeholders.

Keywords: Conceptual Modelling, Virtual Worlds, Process Models, BPMN.

1 Introduction

Conceptual Process Modelling is a visual approach used to represent how an organisation carries out its day to day activities (Rosemann et al., 2006).

Process Modelling uses grammatical notations that represent a conceptual model of the processes within a business, and may include a data model that supports the processes within the enterprise. Business Process Model representations include visual representations for Activities, Choices, Events, Messages, Compound processes. Refined forms of these constructs have been developed into international standards, such as Business Process Modeling Notation (BPMN) (OMG, 2006). Their structure and modelling capabilities are a part of continuing research in the area of BPM (Rosemann et al., 2006).

While these representations are very useful for experts in the field of business process modelling, there is the

problem of using these representations to communicate such processes to the rest of the stakeholders who are not cognisant of the visual grammars used.

It may be stated that such representations are focussed on the abstract components of a business model, for in principal, this is the only requirement for the modelling of information within an enterprise, as the data is an abstracted representation of the real enterprise, and is not a representation of physical things, due to its irrelevance to the information processing required.

This is made apparent by research showing that grammars such as BPMN have difficulty with representing physical things, leading to problems with communicating these models to stakeholders (Recker et al., 2007). The approach of removing physical aspects of a business model in conceptual grammars is problematic on a number of counts.

Firstly, every business object that is modelled in a business process model, has a physical representation within the real world, which is interacted with by stakeholders (Rosemann et al., 2006). Such physical objects have physical properties that form a component of how an enterprise performs. For example, the location, and spatial arrangement of tasks being performed in a business, will bring about effects upon those same activities, due to space limitations in a building, or influences on task planning and resource allocations from distances to be covered.

Secondly, when wanting to communicate this business processes to other stakeholders, the process of communication involves a physical representation component, especially to non-process cognisant stakeholders. People perform their work in the real world, not in a conceptual space. It can be argued that stakeholders will not consider their work in a conceptual manner, but in a "hands-on" manner that involves real artifacts in real spaces. Indeed, this has been noted by other researchers in the field, that have sought to use philosophical techniques drawn from ontological research (Bunge, 1977), applied to the process of defining the specifications for Information Systems (Green and Rosemann, 2004). Their belief is that the absence of such object representational abilities hampers the clear specification of an Information System.

It can be argued that this object representational issue still holds for even process cognisant stakeholders, such as business analysts. In the end, their conceptual models at some level are still drawn from physical artefacts, and therefore are influenced in structure and dynamics by the same said physical artefacts.

Therefore that it is logical that a visual simulation of a Physical Model of the enterprise will support and improve communication processes in Conceptual Modelling, as it has in many other data visualisation domains (Tufte, 1983). It is believed that this visual approach also supports the various underlying Process Communication tasks within the Business Process Life Cycle (van der Aalst, 2004). It can be conjectured that many aspects of the process modelling life cycle (eg. Modelling, Improvement and Monitoring) can be assisted, more or less, by the inclusion of an easy to access simulation of the business, implemented as a 3D Virtual Environment (VE), due to the ease of process communication afforded by such representations.

The main argument of this paper is that simulations of business process models in virtual environments can be an invaluable addition to the toolset available to business analysts, and will support their communication tasks. This paper endeavours to make a first impression on what is potentially a major field of conceptual modelling research, showing a theoretical framework and a proof of concept implementation of a 3D Virtual Environment Modeller.

The rest of the paper is structured as follows. Section Two covers previous work in higher dimensional representations of business processes, utilising richer visual representations than 2D static diagrams. Section Three details a conceptual framework for process models in virtual worlds, indicating their utility and specific tasks and contributions to process modelling. Section Four details our Implementation of a proof of concept BPMN Process Model tool, with an example of a software quality assurance process model developed within the world described in Section Five. Section Six concludes the paper with a discussion on future work.

2 Previous Work

Presently, the state of the art in software technology for conceptual modelling of business processes is embodied in 2D static grammars, implemented via drawing tools. The tools vary between commercial systems such as Visio, to ARIS, over to experimental Grammars developed with research modelling systems (van der Aalst and ter Hofstede, 2005).

Annotations are provided in some cases for non-standard icons, that allow for representations that are closer to real imagery of the activities in question (CaseWise, 2008). Some systems include mapping capabilities, to show the spatial arrangements of process models as reviewed and shown in (Brown and Paik, 2009) and (de Leoni et al., 2008). However, in each case, the work is still limited to 2D.

Three Dimensional representations of process models have been developed. Typically, a graph-based version of process models is extended from 2D to 3D. These have been developed for a number of years now (Schonhage et al., 2000) (Pamplin and Zhu, 2004) (Stefanie Betz et al., 2008) and have been incorporated into swim lane models of process modelling systems (Effinger et al., 2009).

Some commercial experimental systems have investigated 2.5D (oblique projection) and 3D virtual models of process systems (OnMap, 2009) (Interactive-

Software, 2004). While very promising, these implementations have not utilised the power of full 3D virtual worlds for representation and remote collaboration across a network, and still do not incorporate a conceptual model into the visualisation as is shown in this paper.

IBM has pioneered visualisations of process models in 3D games systems via their Innov8 project (IBM, 2008). However, their focus has been on games for training, and not tools for the development of integrated conceptual process modelling grammars for process life cycle stages. As a result, the environments are not for general modelling purposes, as is envisaged in this paper.

What is lacking is a conceptual modelling and visualisation approach that integrates the present grammars into the 3D Worlds to provide the ability to effectively provide a 3D visualisation annotation to a conceptual process model.

In this paper just such a framework is developed which integrates together the best of both worlds; conceptual and physical modelling and simulation, to provide an approach to process modelling that enables the practitioner to more easily communicate the newly developed process model to the viewer, via providing object annotations and representations that communicate the process model more clearly.

3 Conceptual Process Modelling Virtual Reality Framework

The newly developed framework seeks to provide support for communicating process models to naive stakeholders. The reason for this is two fold.

Firstly, it has been reported in a number of fora, the difficulty, and potential redundancy, of grammar components, creating a need to ascertain what the main components of such a conceptual modelling framework need to be (Muehlen and Recker, 2008). Such communication problems are related to the complexity of the representations in conceptual models, so an environment that juxtaposes the model with its original physical space will ease understanding of the structure of the process model.

Secondly, it has been shown in other domains that one of the best strengths of virtual environment models is the area of education and communication (Gallagher et al., 2005). Thus the use of VEs for this task is intuitive due to their success in other communication domains.

As previously described, the intention is to model the physical environment surrounding the process model. So there is a need to ascertain the physical modelling requirements, and how this should be embedded into the virtual world.

For the purpose of this framework, the intention is to cross the boundary between conceptual models and physical models, by placing the grammars in a virtual world simulation of the physical environment of the process model. This provides a method of showing the conceptual model boundaries and how it will interact with a physical reality of the business, and thus will enable a conceptual modeller and a client to communicate on a better footing, as shown in Figure 1. This extends present modelling practices, whereby the conceptual modeller would analyse the physical model of the business process,

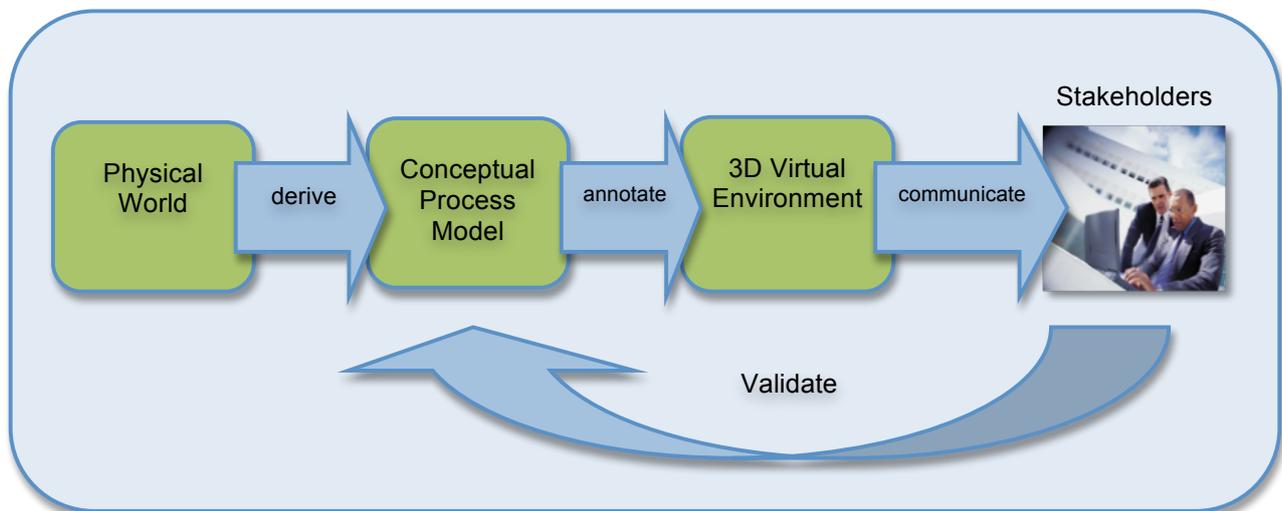


Figure 1 Concept diagram of the Virtual Environment Modelling approach, showing the major processes involved, including the derivation of the Conceptual Process Model, annotation with a 3D Virtual Environment, and using the Virtual Environment to communicate and validate the Process Model.

via monitoring and interviews. In this case the modeller can add a simulation of the physical world to their information sources.

It should be highlighted, that a valuable collaborative feature that comes from the use of such VEs for validation purposes, is the ability to spatially store feedback information about the conceptual process model. So, as well as being able to simulate the business process, we seek to store annotation information in the simulation for validation processes.

We now identify the requirements for a software approach to conceptual modelling in 3D Virtual Environments.

3.1 Task Analysis

The environment can then be used as an additional tool in a number of process modelling tasks that emerge from analysis of the task of communicating. Using a task analysis before deriving visualisations enhances the chances of the visualisations being useful to the stakeholders in question (Treinish, 1999):

- Process Communication - placing the conceptual model in a physical environment, facilitates stakeholder buy in to the new processes by being able to see the process model simulated in that physical environment.
- Validation with Clients - conceptual process models can be embedded in the environment, to facilitate insight in to the conceptual model for the client. And vice versa, the analyst may understand the underlying processes better, if they are able to see the physical reality of the environment, and have it explained to them by the client.
- Spatial Factors - the physical representation of the business in the virtual environment enables insight into how the business site impacts on the implementation of new process models, especially via the spatial location of the activities involved.
- Human Resources - the direct representation of humans as avatars in the VE enables insight into the

how the people perform tasks, as groups, and enables them to see interactions between roles, and potential bottlenecks from the resource perspective.

- Process Change – “as is” and “to be” (Jörg Becker et al., 2003) process models can be represented and compared in the environment for stakeholder buy-in and validation purposes.

3.2 Virtual Environment Functional Requirements

A set of technological requirements has been derived to support the previous major tasks outlined:

1. Spatial modelling of the location of activities, choices and events. Each component of the process model needs to have an X,Y,Z coordinate to embed it into the world.
2. Geometric Modelling of the process model components. Each process model component needs to be converted, even translated, into a 3D geometric model, with features that represent the modelling grammar either directly or indirectly.
3. Modelling of the physical surrounds of the model – buildings, business objects and people. The business model simulation requires a geometric model of the business components, and an appropriate animation to be displayed to indicate its correct usage.
4. Modelling tools for conceptual process models, and their importation and exportation to other systems. Along with the geometric model components, the user interface for model development must be intuitive and able to make the task of modelling easy for a business analyst.
5. Simulation of instances of process models being executed – via translation of avatars representing human resources. Analogs of 2D simulation methods need to be developed that enable the visualisation of running process model simulations.
6. Ability to annotate models with commentary from clients and from analysts, to assist in the validation process. the clients and analysts need to be able to

leave annotations within the environment that indicate problems with the model for later amendment.

7. Ability to represent statistical information alongside the process model, especially if performing Six Sigma or LEAN process improvement approaches. Similar to the text annotations, statistical annotation data should be incorporated into world for process validation, as examples of actual process executions. In addition, the ability to annotate process models with video and audio information from the analyst's analysis of the actual business in question. This allows the production of multi-media representations of interview and data collection information, opening up new possibilities for higher accuracy models due to the quality of feedback information.
8. Collaboration and remote interactions. The environment should support the ability to collaboratively generate a business process model by multiple analysts, and should allow remote access by the client and other stakeholders in order to communicate and analyse the model of the business for efficient feedback purposes.
9. The ability to treat such VE process models as a form of document. In a similar manner to other process model software systems, this framework should allow for creation of process models as living multimedia documents that can be annotated, shared and distributed by stakeholders in the business process.

An early proof of concept BPMN editor prototype has been developed as an implementation of many of the above items. We now describe its major components.

4 BPMN Open Simulator Implementation

BPMN (OMG, 2006) has been chosen as the grammar to implement in this prototype because of its common acceptance as a conceptual process modelling standard. Furthermore, extensions to previous research performed into 2D versions of BPMN modelling tools can thus be transferred and investigate via the development of a 3D BPMN modelling tool. It should be noted that this framework and implementation can be easily modified to suit another conceptual process modelling framework, as the grammar used for the process model is able to be changed relatively easily. This grammar is purely conceptual in nature, however, there are research efforts focussed on developing executable forms of BPMN. We do not address any executable concerns here (including related data structures and functionality), and so the modelling is restricted to high-level conceptual visualisation and communication.

The intention in this prototype is to show how easy it is to develop such environments using the latest VE technology. However, while a system may provide insight and capacity for modelling of new aspects of business systems, there is a need to consider the user base, which will typically be business analysts that do not have high levels of technical skills.

An example of the uptake of such environments by non-technical personnel is found with the Second Life user population. Of the approximately 80,000 users that

may be on the world at any time, a large number are non-technical laity, that have taught themselves to use the environment (Linden, 2009). We argue therefore, that a well designed VE Conceptual Modelling tool is well within the reach of a typical business analyst with a modicum of computer skills.

A short introduction to VEs is now shown, to indicate their capacity for such modelling tasks.

4.1 3D Virtual Environment Technology Introduction

Virtual Environments have emerged as a powerful software technology for supporting collaborative 3D work and entertainment environments (Burdea and Coiffet, 2003). These environments are typically client server systems that support the creation of, and full interaction with, shared and collaborative 3D dimensional spaces.

These 3D spaces allow the creation of geometry representing many things experienced in reality, including buildings, terrain, forms of transportation, weather models, amongst other things. These objects are represented using data structures known as geometric meshes, which have images laid over them known as textures to give the objects visual appearances approximating real object properties (eg. stone, wood, glass). Human modellers can use VEs to model such objects very easily (refer to Figure 3).

One of the most powerful components of such worlds, exploited by the new framework, is the ability to attach programs to the objects, known as scripts, to imbue objects with extra functionality. A major component of the functionality in the new BPMN editor is implemented using such a scripting process (refer to Figure 5).

In addition, one of the most useful facilities available within a Virtual Environment is the representation of the user as an Avatar, the user ego centre so to speak, that allows the humans to have a spatial presence in the Virtual Environment.

This paper shows that this 3D modelling approach is not too large an extension to present toolsets, and thus enables business analysts to utilise such tools with minimal training.

4.2 BPMN Modelling System Components

A lot of Virtual Environment systems may be used for this project. With reference to the functional requirements, the Open Simulator (OpenSimulator, 2008) and Second Life Client (Linden, 2008) were chosen to implement this prototype. The following details the major reasons:

1. Open Simulator is an open source virtual world server (BSD Licence) which works on many hardware platforms and is compatible with the Second Life virtual world viewer. This allows the implementation of such modelling systems on laptop systems that can be used in the field by analysts. In fact, the demonstrator in this paper was developed on a standard Apple MacBook.
2. Using the Second Life open source viewer provides a set of easy to use tools for content development, well suited to non-technical users who are a large

component of the Second Life community, thus facilitating its usage by business analysts (refer to Figure 3);

3. Open Simulator has a powerful scripting language, which can be extended by the use of C# language technology, thus enabling modelling environment interactions, such as process model linking (refer to Figure 5).
4. Open Simulator has been used in a successful executable process modelling system that uses the YAWL workflow package (Brown and Rasmussen, 2009). So Open Simulator had already proven its capabilities.



Figure 2 Showing the major 3D constructs present in the diagrams. From the left, they are, a flow link ball (for long bent flows), activity node, gateway node, event node and a control script embedded in a sphere. The other small items are the components of the BPMN graph flow connectors.

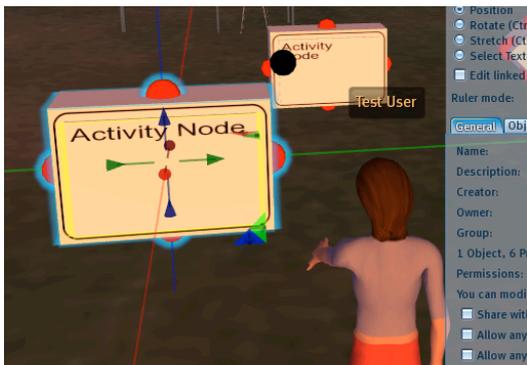


Figure 3 Illustration showing the tools used to position and edit BPMN model objects, in this case an activity node.



Figure 4 Illustration of using texture images to annotate BPMN objects, in this case to create a parallel gateway from a generic gateway.

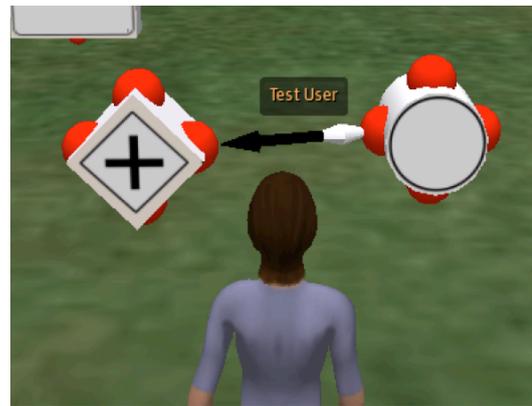


Figure 5 Illustration of linking BPMN process model components, in this case with a Conditional Flow between an event and a Gateway. Links are formed by clicking on pairs of red spheres.



Figure 6 An illustration of the ability to insert textual annotations into the process model. Stakeholders may leave Comment Walls (left) or may use web services to show process statistics (right)

To meet the task and functional requirements, we have developed the following:

1. Object models for the 3D environment to represent the BPMN notation in world – these are Events, Activities, Gateways and Flows. Each have been given a simple 3D representation, and have been modelled with connection nodes for linking as a graph. Each can be dropped into the world from the inventory owned by the user in world, giving an easy to use modelling approach (refer to Figure 2 and Figure 3).
2. Textures for the different annotations – thus developed a complete image database for the annotation of the basic BPMN nodes with image information, as shown in Figure 4.
3. Flow Connection - as the modeller is in essence a graph drawing tool, there is a need to easily create the model flow connections, as per Figure 5.
4. Avatar instance animations. A human resource simulation model is required to show simulations of the processes being executed. This facilitates validation processes by providing an easy to use interface for running simple simulations of instances of processes being executed by an involved stakeholder, to support validation processes with the client (refer to Figure 9).

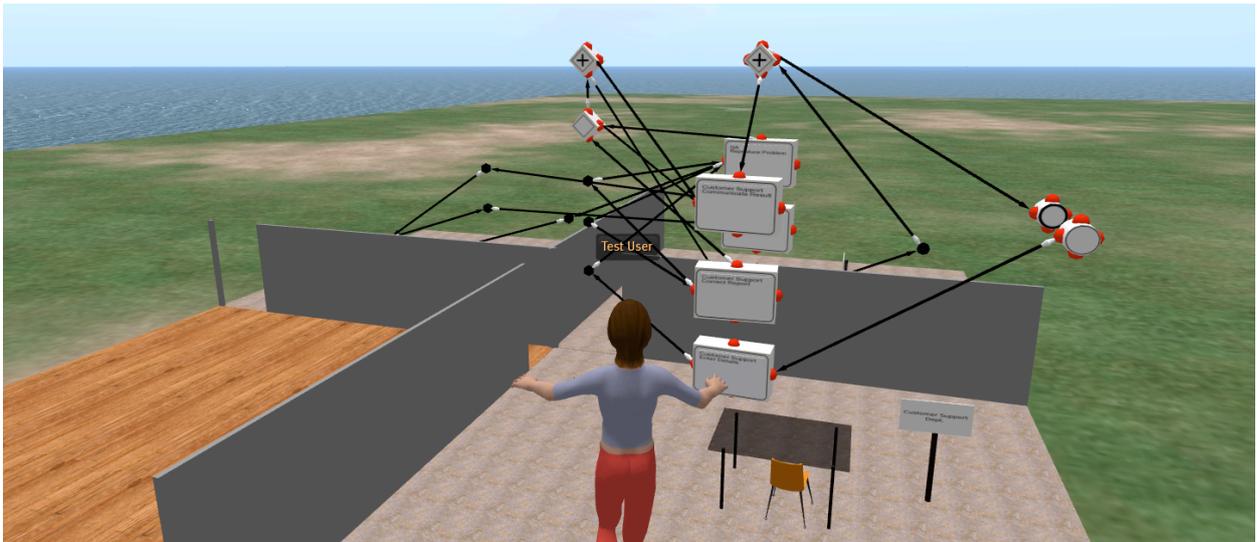


Figure 7 Illustration overview of the full 3D BPMN Software QA example in a Virtual Environment.

5. Annotations are implemented as signpost objects that can be left near the process, and have text inserted to show commentary by various clients who have viewed the model and wish to comment on the validity of the model. Video and audio interview data can be inserted into the world and embedded near the BPMN places of importance, for confirmation purposes, as per Figure 6. Such an approach enables the environment to become a 3D spatial database for the visual mnemonics of video, text and audio information.

- Representations of Human Resource Roles;
- Relationships between other aspects of the physical environment;

5 Software Company Case Study

To illustrate the major features of this approach and prototype implementation, we have developed a 3D BPMN diagram for a Software Quality assurance process. The process involves a description of the interactions of three quality control departments within a software company – Customer Support, QA and Development – and how they interact to deal with software quality issues reported by users.



Figure 9 Showing the room with the Customer Support Activities. Note the sitting avatar, representing a Human Resource performing work.

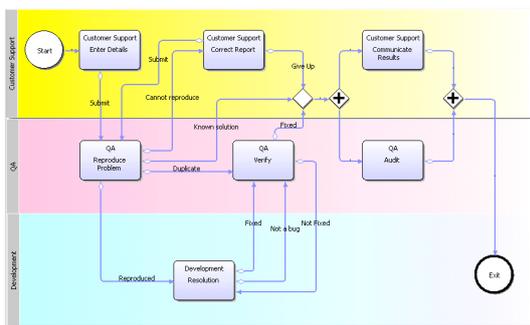


Figure 8 Example BPMN Model for a software quality process (Swenson, 2009). Each swim lane represents a department in the software organisation.

Spatial locations for the processes being enacted are represented in this approach using 3D coordinates. BPMN and other forms of process modelling have a swim lane model of activity representation (OMG, 2006), here we extend that to an actual spatial locations and groupings with reference to the natural spatial structures used in the business, in a full 3D, rather than 2D manner (refer for an overview of the diagram to Figure 7).

We see in the following scenes (Figure 9 and Figure 10) that work is divided up into three departments in three different locations in the building. This is illustrated with a set of rooms aligned with a floor layout.

We use the case study to highlight a number of possibilities with visualisations:

- Spatial Locations;

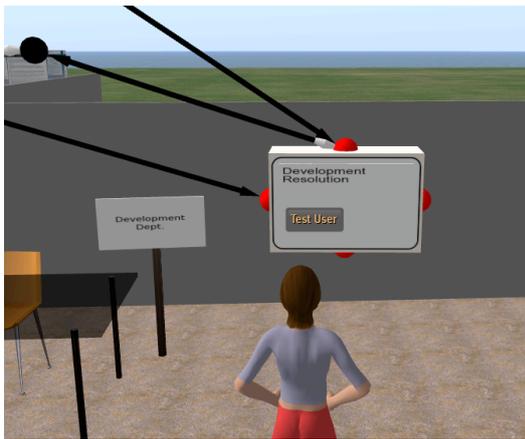


Figure 10 Activities in the Development Department for the example process model.

At each location, the activity is embedded for reference. With this example, the client is thus able to make a direct comparison of what is represented conceptually with a physical model of the environment, to assist the client in mapping conceptual constructs with their more physical understanding of the process. They can use the physical simulation as a mnemonic to help validate a business process model.

Representations of the roles of people doing their tasks can be shown to a client, thus adding insight into the reasons why things are performed in a certain manner, and why certain tasks are either performed badly, or are performed well. This is also useful with process improvement scenarios, where a process model can be labelled or colour coded “as-is” and “to-be”, in order that the client can easily see the change, in a manner similar to 2D modelling systems.

Again, the client can verify whether the business analyst has correctly captured the tasks being performed, so along with the physical location, the client can give feedback on the business analyst on the validity of the process model, from an activity and role based approach using the annotation markers if necessary.

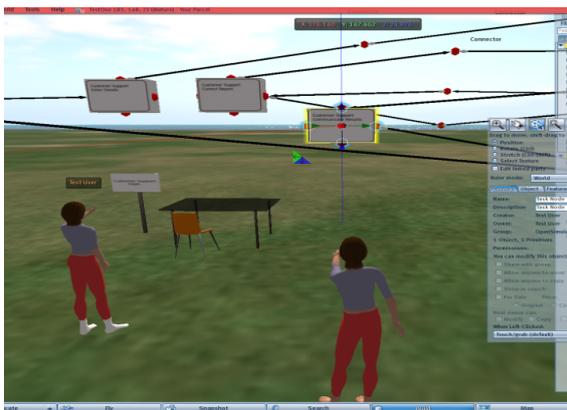


Figure 11 Illustration of collaborative capabilities of VEs, in particular, the ability to collocate avatars, and refer to objects in a natural spatially oriented manner. Both avatars here are discussing and manipulating the activity object (highlighted).

In summary, communication processes can be enhanced by the use of such tools due to their interactive nature. A model of the business can be shown to the client by an analyst, with the added advantage of it being interactive. Therefore, any number of potential scenarios in the business can be visualised and demonstrated with full control, facilitating dialog by allowing clients to ask questions and to step through the environment to examine particular details, to make them clearer.

The other major benefit of such 3D environments is the ease of network communication offered by the use of 3D Avatars. Compared to video collaboration, where the relative location of the client to what they are manipulating is not apparent, 3D VEs represent the person in the dialog, juxtaposed alongside the process model items being discussed. Coupled with a audio chatting capabilities, VEs offer powerful process communication capabilities, especially when dealing with spatially oriented conceptual process models. An example of collaborative modelling is presented in Figure 11.

6 Conclusions

In this paper we have outlined an approach to the use of 3D Virtual Environments as an annotation of process models for process model communication.

We have outlined the motivations and general requirements for the development of 3D Concept Model representations, and have developed a proof of concept BPMN Editor in Open Simulator, which provides most base functionality required to create usable BPMN diagrams in a 3D world.

As this is a first incursion into this research space, we see that there are many extensions to this research work that need to be explored in order to make the general approach and the software that much more usable.

The system successfully implements many of the ideas presented, but needs to be extended in a number of areas.

There needs to be a development of a message and event animation component, to clearly show the actions of the other event and timer constructs within the process model.

Suitable export and import tools need to be developed for integration with other BPMN tools. The 3D Environment can be used as part of a constellation of tools to support the development of 3D Process Model visualisations.

Finally, there needs to be an analysis of benefits via experimentation of insight into process systems. While there is a good argument for the use of visualisation systems in the communication of process models, qualitative and quantitative experiments need to be performed to establish the potential of such environments within the process modelling domain.

7 References

- Brown, R. & Paik, H.-Y. (2009) Multi-faceted Visualisation of Worklists. *Journal on Data Semantics XII*, 153-178.
- Brown, R. & Rasmussen, R. (2009) Virtual Environment Visualisation of Executable Business Process Models

- (accepted). IN RAO, R. (Ed.) *Virtual Technologies for Business and Industrial Applications: Innovative and Synergistic Approaches*. Hershey, USA, IGI Global Press.
- Bunge, M. A. (1977) *Treatise on Basic Philosophy Ontology I - The Furniture of the World, vol. 3.*, Dordrecht, The Netherlands, Kluwer Academic Publishers.
- Burdea, G. & Coiffet, P. (2003) *Virtual Reality Technology*, New Jersey, USA, Wiley.
- Casewise (2008) CaseWise, www.casewise.com, Accessed October 2008.
- De Leoni, M., Van Der Aalst, W. & Ter Hofstede, A. (2008) Visual Support for Work Assignment in Process-Aware Information Systems, Proc. *Business Process Management*, 67-83.
- Effinger, P., Kaufmann, M. & Siebenhaller, M. (2009) Enhancing Visualizations of Business Processes, Proc. *Graph Drawing*, 437-438.
- Gallagher, A. G., Ritter, E. M., Champion, H., Higgins, G., Fried, M. P., Moses, G., Smith, C. D. & M., S. R. (2005) Virtual Reality Simulation for the Operating Room Proficiency-Based Training as a Paradigm Shift in Surgical Skills Training. *Annals of Surgery*, 241, 364-372.
- Green, P. & Rosemann, M. (2004) Applying Ontologies to Business and Systems Modeling Techniques and Perspectives : Lessons Learned. *Journal of Database Management*, 15, 105-117.
- Ibm (2008) Innov8 Web Site, <http://www-304.ibm.com/jct03001c/software/solutions/soa/innov8.html>, Accessed March 2008.
- Interactive-Software (2004) Interactive Software Systems, <http://www.interactive-software.de/>, Accessed 27th August 2004.
- Jörg Becker, Martin Kugeler & Rosemann, M. (Eds.) (2003) *Process Management: A Guide for the Design of Business Processes*, Berlin, Springer-Verlag.
- Linden (2008) Second Life, www.secondlife.com, Accessed October 2008.
- Linden, T. (2009) The Second Life Economy - Third Quarter 2009 in Detail <https://blogs.secondlife.com/community/features/blog/2009/11/02/the-second-life-economy--third-quarter-2009-in-detail>, Accessed November 2009.
- Muehlen, M. Z. & Recker, J. (2008) How Much Language is Enough? Theoretical and Practical Use of the Business Process Modeling Notation, Proc. *20th International Conference on Advanced Information Systems Engineering (CAiSE 2008)*, Montpellier, France, 465-479.
- Omg (2006) Business Process Modeling Notation Specification, www.bpmn.org, Accessed 29th March 2008.
- Onmap (2009) OnMap, <http://www.onmap.fr/>, Accessed August 2009.
- Opensimulator (2008) Open Simulator, http://opensimulator.org/wiki/Main_Page, Accessed October 2008.
- Pamplin, J. A. & Zhu, Y. (2004) Design and Implementation of a Workflow Rendering Engine. *International Conference on Modeling, Simulation and Visualization Methods (MSV)*.
- Recker, J., Indulska, M. & Green, P. (2007) Extending Representational Analysis: BPMN User and Developer Perspectives, Proc. *Business Process Management*, 384-399.
- Rosemann, M., Recker, J., Indulska, M. & Green, P. (2006) A Study of the Evolution of the Representational Capabilities of Process Modeling Grammars. *Advanced Information Systems Engineering*, 447-461.
- Schonhage, B., Van Ballegooij, A. & Elliens, A. (2000) 3D gadgets for business process visualization—a case study, Proc. *Virtual Reality Modeling Language Symposium*, Monterey, California, United States, 131-138.
- Stefanie Betz, Daniel Eichhorn, Susan Hickl, Stefan Klink, Agnes Koschmider, Yu Li, Andreas Oberweis & Trunko, R. (2008) 3D Representation of Business Process Models. *MobIS*, 73-87.
- Swenson, K. (2009) Trouble Ticket Scenario www.xpdl.org/nugen/p/troubleticketscenario/public.htm, Accessed August 2009.
- Treinish, L. (1999) Task-specific visualization design. *Computer Graphics and Applications, IEEE*, 19, 72 - 77.
- Tufte, E. (1983) *The Visual Display of Quantitative Information*, Cheshire, USA, Graphics Press.
- Van Der Aalst, W. (2004) Business Process Management: A Personal View. *Business Process Management Journal*, 10, 248-253.
- Van Der Aalst, W. M. P. & Ter Hofstede, A. H. M. (2005) YAWL: Yet Another Workflow Language. *Information Systems*, 30, 245-275.

Towards A Comprehensive Requirements Architecture For Privacy-Aware Social Recommender Systems

Shan Chen

Mary-Anne Williams

Innovation and Enterprise Research Laboratory
 Centre for Quantum Computation and Intelligent Systems
 Faculty of Engineering and Information Technology
 University of Technology, Sydney
 Email: shanc|mary-anne@it.uts.edu.au

Abstract

Social recommendations have been rapidly adopted as important components in social network sites. However, they assume a cooperative relationship between parties involved. This assumption can lead to the creation of privacy issues and new opportunities for privacy infringements. Traditional recommendation techniques fail to address these issues, and as a consequence the development of privacy-aware cooperative social recommender systems give rise to an important research gap. In this paper we identify key problems that arise from the privacy dimension of social recommendations and propose a comprehensive requirements architecture for building privacy-aware cooperative social recommender systems.

1 Introduction

Content-based filtering (Billsus & Pazzani 2007) vs. collaborative filtering (Goldberg et al 1992) have been dominating the traditional recommender systems. Typically, users are classified by their interests and/or preferences based on some similarity measures. Grouping using these approaches connects users to each other, implicitly or explicitly. Such connections can create new and reveal social contexts for users, and can make the system prone to privacy breaches. On the other hand, the development of on-line social networks has addressed the need to support social recommendations, i.e., provide users the ability to introduce people (i.e., friends) to others or to offer social referrals between users to facilitate network consolidation and expansion. However, such recommendations are either based on existing connections or created using social factors in individual's personal space. For example, many *social network sites* (SNS) - e.g., Facebook (2009), LinkedIn (2009) and Pulse (2009) - provide a list of "People you may know" for users to build and/or expand their networks, or to invite people who were previously unknown. People being introduced in this way are either socially connected explicitly, or identified based on the similarity of some attributes (e.g., interests, geography location, occupation, etc.) This kind of recommendation can inject many privacy issues for parties involved, and as a consequence increase the opportunities for privacy infringements.

In the problem domain of social recommendation, we argue that the privacy issues arise due to the lack of *choice* offered to users as to whether they want to

be introduced or referred - i.e., *consent* - and the ability to *control* "who knows what about me". The problem of *choice* concerns users' rights to choose their preferences and give *consent*. The *control* problem has two important aspects: the *who* and the *what* dimensions - the latter concerns "things about me", while the former concerns "things about others". As a consequence, concerns about users' rights involve both "me" and "others". Privacy issues related to the *control* problem highlight the balance of rights between users. By identifying key problems that arise from the underlying *rights* (i.e., choice, consent and control), this paper studies privacy requirements and proposes a requirements architecture for building privacy-aware social recommender systems.

The rest of this paper is organized as follows. Section 2 studies the problem domain, section 3 identifies architectural requirements, section 4 investigates privacy issue in social connections, section 5 establishes social connection privacy preserving requirements, section 6 describes system requirements, and section 7 presents a discussion and future work.

2 Problem Domain

This section describes a motivating example that reflects a social recommendation service - i.e., *People you may know* (PYMK) - which has been offered by many popular social network sites such as Facebook, LinkedIn and Pulse. It then discusses problems arise from the motivating example to uncover the fundamental privacy problem in social recommendations.

2.1 Motivating Example

Mary joined a social network on MySN site and she received a PYMK list, on which those who went to the same school as her and those who share the same type of profession as her are listed, allowing her to send a message to them. Mary was surprised to find many school friends she had lost contact with on the PYMK list. However, she also felt compromised to be on the MySN because she wanted to keep her professional information disjoint from her personal social network, and to keep her away from those professionals she did not want to network with. She reasoned that if she saw other's information they could also see hers.

2.2 The Right Problem

From the motivating example, above, we can see that if Mary was not asked if she wanted to be on the PYMK list that appeared to others (e.g., to the public or to specific targeted groups), then she had no way of choosing preferences and giving permissions to control her information privacy. We refer to this problem the user's *right of choice*. If choice is offered, the ability to consent on the usage of information is then required

Copyright ©2010, Australian Computer Society, Inc. This paper appeared at the Seventh Asia-Pacific Conference on Conceptual Modelling (APCCM 2010), Brisbane, Australia, January 2010. Conferences in Research and Practice in Information Technology (CRPIT), Vol. 110, Sebastian Link and Aditya K. Ghose, Ed. Reproduction for academic, not-for profit purposes permitted provided this text is included.

to control the information. In other words, the user's *right to consent* and *right to control* their information are essential to fulfill privacy requirements. In this light, the privacy problem in social recommendations mainly involves users' rights of *choice*, *consent* and *control*. We refer to these rights the *3C Rights* (3CR) framework and describe them as follows:

- *choice* - the ability to choose to-be or not-to-be introduced or referred;
- *consent* - the freedom to give permissions of personal information usage to others; and
- *control* - the power to control personal information and ways of sharing it.

The 3CR framework can provide a specification of higher-level privacy requirements to the recommender provider. To fulfill these requirements, lower-level detailed requirements for recommender system implementation are required. To discover fundamental problems behind the 3CR framework, we analyze the interplay between each right of the 3CR using scenarios educed from the motivating example above.

2.2.1 Choice

Scenario MySN adds Mary to the PYMK list with people that share the same type of profession as her and then presents the PYMK list to her boss who is on the MySN. Mary's boss sends a friending request to her and she adds her boss upon his request because she does not want to be impolite. However, after her boss joins her social network, Mary fails to maintain the disjointedness of her personal social network and professional social network.

Problem The user's choice of being introduced or referred will have an impact on his/her information privacy. In the light of 3CR, a shortage of the right to *choice* naturally leads to a deficiency of rights to *consent* and *control* information.

2.2.2 Consent

Scenario MySN offers recommendation choice options: to-be-recommended or not-to-be-recommended. Mary wants to expend her social networks but does not want her boss to be in her social network on MySN. She knows that her boss is on MySN. If she chooses the option to-be-recommended then her boss will know of her existence and might request friending. But if she chooses not-to-be-recommended then she will loose the opportunity to be known to potential social contacts.

Problem Having binary choice options is insufficient for supporting consent and leads to failures of information control because social relationships are not binary: friend or not. Users should have the freedom to give different permissions to different contacts.

2.2.3 Control

Scenario The new version of MySN allows Mary to specify who she will not be recommended to. Mary believes on MySN she can now stay away from her boss because she has specified the name of her boss not to receive recommendation about her. However, two days later she receives a friending request from her boss. She does not know that her MySN new friend Phoebe is her boss's little daughter who shares her online experience with her father.

Problem Even though Mary has sufficient *rights of choice and consent*, she does not have sufficient power to control what information is made available to her contacts - i.e., ways of sharing in the network.

It can be seen that, the magnitude and dimension of 3CR as well as the interplay between the three rights have a major impact on the privacy. We describe the magnitude of 3CR in terms of *3CR values*:

- the range of available choice options,
- the type and detail of consent the user can set, and
- the level of power the users have in controlling his/her information.

These 3CR values and the ways they interoperate can lead to different impacts of the information usage and in turn the privacy. Since the problems reflected in these values are closely related to social problems in social networks, to gain an insight into their privacy implications, we study the related social problems in the next sub-section.

2.3 The Social Problem

Given that social interactions are fundamental activities in social networks and interactions are based on social connections, the context of a social network is framed by social entities (e.g., users) and relationships connecting them. In this light, fundamental to the privacy problem in social networks is philosophy of social relationships - i.e., the relationship privacy is attributed as the primary privacy problem in social networks. Processed in social networks, social recommendations inherit the philosophical privacy problem - i.e., relationship privacy as social recommendation privacy.

Relationship privacy involves several problems that needs to be addressed. In light of the 3CR, these problems typically are:

- the selection of potential parties - who can be considered as appropriate candidate(s) to network with; and
- the selection of specific information to share - who can share some certain information with and in what way.

It has been evidenced that different networks tolerate different connections, reflected in properties such as types, degrees, directions and multiplex (Chen & Williams 2009). The way these properties cohere to balance users' rights in the 3CR space provides a key to the preservation of users' information privacy in social recommendations. However, the dynamic of social networks gives users no way of knowing the status of these properties. Consequently the 3CR problem in social networks leads to several operational issues described in the next sub-section.

2.4 The Operational Problem

Information privacy requires users be aware of the current status of the social network in which they interact with others. One way to address this problem would be to allow users to query the network about self and others. However, to promise a balance of rights between users, queries cannot return comprehensive information to the user that violates others' privacy. On the other hand, queries can potentially reveal the user's privacy because they reflect the querier's intentions. Accordingly, privacy-aware

queries need to be constructed with consideration to the following issues.

- content - i.e., *what information can be retrieved* such that the maximum information can be obtained without violating privacy; and
- behavior - i.e., *how to query* such that the querier's intentions that can reveal or be used to infer privacy are not disclosed while at the same time necessary information can be obtained as complete as possible.

While the social problem concerns current status of a social network, the operational problem gives consideration to the dynamic aspect of the network - i.e., the evolution of the network. This can be reflected in both the content issue and the behavior issue taking privacy implications into account upon each operation. The key to uncover privacy implications in evolving social networks is to learn potential ways the user connects to others - i.e., possible relationships that can be established. Since each candidate is a social entity playing specific social roles in the network, they can have different impacts on their social connections and in turn impact the privacy of those connected to them.

2.5 Summary

The problems described above suggest the privacy problem domain in social recommendations can be divided into *choice*, *consent* and *control* issues, with the core in *relationships*. Consequently it requires an adaptable and extendable choice space, rich relationship semantics, and privacy-aware queries. In the subsequent sections we identify low-level challenges and fine-grained requirements to address these problems.

3 Architectural Requirements

3.1 Choice, Consent & Control

Our architecture is based on three core pillars: choice, consent and control (Williams 2009). Users are given *choice to consent* and to *control* their information privacy. To create consent for recommendations users need to be able to accurately express their needs. In addition, preferences for wishes and interests are preferable because they help to determine users' intentions and in turn privacy management decisions. Since people's desires, wishes and interests are highly situated and can be multiplex, to accurately capture preferences adaptability and extendability of choice are essential. To this end, a capable privacy-aware recommender will provide users with the following:

- *choice options* that allow them to
 - express their needs and preferences accurately, and
 - adjust and change their needs or preferences to new conditions.
- *consent mechanisms* that enable them to
 - learn the context of their own networks in relation to privacy implications, and
 - specify permissions for using their data and obligations attached to the usage.
- *control devices* that provide them ability to
 - control ways of sharing personal information, and

- verify expected controls.

It can be seen that, the ability to preserve privacy largely depends on the power to control information usage. To achieve this, comprehensive guidelines for privacy protection are essential. The set of principles for privacy protection identified by the Organization for Economic Cooperation and Development (OECD) (OECD 2009) is a good candidate because they represent as far as possible a global consensus.

3.2 OECD Privacy Principles (OECD_PP)

The eight principles for privacy protection identified by OECD are as follows:

1. Collection Limitation (CL) limits the collection of personal data.
2. Data Quality (DQ) ensures personal data is relevant to the purposes of used.
3. Purpose Specification (PS) restricts the collected data to the purposes of collection.
4. Use Limitation (UL) restricts data to be used within the permission of the purpose specification.
5. Security Safeguards (SS) ensures data is protected by safeguards.
6. Openness (OP) ensures policies with respect to personal data are open to the user.
7. Individual Participation (IP) ensures individual rights of actions related to own personal data.
8. Accountability (AC) ensures the principles above are complied.

Based on the notion of the 3CR, this set of eight principles are categorized into the 3CR groups and serve as a higher-level guideline for specifications of layer requirements. Fig. 1 shows a two-layer requirement architecture.

3.3 OECD_PP in Recommendations

The increasing number of privacy breaches reported in the media almost everyday has demonstrated that, from a users' perspective, there is insufficient support for the principles of Purpose Specification and Use Limitation in existing SNS. One might argue that these SNS do provide limited access control support. However, users are not made aware of nor do they have the ability to specify the purpose and usage of their information being collected. We argue that these two principles dominate social recommendations because:

- *The Purpose Specification Principle* restricts the use of the collected data to the purposes of collection - it concerns the consistency of data usage and the purpose for which they were collected. For example, relationship information was collected for the purpose of sending social recommendations - i.e., determination of recommendation target/candidates, or personal information was collected for sending social recommendations and not for other types of recommendations like buying or selling.
- *The Use Limitation Principle* restricts data to be used within the permission of the purpose specification - it concerns deviations from specified purposes. For example, the problem of inconsistency where a social relationship exists for the purpose of social interactions, e.g., a religious relationship is for religious interactions and not for trading interactions.

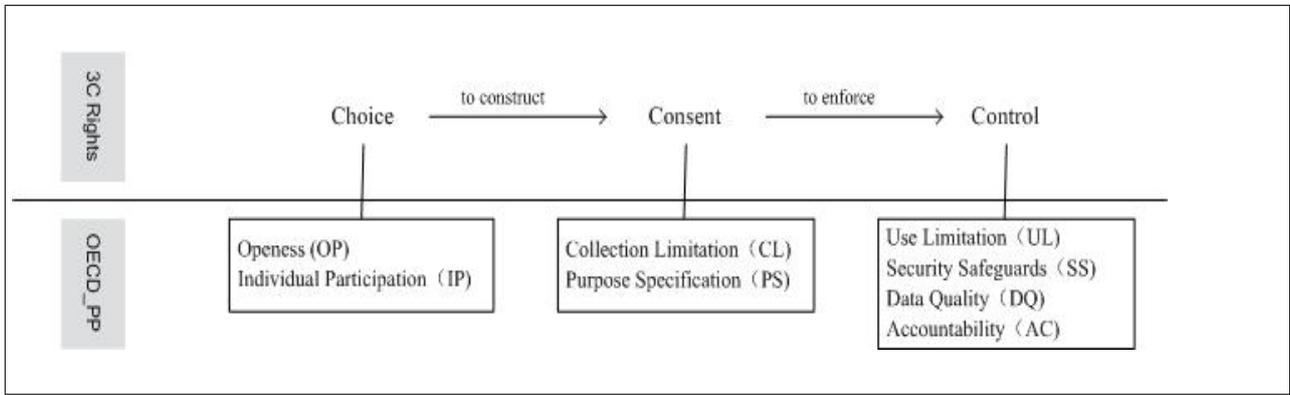


Figure 1: Two-Layer Requirements Architecture

3.4 Layered Requirements

For the above reasons, we aim to address the problem of users' awareness with respect to the principles of PS&UL in the problem domain of 3CR - in this light, Users' awareness is about:

- self information and ability (i.e., power to control their information on the SNS); and
- others' information relevant to his privacy - i.e., users' knowledge about the social context in which he lives, i.e., users' own social networks.

To this end, it is necessary to provide users mechanisms to query the network to alleviate their concerns and to increase their awareness, i.e., *query-answering* as an essential component of the recommender. With the focus on the principles of PS&UL, query-answering addresses the problem of user awareness with respect to privacy. In this regard, the following requirements are established:

- at the 3CR layer, users are provided *choice* options and *consent* to important aspects of *control*.
- at the OECD_PP layer, users are provided conceptual guidelines on what needs to be taken care of in terms of constructing consent and enforcing control.
 - choice, the recommender allows user interactions (IP) for query-answering based on the provided options (OP);
 - consent, the recommender allows user consents set on the collection (CL) and the usage (PS); and
 - control, the recommender allows user information control in usage (UL).
- at the Codes of Practice (CoP) layer, users learn to situate themselves to better establish controls using queries that are based on the set of CoP - in this paper, learning is mainly used to address the problem of social connections, i.e., relationship privacy.

It can be seen that, the CoP layer is a new layer added to the two layered requirements. The purpose of layering requirements is to allow externality (in opposition to “pure” system requirements) input such as law and social norms as higher-level conceptual guidelines for system design situated at a lower-level where CoP applies. The role that CoP plays requires fine-grained requirements of *social connection* and *relationship privacy* be understood and specified.

4 Privacy in Social Connections

4.1 Social Connection and Relationship

To learn the meaning of relationship privacy, we begin from intrinsic properties of a relationship. As we have identified in (Chen & Williams 2009) relationships are multifaceted and can be symmetric or asymmetric. A relationship is symmetric if both ends of the relationship share the same attitude of the relationship, i.e., recognize the relationship under the same conditions; otherwise, the relationship is asymmetric. For example, the relationship between *A* and *B* is symmetric if both set the relationship to the same type under the same conditions. The relationship is asymmetric if *A* sees *B* as a friend but *B* sees *A* as a colleague. The asymmetric property implies the existence of *direction* in a relationship. On the other hand, the same pair of social entities can hold more than one type of relationship. For example, *A* and *B* are siblings, classmates and both are members of club A&B. The connection between *A* and *B* is therefore described as “sibling, classmate and A&B member”.

To better understand the privacy issues that can arise between two social entities, we distinguish the concept of relationship and the concept of social connection. A *social connection* indicates two social entities are connected by some reason. Each involve the connection of two social entities in a *relationship*, which has a type and a direction. A social connection is *multiplex* if there is more than one relationship held between the two entities on the connection.

Social referrals naturally introduce indirect social connections. When one entity connects to another via a referral, the connection is indirect. The concept of *connection degree* is used to indicate the distance between two entities. For example, if *A* connects to *B*, and *B* connects to *C*, then *A* is said to be 2 degrees away from *C*, i.e., the connection degree between *A* and *C* is 2. If there are multiple paths connecting *A* to *B*, technically the degree of *A* and *B* is the length of the shortest path between *A* and *B*.

4.2 Privacy Concerns

In the context of social recommendations, we study what can have an impact on a relationship in a specific context. In the following we begin with a set of scenarios elicited from a social referral.

Assume that on MySN everyone can refer their social contacts to each other. In this example *M* refers *S* to *X*. Scenarios where privacy concerns arise might include the following:

- If *M* introduces *S* to *X* explicitly, then possible privacy costs from such a referral include:

- X knows “ M knows S (the latter may not know the former)” - the existence of the one-way relationship is disclosed;
 - X knows “ M and S know each other (i.e., there is a relationship between them)” - the existence of the two-way relationship is disclosed; or
 - X knows “ M and S are connected by R reason (e.g., working in the same department)” - the type of relationships is disclosed.
- If M refers S to X implicitly by i) anonymity, ii) using a different ID, or iii) via a third-party (i.e., the system or another social entity), if these features available, then privacy costs from such a referral could be:
 - X knows “someone that knows S does not want to be known to X ” (by anonymity);
 - X knows “the owner of the ID knows S ” (by a different ID), or
 - X knows “the third-party knows M does not want to reveal his relationship with S to X ” (by a third-party).

In any of these cases, the referrer (i.e., M , or the third-party) knows that S is known to X .

4.3 Relationship Privacy

Relationship privacy concerns illustrated in this example can be encapsulated in terms of *existence* and *relationship type*:

- *Existence* concerns whether one has a relationship with another. For example, M refers S to X implicitly because of this concern. On the other hand, did M know if S had the same concern?
- *Relationship type* concerns the kind of relationship between two social entities. For example, sibling, friend and colleague are different relationship types.

On the other hand, social entities in a social network are interconnected. Social entities can connect directly or indirectly. A consequence of relationship privacy between the two social entities who hold the connection and their social contacts, existing or potential, is extended to *access control* and *distribution control*:

- *Access control* concerns the kind of social contacts that can be granted access to certain information, or the condition of building a relationship with others that can grant access to the information (recourse). In the context of social recommendation, relationship privacy has three main relationships:
 - the relationship between the recommender agent and the recommended agent,
 - the relationship between the recommended agent and the recommendation recipient, and
 - the relationship between the recommender agent and the recommendation recipient.

In this paper, we give consideration to the recommended agent - i.e., we consider the recommended agent as a privacy priority agent such that *access control* is the recommended agent's power to control his information being accessed by the recommender agent (i.e., to disclose information about the recommended agent) and

the recommendation recipient (i.e., to know information about the recommended agent). For example, does M know if S is willing to make a connection to social entities like X (e.g., X is a member of group GA and if S is concerned about his privacy when having a relationship to GA).

- *Distribution control* concerns the kind of social contacts knowing (i.e., the *existence concern* and the *relationship type concern*) or having access control to some information can also grant permission to distribute information under certain constraints - e.g., before or after accepting the recommendation (i.e., accept S as a social contact for some purpose), can X refer S to others? Can X disclose the relationship between M and S (e.g., when referring S to other social entities)? On the other hand, X also got to know something about M (if not before) - can X distribute M 's information (e.g., refer M to others - i.e., reveal the existence of M - and the relationship between them)?

When each of these concerns on each dimension of a social connection are evident, the privacy issue tends to be multi-layered. Consider the referral example above, If M makes the referral explicitly, the *existence* concern on each dimension extends to

- *direction* - can X know if S and M holds a symmetric or asymmetric relationship? In the case of an asymmetric relationship, who is the dominant partner in the relationship?
- *multiplex* - can X know if S connected to M in various ways and how are they connected?
- *connection degree* - can X know if S is a direct contact of M ? If not, how many degrees away?

It can be seen that these concerns on each dimension can take the problem to a level where more sensitive and negative implications can be discovered. This suggests multi-layer relationship privacy requirements (RPR) for connection-driven social recommenders. The multi-layer RPR serves as a guideline for establishing the codes of practice.

5 Social Connection Privacy Preserving Requirements

Given that our aim is to preserve relationship privacy in social recommenders, the *codes of practice* (CoP) focus on the semantics of relationship privacy on four dimensions categorized as *disclosure* and *control*.

Disclosure

This category concerns the properties of *existence* (EX) and *relationship type* (RT). Let P denote a property of a relationship privacy in Disclosure, such that $P = EX$ for property “existence” and $P = RT$ for property “relationship type”. Then, the CoP for Disclosure are as follows:

- Direction ($P:D$)
 - The disclosure of property P of a symmetric relationship should only be made with the explicit consent of both parties of the relationship.
 - The disclosure of property P of an asymmetric relationship should be made with the consent of the dominant party. If $P = EX$, the consent includes the existence of the relationship and its direction.

- Multiplex ($P:M$)
 - The disclose of property $P = EX$ of each relationship in a multiplex connection should only be made with the consent of the dominant party of each relationship¹.
- Connection Degree ($P:CD$)
 - The disclose of property P of an indirect relationship should only be made with the consent of both parties of the relationship. Such consent includes the value of connection degree and the social entities connected on the path between two ends of the relationship.² If $P = RT$, the consent involves parties on the relationship that RT refers to.

Control

This category concerns the properties of *access* (AC) and *distribution* (DT). Let P denote a property of a relationship privacy in Control, such that $P = AC$ for property “access” and $P = DT$ for property “distribution”. Permissions for control of property P on certain information of an agent can only be granted upon the consent of the agent. Within the scope of a social recommendation, if

- $P = AC$
Permission for agent A to introduce agent B to agent C should only be granted upon B 's consent that allows his (B) certain types of social contacts (i.e., A) to introduce him (B) to certain types of social entities (i.e., C). Permission for agent A to establish a relationship R to agent B should only be granted upon B 's consent that allows certain types of social contacts (i.e., A) to connect to him (B) on a relationship type of R .
- $P = DT$
Permission for agent A to distribute certain information of agent B to agent C (e.g., when making recommendation) should only be granted upon B 's consent that allows his (B) certain types of social contacts (i.e., A) to introduce him (B) to certain types of social entities (i.e., C). The kind of the information of interest must be considered for this property in the CoP below.

Let KI be “the kind of the information of interest”, the CoP for Control are as follows:

- Direction ($P:D$) The type of A is determined by the relationship between A and B ; and KI , if applicable. The type of C is determined by the relationship between A and C , and the potential relationship between B and C ; and KI , if applicable. Direction should be concerned if the relationship is asymmetric.
- Multiplex ($P:M$) The type of A is determined by the set of relationships on the connection between A and B ; and KI , if applicable. The type of C is determined by the set of relationships hold on the connection between A and C , and the potential connection between B and C .
- Connection Degree ($P:CD$) The type of A is determined by the set of relationships held on the set of connections between A and B ; and KI , if

applicable. The type of C is determined by the set of relationships held on the set of connections between A and C , and the potential connection between B and C .

By establishing the CoP above, the layered requirements can be shown in Fig. 2.

It can be seen that, these CoP provide a basis to support consent's construction. Central to the CoP is the problem of “who can see/use what?” (WCS/UW). In the social recommendation problem domain, social entities are identified by their social connections. Subsequently the problem of WCS/UW requires the user not only be able to identify the kind of social entities, but also their connections to the user. This requirement suggests the development of concepts of *abstraction* and *granularity*, where the former reflects the level of detail on social entities - e.g., the abstraction levels of individuals, groups, communities, organizations and networks are from the lowest to the highest, the latter refers to the fineness with which relationship types are categorized on a certain abstraction level - e.g., friends, business partners, family members, classmates, co-workers, etc. In this light, the AC and DT of CoP necessary consider abstraction levels, i.e., groups - i.e., for AC:D, AC:M, AC:CD, DT:D, DT:M and DT:CD, when determining B 's relationship to A and C , criteria should include groups (if any) to which they belong.

6 System Requirements

6.1 Component Requirements

Towards the privacy-aware social recommender system that we propose, components provide functionality to fulfill the requirements established above are: *rights* components, *relationship registry* component and *query-answering* components:

- Rights components: Choice, Consent and Control three components to provide mechanisms for users to express their 3CR;
- Relationship Registry component to store all the relationships and policies;
- Query-Answering components:
 - Query Library to store queries and permissions attached; and
 - Obligation and Permission Reasoner to reason about obligations and permissions.

6.1.1 Choice

The Choice component provides a space for users to establish a basis of recommendation consents. To construct consents for privacy purposes, *choice* requires the following key abilities: *extensibility*, *expressivity* and *adaptability*.

Extensibility: It is essential for the users to accurately express their needs. In the choice space, users are given higher-level options as initial suggestions. Upon selections of these options, users can further detail their needs and preferences if necessary.

Expressivity: Users' requirements that can include short-term and/or long-term needs, and preferences for their wishes and interests, require rich semantics options. From a system design perspective, this requirement shows the need of expressive representations to capture comprehensive requirements.

¹This code is not applicable to the property RT since a relationship is simplex. Thus, $P : M = EX : M$.

²This code is made on the assumption that an indirect relationship is symmetric since otherwise the combination of symmetric and asymmetric leading to the complex consent is out of the scope of this paper.

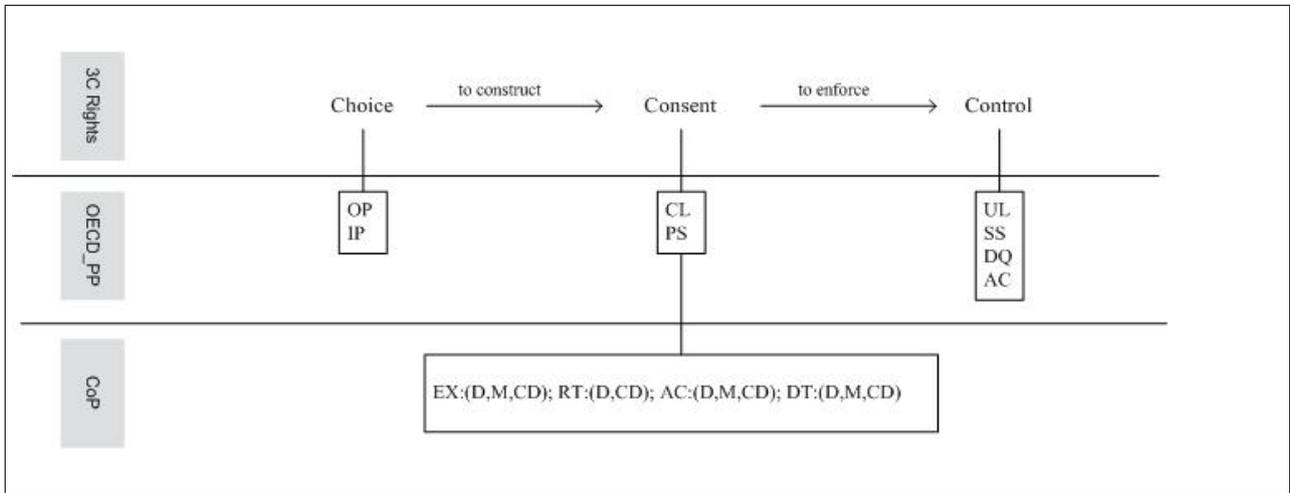


Figure 2: Three-Layer Requirements Architecture

Adaptability: Requirements can evolve over time. On the other hand, with the privacy issue in mind - i.e., relationship privacy (as identified above) - users' not only need to be able to express and control the evolution of their relationships, they may also need to identify their preferences at different level of detail for each relationship. As a result, *adaptability* to allow users to adjust their needs or preferences to new conditions is necessary.

6.1.2 Consent

The Consent component constructs permissions with privacy concerns and obligations in mind. As a prerequisite for *control* - i.e., the output of this component is provided to the Control component to generate controls - it concerns the problem of *fidelity* under the principle of Purpose Specification. As the successor to the Choice in 3C, it transits with the problems of *extensibility*, *expressivity* and *adaptability*.

6.1.3 Control

The Control component manages users' intentions for the privacy of their information. It concerns the problem of *adequacy* under the principle of Use Limitation. As the top level of rights in the 3C, it inherits the problems of *extensibility*, *expressivity* and *adaptability* from its predecessor (i.e., the Consent component).

6.1.4 Relationship Registry

The Relationship Registry component stores all the relationships and associated policies. In this component, *granularity* and *abstraction* are the fundamental requirements and need to be highlighted. The finer granularity relationships tolerance, the more choice the user will have and the more accurate consents can be constructed, which in turn the more fine-grained controls the user will have. This implies *adaptability* and *extensibility* requirements for fine-grained relationships. On the other hand, *abstraction levels* of individuals reflects their relationships to others within a certain scope. This implication not only stresses the need of *adaptability* and *extensibility*, but also requires *scalability* since abstractions can overlap in various degrees - i.e., when one's abstraction level changes, e.g., one can belong to multiple groups where some of these groups are nested and some are overlapped, when the relationship between groups are changed but one's group memberships remain, his relationship

to other members in these group can change. The Relationship Registry is required to be able to adapt and scale to accommodate such changes.

The registry serves as a backbone to support the whole system with central information - i.e., relationships. Each time a new relationship is established, or an existing relationship is evolved, all the related information is required to register with the registry.

6.1.5 Query Library

The Query Library component stores all the queries available and conditions to restrict the kind of users using certain queries. Queries are constructed with the aim to assist users to learn more about their current positions or potential positions in the network in terms of privacy implications - i.e., identifying the choice options available to the user and the CoP for consents construction, queries look for information about the user's control power, and others information related to the user's privacy.

6.1.6 Obligation and Permission Reasoner

To answer queries with maximum information while committing to other users' information privacy preservation - i.e., it is required that the answers to be compliant with these users' privacy policies and not negative consequences. This reasoner serves to fulfill such requirements. It takes users' queries, checks privacy policies of other users that are involved in the answers, then reasons about

- whether the answers are consistent with those policies; and
- whether the querier's obligations and permissions in relation to the queries of interest are compliant with his/her policy and intentions to control own information privacy.

6.2 Overall Functional Requirements

Based on the requirements identified for each component in the previous subsection, the overall functional requirements for a privacy-aware recommender system are illustrated in the form of workflow described based on a social recommendation example as follows:

1. The user is offered two choice options for recommendation: to-be-recommended and not-to-be-recommended. If to-be-recommended is chosen, then the following options are provided.

2. The user is offered choice to query the existing network.

The user has choice to learn about “who is around” - i.e., who is in the network and how they are connected. For social concerns, the user has choice to query about existence of groups and their members on their interests and habits - e.g., who is in his/her profession, who share the same interests, who has something in common (e.g., attending the same primary school), etc. For privacy concern, the user has choice to acquire knowledge about the social connections of those who are of interest, as well as their availabilities in terms of what they can do and what they cannot do in relation to the user’s privacy concern. However, while the user is given choice for querying about the network, rights of social entities of interest to the user must also be taken care of - i.e., these entities’ permissions for use of their information. If answers to the query will not respect to their privacy, i.e., answers conflict with other’s permissions, then such answers should not be given. The Obligation and Permission Reasoner is responsible for checking such conflicts and provide advice in compliance with related entities’ permissions. The CoP are applied to the related entities.

Example

A query could be “are there anyone from my company?” or “anyone in Cooking group working in my company?”. If a member of the Cooking group is working for the company and has specified that he does not want to reveal his professional information on the network, then the answer cannot include any information that can reveal his such information.

3. The user specifies his/her interests with consideration of the existing network - i.e., knowledge learned from Step 2. above.

Example

The user asks to be recommended to “people in the Cooking group”.

4. The user specifies his/her needs. The CoP are applied to the user.

Example

After querying, the user is aware that his boss is a member of the Cooking group. He does not want his boss to know that he is in the Cooking group so he requires to “keep me away from my boss”.

5. The user queries about obligations and permissions on the existing network and potential network in relation to his consent. The CoP are applied to both the user and the related entities.

Example

In the previous step, if the user was told that his boss is not in the Cooking group but somewhere in the network, then he might further query about “is there someone in the Cooking group will share information with my boss (i.e., someone permits or has obligations to share information with his boss).”

6. The user is notified about potential implications with respect to his privacy concern. The CoP are applied to both the user and the related entities.

Example

After step 5, the user is informed that “Member A is connected to your boss B and A allows B to view all her connections”.

7. Based on the information obtained, the user makes a decision regarding whether to change his/her choices, consents and/or expectations of what to control. Then, repeat previous steps if desired and applicable.

Example

Upon obtaining the information from step 6, the user requires “not to be recommended to the Cooking group” (i.e., repeat step 1), or “keep me away from Member A” (i.e., repeat step 4).

7 Discussion and Future Work

The need for privacy-aware social recommendations has been stimulated by the increasing prevalent privacy infringements in current online social networks and the failures of existing recommendation techniques to address such problems. The development of a privacy-aware system requires a privacy-by-design approach that necessarily begins from requirements development. In the area of requirement engineering, a number of contributions to the work on semantic privacy issues have been made. For example, Anton et al (2002) propose using goal taxonomies to structure privacy policies. Liu et al (2003) use a role-based approach to study trust relations and attacker-defender relations. Giorgini et al (2005) propose a goal-oriented secure tropos methodology to address security and trust issues. A common deficiency of these attempts is lack of considerations of the right problem at social level.

In the problem domain of social recommendations, we argue that the fundamental problem is philosophically the *right problem* and sociologically the *social problem*, of which technology solutions are necessary attributed to relationships and multiplex of relationship privacy must be addressed. To this end, this paper presents a preliminary work towards a comprehensive layered requirements architecture for building privacy-aware social recommender systems.

The proposed requirements architecture takes philosophical and sociological needs into account, allowing inputs from external regulations like legislation by using a layer approach for requirements development. Within the scope of social recommendations, requirements are developed for the central problem identified - i.e., social connection and relationship. As a result of lessons learned from existing social recommender business models such as Facebook (2009), LinkedIn (2009) and Pulse (2009) that are inadequate to meet the requirements, functional requirements towards a privacy-aware cooperative social recommender system are also developed. Future work will take steps towards methodologies and techniques for fulfilling the functional requirements to realize privacy-awareness in social recommendations.

References

- Anton, A. I., Earp, J. B., Reese, A. (2002), Analyzing Website Privacy Requirements Using a Privacy Goal Taxonomy. IEEE Joint International Requirements Engineering Conference (RE'02). pp.23-31.
- Billsus, D., Pazzani, M. J. (2007), Content-based recommendation systems. The Adaptive Web, LNCS.4321,325-341.
- Chen, S., Williams, M-A. (2009), Privacy in social networks: A comparative study. *in* PACIS 2009 Proceedings. Paper 81.
- Giorgini, P., Massacci, F., Zannone, N. (2005), Security and Trust Requirements Engineering, *in* A.

Aldini, R. Gorrieri, and F. Martinelli, eds, 'FOSAD 2004/2005', LNCS 3655, pp. 237-272.

Goldberg, D., Nichols, D., Oki, B. M., Terry, D. (1992), Using collaborative filtering to weave an information tapestry. *Communications of the ACM*. 35(12), 61-70.

Liu, L., Yu, E. S. K., Mylopoulos, J. (2003), Security and Privacy Requirements Analysis within a Social Setting. *in Proc. of RE'03*. pp.151-161.

Williams, M-A. (2009), Privacy, the law and global business strategies: A case for privacy driven design. *in Proceedings of the AAAI 2009 Spring Symposium on Social Semantic Web: Where Web 2.0 meets Web 3.0*.

OECD. (2009), http://www.oecd.org/document/18/0,3343,en_2649_34255_1815186_1_1_1_1,00.html. (Accessed: 1 July 2009).

Facebook. (2009), www.facebook.com. (Accessed: 1 July 2009).

LinkedIn. (2009), www.linkedin.com. (Accessed: 1 July 2009).

Pulse. (2009), www.plaxo.com. (Accessed: 1 July 2009).

A Conceptual Modeling Approach for Web Service Composition Supporting Service Re-Configuration

Georg Grossmann

Michael Schrefl

Markus Stumptner

University of South Australia, Advanced Computing Research Centre, Mawson Lakes, SA 5095, Adelaide, Australia. E-mail: {georg.grossmann, michael.schrefl, mst}@cs.unisa.edu.au.

Abstract

Service composition is a recent field that has seen a flurry of different approaches proposed towards the goal of flexible distributed heterogeneous interoperation of software systems, usually based on the expectation that such systems must be derived from higher level models rather than be coded at low level.

We propose a conceptual modelling approach for the composition of service processes into a task workflow and its dynamic adaption to changes during runtime. The contribution of the paper is twofold. Firstly, our approach improves on existing approaches by the separation of configuration into service configuration and service instance configuration which reduces the reconfiguration cycles in case of changes and secondly, we describe a comprehensive modelling concept that combines existing dynamic composition techniques in a novel way.

1 Introduction

The dynamic interaction of the software systems that support modern business processes is among the most important challenges faced by organizations today. A key issue faced by the developers in such situations has always been that, inevitably, separately developed systems rely on disparate models and no matter how flexible the infrastructure, it requires major effort to establish interoperability via the mutual adaption of coexisting business processes, or the integration of legacy applications, or the incorporation of local “grassroots” solutions in individual branches of enterprises that often have thousands of applications and data repositories spread across the organization. In most cases nowadays, this integration task is addressed by manual analysis and laborious development of specific integration solutions.

This problem, traditionally experienced in the context of interaction and distributed database systems, has become particularly poignant in recent times with the explosive development of new styles of infrastructure that support significantly more flexible and dynamic interactions between applications. The so-called Service Oriented Computing (SOC) paradigm uses the idea of flexibly (possibly dynamically) assembling individual application components (possibly directly accessible via the Web) into complex processes. The underlying technology on which SOC is often

assumed to be based is the technology of Web Services, using Web-based communication protocols and XML-based data formats and communication interface definitions, all of which are enjoying increasingly widespread usage. However, unless everyone uses the same data structures and interfaces, a utopian scenario, fully realizing the SOC dream requires overcoming the interoperability problem [25].

Four aspects discussed in related literature in the context of SOC are abstraction, semantic description, service behaviour and dynamic integration [7, 24]. Neřcaský et al. propose the abstraction from underlying XML-based description and using a single ontology derived from existing descriptions to facilitate the semantic integration [24]. Similarly, the BPMO approach proposes the use of an ontology but considers in addition the service behaviour [7]. Our approach goes beyond that by considering the re-configuration of services if the environment changes during runtime.

The context of this paper is the interoperation of several service processes that are dynamically composed into a *task workflow* [12] to achieve a certain business goal. We use the notion “task workflow” because it used in related literature but point out that a task in a task workflow may consists of sub-tasks modelled as activities and subactivities. For example, a simple task workflow could be a travel booking, comprising a flight and a hotel booking. The relevant service processes would be flight reservations and bookings of different airlines and hotel reservations and bookings of different hotels.

Semantic interoperation is the process of bridging and reconciling heterogeneity for combining a set of service processes, selected based on their capabilities, to meet a goal that cannot be met by using a single service. The composition of the services of the combined service processes needs to be planned and the composed services needs to be enacted. This is described by a workflow of itself, the *task workflow* [12].

Pre-planning and manual combination of service processes into complex structures is a significant bottleneck on the way to establish dynamic interactions and interoperability, an ability which is one of the main perceived advantages of the service-oriented approach over other paradigms. The goal of providing automated support for Web service discovery and composition has led to the development of advanced service description approaches, employing languages which permit defining not just inputs and outputs but also parts of the semantics of the service. In practice, however, even apparent front-runners can prove to be difficult to apply [30, 4] and insufficient for describing certain complex functionality [9].

A fundamental decision concerns the selection of the underlying interoperability architecture and knowledge representation of service processes and the composite task workflow. The representation must be rich enough to capture the semantics of service pro-

This research was partially supported by the Australian Research Council (ARC) under grant DP0988961. Copyright ©2010, Australian Computer Society, Inc. This paper appeared at the Seventh Asia-Pacific Conference on Conceptual Modelling (APCCM 2010), Brisbane, Australia, January 2010. Conferences in Research and Practice in Information Technology, Vol. 110. Sebastian Link and Aditya K. Ghose, Eds. Reproduction for academic, not-for profit purposes permitted provided this text is included.

cess profiles in order to facilitate the detection and bridging of semantic heterogeneities, abstracting from syntactic details of Web service descriptions. The architecture should support different target implementations. An approach that was developed with this goal in mind is the so-called Model Driven Architecture (MDA). It offers a layered architecture that permits modeling of system artifacts in a platform independent fashion, with the goal of automatically mapping them later to a specific implementation platform. MDA representations typically (though not always) use variants or subsets of UML. UML has been used for modeling service workflows, e.g., [12], but again only at the structural level. UML does provide more powerful means of expression, such as the Object Constraint Language OCL, which has been successfully used for knowledge representation [9], including in the SOC area.

The contribution of this paper lies in a top-down design approach for the dynamic composition of Web services. It consists of a comprehensive modelling concept that combines existing dynamic composition techniques and provides a multi-level configuration approach on the service and service instance level that reduces the reconfiguration cycles.

The next section describes a motivating example followed by a discussion on related work. Section 4 explains the composition approach in detail followed by the conclusion.

2 The Business Trip Scenario

We use the following business trip booking scenario to demonstrate our composition approach. The requirements for the business trip are as follows:

- r_1 : Make a round trip airline booking from Adelaide to Vienna on either one of the airlines Deepblue or PurpleRed, with preferred departure date 4th of July and preferred return date 14th of July, and make an accompanying hotel reservation with StarHotels.
- r_2 : Spend maximum 2,800 AUD for the airline ticket.
- r_3 : Select a hotel rated as ***-stars or higher, spend a maximum of 1,900 for accommodation.
- r_4 : The overall trip costs should not exceed 4,600. If overall costs can be reduced by more than 10% percent, the travel itinerary may deviate from the preferred dates, with earliest departure 2nd of July, latest departure 6th of July, earliest return 12th of July, latest return 18th of July, a minimum stay of 10 days and maximum stay of 14.

3 Related Work in Dynamic Composition of Services

Semantic Web service composition is the process of combining a number of Web services, selected based on their capabilities, to meet a goal that in general cannot be met by using a single service. Web service compositions are specialized business processes or workflows where every activity or operation in the process is carried out by a Web service. Previous works have examined a number of traditional paradigms such as planning [19, 26, 34, 32], ontology-based matchmaking [23, 11, 20] and constraint-based systems [2, 16] to automate various parts of the composition task. Alternatively, some approaches such as [12, 36] have investigated the effectiveness of using MDA-based technologies to accomplish semantic Web

tasks. In this section, we discuss some related approaches and techniques in the area of service configuration. Further work can be found in the modelling of adaptive workflows in the health care domain [6, 35], concepts and tools for modelling dynamic workflows and process variants [21, 1, 28] and research in self-healing business processes and Web services [15].

Related Approaches: There are multiple approaches developed by the research community in dynamic Web Services composition but only few address a top-down design. We discuss the four approaches that are more closely related to the approach presented in this paper: (1) Processes with Adaptive Web Services (PAWS) ¹ [3], (2) ServiceMosaic [5], (3) the Model Driven Web Service Composition developed by Grønmo et al. [12] and (4) the Service Delivery Life-Cycle for Semantic Service Provisioning [18, pp.14-18].

The Process with Adaptive Web services (PAWS) framework couples adaptation design-time and run-time execution [3]. It presents an approach that deals with service processes and provides several design- and run-time tools such as mediator configurator and mediator engine for coupling modules and supporting message transformation. PAWS differs in achieving flexibility because of providing advanced design tools and separating the design from the run-time part in the composition process. This has the disadvantage that the number of services as well as all possible exceptions must be known prior to execution. Our approach allows to re-configure the design of a service process in case of a run-time failure and to consider new alternative services in the composition.

The ServiceMosaic project targets the development of model-driven adapters for business protocols [5]. In this project Benatallah et al. investigated the automated analysis of compatibility and replaceability of services on the protocol level as basis for the model-driven development of Web service protocol adapters. Adapters are used to mediate between incompatible interfaces of functionality-wise compatible services and adapting Web services in a way so they become compliant with other services. This work relates to one part of our composition approach, the service configuration step described in Section 4.3 in which mediators are developed in order to communicate with the interfaces of service providers.

Grønmo et al. propose a model-driven composition approach for semantic Web services [12]. It considers the discovery and selection of service using an ontology based approach in order to capture the semantics of service functionality. The approach also supports the definition of Quality of Service (QoS) in form of requirements such as maximum costs. However, it does not consider the reconfiguration of the composite process if unforeseen changes of the service environment occur.

Grønmo et al. introduced the concept of an abstract workflow in the composition approach. An abstract workflow is a pre-specified control flow specification between abstract service specifications. UML activity diagrams are used to model the abstract workflow. Abstract service specifications, which are a representation of a task that needs to be realized, are modelled as activities. Given an abstract workflow, the composition process discovers services from a registry that meet the requirements, replaces abstract services with respective concrete services, and finally formulates a concrete workflow where all the abstract service specifications are replaced with a respective concrete service while retaining the control flow from the abstract specification.

¹<http://www.paws.elet.polimi.it/>

Kuropka et al. describe an approach for *adaptive service provisioning* called *service delivery cycle* [18, pp.14-18]. The life cycle considers the iterative planning and enactment of services, where the planning sub-cycle includes matchmaking and composition, the binding sub-cycle includes negotiation and contracting, and the enactment sub-cycle includes invocation, monitoring and profiling. This approach overcomes some limitations of static binding of Web services, e.g., it allows the utilisation of new Web services during the enactment and can accommodate changes such as sudden unavailability of services. Our approach goes beyond this by distinguishing between the composition and matchmaking on the schema and instance level and proposing a configuration rather than a planning approach. A configuration on the schema and instance level has the advantage that a composition can be adapted by reconfiguring bound services in a subcycle rather than performing a new replanning and rebinding cycle with all available services every time a change occurs. For example, in the business trip scenario it is possible to identify an airline that fulfills all the criteria on the service description level. However, it might happen during runtime that there is no flight available on a specific day. Instead of replanning the whole trip, an alternative flight on a different day is searched with the already chosen airline.

The problem that many planning approaches face is that the number or types of services involved in the composition process must be known beforehand because they have a finite set of variables. These predictions are usually difficult to make especially for dynamic composition scenarios. Therefore we propose a configuration approach instead of a planning approach which defines the scenario in form of a *Generative Constraint Satisfaction Problem* [22]. The approach is explained in more detail in Section 4.3 and overcomes the limitation previously mentioned.

Service Configuration: AI planning approaches have been proposed as one option for automated Web service composition [17, 18]. In planning, an initial and final state are provided along with constraints on actions that are available to agents. In a forward chaining approach, the agent starts in the initial state and tries to identify the next action which brings the solution closer to the final state, in contrast to backward chaining which starts in the final state and tries to identify actions backwards in a possible control flow leading to the initial state.

An alternative approach is presented by consistency-based configuration that has been widely studied and applied to a number of industrial problems. Web service composition can be modelled as a constraint satisfaction problem as shown in [2, 16, 33]. In comparison to other approaches this work is robust in handling overspecified profiles which usually lead to less preferable matches and in supporting queries for the non-existence of certain properties.

4 Dynamic Composition and Re-adjustable Execution of Service Processes

The interoperation of business processes is generally acknowledged as one of the most challenging problems faced by the more and more interconnected digital economy, still requiring at this point largely manual intervention resulting in huge costs during the development and over the lifetime of software systems. The shift to more flexible and increasingly virtual technologies does not alleviate the problem, rather it runs the risk of not realizing the potential gains

inherent in these technologies if the fundamental interoperability problem is not tackled. The approach outlined in this section uses a combination of novel extensions of technologies that have, individually, been proven to be very powerful in tackling difficult real-world problems as demonstrated by the authors' earlier work [14, 31, 10].

We propose to use a Model-Driven Architecture (MDA) approach as input to a configuration reasoning engine to create interoperable business processes. Figure 1 shows an overview of the composition during its life-cycle.

The actual integration of services becomes an integral part of a workflow that is seamlessly embellished with matching-level activities as needed. As a result, the workflow is dynamically built and adapted while using a straightforward notation. Once assembled, a workflow is reusable, so that many bookings can theoretically be made subject to successful execution. This successful execution is adaptive to environment changes (e.g., selected hotels having no rooms free once the user has made a decision). However, there is no explicit exception handling, instead we provide first class workflow activities ("reflective workflow") to enable adaptation at runtime such as re-planning a business trip if a hotel room previously quoted is no longer available at booking.

Most importantly, the modeling approach is intuitive: the process is modelled the way a normal clerk would do it, but the cumbersome matching task is automated. The user is in control through providing parameters and priorities. If a library of task workflow patterns exists for a particular domain, different patterns could be selected according to preferences (e.g., book hotel first, or check multiple airlines etc).

Semantic technologies assume the existence of a machine-interpretable description of information and services. In our case, the "semantic" aspects refer to three specific technologies linked up to provide a joint model of the problem area:

- an underlying object model of the entities and relationships in the problem area, amended with declarative constraints between these entities and a specific set of relationships that permit a meaningful distinction between different types of heterogeneities [14].
- a model of the behavior of a set of processes, expressed in a formally defined diagram language (Petri net-like or UML activity diagram extended with states) [27].
- a declarative approach for reasoning about these processes and the constraints in the object model in a seamless fashion [33, 2, 8].

The composition process consists of three levels, (1) abstract level, (2) candidate selection and (3) execution level. They are based on the abstraction levels proposed by the MDA but differ in the way that they might not be executed in a strict sequence but interleaved. Under certain circumstances explained below, it might be necessary to loop back to a previous level. The approach

- begins on the abstract level with a user modeling a task workflow with
 - a pre-specified control flow between "service usage tasks", each tied to an "abstract service object" (e.g., a flight reservation)
 - functional and other individual requirements of every service usage task (e.g., payment in Euro or in AUD), and

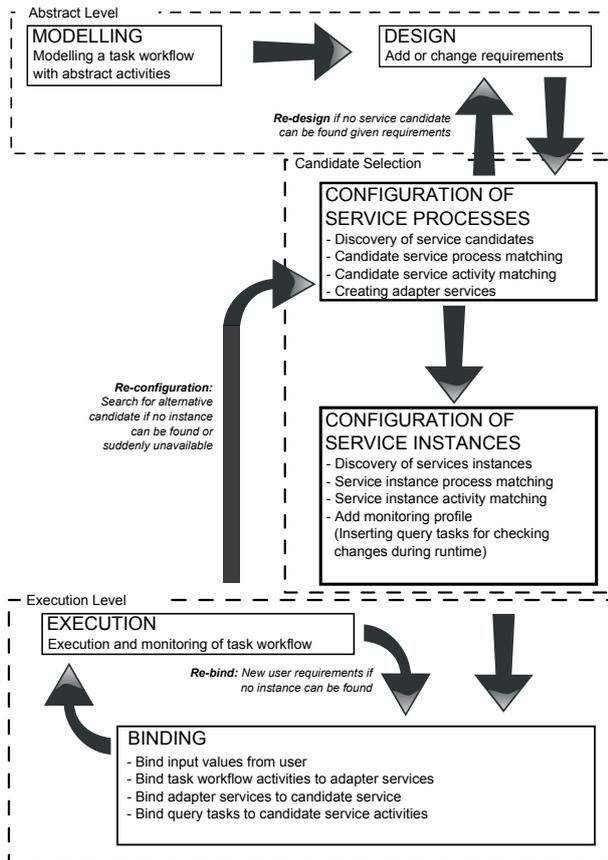


Figure 1: Overview of approach for dynamic composition and reconfiguration during runtime.

- additional global requirements of the whole workflow (e.g., maximum cost)
- proceeds with a candidate selection which includes the identification and interoperable integration of service processes and their instances into the task workflow, whereby
 - semantic heterogeneities are resolved through inclusion of “bridging tasks” (e.g., an activity converting different currencies), or composing tasks into compound non-decomposable task (e.g., payment and ticketing), or re-ordering tasks (e.g., payment before ticketing), if appropriate and acceptable to the user, and
 - “control tasks” are inserted to query service processes for their current service environment, i.e., their available or running service instances, (e.g., available flights or status of flight bookings), to match - at the instance level - service instances that best meet the specified global requirements to abstract service objects, and to configure or re-configure (upon a changed service environment) a corresponding workflow execution schedule (determining which paths to take in the workflow “upstream” of the currently active task),
- proceeds with the execution, whereby
 - user input values (e.g., flight details) are bound to task workflow and abstract service objects are bound to selected service instances, and

- “service usage tasks” are subsequently executed.

Some steps in the composition process may be repeated if changes occur:

- *Re-design*: If no service candidate can be identified during the candidate selection because the requirements cannot be fulfilled by available services, the user has to change the requirements on the abstract level (e.g., change order of service usage tasks).
- *Re-bind*: If no matching service instances can be found by “control tasks” during execution, then the user has to provide different values for the task workflow parameters which are bound to the task workflow (e.g., different dates for flights).
- *Re-configuration*: If the environment changes during execution, e.g., a service is suddenly not available any more, then services processes and service instances need be to re-configured to find an alternative solution. Changes in the environment are identified by “control tasks” in the task workflow.

In the remainder of this section the composition process steps are explained in the context of the business trip example from Section 2.

4.1 Modeling Service Processes

We now provide an MDA-based approach for describing service processes. Service processes represent the functional infrastructure that is offered by providers, and the execution that is provided by the task workflow. For each offered service process its provider publishes a service process description, describing its properties by states and constraints, and its behavior by states and actions. Furthermore, it specifies a set of parameterised queries, referred to as enquires, that select all or a subset of the service instances available and that retrieve the current state of a specific service instance.

Definition 1 (Service Process Description) A service process description D is defined as a tuple (S, T, F, E, A, C) defined on a set of states S , actions T (with a set of arcs $F \subseteq (S \times T) \cup (T \times S)$ describing state transitions via actions), and enquiries E . The properties of each instance of this service process are described by a set of attributes A and constraints C . We assume a subset K of A serves as key to identify individual service instances. (Alternately, a dedicated interaction key k may exist for part of the lifecycle.)

Simple services are a degenerated case of this definition, represented by a Service Process Description with a single action transiting from initial to final state.

Definition 2 (Service Instance) A service instance i of a given service D is described by the current life cycle state inhabited by i , written as $lcs(i) \subseteq 2^{S \cup T}$ and a value for each attribute in A . Let I_D denote the set of service instances of a given service D .

Definition 3 (Enquiry Service Description) An enquiry is a parameterised query over the current process state of a subset of service instances for some service description D (generic enquiry), or for the current process state of a specific service instance (specific enquiry). A generic enquiry service description is defined as a parameterised (first order) predicate

over the set of service attributes A , returning all service instances that satisfy the predicate. A specific enquiry description is parameterised by the interaction key of the specific service instance to be selected. Providing actual parameter values for an enquiry service description results in a concrete enquiry.

Example: Let us consider the service process descriptions of two airlines, DeepBlue and PurpleRed. For simplicity, we assume that our two airlines offer only return tickets and quote only the cheapest price for each itinerary. For brevity, we also consider a simplified service process.

Potential service instances (un-booked flight offers) of both airlines are identified by attributes `fitNo`, `departureDate`, `retFitNo`, `retDate`, and are described by the ticket price. A reserved or booked itinerary is identified by a `reservationNo` that is unique per airline. The reservation number is returned by the first action in a service process e.g., `reserve` with DeepBlue or `book&pay` with PurpleRed. Both airlines offer two enquires: `obtainQuote` (`from`, `to`, `depDate`, `retDate`), which return a set of quotes for the specified itinerary, and `checkStatus`(`reservationNo`) which returns for the current status of a reservation.

Figures 2 and 3 depict the service process descriptions of airlines DeepBlue and PurpleRed, respectively. The figures show actions and associated state transitions. While airline DeepBlue offers the possibility to make reservations (`reserve`) and to cancel or confirm the reservation later, which results in a booking that is paid (`pay`) and used (`checkIn`), airline PurpleRed offers only instant ticket sales (`book&pay`). Once a ticket is sold it may be used `checkIn` or canceled (`cancel`). Canceled tickets may be entitled to a refund, subject to a deducted cancellation fee (`collectRefund`).

The life cycle of the service process of our hotel group StarHotels consists of a single action only, `makeReservation`(`quoteNo`). The service provider StarHotels offers one enquiry service `obtainQuote` (`fromDate`, `toDate`) returning a set of quotes for the indicated period, each one identified by a unique `quoteNo` and described by the hotel name and `dailyRate`.

While flight quotes will be priced in Australian Dollar, hotel quotes will be in Euro.

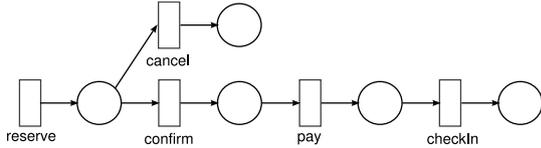


Figure 2: Deepblue Airline Service Process (simplified)

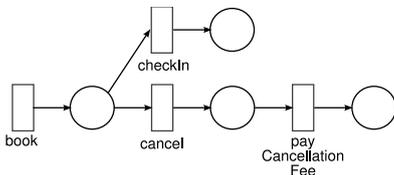


Figure 3: PurpleRed Airline Service Process (simplified)

4.2 Modeling the Task Workflow

The task workflow is the “workhorse” of the approach and serves as the anchor for the process logic and the service matching and instantiation. It is specified by the user on the abstract level of the approach.

Definition 4 (Task Workflow Description) A task workflow description W is defined as a tuple $(S_w, T_w, F_w, E_w, K_w, A_w, C_w, O_w, D_w, M_w, b_w, e_w, v_w, s_w)$ defined on a set of states S , actions T (with a set of arcs $F \subseteq (S \times T) \cup (T \times S)$ describing state transitions via actions T_w), and enquiries E . The properties of each instance of this service process are described by a set of attributes A_w and constraints C_w . $K_w \subseteq T_w \times S_w$ indicates a set of k.o.-links between activity and states, describing what states (referred to as k.o. post states) are entered when an activity completes unsuccessfully. Ordinary links and post states, i.e., those determined by F_w , are called o.k.-links and o.k. post states, respectively.

D_w is the set of service process descriptions from which the task workflow is composed.

O_w is a set of abstract service objects, each potentially assigned to a specific service instance of some service process.

$M_w : T_w \rightarrow D_w \times T_w$ associates with each activity an activity of some service process, where if $M_w(t) = (d, t)$, t is an activity of service process description d .

Partial function $b_w : O_w \rightarrow I$ binds each abstract service object to some service instance.

$E_w : O_w \rightarrow 2^{D_w}$ associates with each abstract service object the set of service process descriptions to whose instances the abstract service object may be bound.

Partial function $e_w : O_w \rightarrow I \times D_w$, where for $e_w(o) = (i, d)$, it holds that $d \in E_w(o)$ and $i \in I_d$, associates each abstract service object o to a pair of service instance and service process.

Partial function $v_w : O_w \times I \times A \rightarrow V$ assigns each abstract service object o for each service instance in $e_w(o)$ a value for each attribute. (Such a value corresponds initially to the attribute value of the service instance, but may become “dirty” if the service instance of the service processes involves independently over time, e.g., flight is booked by someone else).

Function $s_w : O_w \rightarrow SOS$ where $SOS = \{\text{unbound, tentative, committed, canceled, fulfilled}\}$ is the set of possible Service Object States.

The workflow activities T_w are distinguished into service usage activities (SUAs), enquiry activities, bridging activities, and control activities. The insertion of bridging and control activities is system supported if data type heterogeneities or critical execution states of the task workflow are identified. A critical execution state might be the begin or end of a milestone (e.g., after obtaining quotes or booking itineraries) when control activities monitor service processes by querying service instances for state changes that need to be handled (e.g., booking an itinerary failed).

Example: The task workflow of our business trip consists of the service process description described above. The user models a simple task workflow similar to a reference process which is refined automatically in the candidate selection step using configuration techniques. Refinement means that modelled activities are refined with subactivities and new activities may be inserted in a consistent way. Consistency criteria using inheritance which are applicable in this step have been investigated in previous work [31]. Figure 4 shows an example for an initial task workflow modelled by the user.

The task workflow has two abstract service objects, `flightItinerary` and `hotelBooking` which are not explicitly modelled in Figure 4. Each instance of the task workflow is described by the following attributes: `fromCity`, `toCity`, `preferredDepDate`, `preferredRetDate`, `maxAirFare`, `minStars`, `maxHotelCosts`, `maxOverallCosts`, `earliestDep`, `latestDep`, `earliestReturn`, `latestReturn`, `miniumStay`,

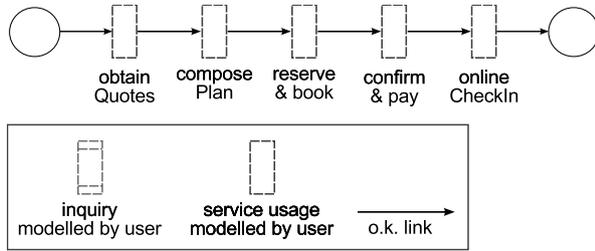


Figure 4: Initial task workflow modelled by the user.

maximumStay, deviateThreshold. The attributes are self explanatory, except `deviateThreshold` which indicates the percentage in savings which justify choosing an alternative itinerary than that of the preferred dates. Attributes are extracted from user-defined requirements. During execution some service providers may require further attributes which are added dynamically during runtime (e.g., frequent flyer number).

We describe the execution semantics of a task workflow in this section prior to the candidate selection and execution level for better understanding.

Execution Semantics: We define the notion of workflow state, formally capturing the current processing state of a workflow instance (which is referred to as life cycle state) and the workflow execution schedule, determining how the workflow should advance from its current processing state through the execution of service usage activities. As the outcome of service usage activities in the workflow execution schedule is not known in advance, i.e., the booking of a quoted hotel room may be successful or not, the different steps of the execution schedule may be ‘guarded’, where a guard is expressed by a life cycle state of the workflow instance. The execution of a workflow schedule and the semantics of guards is explained below.

Definition 5 (Workflow State) *The workflow state of a service instance i of a given service D is described by a tuple $\langle clcs, es \rangle$ consisting of the current life cycle state lcs inhabited by i and the execution schedule $es = \langle s_1, \dots, s_n \rangle$ (which is a list of steps to be executed (started) in order in the future. The current life cycle state $clcs$ of the workflow instance is expressed by the states in which the workflow instance currently resides; if an activity is currently executed, it is also part of the $clcs$. Each step s_i ($i = 1 \dots n$) of the execution schedule es is a triple $\langle lcs, a, b \rangle$, where lcs is a life cycle state that acts as a guard indicating the step to be executed only if it matches the current life cycle state of the workflow instance, a indicates the activity that is to be invoked if the step is executed, and b determines the actual parameters of the activity invocation.*

The execution schedule, the output of the match-making process, is a list of activities with actual parameter bindings. Service execution means calling the individual activity at the head of the execution schedule with the proper parameter binding. If the activity is guarded and the current life cycle state of the workflow instance matches the life cycle state indicated by the guard, the activity is executed. It is ignored otherwise, and the execution schedule is continued with the next step.

Formally, given current lifecycle state $clcs$ and execution schedule $es = \langle s_1, \dots, s_n \rangle$ with $s_1 = \langle lcs_1, a_1, b_1 \rangle$, we have the following situation. If $lcs_1 = \emptyset$ or $lcs_1 = clcs$, then we execute a_1 with b_1 as

parameters. If $lcs_1 \neq clcs$, the activity is discarded. In either case, the new execution schedule es' is defined as $es' = \langle s_2, \dots, s_n \rangle$ (This description does not include the semantics of control activities which alter the execution schedule).

The current workflow object O_w is omitted if it is understood.

To simplify the execution model, we make the assumption that individual activities in the service process (enquiries and actions) are atomic.

Per the definitions of the previous section, we have an environment e_w (potential bindings) plus instantiated (concrete) query (the latter for optimization purposes) for each abstract service object, plus global constraints given with the service description, plus global workflow constraints C_w , plus global attributes A_w (which must be bound to values).

4.3 Candidate Selection

The candidate selection follows the modelling step and consists of the service and service instance configuration.

During service configuration, services are discovered, matched, and introduced into the task workflow, i.e., the task workflow is modified. The remainder of this paper focuses on service instance configuration and shows only one service configuration step.

Service instance configuration consists of instance discovery and matching. For better understanding of the approach, we first explain briefly the service configuration part with an example, then the service instance configuration part and later mediation, matching, and configuration in more detail.

Service Discovery: The identification of relevant service processes is conducted by existing approaches like the ones mentioned in [17]. Here we describe three efforts used in Semantic Web Service discovery that are applicable: First, keyword-based discovery can be applied on a directory, e.g., UDDI, based on the names of the *abstract service objects* defined in a *task workflow description* (cf. Def.4). Second, subsumption-based discovery can be applied by consuming functional requirements defined as *constraints* for each *service usage task* (cf. Def.1), and third, state-based discovery is applicable by using global requirements defined in the task workflow description (cf. Def.4) and service usage task specific constraints like pre- and postconditions (effects) (cf. Def.1).

Example: Abstract service object `flightItinerary` may be bound to a service instance of either `DeepBlue` or `PurpleRed`, and abstract service object `hotelBooking` to a service instance of `StarHotels`.

The binding of the abstract service objects to service instances is constrained by the collected user requirements.

Fig. 5 depicts the life cycle of the task workflow of our business trip after the service configuration was executed and the task workflow from Figure 4 was refined and extended. An instance of the task workflow is created by introduced control activity `collectTravelReq`, which collects the travel requirements from the user and initializes the attributes of the workflow instance accordingly. If additional attributes (e.g., frequent flyer numbers) are required by certain service providers then the control activity takes care of that.

Thereafter, activity `obtainQuotes` is executed which was refined into three enquiry activities, `obtainQuotes_DB`, `obtainQuotes_PR`, `obtainQuotes_SH`, to obtain quotes from service providers `DeepBlue`, `PurpleRed` and `StarHotels`, respectively. The three subtasks may be executed in any order. As hotel quotes are in Euro,

they are converted by a subsequent bridging activity `convertCurrency` into Australian Dollar. The bridging activity was also introduced by the configuration task.

Once these activities have been performed, the quotes are matched and selected by the control activity `composeTravelPlan`.

If a flight itinerary and a hotel quote could be found that meet the specified user requirements, the workflow can proceed with respective bookings.

If the offers cannot meet the user requirements, the workflow is continued by following the *k.o. link* and continued with a control activity that recollects changed user requirements `reCollectTravelReq`.

For better readability of the Figure, only states to which we explicitly refer to in our examples are named.

The example mentioned before describes the result of the service configuration step. In the remainder, the concepts that result task workflow are discussed.

Service Instance Discovery: The task workflow will usually include a set of enquiry activities that query service providers for available service objects, i.e., instances of service processes, that are considered relevant for the subsequent service matching.

Example: In our business trip setting, which should start on 2nd of July at the earliest and conclude on 18th of July at the latest, only flight itineraries between Adelaide and Vienna whose departure is on 2nd of July or later and whose return flight is on 18th of July or earlier, are relevant.

The first table in Figure 6 depicts $e_w(\text{flightItinerary})$, the potential bindings of abstract service object `flightItinerary`. The set of potential bindings is determined by the result of executing in our task workflow the enquiry activities `obtain_DB_quotes` (followed by bridging activity `convertCurrency`) and `obtain_PR_quotes` for available flights in the period between 2nd of July (earliest departure) and 18th of July (latest return). The second table in Figure 6 depicts $e_w(\text{hotelBooking})$, the potential bindings of abstract service object `hotelBooking`. The set of potential bindings is determined by the result of executing in our task workflow enquiry activities `obtainQuotes.SH` for available hotels in the period between 2nd of July (earliest departure) and 18th of July (latest return).

SP	fltNo	date	retFltNo	retDate	price
DB	DB1	4/Jul	DB2	14/Jul	2.780
DB	DB1	5/Jul	DB2	18/Jul	2.390
DB	DB1	2/Jul	DB2	12/Jul	2.660
PR	PR5	4/Jul	PR6	18/Jul	2.640
PR	PR5	6/Jul	PR6	16/Jul	2.800

quoteld	hotel	availability	dailyRate
13	Uni Lodge	2/Jul - 12/Jul	115
21	BlueDanube	2/Jul - 18/Jul	180
23	Budget Motel	6/Jul - 16/Jul	120
27	Park Inn	2/Jul - 18/Jul	190

SP ... service provider

Figure 6: Potential bindings for abstract service object `flightItinerary`, available after obtaining airline quotes, and abstract service object `hotelBooking`, available after obtaining hotel quotes

Service Mediation: For the semantic mediation, we distinguish between data and behavior mediation. For data mediation, an ontology alignment approach is used as proposed by WSMO [29]. This handles mismatches at the data definition and message exchange

protocol level. The handling of heterogeneous business processes is addressed by a catalog of declarative integration patterns defined in previous work [13]. The catalog defines patterns used for the horizontal and vertical integration in Enterprise Application Integration (EAI) and business-to-business (B2B) scenarios. Vertical integration focuses on services in a similar domain and horizontal integration targets services from different domains contributing to a common goal.

Example: A set of airline booking services need to be integrated *vertically* in order to choose the best offer according to some preferences for a specific flight. An airline booking service and a hotel booking service need to be integrated *horizontally* in order to complete a travel booking.

Service Matching: Constraint-based systems, which have been successfully applied to a number of large-scale industrial problems, are potentially useful in the Web service composition context. Previous works, in particular [2, 16], have examined the feasibility of using constraint-based systems to address the Web service composition problem. While [2] provided a constraint-based workflow design based on input and output compatibilities, [16] provided an optimized approach to carry out simple checks on Web service attributes. The need for a matchmaking process, that filters Web service candidates based on the functionality a Web service offers, for use within semantic Web based tools has already been established. However, the previous constraint-based proposals do not consider semantics such as functionality of a Web service while performing candidate selection. Our approach incorporates the semantics (as specified in the abstract workflow, any additional operation specifications, plus functional requirements specified by the user) to generate concrete workflows for any given abstract workflow.

The composition process begins with the design of an abstract workflow. An abstract workflow is a pre-specified control flow specification between abstract service specifications. An abstract service specification is a representation of functional and additional individual requirements that a concrete service has to satisfy. Given an environment of Web services with declarative specifications of their properties such as functionality, semantic service composition is the process of formulating concrete instances of an abstract workflow using the environment such that all the functional, individual and global requirements specified in the abstract workflow are satisfied.

Example: Figure 7 depicts the tentative bindings of abstract service objects `FlightItinerary` and `hotelBooking` after service matching and composition and before the respective reservations or bookings have been made.

SP	fltNo	date	retFltNo	retDate	price
DB	DB1	2/Jul	DB2	12/Jul	2.660

quoteld	hotel	availability	dailyRate
21	UniLodge	2/Jul - 12/Jul	135

Figure 7: Tentative bindings of abstract flight and abstract hotel after service matching

Example: Figure 8 depicts the workflow state and execution schedule after service matching. The workflow is in life cycle state `{ travelPlanComposed }`, which is the post state of control activity `composeTravelPlan` in our task workflow. Two activities are scheduled in accordance with the task workflow description, the reservation of the flight itinerary (unguarded activity `reserve_DB`) with service provider `DeepBlue` for the tentative binding of abstract service object `flightItinerary`

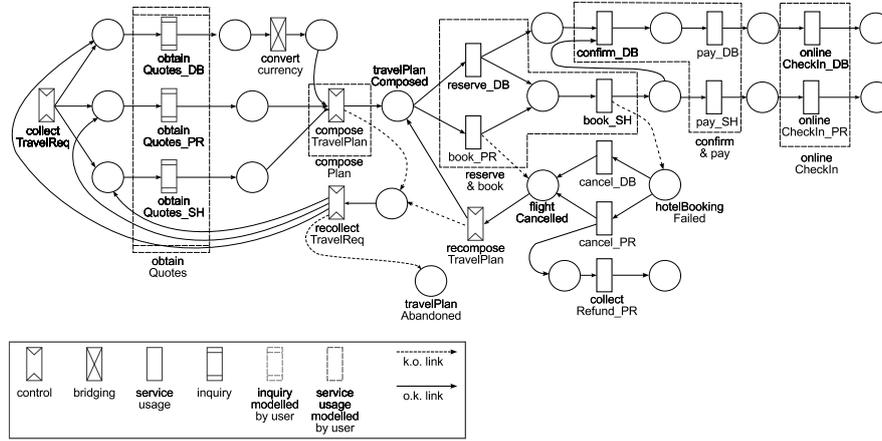


Figure 5: Task Workflow: Business Trip

and the booking (activity guarded activity `book_SH`) of the hotel with service provider `StarHotels` according to the tentative binding of abstract service object `hotelBooking` (cf. Figure 8). The second step of the execution schedule is guarded by life cycle state `{hotelBookingFailed}` which is the *ko-post state* of activity `book_SH`. This means that the third step (activity `cancel_DB`) will be executed only if the requested hotel booking fails once a quoted offer is no longer available. The fourth state `flightCancelled` is entered after `cancel_DB` is executed and enables the execution of activity `recomposeTravelPlan`.

step#	lcs	activity	p.bindings
1		reserve_DB	dept: (DP1,1/Jul), ret: (DP2,11/Jul) quoteld: 21
2		book_SH	
3	{ hbf }	cancel_DB	
4	{ fcd }	rTp	

clcs = { `travelPlanComposed` }
 rTp ... `recomposeTravelPlan`,
 hbf ... `hotelBookingFailed`
 fcd ... `flightCancelled`

Figure 8: Workflow state with execution schedule after service matching, i.e., after execution of control activity `composeTravelPlan`

The profile of a service is an advertisement of the capabilities or functionality that the service claims to offer. Its two parts (functional properties or attributes on one hand and invariants on the other) should be satisfied by all the concrete service instances which are advertised using this profile.

Semantic candidate selection is the process of discovering Web services from one or more registries by performing a matchmaking operation on the advertisement of the Web service capabilities against a set of user specified requirements which in general represents the functionality a user expects the Web service to offer.

Definition 6 (Service Matchmaking Request)
 A service matchmaking request R is a set of constraints R_{cons} usually representing the functionality of a Web service that a user of the matchmaking procedure is searching for.

Definition 7 (Constraint Satisfaction Problem)
 A constraint satisfaction problem (CSP) is defined as a triple $\langle V, D, C \rangle$ where

- V is a set of variables $\{v_1, v_2, \dots, v_n\}$

- $domain(v_i)$ denotes the set of allowable values the variable v_i can be assigned with.
- D is a set of domains $\{D_1, D_2, \dots, D_n\}$ such that the $D_i = domain(v_i)$
- C is a set of constraints that need to be satisfied for any given assignment to the variables in V .

A solution to a CSP is an assignment of values to variables such that all constraints are satisfied.

Frequently, in modeling complex, component-based systems (hardware or software), this basic definition is extended to provide individually identified components with a structure defined by a type hierarchy (i.e., object structures). An object's attributes are variables of the CSP, and an object's references (instance variables referencing other objects) also are interpreted as variables in a CSP. However, their domain is special: it is the set of components (or, in typed systems, the set of components of the proper type). This object-oriented constraint view is the modeling approach used by commercial configuration tools and, e.g., in [2], maps naturally to a service composition problem where service instances, activities in the workflow, and data processed by services are considered first-class objects, and the invariants, constraints and pre- and postconditions (guards) associated with services and data objects are constraints on these objects and their values and references (i.e., relationships).

Service Configuration: The service configuration consists of the sequential execution of the composition task (a) "Identification of pairs of service usage tasks and service candidates" and (b) "Request for user feedback for changing the execution order of service usage tasks in the task workflow description". Composition task (a) performs the configuration using the description of previously discovered services, the task workflow description, all requirements (function, non-functional, global requirements), and any necessary and available bridging tasks as inputs. If the task is successful then it produces a list of pairs that assigns for each service usage task a concrete service and the task workflow description. If the composition task does not find a solution, i.e., it fails, then the task workflow needs to be re-designed by the user, i.e., the user has to change the requirements and the configuration will be executed again.

During configuration the order of the steps in the task workflow may need to be changed due to heterogeneous business processes of the discovered services. The change results from a re-ordering caused by the

instantiation of an integration pattern. In this case composition task (b) is executed which requests confirmation for the re-ordering from the user. If the user does not agree with the re-ordering then the configuration task searches for an alternative solution.

Configuration is finished when a concrete service is assigned to each service usage task in the task workflow. It may be executed again if the environment of a task workflow instance is changed during execution and part of the workflow state enters a compensation task.

4.4 Execution

Under certain circumstances, when an action fails and steps have to be retaken, existing objects may have to be wound up, i.e., finish their lifecycle according to their current condition, as part of the compensation activities for the problem.

Example: Figure 9 depicts the execution stage of our workflow instance after the tentative hotel booking at UniLodge hotel from 2/Jul till 12/Jul (cf. Fig. 7) according to quote no. 21 failed due to the quoted offer no longer being available. The workflow instance has entered the *k.o.* post state `hotelBookingFailed` of activity `book.SH`. The workflow has entered the compensation phase. The flight reserved with airline `DeepBlue` needs to be cancelled (activity `cancel.DB` of step 1) and the travel plan needs to be re-composed. This re-tasking step is represented by control activity `recomposeTravelPlan` that is scheduled as step no. 2 in the workflow execution schedule after the flight has been canceled. This control activity will invoke another match-making process and, thereupon, define another workflow execution schedule and enter state `travelPlanComposed`. If this match-making process fails, the workflow continues with another control activity, `recollectTravelReq`, in which the user can change the travel requirements or abandons the business trip.

step#	lcs	activity	p.bindings
1	{ hbf }	cancel.DB	quoteld: 21
2	{ fclد }	rTp	

```

clcs = { hotelBookingFailed }
hbf ... hotelBookingFailed
fclد ... flightCancelled
rTp ... recomposeTravelPlan

```

Figure 9: Execution stage during compensation phase after tentative hotel booking found unavailable

In some cases, it may be useful to re-task a workflow, i.e., to determine new bindings for abstract service objects, while the compensation phase is still incomplete. In such a case, a *wind-up* copy of the task workflow instance is created that exists concurrently to the current workflow version. The wind-up area may maintain a separate branch until it has reached the end of its lifecycle. A wind-up areas of a task workflow is identified by a set of activities and states in the workflow description.

Example: In our business trip workflow, service usage activity `collectRefund.PR` together with its prestate and its poststate constitutes a wind-up area. Notice that this area is not explicitly emphasized as such in Fig. 5. The wind-up process maintains the binding of abstract service object `flightItinerary` such that a refund can be obtained for the canceled flight booking (service usage activity `collectRefund.PR`). Concurrently, the proper workflow process can be continued by recomposing the travel plan (control activity `recomposeTravelPlan`) and, thus, associating another flight itinerary (service process instance) to the abstract service object `flightItinerary`.

5 Conclusion

We have presented a framework for service composition based on conceptual modeling principles, i.e., the use of high-level models of the data and processes involved in the implementation and execution of collaborative service processes. It considers service composition as part of (and result of) a complex design process that in real-world applications has to permit inclusion of user decisions and to execute processes in a distributed, non-atomic environment. The conceptual model serves to describe schema level information and process templates that can be instantiated for the execution of actual services, a distinction that is routine in classic data modeling, but not normally made so far in service composition. By separating out the schema and instance levels, finer distinction between potential service capabilities and actual runtime executability can be achieved. Most importantly, based on this capability, the conceptual models include primitives for modeling decisions taken during the design stage (i.e., the “including control tasks” phase of the process development and execution cycle).

This provides a structured framework for describing how a process can be readjusted at runtime according to information that in real world scenarios is ultimately dependent on execution of particular service instances and therefore not available at static design time, such as the availability of seats on a particular flight at the moment of booking a whole itinerary that was previously planned. The framework enables the composition of service processes under control of the process itself, including the ability to adjust the process if conditions imposed by the environment have changed.

We plan to implement the conceptual model with a similar visual notation as shown in Figure 5 in the DoME² meta modelling framework. We extended DoME with Web Service execution capabilities and plan to link the prototype to existing Enterprise Service Bus (ESB) implementations. Using such an infrastructure would allow to integrate existing technologies if provided in Web service form.

References

- [1] Michael Adams, Arthur H. M. ter Hofstede, David Edmond, and Wil M. P. van der Aalst. Worklets: A Service-Oriented Implementation of Dynamic Flexibility in Workflows. In *Proc. OTM*, LNCS 4275, pages 291–308. Springer, 2006.
- [2] Patrick Albert, Laurent Henocque, and Mathias Kleiner. Configuration-Based Workflow Composition. In *Proc. ICWS*, pages 285–292. IEEE Press, 2005.
- [3] D. Ardagna, M. Comuzzi, E. Mussi, B. Pernici, and P. Plebani. PAWS: A Framework for Executing Adaptive Web-Service Processes. *IEEE Software*, 24(6):39–46, 2007.
- [4] Steffen Balzer, Thorsten Liebig, and Matthias Wagner. Pitfalls of OWL-S – A practical Semantic Web Use Case. In *Proc. ICSOC*, pages 289–298, 2004.
- [5] Boualem Benatallah and Hamid Reza Motahari-Nezhad. ServiceMosaic Project: Modeling, Analysis and Management of Web Services Interactions. In *Proc. APCCM*, volume 53 of *CRPIT Series*, pages 7–9. ACS, 2006.

²Domain Modelling Environment
(<http://www.htc.honeywell.com/dome/>)

- [6] Eric D. Browne, Michael Schrefl, and James R. Warren. Goal-Focused Self-Modifying Workflow in the Healthcare Domain. In *Proc. HICSS*, 2004.
- [7] Marin Dimitrov, Alex Simov, Sebastian Stein, and Mihaïl Konstantinov. A BPMO Based Semantic Business Process Modelling Environment.
- [8] Alexander Felfernig, Gerhard Friedrich, Dietmar Jannach, and Markus Stumptner. Consistency-based diagnosis of configuration knowledge bases. *Artificial Intelligence*, 152(2):213–234, 2004.
- [9] Alexander Felfernig, Gerhard Friedrich, Dietmar Jannach, Markus Stumptner, and Markus Zanker. Transforming UML Domain Descriptions into Configuration Knowledge Bases. In *Knowledge Transformation for the Semantic Web*, pages 154–168. IOS Press, 2003.
- [10] Gerhard Fleischanderl, Gerhard E. Friedrich, Alois Haselböck, Herwig Schreiner, and Markus Stumptner. Configuring Large Systems Using Generative Constraint Satisfaction. *IEEE Intelligent Systems*, 13(4):59–68, 1998.
- [11] Asunción Gómez-Pérez, Rafael González-Cabero, and Manuel Lama. ODE SWS: A Framework for Designing and Composing Semantic Web Services. *IEEE Intelligent Systems*, 19(4):24–31, 2004.
- [12] Roy Grønmo and Michael C. Jaeger. Model-Driven Semantic Web Service Composition. In *Proc. APSEC*, pages 79–86. IEEE Press, 2005.
- [13] Georg Grossmann. *Horizontal and Vertical Integration of Object Oriented Information Systems Behaviour*. PhD thesis, School of Computer and Information Science, University of South Australia, 2008.
- [14] Georg Grossmann, Michael Schrefl, and Markus Stumptner. Modelling Inter-Process Dependencies with High-Level Business Process Modelling Languages. In *Proc. APCCM*, volume 79 of *CRPIT Series*. ACS, 2008.
- [15] Rachid Hamadi, Boualem Benatallah, and Brahim Medjahed. Self-adapting recovery nets for policy-driven exception handling in business processes. *Distributed and Parallel Databases*, 23:1–44, 2008.
- [16] Ahlem Ben Hassine, Shigeo Matsubara, and Toru Ishida. A Constraint-Based Approach to Horizontal Web Service Composition. In *Proc. ISWC*, pages 130–143, 2006.
- [17] Vipul Kashyap, Christoph Bussler, and Matthew Moran. *The Semantic Web – Semantics for Data and Services on the Web*. Data Centric Systems and Applications (DCSA). Springer, 2008.
- [18] Dominik Kuropka, Peter Tröger, Steffen Staab, and Mathias Weske, editors. *Semantic Service Provisioning*. CAT-SWS-1. Springer, 2008.
- [19] Freddy Lécué and Alain Léger. Semantic Web Service Composition Based on a Closed World Assumption. In *Proc. ECOWS*, pages 233–242. IEEE Press, 2006.
- [20] Lei Li and Ian Horrocks. A Software Framework for Matchmaking Based on Semantic Web Technology. In *Proc. of WWW Conference*, pages 331–339. ACM Press, 2003.
- [21] Peter Mangan and Shazia Sadiq. A Constraint Specification Approach to Building Flexible Workflows. *JRPIT*, 35(1):21–39, 2003.
- [22] Wolfgang Mayer, Rajesh Thiagarajan, and Markus Stumptner. Service Composition as Generative Constraint Satisfaction. In *Proc. ICWS*, pages 888–895, 2009.
- [23] Sheila McIlraith and Tran Cao Son. Adapting Golog for the Composition of Semantic Web Services. In *Proc. KR*, pages 482–493, 2002.
- [24] Martin Necasky and Jaroslav Pokorný. Designing Semantic Web Service using Conceptual Model. In *Proc. of ACM SAC*, pages 2243–2247. ACM Press, 2008.
- [25] Michael P. Papazoglou and Willem-Jan van den Heuvel. Service oriented architectures: approaches, technologies and research issues. *The VLDB Journal*, 16(3):389–415, 2007.
- [26] Marco Pistore, Annapaola Marconi, Piergiorgio Bertoli, and Paolo Traverso. Automated Composition of Web Services by Planning at the Knowledge Level. In *Proc. IJCAI*, pages 1252–1259, 2005.
- [27] Günter Preuner, Christian Eichinger, and Michael Schrefl. Static-Dynamic Integration of External Services into Generic Business Processes. In *Proc. ICSNW*, LNCS 3226, pages 263–277, 2004.
- [28] M. Reichert, P. Dadam, S. Rinderle-Ma, A. Lanz, R. Pryss, M. Predeschly, J. Kolb, L.T. Ly, M. Jurisch, U. Kreher, and K. Goeser. Enabling Poka-Yoke Workflows with the AristaFlow BPM Suite. In *Proc. BPM Demonstration Track*, CEUR WS Vol.489, 2009.
- [29] Dumitru Roman, Uwe Keller, Holger Lausen, Jos de Bruijn, Rubn Lara, Michael Stollberg, Axel Polleres, Cristina Feier, Christoph Bussler, and Dieter Fensel. Web Service Modeling Ontology. *Applied Ontology*, 1(1):77–106, 2005.
- [30] M. Sabou, D. Richards, and S. van Splunter. An experience report on using DAML-S. In *Proc. of WWW workshop on E-Services and the Semantic Web (ESSW)*, 2003.
- [31] Michael Schrefl and Markus Stumptner. Behavior-consistent Specialization of Object Life Cycles. *ACM TOSEM*, 11(1):92–148, 2002.
- [32] Evren Sirin, Bijan Parsia, Dan Wu, James Hendler, and Dana Nau. HTN Planning for Web Service Composition Using SHOP2. *Journal of Web Semantics*, 1(4):377–396, 2004.
- [33] Rajesh Thiagarajan and Markus Stumptner. Service Composition With Consistency-based Matchmaking: A CSP-based Approach. In *Proc. ECOWS*, pages 23–32, 2007.
- [34] Michele Trainotti et al. ASTRO: Supporting Composition and Execution of Web Services. In *Proc. ICSOC*, pages 495–501, 2005.
- [35] Kees van Hee et al. Adaptive Workflows for Healthcare Information Systems. In *Proc. BPM Workshops*, LNCS 4928, pages 359–370. Springer, 2008.
- [36] Bin Yu, Chen Zhang, and Yang Zhao. Transform from Models to Service Description Based on MDA. In *Proc IEEE APSCC*, pages 605–608. IEEE Press, 2006.

The *E-Decisional Community*: An Integrated Knowledge Sharing Platform

Leonardo Mancilla-Amaya¹, Cesar Sanín², Edward Szczerbicki²

School of Engineering, Faculty of Engineering and Built Environment
The University of Newcastle, Australia
University Drive, Callaghan, 2308, NSW

¹leonardo.mancilla@studentmail.newcastle.edu.au

²{Cesar.Sanin, Edward.Szczerbicki}@newcastle.edu.au

Abstract

Knowledge Management (KM) has become a key success factor in diverse fields, given the importance of knowledge as a significant organizational asset. In order to solve problems and support complex decision-making processes, knowledge and experience have to be transmitted across individuals, business units and organizations. Thus, *Knowledge Sharing (KS)* can be considered as the basic element of any knowledge-oriented process: *KS* fosters collaboration in complex environments, and facilitates experiential knowledge discovery, distribution and use. This paper presents a proposal for the development of a *Community of Practice (CoP)* called the *E-Decisional Community*. This community uses a domain-independent knowledge representation called *Set of Experience Knowledge Structure (SOEKS)*, and captures the decisional fingerprint inside organizations using *Decisional DNA*. It is based on principles from different technologies namely *Software Agents*, *Grid* and *Cloud* computing, in order to provide an autonomous, intelligent and coordinated large-scale *KS* environment. The *E-Decisional Community* biggest concern is to promote experiential knowledge evolution and sharing through generations of decision-makers, aiming at the creation of a marketplace where knowledge is provided as a service.

Keywords: Knowledge Management, Software Agents, Grid Computing, Cloud Computing, Knowledge Engineering, Smart Knowledge Management System, Decisional DNA, Set of Experience Knowledge Structure.

1 Introduction

Acquiring knowledge, representing it in an explicit and formal way, and supplying suitable mechanisms to re-use it and improve it inside organizations is a complex task. The *Smart Knowledge Management System (SKMS)* defines the processes and components required to capture,

store, improve, re-use, and transmit experience through generations of decision makers.

Nowadays, organizations need to share their knowledge with others when pursuing a common objective. As a result *KM* has become a critical element for organizations, given the importance of knowledge as a major asset that guarantees competitive advantage in a rapidly changing, economic-driven environment (Zhang et al., 2008a).

Several theories and proposals on *Knowledge Sharing (KS)* can be found in literature, and all of them are a valuable contribution to the area of *Knowledge Engineering*. They are concerned with providing technical support for *KS* between entities in various ways and using different approaches such as *software agents*, folksonomies, social networks and many more.

However, there is not a *KS* approach that captures the experience from multiple formal decision events, and transmits it across different generations of workers, using the vision of the *Set of Experience Knowledge Structure* and *Decisional DNA* (Maldonado Sanín, 2007). To achieve this, *SKMS* proposes a *Community of Practice (CoP)*. In this *CoP*, similar systems interact with each other creating new knowledge, and thus, new experience.

This paper describes the *E-Decisional Community*, our proposal for *KS* among individuals and organizations. This is a work currently developed by the *Knowledge Engineering Research Team (KERT)*, at The University of Newcastle, Australia.

The following sections are structured as follows: section two presents a theoretical background on basic concepts around our work; section three describes the vision, features and design for the *E-Decisional Community*. Finally, section four presents some conclusions and future work.

2 Background

This section presents the conceptual elements constituting the foundations on which our proposal is based. We will briefly describe topics in *Knowledge Management (KM)*, *software agents*, *Grid* and *Cloud* computing

2.1 Knowledge Sharing

These days, economy and competition are based on knowledge. The ability to learn from past experiences and adapt to rapidly changing conditions determines which organizations will prevail. Consequently, managers are more conscious about the importance of knowledge in their enterprises; thus, giving a higher priority to *KM* related activities.

Copyright (c)2010, Australian Computer Society, Inc. This paper appeared at the Seventh Asia-Pacific Conference on Conceptual Modelling (APCCM 2010), Brisbane, Australia, January 2010. Conferences in Research and Practice in Information Technology, Vol. 110. Sebastian Link and Aditya K. Ghose, Eds. Reproduction for academic, not-for profit purposes permitted provided this text is included.

One of those activities is *Knowledge Sharing (KS)* and dissemination. *KS* is a key factor, because knowledge is only useful if it is accessible to all users and can be used to solve problems and make decisions (Lao et al., 2008)

According to Hustad (2004), *KS* can be performed at different levels: between individuals, from individuals to groups, between groups and from groups to organizations. In fact, Vidou et al. (2006) state that an organization is comprised of many interconnected groups of interest, called *Communities of Practice (CoPs)*. *CoPs* are: "groups of people who share a passion for something that they know how to do, and who interact regularly in order to learn how to do it better" (Wenger, 2004: p. 2).

Nowadays, such groups can perform their activities using the latest advances in technology. Many tools and proposals have been developed in order to support collaboration and knowledge sharing using different approaches: ontologies, folksonomies, wikis and social networks are some models that support *KS* (Kings et al., 2007). Research projects like SQUIDZ (Kings et al., 2007), Knowledge Spaces (Zhang et al., 2008b), Palette (Vidou et al., 2006), Jasper (Davies et al., 1998) or Wikipedia (Wikimedia-Foundation, 2009) are just a few clear examples of *KS* technologies of everyday use.

Improving *KS* by means of automated distribution mechanisms, and using a single and domain independent knowledge representation still remains as a research area to be explored.

2.2 Smart Knowledge Management System, SOEKS and Decisional DNA

Managers, and decision-makers in general, base their current decisions on lessons learned from previous similar situations (Sanin and Szczerbicki, 2005a). However, much of an organization's experience is not properly capitalized because of inappropriate knowledge administration; this leads to high-response times and lack of flexibility to adapt in dynamic environments.

The *Smart Knowledge Management System (SKMS)* is a platform that defines a set of four macro-processes and components with the objective of supporting experiential knowledge creation, store, re-use, improvement and distribution inside organizations (Maldonado Sanín, 2007). The *SKMS* dynamically transforms large amounts of data and information from diverse sources into knowledge, supporting decision making processes at any level of the organizational hierarchy.

This platform uses a standard, flexible and domain-independent knowledge representation called *Set of Experience Knowledge Structure (SOEKS)* or shortly *SOE*. Each *SOE* represents a single formal decision event, and after being transformed by the *SKMS* macro-processes, many *SOEKS* comprise a *Decisional DNA* strand of an organization. Consequently, *Decisional DNA* captures the inference strategies of enterprises (Sanin and Szczerbicki, 2008).

Given that formal decision events are meant to be kept in an explicit way inside the *SKMS* standard representation means, for instance *Ontology Web Language (OWL)* and *XML*, are used for such matter. These representation mechanisms facilitate knowledge sharing and transportation, mainly because they describe

human-cognition with a high level of abstraction and are broadly accepted standards (Sanin and Szczerbicki, 2005b; Sanin et al., 2007).

The *SOE* has been successfully applied in industrial environments, specifically for maintenance purposes, in conjunction with *Augmented Reality (AR)* techniques (Toro et al., 2007). Additionally, implementations in the fields of finances and energy research have evaluated the performance of the *SOE* and demonstrated that it is an optimal multi-domain knowledge representation (Sanin et al., 2009).

2.3 Software Agents

Software Agents (or simply *agents*) represent an active research area where many efforts have been made to develop human-like behavior in computer systems. Basically, an *agent* is a software or hardware component that acts without external intervention to achieve a set of well-defined goals on behalf of its user (Nwana, 1996). According to Wooldridge and Jennings (1995), this technology possesses some features that have made it a suitable approach for modeling complex systems; these characteristics are: (i) Autonomy, (ii) Social ability, (iii) Reactivity, (iv) Pro-activeness, (v) Mobility, (vi) Veracity, (vii) Benevolence and (viii) Rationality. As a consequence, *agents* are used in many *KM* approaches because they provide an appropriate way for modeling organizational knowledge. As mentioned by Van Elst et al. (2004), knowledge distribution, low-priority *KM* goals, complex interactions and dynamic environments are some characteristics that justify the use of *agents* in knowledge management solutions.

Previous work on *KM* based on *agents* was concerned with text mining, automated suggestions and smart document access (De Rezende et al., 2007; Kim et al., 2007), Distributed Organizational Memories (Abecker et al., 2003; Gandon and Dieng-Kuntz, 2002), *agent*-based architectures (Vizcaino et al., 2007), and use of ontologies in *Multi-Agent Systems (MAS)* among others.

However, to the best of our knowledge, existing proposals do not address experiential knowledge representation, management, evolution and distribution in the way the *SKMS* does using *SOEKS* and *Decisional DNA*.

2.4 Grid Computing

Grids provide a robust and highly scalable infrastructure for multi-purpose problem solving tasks. This is achieved by sharing resources and coordinating efforts in *Virtual Organizations (VOs)* (Foster, 2002).

As the complexity of problems undertaken by users everyday increases, the requirements surrounding the *Grid* have become more complex and demanding. Efforts like the *Semantic Grid* provide new capabilities to users, and also, as mentioned by De Roure et al. (2005), new research opportunities: semantic service description, smart interaction, autonomous behavior, knowledge technologies, among others, are topics that should be addressed by future efforts.

In fact, knowledge technologies for *Grid* environments are getting more attention from the scientific community. Approaches like the *Knowledge Grid* presented by Zhuge

(2008) propose a highly distributed collaborative environment, where explicit knowledge resources are managed to support decision-making processes and cooperative work.

Regardless of its powerful attributes, *Grid* technology concepts need to be improved with ideas from other areas in order to fulfill the elements proposed by De Roure et al. (2005). *Software agents*, as described in the previous section, possess some unique attributes which are common to the research topics defined for the *Grid* (Foster et al., 2004). Elements like autonomous behavior, community management and advanced coordination and negotiation techniques are being used in the *Grid* to make it more resilient and efficient; two examples of this are presented by Gil (2006) and Norman et al. (2004), who described new ways to make *Grid* environments more robust and to dynamically manage *VOs* in electronic commerce scenarios, both based on *agents*.

In conclusion, *Grids* have the potential to provide large-scale knowledge oriented services, to support critical decision-making processes. This may be achieved by developing advanced mechanisms based on other technologies.

2.5 Cloud Computing

There are many definitions about *Cloud Computing (CC)*, but until now, there is no consensus on what *CC* is (Foster et al., 2008; Mc Evoy and Schulze, 2008; Youseff et al., 2008).

CC is closely related to *Grids*, and according to Foster et al. (2008), *Clouds* are an evolution of *Grid* technology, but with different requirements in areas such as business model, applications and abstractions. *CC* is a service oriented approach that takes full advantage of current developments in virtualization, Semantic Web and *Grid* computing to provide different services on-demand. These services most frequently are: applications (i.e. *SaaS: Software as a Service*), platforms (i.e. *PaaS: Platform as a Service*), and hardware/software infrastructures (i.e. *IaaS: Infrastructure as a Service*).

There is an increasing interest in the scientific community regarding *CC*. Some proposals for the implementation of *Cloud* based environments have been developed, for example Zhan et al. (2009) presented a cloud computing system software to consolidate heterogeneous workloads in organizations. Others propose the use of human organizational principles to develop client *CC* environments, which facilitate knowledge and experience transfer between people (Hewitt, 2008).

Moreover, *KM* systems based on the *Cloud* vision have been envisaged by Delic and Riley (2009); knowledge *Clouds* will interconnect users across several organizations and data centers, thus supporting the “*Intelligent Enterprise*”. This *Intelligent Enterprise* is an entity (*agent*) that behaves intelligently and used the Internet as its base for providing services and performing operations.

In spite of the existing work, more research needs to be done on *CC* and the way it supports knowledge based tasks inside organizations.

3 The E-Decisional Community Proposal

Due to the increasing need of many organizations to share knowledge, not only among them, but inside their different business units, the *SKMS* proposes a *Community of Practice (CoP)* called *E-Decisional Community*. In this *CoP* similar systems interact with each other creating new knowledge and experience, and thus, extend the limits of the *SKMS*. This section presents different aspects and features of the *E-Decisional Community* proposal, such as its global vision and suggested design.

3.1 Vision

The *E-Decisional Community* is concerned with experiential knowledge represented as *Decisional DNA* and *SOEKS*, and the way this novel knowledge representation is passed on and evolves through generations of decision makers.

Extending the limits of the *SKMS*, autonomous and smart knowledge sharing mechanisms must be developed. This will make the *SKMS* capable of discovering knowledge based on real-world data and information provided by users. However, the *E-Decisional Community* is not a data/text-mining tool, or just a smart document-repository. It is meant to be a dynamic and scalable platform for problem solving activities among individuals and organizations.

Our proposal is based on concepts from *software agents*, *Grid* and *Cloud* computing. Modeling of complex human interactions, coordinated knowledge sharing, team formation, autonomous actions and on-demand services, are just some of the concepts surrounding the *E-Decisional Community*.

Figure 1 illustrates the global vision for the *E-Decisional community*. Our proposal for a *CoP* allows individual agents (i.e. workers), as well as groups (i.e. business units), to contribute with their experience to the construction of collective knowledge. To achieve this, the entities involved must share knowledge and interact in a coordinated fashion. Continuous participation in the *CoP* to solve problems promotes experience transmission, and knowledge discovery and re-use.

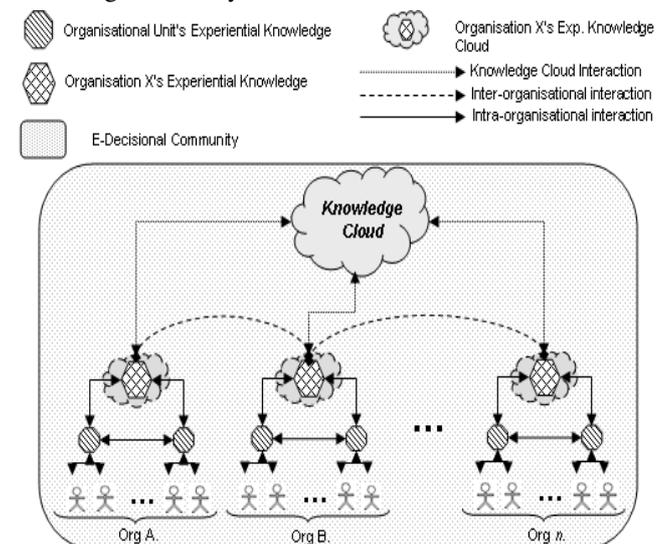


Figure 1: The E-Decisional Community Vision

In our approach, organizations may create their own *clouds*, interacting with other and providing knowledge-based on-demand services. These interactions are motivated by economic principles for instance alliances, and producer-consumer relationships. In fact, the interconnection of different business partners will generate a much larger *Knowledge Cloud*, stimulating the creation of a large-scale marketplace, in which knowledge is the main asset, and it is sold or exchanged as part of collective strategies.

3.2 Features

In order to support decision-making processes in organizations, the *E-Decisional Community* provides the following features:

–*People-oriented*: the platform is a tool that provides knowledge-oriented problem solving, in which people can take advantage of today's computational improvements to support complex decision-making processes in organizations.

–*Agent-based capabilities*: characteristics from software agents are provided to model complex human interactions and support intelligent *KM* processes, in highly distributed and complex environments.

–*Constant Evolution*: knowledge evolution and refinement is an intrinsic characteristic of *SKMS*, and is achieved by constantly updating existing experiences with data from the real world which is fed by the users.

–*Community formation*: there are tasks that cannot be executed successfully in an individual fashion; therefore, grouping, based on objectives and knowledge, is supported, both in a static and dynamic manner.

–*Well-defined interactions*: agents and services participating interact in an orderly fashion. Therefore, protocols and interaction schemes are defined to establish proper communication, role assignment and permission policies.

–*Conflict resolution*: negotiation techniques and conflict resolutions mechanisms are provided to solve disputes caused by accessing scarce resources or by conflicting beliefs and experiences.

–*Multimedia information*: multimedia information should be a source for experiential knowledge extraction in today's organizations. The *E-Decisional Community* aims at including multimedia resources as part of the decisional fingerprint of organizations.

–*Security, trust and provenance*: it is clear that a secure environment is a key requirement for any distributed system these days; moreover when Internet is used as the primary communication channel. Also, knowledge collected must be reliable to make the right decisions; that is why the concept of *Decisional Trust* (Sanin and Szczerbicki, 2008) is extended to include more features that reflect human-like behavior. Finally, as knowledge is transformed, it might be useful to have a way to trace the derivation history of it, which can be used to generate performance measures for the system, or for possible auditing and/or legal purposes.

3.3 Design

3.3.1 Conceptual Model

The proposed model for the *E-Decisional Community* is comprised by four layers: Knowledge-based Application Layer, Collective and Individual Management Layers, and Knowledge-Oriented Services Layer. The platform is built on top of the *SKMS*, extending its limits always using *Decisional DNA* and *SOEKS* for knowledge transmission and representation. Figure 2 depicts the conceptual model for the *KS* platform.

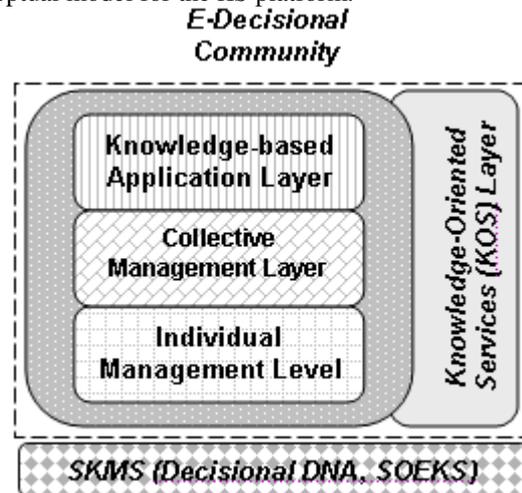


Figure 2: E-Decisional Community Conceptual Model

All the comprising layers make extensive use of *Knowledge Oriented Services (KOS)* to provide appropriate *KS* capabilities. Each layer has a set of responsibilities and characteristics as follows:

–*Knowledge-based Application Layer (KAL)*: this layer provides end-user access to the platform functionality. At this level, Web 2.0 or mobile applications may be used by workers to interact with other individuals or groups in order to solve problems and make decisions, as well as to feed the system with data based on their daily activities. Knowledge-based applications can use complementing technologies such as software agent or *Augmented Reality (AR)*, to promote interaction with the environment and capture experiential data from different sources.

–*Collective Management Layer (CML)*: dynamic teamwork management, inter and intra-organizational interactions, cooperation and global policies, among other mechanisms, are provided by this layer to support collaborative work. Groups and organizations are represented as heterogeneous *MAS*, thus, multiple *MAS* can interact between each other using well-defined protocols and policies provided by the *CML*. During the interaction process, new collective experiential knowledge is created or inferred, increasing the expertise level of the entire enterprise. *VO* formation and management based on knowledge objectives is also supported at this level.

–*Individual Management Layer (IML)*: individuals in an organization are represented by *software agents*. As a consequence, knowledge exchange, collaboration and dynamic teamwork formation can be performed in an autonomous fashion, resembling human behavior.

Moreover, *agents* can remember users' behavior in order to proactively initiate knowledge-based tasks. In this layer, *agents* are an entry point to the knowledge-oriented services provided by the platform, and are able to create an individual's decisional fingerprint, that can be used, for example, as a performance or reputation indicator. This layer provides all the required mechanisms to support the aforementioned functionality.

–*Knowledge-Oriented Services Layer (KOS)*: knowledge-oriented services deliver a wide range of features oriented to promote proper *KS* inside organizations. Access to *Decisional DNA* and *SOEKS* repositories, yellow and white pages directories, role definitions, trust and reputation services, among others, are provided by the *KOS* layer. Additionally, this layer defines the interoperability elements required to perform inter-cloud communication in order to provide on-demand access to users across different organizations. Coordinated execution of *KOS* is defined by the interaction protocols of the *CML* and *IML*.

–*SKMS, Decisional DNA and SOEKS*: this is not a layer of the *E-Decisional Community*; however the four macro processes defined by the *SKMS* (diagnosis, prognosis, solution and knowledge), along with its knowledge capturing and representation mechanisms, constitute the foundation on which the *KS* will be constructed. More details about the *SKMS* proposal can be found in (Maldonado Sanín, 2007; Sanin and Szczerbicki, 2005a; Sanin and Szczerbicki, 2005b; Sanin and Szczerbicki, 2008; Sanin et al., 2007)

3.3.2 Conceptual Architecture

Figure 3 illustrates the proposed conceptual architecture, based on the model previously described. The objective of the conceptual architecture is to identify the elements and global relationships that might be present in the *SKMS* knowledge sharing environment.

In the *E-Decisional Community*, users access the *KS* platform by means of knowledge-based applications. As mentioned in the *KAL* description, Web 2.0 applications are used for this purpose, as a consequence, protocols like HTTP or SOAP, and architectural approaches such as REST (Representational State Transfer) must be employed.

Applications in the *E-Decisional Community* make use of *software agents*; each user is represented by a *Personal Agent (PA)* which acts in his/her behalf inside the community, and provides access to *KOS*. *PAs* know about their roles, interaction restrictions, trust relationships and reputation of other entities by means of specialized *KOS*.

Numerous *PAs* may share a temporary, or permanent, interest for a specific topic which leads to a dynamic group formation. When various *agents* form a coalition (i.e. a *MAS*), they are represented by a *Group Agent (GA)*. Consequently, multiple *MAS* are viewed as complex agents that interact similarly to how individual *PAs* do, but with higher-level goals and interests. Also, *GAs* are able to solve more complex problems or take critical decisions, because they count on the experience from many individuals. Dynamic teamwork formation and life-cycle management is also supported by *KOS*.

Interaction between enterprises is also carried out using *GAs* that represent each individual organization. At

this level, the strategic experience of each organization may be shared or sold using a *Cloud* scheme for on-demand usage.

The architecture defines six *KOS* categories oriented to facilitate experience diffusion. Services may be provided on-demand for external or internal entities, and an organization can provide one specific type of *KOS* over the *Cloud* as a profit opportunity. The service categories are:

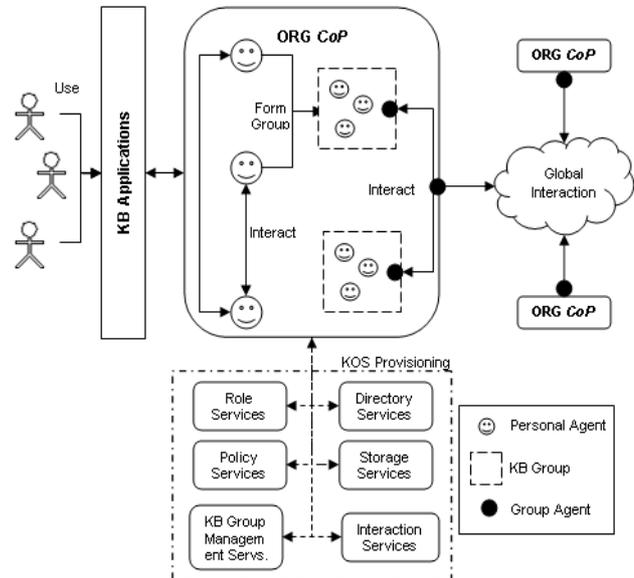


Figure 3: E-Decisional Community Conceptual Architecture

–*Role Services*: this service category acts as a repository where organizational roles are mapped, defining the corresponding behaviors, responsibilities, capacities, goals and permissions. Roles can be dynamically queried and executed by any entity.

–*Directory Services*: provides white and yellow pages services, in order to query for individuals or knowledge resources/services, respectively.

–*Policy Services*: stores the organizational policies for dealing with different issues. For example, policies for uncertainty management, service distribution, rewards/punishment, and others, are stored for dynamic querying.

–*Knowledge Storage Services*: these services provide storage and retrieval capabilities for individual, collective knowledge and organizational experiential knowledge. Providing secure, reliable, location-independent and fast access to *SOEKS* and *Decisional DNA* structures is the main concern of this service category.

–*Knowledge-based Group Management Services*: dynamic formation of groups based on knowledge objectives is a key feature of the *E-Decisional Community*. As a consequence, a specific category of services is devoted to support this aspect. Trust, negotiation, reputation, quality of service and service level agreements constitute the key elements that are provided to support cooperative problem-resolution and decision-making.

–*Interaction Services*: these set of services contains the definition of all the protocols that are used inside the *CoP*, to guarantee orderly interaction. These protocols are

employed to coordinate the communication flow between organizational units, organizational units and the enterprise, and among enterprises.

4 Conclusions and Future Work

We presented a design proposal for a *CoP* that allows sharing experiential knowledge across different organizational levels. It uses a standard knowledge representation called SOEKS, which in turn, comprises Decisional DNA (i.e. a collection of multiple SOEKS).

This proposal, called *E-Decisional Community*, is based upon the principles of different computing technologies, namely: *software agents*, *Grid* and *Cloud* computing. Modeling of complex human interactions, autonomous and intelligent behavior, coordinated knowledge sharing, and on-demand service provisioning are some of the concepts behind the *CoP*. As a consequence of this approach, eight global features have been presented as the main concerns in our work.

Since the *E-Decisional Community* proposal is at its early development stages, further research and refinement of the elements presented in this paper remains to be done. Some future tasks are:

- Evaluation of different *agent* architectures to determine which approach is more appropriate for smart SOEKS and *Decisional DNA* sharing. Also, validation of the candidate architectures with a case study implemented using a Java-based *agent* framework is required. Java is preferred because the first version of the SOEKS API was developed using this language, and it allows for OS independency.

- Comparison and evaluation of the different interaction, negotiation, coordination and conflict resolution protocols for *agents* that might be used inside the *E-Decisional Community*. If new protocols are required for knowledge-based collaboration, then a proposal will be formalized.

- Refinement of the requirements for dynamic knowledge-based teamwork formation. Requirements like protocols, policies and life cycle management need to be described in detail.

- Establish appropriate human-machine interaction mechanisms to capture experience from different sources other than data warehouses or files. Currently, *ARTag* (Fiala, 2009) is being evaluated by the *KERT* as an *Augmented Reality* tool for this purpose.

- Technical review of *Cloud* and *Grid* tools and middleware and design principles to determine their viability for a future implementation of the *CoP*, both at a conceptual and technical level.

- Research on experience extraction and inference from multimedia files.

- Complement the work around *Decisional Trust*, to include elements that allow entities to trust others, but not only in the virtual world. People-system trust relationships should be bidirectional, based on reputation and other measures that can be applied to human and virtual workers alike, and that can be used by either one of them to assess critical decisions.

5 References

- Abecker, A., Bernardi, A. and Van Elst, L. (2003) Agent technology for distributed organizational memories. *Proceedings of the 5th International Conference on Enterprise Information Systems - ICEIS 2003*. Angers, France.
- Davies, N. J., Stewart, R. S. and Weeks, R. (1998) Knowledge Sharing Agents Over the World Wide Web. *BT Technology Journal*, **16**, 104-109.
- De Rezende, J. L., Pereira, V. B., Xexeo, G. and De Souza, J. M. (2007) Olympus: Personal Knowledge Recommendation Using Agents, Ontologies and Web Mining. In *Computer Supported Cooperative Work in Design III*. 53-62. Shen, W., Luo, J., Lin, Z., Barthès, J.-P. A. and Hao, Q. (Eds.), Springer-Verlag Berlin Heidelberg.
- De Roure, D., Jennings, N. R. and Shadbolt, N. R. (2005) The Semantic Grid: Past, Present, and Future. *Proceedings of the IEEE*, **93**, 669-681.
- Delic, K. A. and Riley, J. A. (2009) Enterprise Knowledge Clouds: Next Generation KM Systems? *eKNOW '09. International Conference on Information, Process, and Knowledge Management*. Cancun, Mexico.
- ARTag, Fiala, Mark <http://www.artag.net/>. July 30, 2009
- Foster, I. (2002) What is the grid? A three point checklist. *GRID today*, **1**, 22-25.
- Foster, I., Jennings, N. R. and Kesselman, C. (2004) Brain Meets Brawn: Why Grid and Agents Need Each Other. *AAMAS '04: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*. New York, New York, IEEE Computer Society.
- Foster, I., Yong, Z., Raicu, I. and Lu, S. (2008) Cloud Computing and Grid Computing 360-Degree Compared. *Grid Computing Environments Workshop, 2008. GCE '08*. Austin, Texas.
- Gandon, F. and Dieng-Kuntz, R. (2002) Distributed Artificial Intelligence for Distributed Corporate Knowledge Management. In *Cooperative Information Agents VI. 6th International Workshop, CIA 2002 Madrid, Spain, September 18-20, 2002 Proceedings*. 202-217. Klusch, M., Ossowski, S. and Shehory, O. (Eds.), Springer-Verlag. 735865
- Gil, Y. (2006) On agents and grids: Creating the fabric for a new generation of distributed intelligent systems. *Web Semantics: Science, Services and Agents on the World Wide Web*, **4**, 116-123.
- Hewitt, C. (2008) ORGs for scalable, robust, privacy-friendly client cloud computing. *IEEE Internet Computing*, **12**, 96-99.
- Hustad, E. (2004) Knowledge networking in global organizations: the transfer of knowledge. *Proceedings of the 2004 SIGMIS conference on Computer personnel research: Careers, culture, and ethics in a networked environment*. Tucson, AZ, USA, ACM.
- Kim, H. L., Choi, J. H., Kim, H. G. and Hwang, S. H. (2007) WANT: A Personal Knowledge Management

- System on Social Software Agent Technologies. *Proceedings of the 1st KES International Symposium on Agent and Multi-Agent Systems: Technologies and Applications*. Wroclaw, Poland, Springer-Verlag.
- Kings, N. J., Gale, C. and Davies, J. (2007) Knowledge Sharing on the Semantic Web. In *The Semantic Web: Research and Applications. 4th European Semantic Web Conference, ESWC 2007, Innsbruck, Austria, June 3-7, 2007. Proceedings* 281-295. Springer Berlin / Heidelberg.
- Lao, G., Xiao, L., Wang, Q. and Qin, Z. (2008) Research on organizational knowledge sharing framework based on CAS theory. *2008 International Conference on Service Systems and Service Management*.
- Maldonado Sanín, C. A. (2007) *Smart knowledge management system*. PhD Thesis, Faculty of Engineering and Built Environment - School of Mechanical Engineering, University of Newcastle. Supervisor: Szczerbicki, E. 177 p
- Mc Evoy, G. V. and Schulze, B. (2008) Using clouds to address grid limitations. *Proceedings of the 6th international workshop on Middleware for grid computing*. Leuven, Belgium, ACM.
- Norman, T. J., Preece, A., Chalmers, S., Jennings, N. R., Luck, M., Dang, V. D., Nguyen, T. D., Deora, V., Shao, J. and Gray, W. A. (2004) Agent-based formation of virtual organisations. *Knowledge-Based Systems*, **17**, 103-111.
- Nwana, H. S. (1996) Software agents: An overview. *Knowledge engineering review*, **11**, 205-244.
- Sanin, C. and Szczerbicki, E. (2005a) Set of Experience: A Knowledge Structure for Formal Decision Events. *Foundations of Control and Management Sciences*, **3**, 95-113.
- Sanin, C. and Szczerbicki, E. (2005b) Using XML for implementing set of experience knowledge structure. *International Conference on Knowledge-Base and Intelligent Information and Engineering Systems - KES*. Melbourne, Australia, Springer.
- Sanin, C., Szczerbicki, E. and Toro, C. (2007) An OWL ontology of set of experience knowledge structure. *Journal of Universal Computer Science*, **13**, 209-223.
- Sanin, C. and Szczerbicki, E. (2008) A Decisional Trust Implementation on a Maintenance System by the Means of Decisional DNA and Reflexive Ontologies. *WI-IAT '08. IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, 2008*.
- Sanin, C., Mancilla-Amaya, L., Szczerbicki, E. and Cayfordhowell, P. (2009) Application of a Multi-domain Knowledge Structure: The Decisional DNA. In *Intelligent Systems for Knowledge Management*. Nguyen, N. T. and Szczerbicki, E. (Eds.), Springer. In printing -To be published October 2009.
- Toro, C., Sanin, C., Vaquero, J., Posada, J. and Szczerbicki, E. (2007) Knowledge based industrial maintenance using portable devices and augmented reality. In *Knowledge-Based Intelligent Information and Engineering Systems. 11th International Conference, KES 2007, XVII Italian Workshop on Neural Networks, Vietri sul Mare, Italy, September 12-14, 2007. Proceedings, Part I*. 295-302. Apolloni, B., Howlett, R. J. and Jain, L. (Eds.), Springer Berlin / Heidelberg.
- Van Elst, L., Dignum, V. and Abecker, A. (2004) Towards agent-mediated knowledge management. In *Agent-Mediated Knowledge Management. International Symposium AMKM 2003, Stanford, CA, USA, March 24-26, Revised and Invited Papers* 1-30. Van Elst, L., Dignum, V. and Abecker, A. (Eds.), Springer Berlin / Heidelberg.
- Vidou, G., Dieng-Kuntz, R., El Ghali, A., Evangelou, C., Giboin, A., Tifous, A. and Jacquemart, S. (2006) Towards an ontology for knowledge management in communities of practice. In *Practical Aspects of Knowledge Management*. 303-314. Springer Berlin / Heidelberg.
- Vizcaino, A., Soto, J. P., Portillo, J. and Piattini, M. (2007) A Multi-agent Model to Develop Knowledge Management Systems. *HICSS 2007. 40th Annual Hawaii International Conference on System Sciences*.
- Wenger, E. (2004) Knowledge management as a doughnut: Shaping your knowledge strategy through communities of practice. *Ivey Business Journal*, **68**, 1-8.
- Wikipedia, Wikimedia Foundation. <http://www.wikipedia.org/>. July 28, 2009
- Wooldridge, M. and Jennings, N. R. (1995) Intelligent agents: Theory and practice. *Knowledge engineering review*, **10**, 115-152.
- Youseff, L., Butrico, M. and Da Silva, D. (2008) Toward a Unified Ontology of Cloud Computing. *Grid Computing Environments Workshop, 2008. GCE '08*.
- Zhan, J., Wang, L., Tu, B., Li, Y., Wang, P., Zhou, W. and Meng, D. (2009) Phoenix Cloud: Consolidating Heterogeneous Workloads of Large Organizations on Cloud Computing Platforms. *CoRR*, **abs/0906.1346**.
- Zhang, C., Tang, D., Liu, Y. and You, J. (2008a) A Multi-agent Architecture for Knowledge Management System. *FSKD '08: Fifth International Conference on Fuzzy Systems and Knowledge Discovery*.
- Zhang, G., Qu, Z. and Guo, A. (2008b) Using Folksonomies to Improve Peer-to-Peer Knowledge Sharing. *2008 International Symposiums on Information Processing (ISIP)*.
- Zhuge, H. (2008) The Knowledge Grid Environment. *Intelligent Systems, IEEE*, **23**, 63-71.

Hetero-Homogeneous Hierarchies in Data Warehouses

Bernd Neumayr¹

Michael Schrefl¹

Bernhard Thalheim²

¹ Department of Business Informatics - Data & Knowledge Engineering
Johannes Kepler University Linz, Austria
E-Mail: {neumayr,schrefl}@dke.uni-linz.ac.at

² Christian-Albrechts-University Kiel, Institute of Computer Science, Kiel, Germany
Email: thalheim@is.informatik.uni-kiel.de

Abstract

Data Warehouses facilitate multi-dimensional analysis of data from various data sources. While the original data sources are often heterogeneous, current modeling and implementation techniques discard and, thus, cannot exploit these heterogeneities.

In this paper we introduce *Hetero-Homogeneous Hierarchies* to model dimension hierarchies and cubes with inherent heterogeneities. Hetero-homogeneous hierarchies are hierarchies that are heterogeneous in regard to the schema of sub-hierarchies and homogeneous in regard to a minimal common schema shared by all sub-hierarchies.

Sub-dimension-hierarchies can be specialized to contain additional levels and additional non-dimensional attributes. Sub-cubes can be specialized towards additional measures, more fine-grained facts, and differing units of measure. We show how scale differences and conflicts due to multi-dimensional inheritance can be avoided and solved. We provide a formal definition of our approach together with a query/cube algebra.

Keywords: Multidimensional conceptual modeling, abstraction, specialization; Heterogeneous information; OLAP

1 Introduction

Data Warehouses facilitate multi-dimensional analysis of data integrated from various data sources. Available and interesting data is often heterogeneous concerning available measures, granularity of measures, units of measures, applicable rollup-levels, and interesting secondary information (non-dimensional attributes). However, to ease querying and storing multi-dimensional data, current modeling and implementation techniques force to fully homogenize available data according to a global multi-dimensional schema.

Our approach is summarized by the oxymoron term *hetero-homogeneous hierarchies*. A hetero-homogeneous hierarchy is a hierarchy with a single root node that is (1) homogeneous in regard to a minimal common schema shared by all sub-hierarchies, where a sub-hierarchy is a hierarchy rooted in a child of the root node, (2) heterogeneous in regard to the specialized schemas of sub-hierarchies.

We discuss our approach by a running example, starting with a homogeneous schema that can be

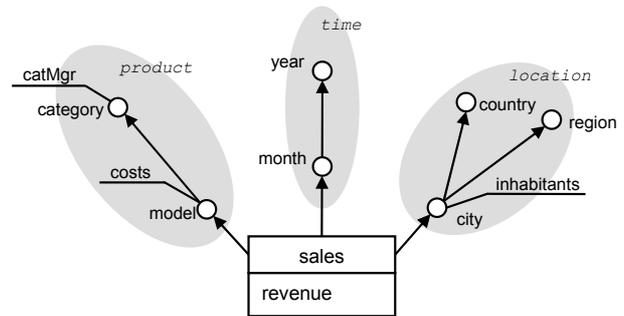


Figure 1: Homogeneous cube modeled with the Dimensional Fact Model

modeled using the Dimensional Fact Model (Golfarelli et al., 1998) (see Fig. 1). Consider a homogeneous *sales*-cube with dimensions *product*, *time*, and *location*. Dimension *product* defines dimension level *category* with non-dimensional attribute *catMgr* and dimension level *model* with non-dimensional attribute *costs*. The level-hierarchy of *product* defines *category* to be above *model*. Dimension *time* defines levels *year* above level *month*. Dimension *location* defines level *country* and level *region*, and level *city* with non-dimensional attribute *inhabitants*. Level *city* is below levels *country* and *region*, which are in no order to each other. The *sales*-cube defines a measure *revenue*.

Dimension hierarchies can be hetero-homogeneous with regard to

- *non-dimensional attributes*, e.g., in sub-hierarchy *Car* of dimension *product*, dimension instances at level *model* have an additional non-dimensional attribute *maxSpeed*.
- *additional levels*, e.g., in sub-hierarchy *Switzerland* of dimension *location* there is an additional level *kanton* between *city* and *country* and an additional level *store* below *city*.

Cubes can be hetero-homogeneous in that

- sub-cubes, such as *car sales in Switzerland 2009* may have *additional measures*, e.g., *quantity sold*,
- different sub-cubes may give *different units* for the same measure, e.g., values of measure *revenue* are provided in *swiss francs*
- base facts for various measures are provided at *mixed granularities*, e.g., base facts for *cheapestOffer* are provided at level *category*, *year*, *country* while base facts for measure *revenue* are provided at level *model*, *month*, *year*.

Copyright ©2010, Australian Computer Society, Inc. This paper appeared at the Seventh Asia-Pacific Conference on Conceptual Modelling (APCCM 2010), Brisbane, Australia, January 2010. Conferences in Research and Practice in Information Technology, Vol. 110. Sebastian Link and Aditya K. Ghose, Eds. Reproduction for academic, not-for profit purposes permitted provided this text is included.

- different sub-cubes may provide base facts for the same measure at *different granularities*, e.g., measure *revenue* originally defined for level *model*, *month*, and *city* is now available more detailed at level *model*, *month*, and *store*.

To represent such kind of situations one needs a design approach to represent hetero-homogeneous hierarchies in dimensions and cubes. Such an approach should allow for an instance-based specialization of dimensions and cubes.

In previous work (Neumayr et al., 2009) we introduced multilevel-objects (m-objects) and multilevel-relationships (m-relationships) to represent objects and relationships at multiple levels of abstraction. In this paper, we show how hetero-homogeneous hierarchies in data warehouses can be modeled by a revised and extended form of m-objects and m-relationships.

Hetero-homogeneous dimension hierarchies are modeled as concretization hierarchies of m-objects. Thereby an m-object encapsulates and arranges dimension levels in a partial order from abstract to concrete. Thereby, it describes itself and the common properties of the objects at each level of the dimension hierarchy beneath itself. An m-object that concretizes another m-object inherits dimension levels and non-dimensional attributes of the parent. It may also introduce additional levels and additional non-dimensional attributes. For modeling dimension hierarchies, we extend the original definitions of m-objects to partially ordered level hierarchies and consistency criteria that avoid conflicts due to multiple concretization.

Cube schemas as well as facts are modeled as m-relationships. M-relationships are analogous to m-objects in that they describe relationships between m-objects at multiple levels of abstraction. For modeling cube schemata and facts, we extend m-relationships from binary m-relationships (Neumayr et al., 2009) to n-ary m-relationships that may define measures and assert measure values. We define consistency criteria that avoid conflicts due to multiple inheritance and avoid overlapping primary fact instances.

Most current approaches to data warehousing are centered around the notion of a *cube*, our conceptual approach to modeling and querying data warehouses is centered around multi-level cubes (m-cubes). An *m-cube* represents a cube of cubes, given by the cartesian product of dimension levels and, on a more fine-grained level, a set of coordinates, given by the cartesian product of dimension instances.

We also introduce an m-cube-algebra with closed m-cube operations *dice*, *slice*, *import-union*, and *projection*; together with *fact-* and *cube-extraction* operations. Other common data warehouse operations like roll-up, drill-down, drill-across are subsumed by these operations. To cope with heterogeneous measure units we also support unit conversion. In order to exploit heterogeneities in m-cubes queries are typically double-staged: after selecting a sub-m-cube, using *dice*, the query can make use of additional schema information like additional measures, refined granularity, additional non-dimensional attributes, and additional cube levels.

The paper is structured as follows: in Sec. 2 and Sec. 3 we show how to model hetero-homogeneous dimension hierarchies and hierarchies of m-cubes, respectively, and provide structural definitions and consistency criteria. In Sec. 4 we show how to query m-cubes and introduce an m-cube-algebra. In Sec. 5 we briefly survey related work. In Sec. 6, which concludes the paper, we give an outlook on future work.

2 Hetero-Homogeneous Dimension Hierarchies

In this section we first revisit and extend m-objects (Neumayr et al., 2009) and, then, we show how to model hetero-homogeneous dimensions with them.

2.1 M-Objects revisited

An m-object, as originally introduced, encapsulates and arranges abstraction levels in a linear order from the most abstract to the most concrete one. Thereby, it describes itself and the common properties of the objects at each level of the concretization hierarchy beneath itself. An m-object specifies concrete values for the properties of its top-level. This top-level describes the m-object itself. All other levels describe common properties of m-objects beneath itself.

We now give revised definitions that support m-objects with a partial (non-linear) order of levels.

Definition 1 (M-Object). *An m-object o is described by a 6-tuple $(L_o, A_o, P_o, l_o, d_o, v_o)$ where $L_o \subseteq L_D$ is a set of levels from a universe of levels and $A_o \subseteq A_D$ is a set of attributes from a universe of attributes. The levels L_o are organized in a partial order, as defined by parent relation $P_o \subseteq L_o \times L_D$, which associates with each level its parent levels. Each attribute is associated with one level, defined by function $l_o : A_o \rightarrow L_o$, and has a domain, defined by function $d_o : A_o \rightarrow \text{datatypes}$. Optionally, an attribute has a value from its domain, defined by partial function $v_o : A_o \rightarrow V$, where V is a universe of data values, and $v_o(a) \in d_o(a)$ iff $v_o(a)$ is defined.*

An m-object has a single top-level, $\hat{l}_o := l \in L_o : \nexists l' \in L_o : (l, l') \in P_o$.

We say o is at level l , if l is its top-level. We further say level l' is a child of level l iff $(l', l) \in P_o$, and l' is a descendant of, or below, l iff $(l', l) \in P_o^+$, where P_o^+ is the transitive closure of P_o , and l' is a descendant of or the same as l iff $(l', l) \in P_o^*$, where P_o^* is the transitive-reflexive closure of P_o .

M-objects, levels, and attributes have names, defined by function $\text{name} : O \cup L \cup A \rightarrow \text{names}$, where names is the universe of names. Names of m-objects, attributes, and levels are unique within one dimension.

Example 1 (M-Object Car). Product category *car* (see Fig. 2) has three levels *category*, *brand*, and *model* and defines a value for attribute *catMgr*.

An m-object can *concretize* another m-object, which is referred to as its parent, by introducing new levels, introducing new attributes, and providing values for attributes. The concretizes-relationship comprises classification, generalization and aggregation. A concretization relationship between two m-objects does not reflect that one m-object is at the same time an *instance of*, *component of*, and *subclass of* another m-object as a whole. Rather, a concretization relationship has to be interpreted in a multi-faceted way. This is exemplified by the following example.

Example 2 (Concretization). M-object *Car* concretizes *Product*. The concretization relationship is to be interpreted in a multi-faceted way: m-object *Car* is instance of level *category* of m-object *Product* because level *category*, which is the first non-top-level of m-object *Product*, is its top-level. It also specifies a value for its attribute *catMgr*. M-object *Car* specializes m-object *Product* by introducing a new level *brand* and adding attribute *maxSpeed* to level *model*. The level *model* of m-object *Car* is regarded as a subclass of level *model* of m-object *Product*.

A child m-object o' chooses its single top-level from the common second-top-levels of its parent m-objects. It 'inherits' from each parent m-object o all levels below its own top-level, together with the relative order of these common levels. It also 'inherits' attributes associated with common levels, together with the properties of these attributes, as defined by functions l_o , d_o , and v_o . In the case of multiple concretization the top-level of the child m-object must be a common second-top-level of the parent m-objects.

For simplicity, we do not define this inheritance mechanism and assume that each m-object is fully described. We summarize the consistency criteria in the following definition.

Definition 2 (Consistent Concretization). *An m-object o' is a consistent concretization of another m-object o iff*

1. The top-level of o' is a second-top-level in o : $(\hat{l}_{o'}, \hat{l}_o) \in P_o$
2. Each level of o , from $\hat{l}_{o'}$ downwards, is also a level of o' : $l \in L_o : (l, \hat{l}_{o'}) \in P_o^* \Rightarrow l \in L_{o'}$ (level containment)
3. All attributes of o , associated with a level that is shared by o and o' , also exist in o' , $\{a \in A_o \mid l_o(a) \in L_{o'}\} \subseteq A_{o'}$ (attribute containment)
4. The relative order of common levels of o and o' is the same: $l, l' \in (L_{o'} \cap L_o) : (l, l') \in P_{o'}^+ \Leftrightarrow (l, l') \in P_o^+$ (level order compatibility)
5. Levels newly introduced in o' have parents only within o' : $\forall (l, l') \in P_{o'} : l \in (L_{o'} \setminus L_o) \Rightarrow l' \in L_{o'}$ (locality of level order).
6. Common attributes are associated with the same level, have the same domain, and the same value, if defined: For $a \in (A_{o'} \cap A_o)$:
 - (a) $l_o(a) = l_{o'}(a)$ (stability of attribute levels)
 - (b) $d_o(a) = d_{o'}(a)$ (stability of attribute domains)
 - (c) $v_o(a)$ is defined $\Rightarrow v_o(a) = v_{o'}(a)$ (compatibility of attribute values)

2.2 Modeling Hetero-Homogeneous Dimension Hierarchies with M-Objects

We now describe how a homogeneous dimension hierarchy can be modeled by m-objects: (1) The dimension is represented by a hierarchy of m-objects. (2) Each dimension level corresponds to a level of the root m-object. (3) Each level schema is represented by the attributes associated with that level of the root m-object. (4) A dimension instance of some dimension level is represented by an m-object, whose top-level is the dimension-level. (5) Attribute values associated with the top-level of an m-object describe the dimension instance that the m-object represents.

Example 3 (Homogeneous Dimension Hierarchies). Consider Fig. 4 ignoring all relationship symbols. M-objects *Product*, *Time*, and *Location* represent the dimensions of the Dimensional Fact Model depicted in Fig. 1. The m-object beneath the gray line depict dimension instances.

Additional non-dimensional attributes can be introduced at various levels for the successors of some dimension instance as follows: The m-object representing this dimension instance is extended by attribute definitions at that level; the m-object now

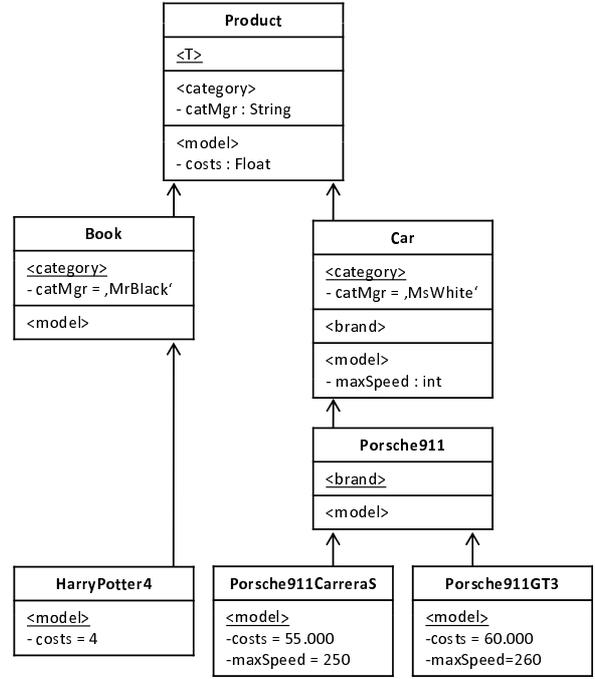


Figure 2: Hierarchy of m-objects representing hetero-homogeneous dimension hierarchy *product*. Attributes are only shown at m-objects where they are introduced or instantiated

serves also as dimension schema for the sub-hierarchy rooted at this dimension instance.

Additional levels can be introduced for the successors of some dimension instance as follows: The m-object representing this dimension instance is extended with additional levels and now serves also as dimension schema for the sub-hierarchy rooted in this dimension instance.

Example 4 (Hetero-homogeneous dimension hierarchy). In the dimension hierarchy *product* (see Fig. 2), m-object *car* introduces additional attribute *maxSpeed* at level *model* and additional level *brand*.

A data warehouse comprises multiple dimensions. Each dimension D organizes a set of m-objects $O_D \subseteq O$ in a hierarchy H_D , with levels L_D , taken from a universe of levels L , and describes m-objects using attributes A_D , taken from a universe of attributes A . Each m-object, but the root-m-object, has one or more parent-m-objects as defined by acyclic relation $H_D : O_D \times O_D$. Let $o, o' \in O_D$, then o' is said to be a *direct concretization* of o or o' concretizes o , iff $(o', o) \in H_D$, to be an *indirect concretization* of o iff $(o', o) \in H_D^+$, to be equal to or an indirect concretization of o iff $(o', o) \in H_D^*$. H_D^+ and H_D^* denote the transitive and transitive-reflexive closure, resp., of H_D .

In case of multiple concretization, stemming from level hierarchies that are not in a total but only in a partial order (see Fig. 3), we avoid conflicts due to 'multiple inheritance' by ensuring that each attribute and each level is inducted at exactly one m-object. We only consider dimensions with such concretizations of m-objects to be consistent.

Definition 3 (Consistent Dimension). *A dimension $D = (O_D, A_D, L_D, H_D)$ is consistent, iff*

1. Each $o \in O_D$ is an m-object according to Definition 1.
2. For each pair of m-objects $(o', o) \in H_D$, o' is a consistent concretization of o according to Definition 2.

3. Each attribute and level is introduced at only one m-object:

(a) $a \in (A_o \cap A_{o'}) : \exists \bar{o} \in O : (o, \bar{o}) \in H_D^* \wedge (o', \bar{o}) \in H_D^* \wedge a \in A_{\bar{o}}$ (unique induction rule for attributes)

(b) $l \in (L_o \cap L_{o'}) : \exists \bar{o} \in O : (o, \bar{o}) \in H_D^* \wedge (o', \bar{o}) \in H_D^* \wedge l \in L_{\bar{o}}$ (unique induction rule for levels)

4. If an m-object o' with top-level l is a direct or indirect concretization of m-object o where $(l, l') \in P_o$ then o' must concretize an m-object \hat{o} with top-level l' .

5. An m-object o may not directly or indirectly concretize two m-objects o', o'' that are at the same level, i.e., $(o, o') \in H^* \wedge (o, o'') \in H^* \Rightarrow \hat{l}_{o'} \neq \hat{l}_{o''}$ (unique level predecessor)

Levels in a dimension, L_D , are implicitly partially ordered. This follows from the unique induction rule for levels and level order compatibility. We say, $l' \in L_D$ is a descendant of $l \in L_D$, written as $l' \prec l$, if there is an m-object $o \in O_D$ in which l' is a descendant of l . We write $l' \preceq l$ to denote that l' is either descendant of or equal to l . Also note that \prec and \preceq are transitive, i.e.: $\forall l', l \in L_D : (\exists o \in O_D : (l', l) \in P_o^*) \vee (\exists l'' \in L_D : l' \preceq l'' \wedge l'' \preceq l) \Rightarrow l' \preceq l$.

Example 5 (Consistent Hetero-homogeneous dimension hierarchy). Consider dimension hierarchy *location* in Fig. 3, m-object Lausanne is an indirect concretization of m-object *location* via kanton *Vaud* and country *Switzerland*. As level *region* is also a parent level of level *city* in m-object *location*, Lausanne must also concretize an m-object at level *region*. This is with m-objects *Alps* the case.

3 Hetero-Homogeneous Cubes

In this section we revisit and extend definitions of m-relationships (Neumayr et al., 2009) and, then, we show how to model hetero-homogeneous cubes with them.

3.1 M-Relationships revisited

M-relationships as introduced in (Neumayr et al., 2009) are analogous to m-objects in that they describe relationships between m-objects at multiple levels of abstraction. They have the following features: (1) M-relationships at different abstraction levels can be arranged in concretization hierarchies, similar to m-objects. (2) An m-relationship represents different abstraction levels of a relationship, namely one relationship occurrence and multiple relationship classes. Such a relationship class collects all descending m-relationships that connect m-objects at the respective levels. (3) An m-relationship implies extensional constraints for its concretizations at multiple levels. (4) M-relationships can cope with heterogenous hierarchies and (5) m-relationships can be exploited for querying and navigating.

While our original approach considered only binary m-relationships without relationship attributes, the revised definition below covers for n-ary m-relationships that are described by attributes. Taking into account the data warehouse context the attributes are measures, have an associated aggregation function, and a connection level indicating at which detail measure values are provided.

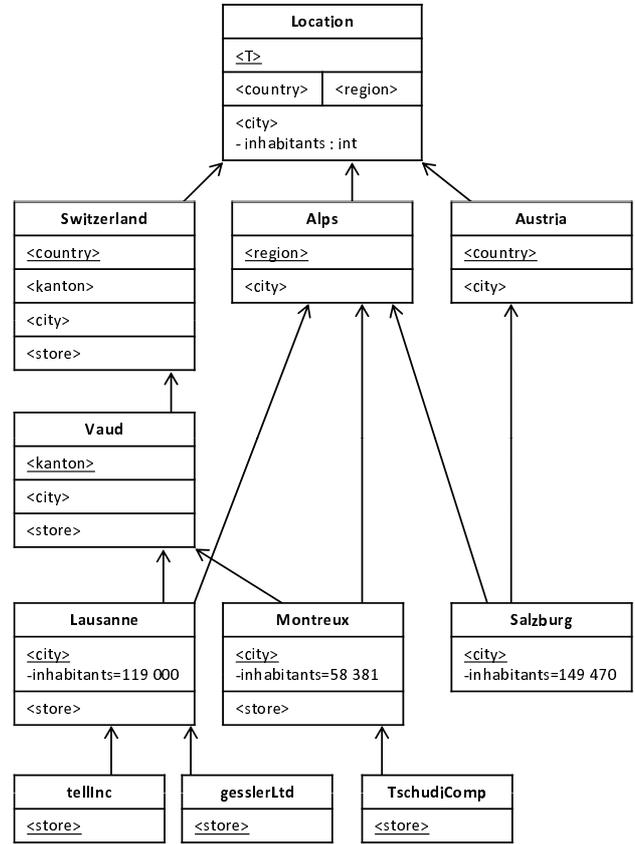


Figure 3: Hetero-homogeneous dimension hierarchy *location* with multiple concretization

Definition 4 (M-Relationship). An m-relationship $r = (o_1, \dots, o_n; M, b, u, f, v)$ between m-objects o_1, \dots, o_n , its coordinate (denoted also by $\text{coord}(r)$), is described by a set of measures M . Its top-connection-level \hat{l}_r is implicitly given by the top-levels of the referenced m-objects, i.e., $\hat{l}_r := (\hat{l}_{o_1}, \dots, \hat{l}_{o_n})$. Each measure $m \in M$ is described by

1. a connection-level, as defined by total function $b : M \rightarrow (L_{o_1} \times \dots \times L_{o_n})$
2. a unit of measure, as defined by total function $u : M \rightarrow U$, where U is a universe of measure units.
3. a distributive aggregation function, as defined by total function $f : M \rightarrow \{\text{SUM}, \text{MAX}, \text{MIN}\}$.
4. an asserted value (primary fact), as defined by partial function $v : M \rightarrow V$. A measure $m \in M$ has an asserted value iff the connection-level of m is equivalent to the top-connection-level of r , i.e.: $v(m)$ is defined $\Leftrightarrow b(m) = \hat{l}_r$.

When talking about different m-relationships, e.g. r and r' , we alternatively use subscripts (e.g., M_r and $M_{r'}$) or quotes (e.g. M, b , being features of r and M', b' being features of r') to denote the context of sets and functions.

Definition 5 (Measure Units and Measure Types). Each measure unit $u \in U$ is member of one measure type $t \in T$, where T is a universe of measure types, as defined by total function $\text{type} : U \rightarrow T$.

Example 6 (M-Relationship). Consider m-relationship *sales* in Fig. 4 between m-objects *Product*, *Time*, and *Location*. It defines measure *revenue* at connection-level $\langle \text{model}, \text{month}, \text{city} \rangle$ with unit of measure € and aggregation function SUM .

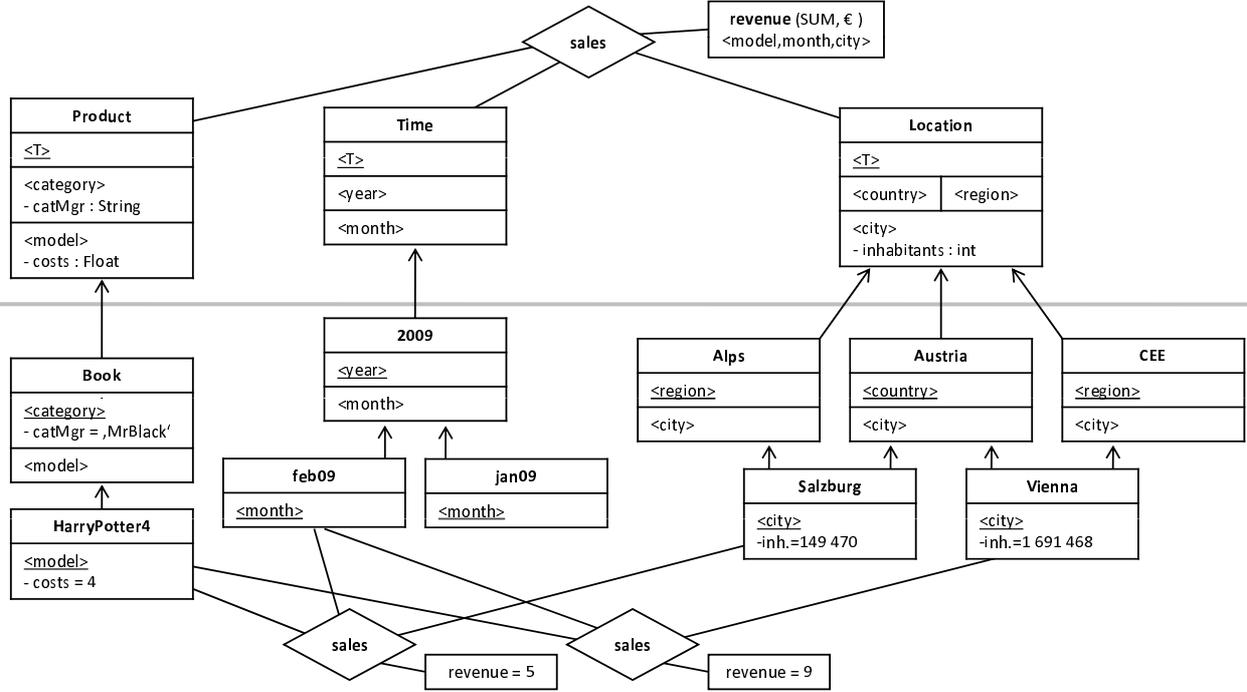


Figure 4: Homogeneous data warehouse modeled with m-objects and m-relationships

To discuss concretization of m-relationships we need the notions of *partial order of connection levels* and *partial order of coordinates*.

Definition 6 (Partial Order of Connection Levels). Given a coordinate (o_1, \dots, o_n) and the levels of the m-objects of that coordinate, L_{o_1}, \dots, L_{o_n} , and two connection-levels $(l'_1, \dots, l'_n), (l_1, \dots, l_n) \in (L_{o_1} \times \dots \times L_{o_n})$. We say (l'_1, \dots, l'_n) is a descendant of (l_1, \dots, l_n) , written as $(l'_1, \dots, l'_n) \preceq (l_1, \dots, l_n)$, iff for $i=1..n$ each level l'_i is a descendant of l_i , i.e., $(l'_1, \dots, l'_n) \preceq (l_1, \dots, l_n) \Leftrightarrow l'_1 \preceq l_1 \wedge \dots \wedge l'_n \preceq l_n$.

Definition 7 (Partial Order of Coordinates). Given n dimensions, D_1, \dots, D_n , of n disjoint sets of m-objects, O_{D_1}, \dots, O_{D_n} . We say coordinate $(o'_1, \dots, o'_n) \in (O_{D_1} \times \dots \times O_{D_n})$ is a descendant of or equal to coordinate $(o_1, \dots, o_n) \in (O_{D_1} \times \dots \times O_{D_n})$, written as $(o'_1, \dots, o'_n) \preceq (o_1, \dots, o_n)$, iff $\forall_{i=1}^n : (o'_i, o_i) \in H_{D_i}^*$. In this case we also speak of a sub-coordinate.

Coordinate (o'_1, \dots, o'_n) is a descendant of - or proper sub-coordinate of - coordinate (o_1, \dots, o_n) , written as $(o'_1, \dots, o'_n) \prec (o_1, \dots, o_n)$, iff for all dimensions $i=1..n$, o'_i is a descendant of or is equal to o_i , and for at least one dimension j , o'_j is a concretization of o_j , i.e., $\forall_{i=1}^n : (o'_i, o_i) \in H_{D_i}^* \wedge \exists_{j=1}^n : (o'_j, o_j) \in H_{D_j}^+$.

Coordinate (o'_1, \dots, o'_n) overlaps with coordinate (o_1, \dots, o_n) , written as $(o'_1, \dots, o'_n) \overline{\cap} (o_1, \dots, o_n)$, iff they have some (sub-)coordinates in common, that is, for all dimensions, $i = 1..n$, the respective dimension m-objects o'_i, o_i are either equal or in a concretization relationship: $(o'_1, \dots, o'_n) \overline{\cap} (o_1, \dots, o_n) \Leftrightarrow (\forall_{i=1}^n : (o'_i, o_i) \in H_{D_i}^* \vee (o_i, o'_i) \in H_{D_i}^*)$

An m-relationship is concretized by substituting one or more of the m-objects in its coordinate by descendant m-objects. The descendant m-relationship must provide values for the measures at its top-connection-level and may add measures, and move the connection-level of a measure to a more specific connection-level.

Definition 8 (Consistent Concretization of M-Relationships). A m-relationship $r' = (o'_1, \dots, o'_n; M', b', u', f', v') \in R$ is a consistent concretization of another m-relationship $r = (o_1, \dots, o_n; M, b, u, f, v) \in R$, iff

1. $(o'_1, \dots, o'_n) \prec (o_1, \dots, o_n)$
2. every measure m of r , $m \in M$, with a base-level that is below or equal to the top-level of r' , is also a measure of r' , every other measure of r is not a measure of r' (measure containment):
 $\{m \in M \mid b(m) \preceq \hat{l}_{r'}\} \subseteq M'$
 $\{m \in M \mid b(m) \not\preceq \hat{l}_{r'}\} \cap M' = \emptyset$
3. for each measure m shared by r and r' , the base-level of m at r' is the same or below the base-level of m at r : $\forall m \in (M \cap M') : b'(m) \preceq b(m)$ (assured granularity)
4. Common measures are associated with measure units of the same measure type and the same aggregation function: For $m \in (M \cap M')$:

- (a) $type(u'(m)) = type(u(m))$ (stability of measure types)
- (b) $f'(m) = f(m)$ (stability of aggregation functions)

Example 7 (Concretization of M-Relationships). M-relationship *sales* between m-objects *HarryPotter4*, *feb09*, and *Salzburg* concretizes *sales* between m-objects *Product*, *Time*, *Location* and its top-connection level is $\langle model, month, city \rangle$, thus it defines a value for measure *revenue*. An example for introducing additional levels and moving measures to more specific connection-levels will be given later.

3.2 Modeling Hetero-Homogeneous M-Cube-Hierarchies with M-Relationships

We first describe how a homogeneous cube of n dimensions can be modeled by m-relationships: (1) A cube is represented by a concretization hierarchy of

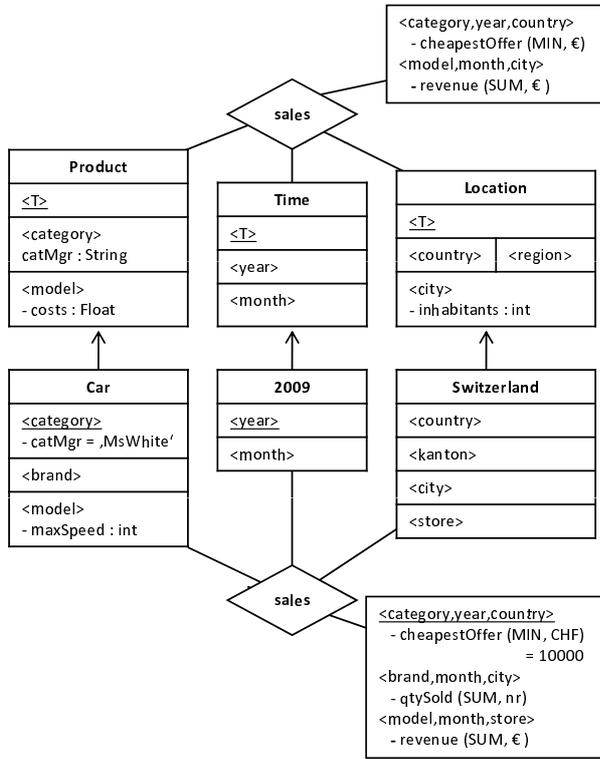


Figure 5: Concretization of m-relationship sales, also representing a hetero-homogeneous cube

n-ary m-relationships. (2) The root m-relationship connects the root m-objects of these *n* dimensions. (3) The root m-relationship has measures associated with a single connection-level which consists of the bottom levels of these *n* dimensions and gives the measures of the cube. (4) The cells or facts of the cube are represented by m-relationships that concretize the root m-relationship and connect *n* m-objects that are at the connection-level for which the measures of the root-m-relationship are defined and give values for these measures.

Example 8 (Homogeneous Cube). Fig. 4 depicts a homogeneous cube schema *sales* (above the gray horizontal line) and its facts (below the gray line). Note, while the m-cube approach provides a coherent model both for cube- and dimension-schemas as well as their instances, its graphical representation is obviously not meant to be used to fully model a cube with all its facts and dimension instances; it is rather used, analogously to object diagrams in UML, to model exemplary dimension instances and facts together with dimension and cube schema. The cube schema corresponds to the Dimensional Fact Model depicted in 1. The cube extension has two facts.

We now describe how a hetero-homogeneous cube of *n* dimensions can be modeled by m-relationships: Cubes can be hetero-homogeneous in that (1) sub-cubes have *additional measures*, (2) different sub-cubes may give *different units* for the same measure (3) various measures are provided at *mixed granularities* (4) different sub-cubes may provide the same measure at *different granularities* (see examples given in the Introduction).

Additional Measures can be introduced at a sub-cube identified by a coordinate (o_1, \dots, o_n) as follows: An m-relationship for this coordinate is introduced. This m-relationship defines a measure for the connection-level at which values for the measure are provided.

Different sub-cubes with *different units* for the same measure are supported as follows: An m-relationship for the coordinates of each sub-cube is introduced and gives a different unit of measure.

Cubes with measures that are provided at *mixed granularities* can be represented as follows: An m-relationship is introduced that associates these measures with different connection levels.

Cubes in which different sub-cubes provide the same measure at *different granularities* are represented as follows: An m-relationship is introduced for the cube and gives measure at some connection-level. For each sub-cube that provides this measure at a more detailed granularity, an m-relationship is introduced and associates this measure with a more specific connection level.

Example 9 (Hetero-Homogeneous Cubes). Fig. 5 depicts a fragment of a hetero-homogeneous cube. - Note, this example is different from previous ones for sake of presentation and simplicity. - M-relationship *sales* between m-objects *product*, *time*, and *location* introduces two measures at mixed granularities. Measure *cheapestOffer* for connection-level $\langle category, year, country \rangle$ and measure *revenue* for connection-level $\langle model, month, city \rangle$. M-relationship *sales* between category *car*, year *2009*, and country *Switzerland* concretizes the above m-relationship as follows: (1) It introduces an additional measure *qtySold* for connection-level $\langle model, month, city \rangle$. (2) It moves measure *revenue* from connection-level $\langle model, month, city \rangle$ to $\langle model, month, store \rangle$. Thus it provides for different granularity of measure *revenue*: the cube will have stored revenue values for models of cars, months in 2009, and stores in Switzerland, but not for other product categories, months in other years, stores in other countries. (3) It provides a different unit of measure for *cheapestOffer*, that is swiss francs instead of €.

The notion of a multi-level cube (m-cube), as defined below, generalizes the cube in the Dimensional Fact Model (Golfarelli et al., 1998).

Definition 9 (Multi-Level Cube). A multi-level cube $C = (D_1, \dots, D_n; S, R)$ connects *n* dimensions, D_1, \dots, D_n . Its root-coordinate *S* is identified by a tuple $(o_1, \dots, o_n) \in O_{D_1} \times \dots \times O_{D_n}$. *R* is a set of m-relationships which represent the measure schema and the base facts of *C*.

The m-relationships of *C* that provide a measure-value are called the base facts or base cells of *C*.

When talking about different m-cubes, e.g. *C* and *C'*, we alternatively use subscripts (e.g., X_C) or quotes (e.g. D_1, S , being features of *C* and D'_1, S' being features of *C'*) to denote the context of sets and functions. Whenever the context is clear we use unquoted variables (e.g. D_1, S).

We now define consistency criteria that avoid conflicts due to multiple inheritance and avoid overlapping facts. For this definition we use the set \hat{R}_r of *directly subsuming m-relationships* of a m-relationship *r*, $\hat{R}_r := \{r' \in R \mid r \preceq r' \wedge \nexists r'' \in R : r \preceq r'' \prec' r\}$.

Definition 10 (Consistent M-Cube). A multi-level cube $C = (D_1, \dots, D_n; S, R)$ with root-coordinate $S = (o_1, \dots, o_n)$ is consistent iff

1. there is one m-relationship in *R* that corresponds to root-coordinate *S*.
2. for each cell $x \in X$ there is at most one corresponding m-relationship in *R*.
3. For each pair of m-relationships $r, r' \in R$, if r' is a concretization of *r*, $r' \preceq r$, then r' is a consistent concretization of *r* according to Def. 8.

4. Each measure is introduced at only one m-relationship: $\forall r, r' \in R : \exists m \in \{M_r \cap M_{r'}\} \Rightarrow \exists r'' \in R : m \in M_{r''} \wedge \text{coord}(r) \preceq \text{coord}(r'') \wedge \text{coord}(r') \preceq \text{coord}(r'')$ (unique induction rule for measures)
5. For each measure m shared by two overlapping m-relationships r and r' , $\text{coord}(r) \not\leq \text{coord}(r')$, if r defines a value for m than r' must not define a value for m : $v_r(m)$ is defined $\Rightarrow v_{r'}$ is not defined. (unique assertion of values)
6. For each non-empty cell x , for each pair r, r' of direct subsuming m-relationships of x that contain a measure m with base-level below or equal to the level of x , the measure unit and the base level for m are the same at r and r' :
 $\forall x \in X, \forall r, r' \in \hat{R}_x, \forall m \in M_r \cap M_{r'} :$
 $(\exists r \in R : \text{coord}(r) \preceq x) \wedge (b_r(m) \preceq \hat{l}_x \vee b_{r'}(m) \preceq \hat{l}_x) \Rightarrow$
 - (a) $u_r(m) = u_{r'}(m)$ (unit conflict avoidance)
 - (b) $b_r(m) = b_{r'}(m)$ (base level conflict avoidance)

Unit conflict avoidance and base level conflict avoidance (Def. 10, item 6) ensure that possible conflicts due to multi-dimensional concretization are solved explicitly by an m-relationship directly beneath the conflicting m-relationships.

An m-cube represents hetero-homogeneous base facts as possibly extracted and loaded from various source OLTP databases. An m-cube defined between with root-coordinate (o_1, \dots, o_n) implicitly also represents a cube of cubes. This cube of cubes consists of a set of homogeneous cubes, one for each n -tuple of levels in the cartesian product of the levels of the m-objects of the root coordinate. The cells of such a cube are given by the cartesian product of m-objects at those levels.

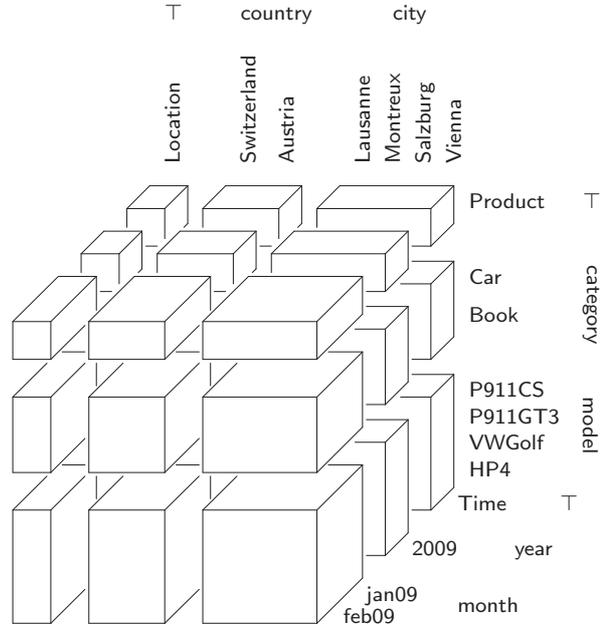
Example 10. Fig. 6 depicts the homogeneous cubes of m-cube *sales*; Fig. 7 shows a sample whereby we ignore dimension *time* for simplicity.

A hetero-homogeneous cube exists for each sub-coordinate and consists of those m-relationships of the given cube that are descendants of that sub-coordinate.

Example 11. Sub-m-cube *sales(Car, Time, Switzerland)* takes a closer look at car sales in Switzerland. Fig. 8 depicts the homogeneous cubes of this sub-m-cube ignoring dimension *time*. Note, that the dimension levels identifying these cubes are not shown. Additional cubes become available for the additional dimension levels *kanton* and *store* defined for country *Switzerland* (see Fig. 3) and additional level *brand* defined for category *car*. Further, additional measure *qtySold* is available for the cubes at connection-level $\langle \text{brand}, \text{city} \rangle$ and above, since this measure has been defined for cars in Switzerland for this level (see Fig. 5). Note that for descendant connection-levels the cubes show a null-value for this measure.

The aggregate cell (or fact) has the coordinate of the m-cube and a value for each measure that is provided for this coordinate or can be calculated from the base cells of the m-cube.

Example 12. The top-left entry in Fig. 7 and the top-left entry in Fig. 8 represent the aggregate cells of coordinates $\langle \text{product}, \text{location} \rangle$ and $\langle \text{car}, \text{switzerland} \rangle$ respectively.



T	Product	T	country		city			
		Location	Switzerland	Austria	Lausanne	Montreux	Vienna	Salzburg
		20	13	7	6	7	7	4
category	Car	13	10	3	5	5	3	4
	Book	7	3	4	1	2	4	4
model	P911CS	7	5	2	3	2	2	4
	P911GT3	7	5	2	3	2	2	4
	VWGolfXY	6	5	1	2	3	1	4
	HP4	7	3	4	1	2	4	4

Figure 7: Homogeneous cubes of a sample m-cube sales showing values for measure revenue as defined by m-relationship sales between *Product* and *Location*, ignoring dimension *Time* and level *region*.

	country		city		store		
	Switzerland	kanton	Lausanne	Montreux	tellinc	gesslerLtd	TschudiComp
Car	28/17	28/17	18/11	10/6	9/	9/	10/
P911	19/10	19/10	11/6	8/4	5/	6/	8/
VWGolf	9/7	9/7	7/5	2/2	4/	3/	2/
P911CS	10/	10/	5/	5/	3/	2/	5/
P911GT3	9/	9/	6/	3/	2/	4/	3/
VWGolfXY	9/	9/	6/	3/	4/	3/	2/

Figure 8: Homogeneous cubes of sub-m-cube with root-coordinate (*Car,Switzerland*) of m-cube sales depicting measures revenue and *qtySold* where available

The *query* consists of (i) optionally a set of boolean *predicates* to narrow the analysis on cells whose m-objects fulfil the predicate (corresponds to operation *slice* in Def. 15) (ii) optionally a set of *measures of interest* (corresponds to operation *projection* in Def. 12), (iii) optionally a *measure unit* for each measure and (iv) a *cell coordinate* to retrieve facts of a single cell, or a *cube coordinate* to retrieve facts of all cells within the specified cube (corresponds to operations *fact extraction* in Def. 20, and *cube extraction* in Def. 22, respectively). If not specified explicitly all available measures are considered and values are converted to the measure unit specified at the specified (sub-)m-cube (see Fig. 9 for an example query and its results).

4.1 Closed M-Cube Operations

The *dice*-operator selects a sub-m-cube from an m-cube.

Definition 11 (Dice δ). *Given an input m-cube $C = (D_1, \dots, D_n, S, R)$, coordinate (o_1, \dots, o_n) , and that there is a m-relationship $r = (o_1, \dots, o_n, M, b, u, f, v) \in R$, then $\delta_{o_1, \dots, o_n} C$ results in output-cube $C' = (D_1, \dots, D_n, S', R')$ with $S' = (o_1, \dots, o_n)$ $R' = \{r' \in R \mid coord(r') \preceq (o_1, \dots, o_n)\}$*

Example 13. Dice operation $\delta_{(Car, 2009, Switzerland)} sales$ retrieves a sub-m-cube *car09SalesCH* containing m-relationships with coordinates that are descendants of (*Car, 2009, Switzerland*). Fig. 8 depicts the cube of cubes of this m-cube.

	tellinc	gesslerLtd
P911CS	3	2
P911GT3	2	4
VWGolfXY	4	3

Figure 9: Homogeneous cube of sales revenue for car sales in 2009 in Switzerland in big cities with cells at level $\langle model, Time, store \rangle$

The projection operator applied on an m-cube, returns an m-cube with a reduced set of measures.

Definition 12 (Projection π). *Given an input m-cube $C = (D_1, \dots, D_n, S, R)$, and a set of measures $\mathcal{M} \in M_C$, then $\pi_{\mathcal{M}} C$ results in output-cube $C' = (D_1, \dots, D_n, S, R')$, with R' defined as follows: for each $r = (o_1, \dots, o_n; M, b, u, f, v) \in R$ there is a $r' = (o_1, \dots, o_n; M', b', u', f', v') \in R'$, with $M' := M \cap \mathcal{M}$, and for each $m \in M'$: $b'(m) := b(m)$, $u'(m) := u(m)$, $f'(m) := f(m)$, $v'(m) := v(m)$.*

As prerequisites for predicates used as selection criteria in slice-operation we define the notions of stable upward navigation and class extension.

Definition 13 (Upward Navigation). *The ancestor m-object of m-object $o \in O_D$ at level $l \in L_o$, denoted as $o[l]$, is defined by*

$$o[l] \stackrel{\text{def}}{=} \{o' \mid (o, o') \in H_D^* \wedge \hat{l}_{o'} = l\}.$$

An m-object represents for each level of direct or indirect descendants the class of descendant m-objects of that level. To refer to the set of m-objects at level l beneath m-object o , we write $o\langle l \rangle$. For example, *car* $\langle model \rangle$ refers to the set of m-objects at level *model* beneath m-object *Car*.

Definition 14 (Class Extension). *The class of m-objects of m-object $o \in O_D$ at level $l \in L_o$, denoted as $o\langle l \rangle$, is defined by*

$$o\langle l \rangle \stackrel{\text{def}}{=} \{o' \mid (o', o) \in H_D^* \wedge \hat{l}_{o'} = l\}.$$

A *predicate* is a boolean expression over attributes of a class of m-objects, $o\langle l \rangle$ and of its ancestors (using upward navigation). Note, that predicates could be predefined at m-objects and associated with a level like attributes. Then these predicates could be over-written in concretizations.

A *slice*-operation on a given m-cube selects all coordinates at a given level that fulfill the given criteria and returns an m-cube with all m-relationships from the given m-cube that are between descendants of the given coordinates, between ancestors of the given coordinates, or are at these coordinates. Dimensions D_1, \dots, D_n and root-coordinate $S = (o_1, \dots, o_n)$ are the same in both the input m-cube and the output m-cube. An outer-slice has the same output but additionally consists of all m-relationships from the input m-cube that are above the cube-level of the selection.

Definition 15 (Slice σ). *Given an input m-cube $C = (D_1, \dots, D_n, S, R)$ with $S = (o_1, \dots, o_n)$ and selection predicates $(p_1, l_1), \dots, (p_n, l_n)$. For $\sigma_{(p_1, l_1), \dots, (p_n, l_n)} C$ to be applicable on C , there must not be an m-relationship in R with an asserted measure value above cube-level (l_1, \dots, l_n) . The slice operation $\sigma_{(p_1, l_1), \dots, (p_n, l_n)} C$ results in output cube $C' = (D_1, \dots, D_n, S, R')$ where R' is given as follows. Let the selected cells be given by $\bar{X} := \{o \in o_1\langle l_1 \rangle \mid p_1(o)\} \times \dots \times \{o \in o_n\langle l_n \rangle \mid p_n(o)\}$; and let the included m-relationships be given by $\bar{R} := \{r \in R \mid \exists x \in \bar{X} : r \preceq x\}$. Then $R' := \bar{R} \cup \{r \in R \mid \exists \bar{r} \in \bar{R} : \bar{r} \preceq r\}$.*

Example 14 (Slice). Slice-operation $\sigma_{(\text{inhabitants} > 100000, \text{city})} \text{car09SalesCH}$ selects m-cube $\text{car09SalesCHinBigCities}$, which comprises m-relationships representing car sales of 2009 in Switzerland in cities with more than 100000 inhabitants.

Definition 16 (Outer Slice $\bar{\sigma}$). Outer Slice is defined as Slice in Def. 15 with the difference that R' is defined as follows:

$$R' := \bar{R} \cup \{r \in R \mid (l_1, \dots, l_n) \preceq \hat{l}_r\}$$

Import Union inserts a cube into an existing cube. It can be seen as a bulk operation for inserting m-relationships. The resulting cube needs to be consistent according to Def. 9.

Definition 17 (Import Union \cup_i). Given two input cubes, main cube $C = (D_1, \dots, D_n, S, R)$ and to-be-imported cube $C' = (D_1, \dots, D_n, S', R')$, with $\nexists r \in R : r \preceq S'$ and $S' \preceq S$, then $C \cup_i C'$ results in output cube $C'' = (D_1, \dots, D_n, S, R'')$ with $R'' := R \cup R'$.

4.2 Fact and Cube Extraction

Before defining fact and cube extraction operators we need to investigate which measures are available for a given coordinate. A measure at a given coordinate may be provided by a m-relationship of the m-cube, i.e., be an asserted fact, or be derived through application of the aggregation function provided with the measure definitions.

Definition 18 (Common Measures at Coordinates). Given a coordinate $x = (o_1, \dots, o_n)$ from a consistent m-cube $C = (D_1, \dots, D_n; S, R)$, its set of measures, M_x , is given by the union of measures of its direct subsuming m-relationships \hat{R}_x , given that the measures connection-level is below or equal to the level of x : $M_x :=$

$$\{m \in \bigcup_{r \in \hat{R}_x} M_r \mid \forall r \in \hat{R}_x : b_r(m) \preceq (\hat{l}_{o_1}, \dots, \hat{l}_{o_n})\}$$

For each measure $m \in M_x$, given one of its direct subsuming m-relationships $r \in \hat{R}_x$ that contains m , $m \in M_r$, the base-level, unit-of measure, and aggregation function are those defined at r :

1. $b_x(m) := b_{r'}(m)$
2. $u_x(m) := u_{r'}(m)$
3. $f_x(m) := f_{r'}(m)$

Conversion between measure units is facilitated by multi-polymorphic function *conv*. It applies, dependent on the pair of source and target measure units, a simple arithmetic expression on the numeric input value to produce an output value. We assume, that there is a conversion expression for each pair of measure units that are members of the same measure type. Context-sensitive unit conversion, e.g. time-dependent currency conversion, is facilitated by extending function *conv* to take dimension objects, i.e. a cell-coordinate, as additional parameters. The extended *conv*-method is multi-polymorphic in the two measure-units and in these dimension-objects. For space-limitations we do not further discuss this extension and refer the interested reader to (Schrefl et al., 1998).

Given a source measure unit $u_s \in U$, a target measure unit $u_t \in U$, with $\text{type}(u_s) = \text{type}(u_t)$, and an input value $v \in V$, operation $\text{conv}(u_s, u_t, v)$ returns a value that is the conversion of value v from measure unit u_s to measure unit u_t .

We now define how measure values are derived from asserted facts.

Definition 19 (Aggregation of Measures *val*). Given an m-cube $C = (D_1, \dots, D_n, S, R)$, a cell $x = (o_1, \dots, o_n)$ with $\exists r \in R : r \preceq x$, measure $m \in M_x$, and measure unit $u \in U$, with $\text{type}(u) = \text{type}(u(m))$, then the value of measure m at coordinate x converted to unit u , $\text{val}(m, x, u)$, is calculated by applying aggregation function $f_x(m)$ on the set of converted m-values of m-relationships below or at cell x , given by $R_x := \{r \in R \mid r \preceq x\}$; or null if this set is empty, i.e.:

$$\text{val}(m, x, u) := \begin{cases} f_x(m)(\bigcup_{r \in R_x} \text{conv}(& \text{if } \exists r \in R_x : \\ u_r(m), u, (v_r(m))) & (v_r(m) \text{ is defined}) \\ \text{null} & \text{otherwise} \end{cases}$$

We are now ready to define the fact extraction operator.

Definition 20 (Fact Extraction φ). Given an m-cube $C = (D_1, \dots, D_n, S, R)$, a cell $x = (o_1, \dots, o_n)$ with $\exists r \in R : r \preceq x$, and a mapping from measures to measure units $(m_1 \mapsto u_1, \dots, m_k \mapsto u_k)$, then fact extraction operation $\varphi_{(o_1, \dots, o_n), (m_1 \mapsto u_1, \dots, m_k \mapsto u_k)} C$ returns a relation with schema $(D_1, \dots, D_n, m_1 : u_1, \dots, m_k : u_k)$ and an instance consisting of one tuple $(o_1, \dots, o_n, \text{val}(m_1, x, u_1), \dots, \text{val}(m_k, x, u_k))$.

When leaving out the mapping from measures to measure units, fact extraction results in a relation with all measures that are available at the respective cell and converts each measure to the respective unit of measure defined at this cell (see Def. 21).

Definition 21 (Fact Extraction Shorthand φ). Given cell $x = (o_1, \dots, o_n)$ with $\exists r \in R : r \preceq x$ with measures $M_x = \{m_1, \dots, m_k\}$ and units of measures $u_x = \{m_1 \mapsto u_1, \dots, m_k \mapsto u_k\}$, then $\varphi_{o_1, \dots, o_n} C$ is a shorthand for $\varphi_{(o_1, \dots, o_n), (m_1 \mapsto u_1, \dots, m_k \mapsto u_k)} C$.

A cube extraction operation returns a homogeneous cube, consisting of a tuple for each non-empty cell at a given cube-level.

Definition 22 (Cube Extraction κ). Given an m-cube $C = (D_1, \dots, D_n, S, R)$, a cube-level $(l_1, \dots, l_n) \in (L_{D_1}, \dots, L_{D_n})$, and a mapping from measures to measure units $(m_1 \mapsto u_1, \dots, m_k \mapsto u_k)$.

The set of non-empty cells of C at level (l_1, \dots, l_n) , denoted as $C(l_1, \dots, l_n)$, is given by $\{(o_1, \dots, o_n) \in X \mid (\exists r \in R : r \preceq (o_1, \dots, o_n)) \wedge \hat{l}_{o_1} = l_1 \wedge \dots \wedge \hat{l}_{o_n} = l_n\}$

The result of cube extraction operation $\kappa_{(l_1, \dots, l_n), (m_1 \mapsto u_1, \dots, m_k \mapsto u_k)} C$ is the relation given by union of facts of all non-empty cells at level (l_1, \dots, l_n) :

$$\kappa_{(l_1, \dots, l_n), (m_1 \mapsto u_1, \dots, m_k \mapsto u_k)} C := \bigcup_{x \in C(l_1, \dots, l_n)} (\varphi_{x, (m_1 \mapsto u_1, \dots, m_k \mapsto u_k)} C)$$

Example 15. Given our m-cube $\text{car09SalesCHinBigCities}$ of car sales, the homogeneous cube with measure revenue of sales rolled up to level *model,store* can be extracted by applying projection and subsequent cube extraction operators, e.g., $\kappa_{(\text{model,store}), (\text{revenue} \mapsto \text{€})} \pi_{\text{revenue}} \text{car09SalesCHinBigCities}$. Fig. 9 depicts the result of this query as cross table.

In order to retain measures that are available at some but not all cells of a cube, we use outer union (Codd, 1979) on facts extracted according to Def. 21. Note that we accept null values and heterogenous measure units in the resulting cube (see Def. 23).

Definition 23 (Outer Cube Extraction $\bar{\kappa}$). The result of $\bar{\kappa}_{(l_1, \dots, l_n)} C$ is the relation given by outer union, denoted as $\bar{\cup}$, on facts of all non-empty cells, at level (l_1, \dots, l_n) :

$$\bar{\kappa}_{(l_1, \dots, l_n)} C := \bar{\bigcup}_{x \in C(l_1, \dots, l_n)} (\varphi_x C)$$

5 Related Work

Heterogeneities in data warehouses are widely acknowledged as an important research direction and have received considerable attention in the literature, especially on data warehouse integration (Torlone, 2008; Berger and Schrefl, 2008), summarizability (Hurtado and Mendelzon, 2001), OLAP visualization (Mansmann and Scholl, 2006; Cuzzocrea and Mansmann, 2009), and conceptual modeling (Malinowski and Zimányi, 2006). These works especially discuss heterogeneities in dimension hierarchies, such as non-covering, non-strict, and asymmetric hierarchies. However, to the best of our knowledge, none of these approaches provides for a top-down modeling approach of hetero-homogeneous dimension and cube hierarchies.

Conceptual data warehouse design has attracted a lot of work, various approaches are based on entity-relationship modeling, such as (Song et al., 2008), on the UML, such as (Trujillo et al., 2001), or on abstract state machines (Zhao and Schewe, 2004). The well-established Dimensional Fact Model (Golfarelli et al., 1998) has been used in this paper as starting point to illustrate homogeneous data warehouse schemas and how hetero-homogeneous hierarchies extend them.

An important area of work concerns summarizability (Lenz and Shoshani, 1997; Hurtado and Mendelzon, 2001) and formal aspects of aggregation in data warehouses (Lenz and Thalheim, 2001). In this context (Gray et al., 1997) introduce the notions of distributive, algebraic, and holistic aggregation functions. In this paper we only considered measures based on distributive aggregation functions, a restriction we will relax in future work.

6 Conclusion

In this paper we introduced hetero-homogeneous hierarchies and discussed their application to data warehousing. We provided structural definitions and consistency criteria based on m-objects and m-relationships.

We believe that hetero-homogeneous hierarchies are a very promising approach to modeling and querying data warehouses. Interesting issues which we will investigate in the future are:

- *Aggregation operations.* In this paper we limited the discussion on measures based on distributive aggregation functions SUM, MAX, MIN. We excluded operation COUNT due to the lack of a meaningful definition of its semantics in the presence of different and mixed granularities. Future work needs to address peculiarities of aggregation operations in multi-level cubes, in the flavor of (Lenz and Thalheim, 2001), especially concerning empty cells, as well as algebraic and holistic aggregation operations.
- *Prototype.* Future work needs to provide a proof-of-concept prototype. We will investigate how our m-cube approach can be implemented on top of object-relational DBMS.
- *Efficiency.* In this paper we discussed a conceptual modeling and querying approach, disregarding optimization issues. In the future we also want to investigate how hetero-homogeneous hierarchies can be implemented and queried efficiently.

References

- Abelló, A., Samos, J. and Saltor, F. (2006), YAM²: a multidimensional conceptual model extending UML, *Inf. Syst.* **31**(6), 541–567.
- Berger, S. and Schrefl, M. (2008), From federated databases to a federated data warehouse system, *HICSS 2008*.
- Codd, E. F. (1979), Extending the database relational model to capture more meaning, *ACM Trans. Database Syst.* **4**(4), 397–434.
- Cuzzocrea, A. and Mansmann, S. (2009), OLAP visualization: Models, issues, and techniques, in J. Wang, ed., ‘Encyclopedia of Data Warehousing and Mining, Second Edition’, Information Science Reference.
- Golfarelli, M., Maio, D. and Rizzi, S. (1998), The dimensional fact model: A conceptual model for data warehouses, *Int. J. Cooperative Inf. Syst.* **7**(2-3), 215–247.
- Gray, J., Chaudhuri, S., Bosworth, A., Layman, A., Reichart, D., Venkatrao, M., Pellow, F. and Pirahesh, H. (1997), Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub totals, *Data Min. Knowl. Discov.* **1**(1), 29–53.
- Hurtado, C. A. and Mendelzon, A. O. (2001), Reasoning about summarizability in heterogeneous multidimensional schemas, *ICDT 2001*, pp. 375–389.
- Lenz, H.-J. and Shoshani, A. (1997), Summarizability in OLAP and statistical data bases, *SSDBM 1997*, pp. 132–143.
- Lenz, H.-J. and Thalheim, B. (2001), OLAP databases and aggregation functions, *SSDBM 2001*, IEEE Computer Society, pp. 91–100.
- Malinowski, E. and Zimányi, E. (2006), Hierarchies in a multidimensional model: From conceptual modeling to logical representation, *Data Knowl. Eng.* **59**(2), 348–377.
- Mansmann, S. and Scholl, M. H. (2006), Extending visual OLAP for handling irregular dimensional hierarchies, in A. M. Tjoa and J. Trujillo, eds, ‘DaWaK’, Vol. 4081 of *Lecture Notes in Computer Science*, Springer, pp. 95–105.
- Neumayr, B., Grün, K. and Schrefl, M. (2009), Multi-level domain modeling with m-objects and m-relationships, *APCCM 2009*.
- Schrefl, M., Kappel, G. and Lang, P. (1998), Modeling collaborative behavior using cooperation contracts, *Data Knowl. Eng.* **26**(2), 191–224.
- Song, I.-Y., Khare, R., An, Y., Lee, S., Kim, S.-P., Kim, J. and Moon, Y.-S. (2008), Samstar: An automatic tool for generating star schemas from an entity-relationship diagram, *ER 2008*, pp. 522–523.
- Torlone, R. (2008), Two approaches to the integration of heterogeneous data warehouses, *Distributed and Parallel Databases* **23**(1), 69–97.
- Trujillo, J., Palomar, M., Gómez, J. and Song, I.-Y. (2001), Designing data warehouses with OO conceptual models, *IEEE Computer* **34**(12), 66–75.
- Zhao, J. and Schewe, K.-D. (2004), Using abstract state machines for distributed data warehouse design, *APCCM 2004*, pp. 49–58.

A Researcher Expertise Search System using Ontology-Based Data Mining

Ravikarn Punnarut and Gridaphat Sriharee

Department of Computer and Information Science
King Mongkut's University of Technology North Bangkok
1518 Pibulsongkram, Bangsur, Bangkok 10800, Thailand

Email: [ravikarn_pu | gridaphat] @kmutnb.ac.th

Abstract

This paper proposes an approach to discover the expertise of researchers using data mining with skill classification ontology. The skill classification ontology is an information model containing skills of doing research in the area of computer and information science. A methodology to build the ontology is presented. The expertise search system is developed, which uses the skill classification ontology, researcher profiles and research profiles in the retrieving process. These profiles and ontology are expressed by OWL. Also, the matching and ranking processes are proposed and these follow semantic-based matching. We explored the evaluation of the retrieving process and the result shows that the proposed approach enables the expertise search system to be efficient regarding accuracy.

Keywords: Expertise Search, Ontology, Matching, Ranking, Data Mining.

1 Introduction

Expertise is the embodiment of knowledge and skills within individuals (Crowder et al., 2002). An individual may have different levels of expertise about different topics, and the expertise distinguishes experts from less experienced people and novices. Expertise search describes the process of seeking to find the people who might have the desired knowledge and skills. The seeking requires a range of information relating to levels of knowledge or experience possessed. There are many techniques available to obtain the knowledge related to expertise. For example, Ehrlich and Shami (2008) analyse the tools that people use to search for an expert such as questioning through dialog with an expert, personal networks and directories. Some systems use the profiles that indicate their expertise; such profiles may be obtained from different information sources such as curriculum vitae, publications, blogs, web sites and research project details. Combining such different data together may require gluing of vocabulary (Aleman-Meza et al., 2006). The expertise search is thus a mechanism to support expert search.

Currently, there are a number of expertise search systems and that are implemented for particular systems or online communities. In the former case, the system may have a limited data set and searching is by keyword/field matching according to the used technologies (e.g. the databases or the profiles). For expertise search in an online community, correlation of the data and the analysis of them are important. Therefore, it is hard to locate particular expertise by searching in an online community because of the vast amount of data. The expert search system may also provide an initial analysis of information by assigning scores based on degree of expertise. Also, ranking relevant expertise is focused.

Data mining is the process of discovering knowledge and their associations from a large amount of the data. The data mining with ontology is using ontology to represent the results (Nigro et al., 2007). The system is able to determine the knowledge at different concept levels (Han, 1995). The most representative applications relate to many research areas such as Medicine, Biology and Spatial Data, etc. Using ontology, an analysis of information relevancy is implemented by considering on the semantic relations of terms defined in ontology. Also, the degree of the relevancy is defined. Applying data mining, the expertise search system develops some algorithms to extract association rules or knowledge from a large collection of data.

In this paper, we propose an approach of using ontology to determine the expertise of the researcher and for retrieving process. We developed an expertise search system that used skill classification ontology to analyse the expertise of the researcher. Here, skill represents the expertise of the researcher and is analysed based on the research conducted by that researcher. We propose some contributions with detailed descriptions as follows.

- (i) A process to build the skill classification ontology. The skill classification ontology is an information model that contains terms related to various types of expertise in the area of computer and information science.
- (ii) A methodology of data mining to determine expertise of the researcher using the skill classification ontology. Probability value of the determination is defined.
- (iii) The matching and ranking methodology in retrieving the relevant researchers who may have competency matched to the desired expertise. The matching follows semantic-based matching according to the skill

Copyright ©2010, Australian Computer Society, Inc. This paper appeared at the Seventh Asia-Pacific Conference on Conceptual Modelling (APCCM 2010), Brisbane, Australia, January 2010. Conferences in Research and Practice in Information Technology, Vol. 110. Sebastian Link and Aditya K. Ghose, Eds. Reproduction for academic, not-for profit purposes permitted provided this text is included.

classification ontology. Also, the evaluation is presented.

The rest of the paper is organised as follows. Section 2 discusses some related works. Section 3 presents the development of the expertise search system and describes the data set used in this work. Section 4 describes the skill classification ontology and the methodology to create it. Section 5 presents the determination of expertise of the researchers. Section 6 presents the matching and ranking process in retrieving the researchers according to the desired expertise. Section 7 illustrates the developed expertise search system and the evaluation is discussed. Section 8 is a conclusion.

2 Related Work

There are various approaches related to expertise search. Zhang et al. (2007) utilise an online community to find the people who may have expertise for answering a particular question. They analyse the experts by considering interactions of the people in questioning and answering the questions. Sim and Crowder (2004) use existing organisationally heterogeneous information sources to locate the experts. They provide an expertise model that defines the relationship of different information sources to locate the experts. Another approach of locating expertise is the work of McDonald and Ackerman (2000). They propose making referrals of information to assist the people to find the experts. Ackerman et al. (2003) present an approach that allows everyone to contribute their competencies by labelling their expertise with relative concepts that describe the expertise. Macdonald and Ounis (2008) propose an approach that uses a range of documents in searching for expertise. The degree of the document is defined and used in the ranking process. Tang et al. (2007) propose an expertise search system that analyses information from a web community. They use ontology to determine the correlation between information collected from different sources.

Our work is close to that of Tang et al. (2007) by which the ontology is used to determine the expertise of the researchers. However, our work focuses on the conceptual level of the expertise according to the ontology. We address the ranking of relevant results – similar to the work of Macdonald and Ounis (2008) – in which the number of profiles related to the person is one of the criteria for ranking. In contrast to the works mentioned above, we use ontology for expertise analysis and the matching and ranking relevant results are addressed according to ontology and subsumption relationship (Brachman, 1983) of the concepts that represent the expertise.

3 The Process and Data Set

3.1 The Overview of the Process

Figure 1 represents the process of the development of the expertise search system. There are various steps involved with details as follows.

- (i) Collecting the data set. The data set consists of the description of the research papers and research projects (described in Section 3.2).

- (ii) Creating skill classification ontology. The skill classification ontology consists of CCS ontology and support ontology. The CCS ontology follows the ACM category (ACM, 1998). The support ontology is an extension model of the CCS ontology. The skill classification ontology is used for retrieving expertise.
- (iii) Expertise analysis of the researcher. We use the skill classification ontology to analyse the related expertise of the researcher (described in Section 5).
- (iv) Creating researcher and research profile. In this step, we provided a generator to create the profiles that are expressed by OWL (OWL, 2004).
- (v) Developing the system and including the matching and ranking process for retrieving expertise.

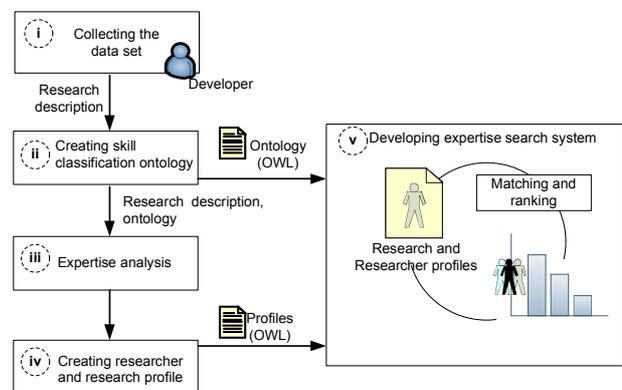


Figure 1: The process

3.2 Data Set

We collect description for use in the system from two sources: the proceedings of the JCSSE and NCSEC conferences and the research report of NRCT. The JCSSE (International Joint Conference on Computer Science and Software Engineering) and NCSEC (National Computer Science and Engineering Conference) are well-known conferences for national researcher collaboration in the areas of computing, computer engineering and information science. The proceedings of JCSSE are available online at <http://jcsse.cp.eng.chula.ac.th> and the proceedings of NCSEC are available in hard copy. The NRCT (National Research Council of Thailand) provides information on conducted research that is funded by the council. The information includes project description and researcher description, and both are provided in terms of XML documents. The project description is a description of research projects that are conducted and reported in a particular year and the researcher description is a description of the researchers that have registered with the NRCT.

4 Creating Skill Classification Ontology

4.1 The Process

Currently, there is no ontology available that represents expertise. Therefore, we built a skill classification

ontology. Figure 2 depicts the building process of the skill classification ontology. We extracted terms from the titles of the conducted researches of the data set (a). The extracted terms are considered in matching (b) with XML-based ACM category (ACM, 1998) (see Figure 3). The mismatched terms are the terms that are not matched to ACM category, but such terms can be considered to be defined into the support ontology since these indicate some skills. We consider the terms to be defined into the support ontology using FOLDOC computer dictionary (FOLDOC, 1993) (e). The support ontology and CCS ontology are created and these are expressed by OWL. The CCS ontology consists of terms defined in the ACM category. The skill classification ontology is used for determining expertise of the researcher, matching and ranking in the expertise search system.

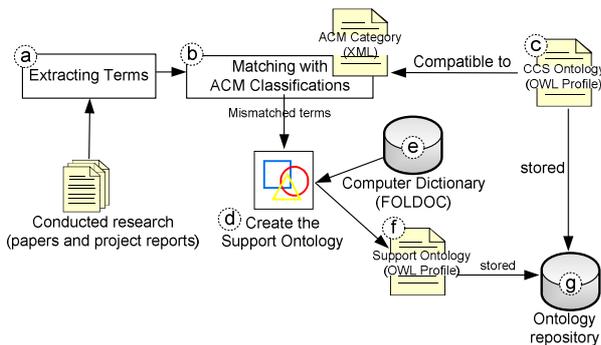


Figure 2: Building skill classification ontology

4.2 CCS Ontology

The ACM Computing Classification System (ACM, 1998) is adopted to create the CCS ontology. It provides indexing for computing publications so that the users can search through ACM’s digital library. The ACM Computing Classification System represents a four-level tree that has three coded levels of terms. The reader can see the CCS model at <http://www.acm.org/class/>. Figure 3 is an example of an ACM category and their sub-categories and related terms.

We formalised the ACM category. For example (see Figure 3), the category H.2.4 is classified into sub-categories: *H.2.4.1 Concurrency*, *H.2.4.2 Distributed databases*, *H.2.4.3 Multimedia Database* and so on. The formalised terms are defined into the CCS ontology.

4.3 Support Ontology

The support ontology contains skill terms related to expertise, and it is an extension model of the CCS ontology. We conducted an experiment to extract terms from the title of the research paper and project. Of the results explored, 1,272 research titles are used for matching to the ACM category (ACM, 1998). Table 1 represents an example of the matched terms and extension terms.

The associations of the terms defined in the support ontology and the CCS ontology are defined with OWL properties such as *owl:equivalentClass* and *rdfs:subClassOf*. Figure 4 represents an example of the association between the term *Authentication* in the support ontology and in the CCS ontology.

```
<node id="acmccs98" label="ACMCCS98">
  <isComposedBy>
    <node id="A." label="General Literature">...</node> ...
    <node id="H." label="Information Systems">
      <isComposedBy>
        <node id="H.0" label="GENERAL"/>
        <node id="H.1" label="MODELS AND PRINCIPLES">
          <node id="H.2" label="DATABASE MANAGEMENT">
            <isRelatedTo><node id="E.5"/></isRelatedTo>
            <isComposedBy>
              <node id="H.2.0" label="General"> ... </node>
              ...
              <node id="H.2.4" label="Systems">
                <isComposedBy>
                  <node label="Concurrency" />
                  <node label="Distributed databases" />
                  <node label="Multimedia databases" />
                  <node label="Object-oriented databases" />
                  <node label="Parallel databases" />
                  <node label="Query processing" />
                  <node label="Relational databases" />
                  <node label="Rule-based databases" />
                  <node label="Textual databases" />
                  <node label="Transaction processing" />
                </isComposedBy>
              </node>...
            </isComposedBy>
          </node>...
        </isComposedBy>
      </node>...
    </isComposedBy>
  </node>
```

Figure 3: Example of ACM category

CCS Categories	Ex. of Matched terms	Ex. of extension terms (defined in support ontology)
A. General Literature	dictionaries	-
B. Hardware	microcomputers, circuits, channel	clock gating, VHDL, OFDM
C. Computer Systems Organization	Security, asynchronous, transfer mode, protocols	Satellite, time division multiple access, radio wave
D. Software	enhancement measurement, prototyping	UML, XML
E. Data	data encryption standard (DES), compression, encoding	data migration, data reduction, video coder, decision tree
F. Theory of Computation	Mathematical, neural networks, routing	mathematical models
G. Mathematics of Computing	Statistic, time series analysis, wavelets	brownian motion, location estimation, sequential random
H. Information Systems	information services, library automation, GIS	knowledge base, sound synthesis, data fusion
I. Computing Methodologies	three dimensional, artificial intelligence, fuzzy	message passing interface (MPI), hidden markov model
J. Computer Applications	sciences, financial, manufacturing	Hazard and operability, quality control
K. Computing Milieux	CAI, software development, policy	electronic learning, watermarking, open source

Table 1: Matching terms with CCS

The association is defined with equivalent property where the concepts *Fingerprint*, *Watermarking*, *Electronic Signature* and *Face Detection* are defined as subconcepts (i.e. specific concepts) of *Authentication*. Combining the support ontology and the CCS ontology forms the skill classification ontology.

The skill classification ontology contains 1,644 terms (unduplicated concepts) in which 420 terms are defined in the support ontology and the rest are defined in the CCS ontology.

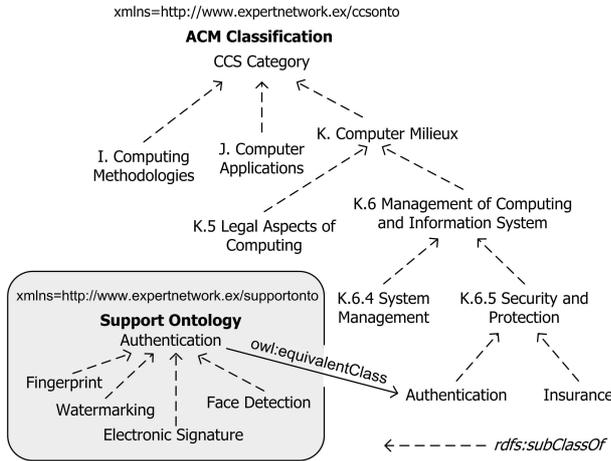


Figure 4: Part of skill classification ontology

5 The Expertise Determination

Using skill classification ontology, the system is able to determine the expertise of the researcher. Also, the system defines a score to represent the degree of expertise possessed by an individual researcher. Figure 5 depicts the research entitled “*Biochemistry Multimedia Learning on Web*” and the extracted terms: *Multimedia*, *Learning* and *Web* to be analysed. The extracted terms from the research title are denoted as follows.

$$T = \{t_1, t_2, \dots, t_n\} \quad \dots(a)$$

Where T represents a set of extracted terms (t_1 to t_n) of a particular research title.

The detailed description of the determination in each step is as follows.

Step 1: Matching skill. This step aims to discover the skill categories related to the terms of the research. In this example (see Figure 5), there are skill categories H.2.4.3 and H.5.1 matched to the term *Multimedia*; I.2.6, I.2.6.1, I.2.6.7, K.3.1.0, and K.3.1.3 matched to the term *Learning*; and H.3.5.2 and H.5.3.7 matched to the term *Web*. The matched terms of each extracted term (from formula (a)) are the set of terms defined in the skill classification ontology. This is denoted as follows.

$$t_i = \{t_{i1}, t_{i2}, \dots, t_{in}\} \quad \dots(b)$$

Where t_{i1} to t_{in} are the matched terms of the extracted terms t_i (from formula (a)).

Step 2: Computing probability for relevant skill categories. This step aims to define the probability for matched skill categories. This is denoted as follows.

$$Prob(t_i) = 1 / N \quad \dots(c)$$

Where N is the number of matched categories relating to the extracted term t_i .

In this example, the term *Web* relates to two skill categories such as *H.3.5.2 Online Information Services* and *H.5.3.7 Group and Organization Interfaces*. Hence, the probability of each matched category is defined as 0.5. This means the term *Web* may relate to category H.3.5.2 or category H.5.3.7 with equivalent probability. The probability represents the degree of the matched category of the extracted term from the research title.

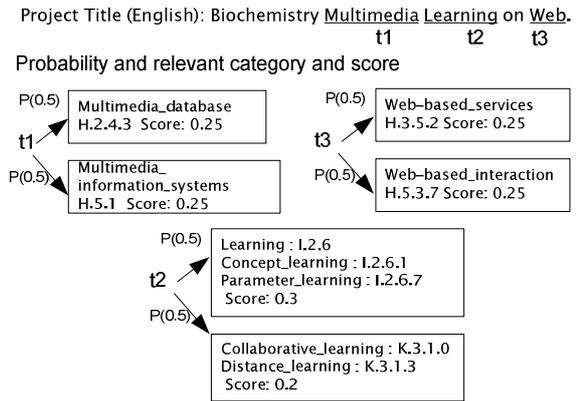


Figure 5: Example of expertise and score

Step 3: Computing score for relevant expertise. This step aims to assign a score for each relevant expertise, and the score varies with respect to the number of relevant skills of the categories. The score of a relevant expertise can be computed by multiplying the probability of the category by the match-term ratio. The match-term ratio is the number of matched terms of each category compared with the number of matched terms related to the extracted term. This is represented by the formula below.

$$Score_c = \frac{Prob(t_i) \times M_c}{M} \quad \dots(d)$$

Where M_c is the number of matched terms for particular category (c) that is related to the extracted term t_i , M is the number of matched terms (t_{i1} to t_{in} in formula (b)) of term t_i and $Prob(t_i)$ computed from formula (c).

For example (see Figure 5), the term *Learning* has the defined probability 0.5 and 5 relevant skill terms, of which 3 terms related to I.2.6 and 2 terms related to K.3.1; thus, the computed scores are 0.3 (i.e. $0.5 \times (3/5)$) and 0.2 (i.e. $0.5 \times (2/5)$), respectively. The computed scores are assigned to each matched term in particular category. The matched term and its score represent the defined expertise judged by the system and these are considered in matching and ranking later. Note that the computed score in this step does not represent any significance regarding the degree of expertise. In this work, the number of the conducted research is considered regarding the degree of expertise (see Section 6.2.1).

The obtained expertise and its relevant score are defined into the research profile. In case the conducted research has co-researchers, the system defines the determined expertise and scores to all of them.

Figure 6 depicts an example of a researcher profile with the linkages to the research profile (see Figure 7) conducted (indicated by <projectMember>) by two researchers. Both profiles are expressed by OWL (OWL, 2004). In this work, we built an OWL profiles generator to create the research and researcher profiles, and the generator is developed with integrating Jena API (Jena, 2001).

```
<position rdf:datatype="xsd:string">
    รองศาสตราจารย์ ระดับ 9</position>
<highestEducation rdf:datatype="xsd:string">
    Doctoral Degree </highestEducation>
<workOrg rdf:datatype="xsd:string">
    มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี</workOrg>
<conductProject rdf:resource="#P11000264"/>
<know rdf:resource="#R11000058"/>
</Researcher>
```

Figure 6: An example of the researcher profile

6 Matching and Ranking

6.1 Expertise Matching

We define two modes for retrieving: exact match mode and flexible match mode. These matching are defined based on subsumption relationship (Brachman, 1983) with details as follows.

(i) *Exact match mode.* Given the query with the desired expertise E (i.e. $Query(E)$), the system retrieves the researcher who may have the skill T (i.e. $Retrieve(T)$) where T is equivalent term of E . This is denoted as follows.

$$Query(E) = Retrieve(T) \text{ where } T \equiv E \dots(e)$$

(ii) *Flexible match mode.* Given the query with the desired expertise E , the system retrieves the researcher who may have the expertise T where T is a specialised concept of E . This is denoted as follows.

$$Query(E) = Retrieve(T) \text{ where } T \sqsubseteq E \dots(f)$$

In the case where there is no specialised concept of E , the system will perform a generalised match in which the generalised concept T corresponding to E will be retrieved. This is denoted as follows.

$$Query(E) = Retrieve(T) \text{ where } E \sqsubseteq T \dots(g)$$

In the case where the query is specified with multiple terms (expertise), the matching is considered for each particular term. For multiple terms query, the researcher profile satisfies the query if the set of retrieved skill matches to the set of expertise specified in the query. This is denoted as follows.

$$Query(\mathcal{E}) = Retrieve(\tau) \dots(h)$$

Where \mathcal{E} is a set of expertise specified in the query and τ is a set of retrieved expertise.

```
<Researcher rdf:ID="R42040150">
  <researcherNameThai rdf:datatype="xsd:string">
    ดาวัลย์ ชิมภู</researcherNameThai>
  <researcherNameEng rdf:datatype="xsd:string">
    Dawan Shimbhu, Mrs. </researcherNameEng>
</Researcher>
<ResearchProject rdf:ID="P11000264">
  <projectNameThai rdf:datatype="xsd:string">
    การสร้างสื่อการเรียนวิชาชีวเคมีผ่านระบบเครือข่ายคอมพิวเตอร์</projectNameThai>
  </projectNameThai>
  <projectNameEng rdf:datatype="xsd:string">
    Biochemistry Multimedia Learning on Web</projectNameEng>
  </projectNameEng>
  <group rdf:datatype="xsd:string">NRCT</group>
  <keyWord rdf:datatype="xsd:string">
    multimedia, learning, web</keyWord>
  <hasSkill rdf:parseType="Resource">
    <skillCCS rdf:resource="#H.2.4.3"/>
    <score rdf:datatype="xsd:float">0.25</score>
  </hasSkill>
  <hasSkill rdf:parseType="Resource">
    <skillCCS rdf:resource="#H.5.1"/>
    <score rdf:datatype="xsd:float">0.25</score>
  </hasSkill>
  ...More defined skill ...
  <ProjectMember rdf:resource="#R42040150"/>
  <ProjectMember rdf:resource="#R11000058"/>
</ResearchProject>
```

Figure 7: An example of the research profile

Figure 8 represents an example of the matching when the query is specified with the term $H.2$ Database_management. The expertise of the researcher B is exact match to the query and the researcher A can be retrieved regarding specialised match. In case the query is specified with the term $H.2.4.3$ Multimedia_database, the researcher A is an exact match to the query and the researcher B is a generalised match to the query.

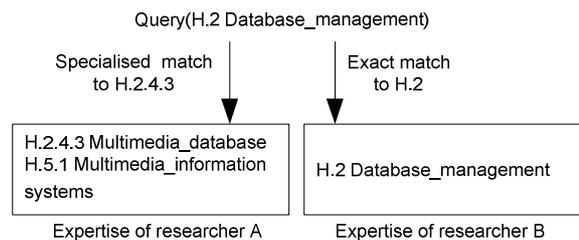


Figure 8: An example of matching

6.2 Ranking the Retrieved Researchers

We break down ranking into two steps: ranking according to *rank-score* and ranking according to *path-rank*. The former relates to the score defined in formula (d) (see Section 5) and the latter relates to the hierarchy of the skill classification ontology.

6.2.1 Rank-Score

Ranking can be considered from rank-score denoted as follows.

$$RankScore(E) = \text{Maximum}(Score(T_1) \times N_1, \dots, Score(T_n) \times N_n) \dots (i)$$

Where N_1 to N_n is the number of papers related to the retrieved skill T_1 to T_n that are the matched terms of E .

Note that the scores that are considered the maximum score are obtained from formula (d) (see Section 5).

Figure 9 depicts an example of retrieved results in flexible match mode when the category I.2.8 is specified in the query. The researchers A, B and C are retrieved since their expertise are specialised match to the category I.2.8.

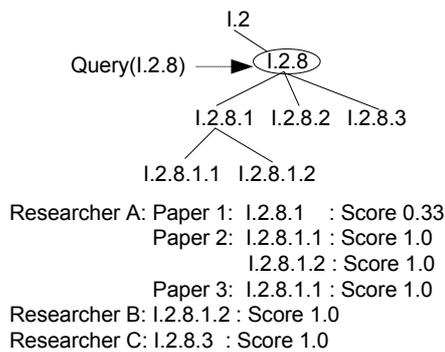


Figure 9: An example of matching

With regard to formula (i), the rank-score related to the matched expertise is computed as follows.

$$Researcher A: RankScore(I.2.8) = \text{Maximum}(0.33 \times 1, 1 \times 2, 1 \times 2, 1 \times 1) = 2$$

$$Researcher B: RankScore(I.2.8) = \text{Maximum}(1 \times 1) = 1$$

$$Researcher C: RankScore(I.2.8) = \text{Maximum}(1 \times 1) = 1$$

In this example, the researcher A is ranked in a higher order than the researchers B and C because of the number of the papers. The researchers B and C are in the same rank. According to the query with multiple terms, ranking can be considered from the summation of the rank-score of the set of retrieved expertise that matches to the query.

6.2.2 Path-Rank

Since rank-score may not be able to distinguish between different types of researcher expertise, path-rank is needed. We define the path-rank as the distance between

the desired expertise and the matched term according to the hierarchy of skill classification ontology.

We focused retrieving based on user preference. The user may specify the query with the generic term or specific term (narrower semantics) according to terms defined in the ontology. With generic term, it represents that the user may have no knowledge of the expertise in the deep level or may not expect the narrower expertise in querying. In contrast, the query with the specific term represents that the user searches with particular requirement. In this work, the system ranks the results by giving the significance to the shortest distance between terms specified in the query and the retrieved expertise. This is denoted as follows.

$$PathRankScore(E) = PathRank(E \rightarrow T) \dots (k)$$

Where the path-rank is a value according to the distance from E to T in the hierarchy of skill classification ontology.

Note that path-rank is implemented in two retrieving modes: exact match mode, and flexible match mode with specialised match only.

In the previous step (Section 6.2.1), the researchers B and C are retrieved in equivalent rank because their rank-score are the same i.e. 1.0. Regarding the path-rank, the distance from the node I.2.8 to I.2.8.3 is shorter than the distance from the node I.2.8 to I.2.8.1.2. Thus, the system ranks the researcher C in a higher order than the researcher B. The long distance represents a poor rank. The system hence ranks the results by considering the ascending order of the path-rank. The query with multiple terms can be considered according to the summation of path-rank scores of the set of retrieved expertise that matches to the query.

7 The Development and Evaluation

We performed the evaluation by giving a set of expertise terms corresponding to the skill classification ontology and evaluated the results by investigating the retrieved results. The results are retrieved according exact and specialised match. The evaluation is conducted (by human) with regard to the following cases.

- (i) The system is able to retrieve the researchers who may have expertise matched to both the term specified in the query and the keyword specified in the research.
- (ii) The system is able to retrieve the researchers who may have expertise matched to the query but the keywords indicated in the research are not relevant to the query. However, the research paper is relevant to the desired expertise specified in the queries. In this case, the author may have specified keywords that are not appropriate to the researchers.
- (iii) The system retrieves irrelevant results. This means the results are not relevant to the desired expertise.

The evaluation described above assesses the efficiency of the system and the proposed approach. For example, evaluation results represented by cases (i) and (ii) represent positive outcomes whereas case (iii) represents a negative outcome.

The data set for retrieving contains 1,046 research profiles (OWL) with the defined expertise (by system) and specified keywords. There are 1,693 researcher profiles that are used in the system and that have the linkage to the research profiles. Within the data set used, 245 research papers have their keywords (from ACM) specified by the authors and these keywords can be considered as the authors' description of their own skills. We also specified the keywords (from skill classification ontology) to 801 researches (papers/projects) that are not specified keywords before. Such these keywords are judged by the developer whether the results are relevant or not relevant to the test query.

Table 2 depicts the results of the evaluation. The evaluation can be summarised in terms of precision and recall. The recall value is computed from the ratio between the number of retrieved objects and the number

Query(E)	Retrieved objects (1)	Relevant to author keyword and query (+) (2)	Irrelevant to author keyword/ no keywords but relevant to query (+) (3)	Irrelevant to query (-) (4)	Precision = (2)/(3)+(4) (1)
Authentication	21 (31)	3 (5)	11 (15)	7 (11)	0.67 (0.65)
Cellular architecture	57 (92)	14 (27)	33 (51)	10 (14)	0.82 (0.85)
Cryptographic controls	7 (22)	1 (1)	4 (16)	2 (5)	0.71 (0.77)
Data encryption	15 (26)	2 (2)	8 (15)	5 (9)	0.67 (0.65)
Data mining	54 (72)	25 (28)	16 (15)	13 (29)	0.76 (0.60)
Data warehouse and repository	6 (18)	2 (5)	3 (12)	1 (1)	0.83 (0.94)
Network protocol	38 (52)	2 (4)	21 (28)	15 (20)	0.61 (0.62)
Programming techniques	23 (33)	0 (0)	10 (19)	13 (14)	0.43 (0.58)
Scene Analysis	30 (54)	1 (2)	16 (31)	13 (21)	0.57 (0.61)
Semiconductor Memories	4 (5)	0 (0)	4 (5)	0 (0)	1.00 (1.00)
Sound and Music Computing	20 (29)	1 (3)	11 (21)	8 (15)	0.60 (0.83)
Fuzzy set	25 (39)	4 (9)	10 (16)	11 (14)	0.56 (0.64)
Average					0.69 (0.73)

Table 2: Matching terms with CCS

of relevant objects in the data set. Note that the number specified without parenthesis represents the number of papers/projects while the number within parenthesis represents the number of researchers. The second column (indicated by (1)) represents the number of the retrieved objects according to the query in the first column. The third to the fifth columns represent the results corresponding to the cases (i) - (iii) mentioned above, and the final column represents the precision value.

From Table 2, the system returned the recall value that is 1.0 which means the number of relevant objects in the

data set is equal to the number of retrieved objects. Also, the precision value is 0.69 and 0.73 regarding the number of retrieved papers/projects and the number of the researchers respectively.

Figure 10 and 11 depict the examples of the user interface for the developed expertise search system. The user can specify the desired expertise through the provided ontology browser implemented by AJAX technology. The system is implemented by JSF framework integrating with Jena API version 2.1 (Jena, 2001) for querying.

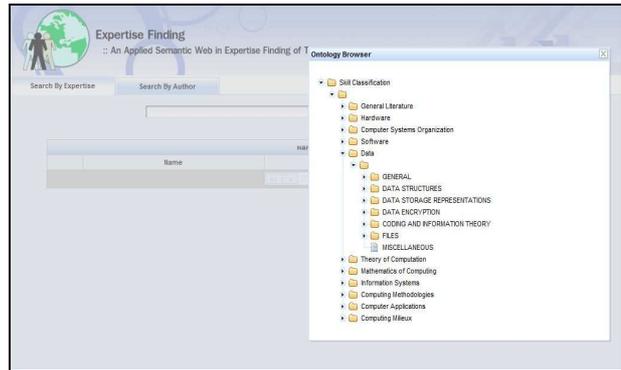


Figure 10: User interface for query



Figure 11: User interface expertise search system

8 Conclusion

In this work, we proposed a methodology for building a skill classification ontology by extracting text from research titles. With the skill classification ontology, the system is able to determine the related expertise of the researcher, and that expertise is analysed in regard to both broad skills and deeper skills. Matching and ranking processes are presented and these are conducted according to a semantic-based approach. Currently, the system supports the query with single term and two terms. From our experimentation, the proposed matching and ranking algorithm is practical and enables efficient search. We also evaluated the system using others set of the test query and the system returned approximately precision value 0.70.

Our work is focused on the research papers. However, the proposed approach may be applied to use with other information that share topics of interest of the people. It is

possible to consider the different degree of expertise according to the positions of conducting research. For example, the leader of the conducted project or being the first author and second author respectively. In our system, the number of conducted researches is the most significant factor that yields well-justification of the degree of expertise. We did not focus on satisfaction of the users but rather focused on efficiency of the search system regarding accuracy. However, the evaluation on satisfaction can be implemented later.

References

- Ackerman, M.S., Wulf, V., and Pipek, V. (2003): Sharing Expertise: Beyond Knowledge Management. *MIT Press*, Cambridge, MA.
- ACM (1998). ACM Computing Classification System. Available at <http://www.acm.org/class/>
- Aleman-Meza, B., Bojars, U., Boley, H., Breslin, J.G., Mochol, M., Nixon, L.J., Polleres, A. and Zhdanova, A.V (2007): Combining RDF Vocabularies for Expert Finding. *The Semantic Web: Research and Applications*, Springer Berlin / Heidelberg, Volume 4519.
- Brachman, R. (1983): What IS-A Is and Isn't: An Analysis of Taxonomic Links in Semantic Networks. *IEEE Computer*, 16(10): 30-36.
- Crowder, R., Hughes, G., and Hall, W. (2002): An Agent Based Approach to Finding Expertise. The 4th International Conference on Practical Aspects of Knowledge Management, LNAI 2569, Vienna, Austria, December 2-3: pp.179–188.
- FOLDOC (1993): Free Online Dictionary of Computing. Available at <http://foldoc.org/>
- Han, J. (1995): Mining Knowledge at Multiple Concept Levels. Proceedings of the 4th International Conference on Information and Knowledge Management (pp. 19-24). New York.
- Jena (2001): Jena A Semantic Web Framework for Java. Available at <http://jena.sourceforge.net/>
- Ehrlich, K., Shami, N.S. (2008). Searching for Expertise. Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems, Florence, Italy.
- McDonald, D.W., Ackerman, M.S. (2000): Expertise Recommender: A Flexible Recommendation System and Architecture. CSCW 2000, Philadelphia, PA, December 2-6.
- Macdonald, C., Ounis, I. (2008): Searching for Expertise: Experiments with the Voting Model. *The Computer Journal on Expertise Profiling*, March 6.
- Nigro, H.O., Elizabeth, S., Cisaro, G., Xodo, D.H. (2007): Data Mining with Ontologies: Implementations, Findings, and frameworks, Information Science Reference, ISBN: 978-1-59904-618.
- NRCT, <http://www.riclib.nrct.go.th/>
- OWL (2004): OWL Web Ontology Language Guide. Available at <http://www.w3.org/TR/owl-guide/>
- Sim, Y., Crowder, R. (2004): Evaluation of an Approach to Expertise Finding. The 5th International Conference on Practical Aspects of Knowledge Management, vol. 3336, Vienna, Austria, December 2-3: pp. 141-152.
- Tang, J., Zhang, J., Zhang, D., Yao, L. and Zhu, C. (2007): ArnetMiner: An Expertise Oriented Search System for Web Community. Semantic Web Challenge. In Proceedings of the 6th International Conference of Semantic Web (ISWC'2007).
- Zhang, J., Ackerman, M.S., Adamic, L. (2007): Expertise Networks in Online Communities: Structure and Algorithms. The 16th international conference on World Wide Web, Banff, Alberta, Canada, May 8-12.

Business Modeling for Service Descriptions: A Meta Model and a UML Profile

Gregor Scheithauer^{1,2}Guido Wirtz²

¹ Siemens Corporate Technology,
Information & Communication,
Otto-Hahn-Ring 6, 81379 Munich, Germany,
Email: gregor.scheithauer.ext@siemens.com

² University of Bamberg
Distributed Systems Group,
Feldkirchenstraße 21, 96052 Bamberg, Germany,
Email: guido.wirtz@uni-bamberg.de

Abstract

The evolution of service-oriented architectures toward market places for business services in the Internet, raises the need for rich service descriptions with respect to service proposition and service discovery. Service providers face the challenge of business-oriented development of service descriptions for there is no conceptual formalism, a wide range and overlapping IT standards, and low alignment between business and IT. This paper reports from a research project which develops a service description method that allows documenting, communicating, and reasoning about service descriptions on various levels depending on intention and abstraction. It introduces the concepts of service market places, offers a business service meta model, and shows a valid UML Profile for it. Furthermore, a case study in the IT outsourcing domain demonstrates the strengths and weaknesses of this approach.

Keywords: Service Description, Business Model, Method, UML Profile

1 Introduction

Globalization, technological change, and an increasing demand for services (Peneder et al. 2003) transform countries from industry economies toward service economies. Regarding this trend, it becomes clear that services and their development play an important role in today's and tomorrow's business. In line with this trend, service ecosystems emerge, as an evolution of service orientation (Papazoglou 2003) that takes services from merely integration purposes to the next level by making them available as tradable products on service market places (Barros & Dumas 2006), such as *StrikeIron* and *SalesForce.com*. These providers aim at trading services over the Internet between different legal bodies, compose complex services from existing ones, and build platforms for IT-supported service provisioning (Janiesch et al. 2008). This development raises the need for rich service descriptions to enable service trade.

Figure 1 depicts steps where service descriptions contribute to service trade (cf. (Kuropka et al. 2008)). By means of service proposition, service

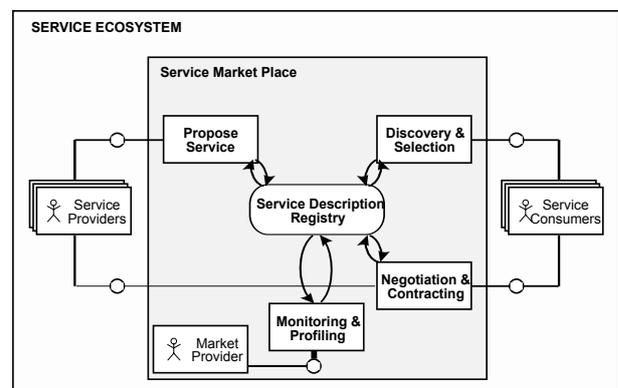


Figure 1: Trade in Service Ecosystems

providers advertise their services toward potential consumers, whereas during discovery & selection, service consumers specify their service preferences toward providers. If a service consumer selects an appropriate service, providers and consumers negotiate and finally agree on service levels (SLA) which are monitored throughout value exchange. In the event that service levels are not met, compensations have to be triggered. During service profiling, valuable information on services' performance is stored, which is gathered through value exchange and monitoring.

From the perspective of service providers, a business-oriented development of service descriptions becomes a crucial part of the service development process, which is impeded for the following reasons. Firstly, there exists no formalism for defining service descriptions on the conceptual level (Kuropka et al. 2008). Secondly, service descriptions embody divergent information and need the involvement of different subject-matter-experts. Thirdly, there do exist ample technical specifications how to describe web services with overlapping domains, which employ first-order logic, predicates, and XML, such as WSDL, WSMO, and SA-WSDL. Fourthly, there is no real alignment between business and IT. These reasons indicate that the service description development process is prone to errors, slow, and irreproducible. While recent work concentrates on *business process modeling* with a focus on how to formalize the relationship between conceptual business requirements and how to implement them with service-oriented architectures (cf. (Ouyang et al. 2006)), no attempt has been made for enhancing (the process of providing) service descriptions.

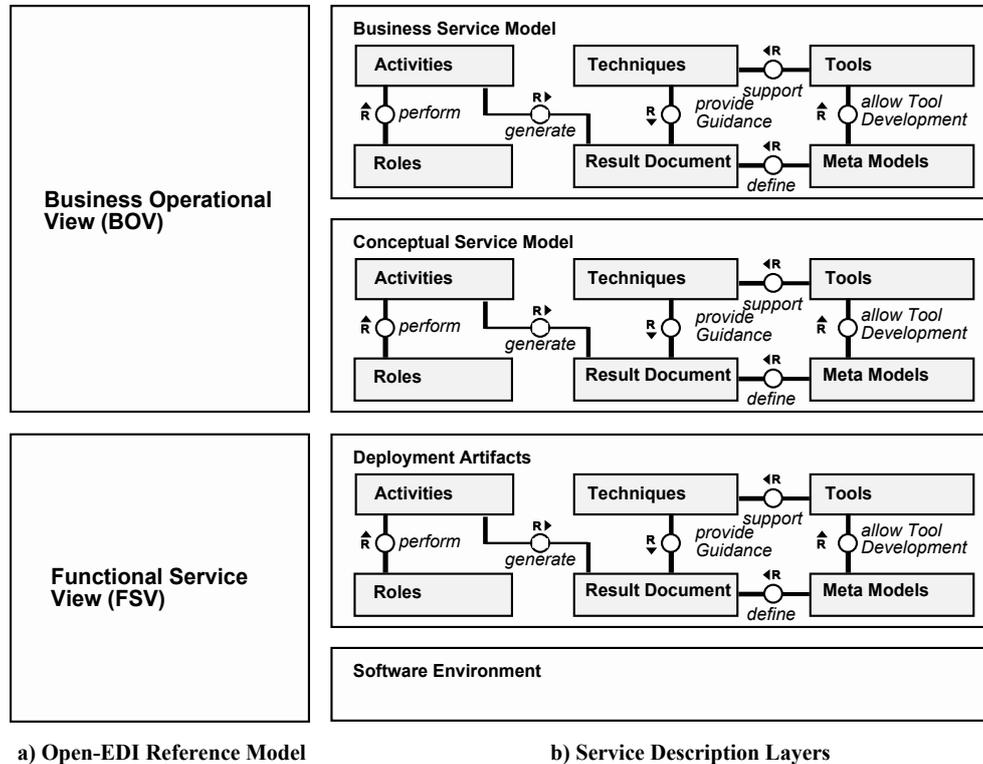


Figure 2: Open-EDI Reference Model & Service Description Layers.

Service providers would benefit from a method that allows for documenting, communicating, and reasoning about service descriptions on different levels of abstraction and, hence, support a much smoother development process. This paper tackles these problems by proposing a service description method that helps to overcome the gap between business and IT (cf. section 2). Section 3 details the Business Service Model, which comprises a knowledge structure, a notation, activities, and tools for service modeling. The approach is tested against a case study in the IT outsourcing domain in section 4. Section 5 discusses related work. Finally, section 6 concludes this work as well as offers prospects about future work.

2 Toward a Service Description Method

This section outlines the Service Description Method for Service Ecosystems (SDM4SE) (Scheithauer 2009). SDM4SE is a method to support defining service descriptions in a business-oriented fashion and to transform them into technical specifications and, hence, to ease and fasten the service description development process. Figure 2 depicts the method's cornerstones, which comprise the open-EDI reference model and method engineering, which both will be briefly explained.

2.1 Reference Model

The reference model differentiates several service description modeling phases. It is based on the open-EDI reference model (International Organization for Standardization (ISO) 2004) and work of (Dorn et al. 2007). The open-EDI reference model distinguishes between the Business Operation View (BOV) and the Functional Service View (FSV). BOV comprises business data semantics as well as business transaction rules, such as agreements and obligations between business partners. FSV, on the other hand, focuses on information technology which includes interfaces, functional capabilities, and protocols.

Dorn et al. add subtle refinements to the open-EDI reference model. They refine BOV into a business model and a process model. Business models express value exchange between different actors and business analysis. Process models represent how each actor realizes value exchanges. Likewise, they refine FSV into deployment artifacts and software environments. Deployment artifacts address implementations of business processes with technical specifications, e.g., BPEL (Alves et al. 2007). Software environments describe runtimes to execute technical artifacts. This refined model serves as a classification system for concepts and modeling notations as well as to define means to bridge gaps between different layers.

Figure 2b shows an adapted version of this reference model. Whereas Dorn et al. focuses mainly on process descriptions, this work proposes a service reference model in that the *process model* changes to *Conceptual Service Model*.

2.2 Method Engineering

Method engineering is a theory about the development of methods in the IT domain. Such methods comprise existing experience and knowledge in a domain and offer a structured approach in terms of guidance as well as documentation. Method engineering supports the formalization of this knowledge and to share it among practitioners.

According to (Gutzwiller 1994), a method embodies (1) meta models for result document specification, (2) activities to guide the modeling process, (3) role definitions, (4) tools specification, and (5) techniques (cf. figure 2b).

Result Documents embody necessary knowledge gathered throughout the engineering process. This includes, e.g., a requirement document or an architecture document. Result documents can be decomposed into sub-documents. *Meta models* define result documents by specifying a knowledge structure by means of concepts and their relationships. *Activities* comprise knowledge about which steps are to

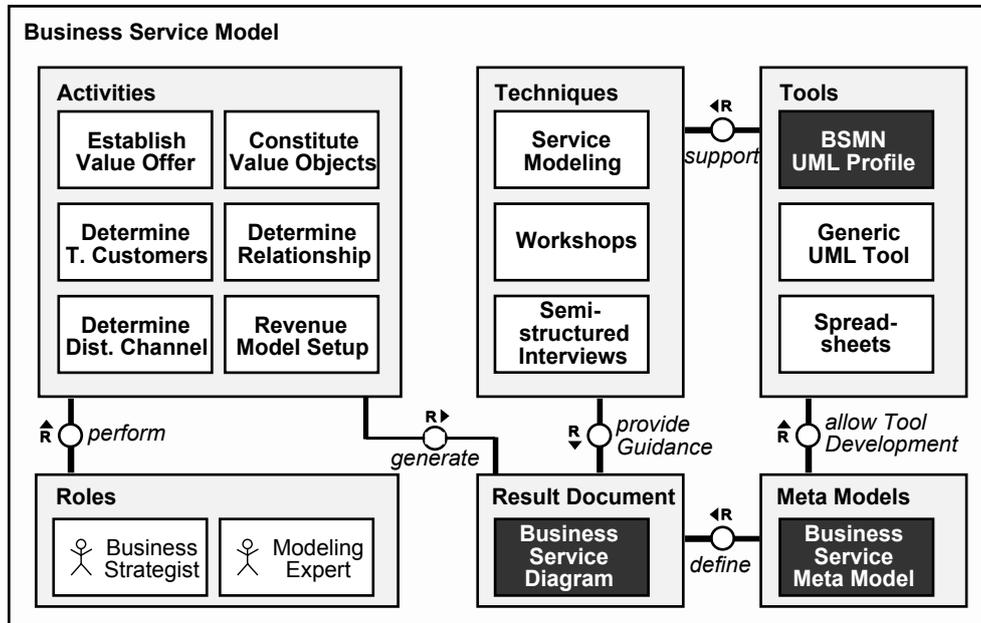


Figure 3: SDM4SE Overview for Business Service Model.

be performed in order to generate result documents. During the performance of activities semi-final result documents may be used as input. Activities may be disaggregated into sub-activities. Furthermore, sequences keep activities linked to each other. *Roles* acknowledge the fact that people with different skills are needed at certain stages in a method. A role defines a specific set of human skills which are needed for an activity. *Techniques* describe theories which are helpful to complete result documents, which include, e.g., data modeling, workflow modeling, and interviews, just to name a few. *Tools* lastly, provide support for techniques.

2.3 SDM4SE: Service Description Method for Service Ecosystems

This subsection outlines the Service Description Method for Service Ecosystems that intends to remedy the issues involved with the development of service descriptions. Figure 2b shows the combination of the reference model and method engineering. (Scheithauer et al. 2009b) argue that service properties in the *Business Service Model* layer own a strategic semantics and take into account services' final purpose and context. The next layer, the *Conceptual Service Model*, represents the actual modeling purpose of service descriptions. Service properties on this layer reflect a firm establishment with concrete values. The result is a value proposition toward potential customers. *Deployment Artifacts* describe technical-related specifications to implement service properties. Each layer features the artifacts from method engineering: activities, roles, techniques, result documents, tools, and meta models.

The service description layers offer an appropriate work-break-down structure in order to reduce complexity and to establish a bridge between business and IT. The definition of method engineering artifacts provides a conceptual formalism for service descriptions. Figure 2b shows that method engineering artifacts need to be defined for each layer. This is due to the fact that each layer presents a discreet phase in the service description development process. By defining the method engineering artifacts for each layer, it is possible to acknowledge different subject-matter-experts involved in describing services by codifying best-practices, to manage and generate IT spec-

ifications, and to offer cohesion between business and IT, which in turn results in less errors, fasten the development process, and makes it comprehensible. The following paragraphs briefly describe each layer.

Business Service Model: Figure 3 depicts an overview of this layer. Its purpose is to grasp services' core idea. Its meta model (BSMM), which is applied in section 3.1, holds information about target customers, distribution channels, value objects, and revenue models. The corresponding modeling notation (BSMN) is a semi-formal graphical notation based on a specific UML Profile (cf. section 3.2) which is used to document business service models. Business strategists with the capability to elicit and judging opportunities in the service market are the main actors for this layer. Typical abstract activities for business service modeling are outlined in section 4; for details see also (Scheithauer et al. 2009b)). This layer is discussed in more depth in section 3.

Conceptual Service Model: This layer's purpose is to transform service ideas into concrete service offers. The layer's meta model is described in previous work (Scheithauer et al. 2008, 2009b); it holds information about functionality, QoS, marketing, legal, as well as financial aspects. A modeling notation for this layer does not yet exist, but is planned for future work. Business analysts with knowledge about service markets and products take service ideas from the Business Service Model and use them in order to model service offers. Guiding activities for this model have already been developed (cf. (Scheithauer et al. 2009b)).

Deployment Artifact: This layer implements service offerings with a deployable technical language. IT architects are responsible for this layer. No specific meta model is needed for the conceptual service's meta model applies here as well. Possible technical languages (result documents) include: (1) WSDL (Chinnici et al. 2007), (2) OWL-S (Martin et al. 2004), (3) WSMO (Roman et al. 2005), (4) SA-WSDL (Farrell & Lausen 2007), and (5) WSLA (Keller & Ludwig 2003). Guiding activities are yet to be developed.

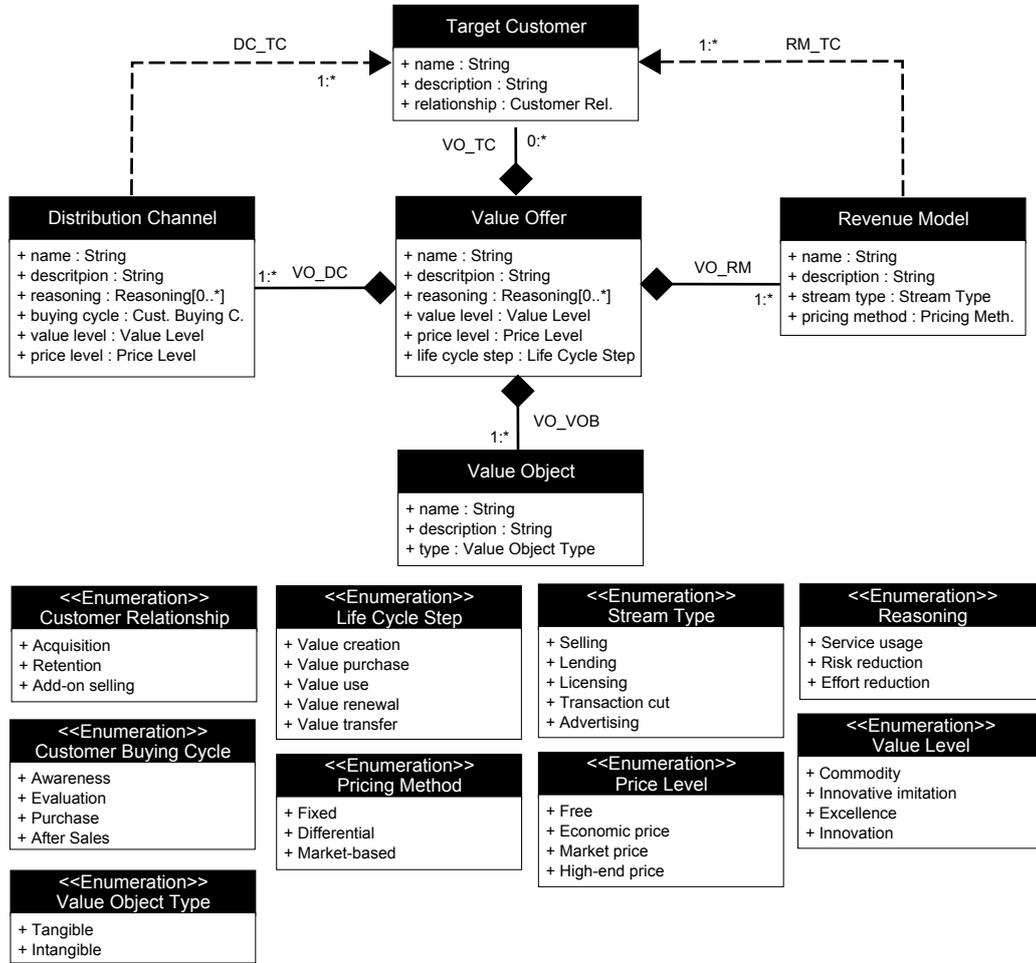


Figure 4: Business Service Meta Model (BSMM). Serves as input for UML Profile generation.

Software Environment: The most technical layer serves as a runtime for service descriptions and for deploying, discovering, and reasoning about services. Possible registries are UDDI (Organization for the Advancement of Structured Information Standards (OASIS) 2004) or WSMX (Roman et al. 2006) for semantic web services.

3 BSM: Business Service Model

Whereas the previous section provided a complete method overview, this section elaborates on the Business Service Model with its artifacts.

Figure 3 depicts a detailing view of the Business Service Model (cf. figure 2b) as well as the corresponding method engineering artifacts. In order to complete the first layer, six *activities* need to be performed: (1) establish value offer, (2) constitute value objects, (3) determine target customers, (4) determine relationship for each target customer, (5) determine distribution channel, and (6) setup appropriate revenue models. The two *roles* business strategist and modeling expert perform these six activities in collaboration. Business strategists are subject-matter-experts in a service domain and possess valuable knowledge of service markets, marketing in general, and service trends. Modeling experts, on the other hand, have the ability to elicit and to document this knowledge of business strategists. For doing so, modeling experts rely on a set of *techniques*: service modeling, workshops, and semi-structured interviews. *Tools* such as UML Profiles, UML in general, or spreadsheets support these techniques.

The black-shaded method engineering artifacts indicate the focus of this paper. (Scheithauer et al.

2009b) elaborate on the other artifacts. The first subsection introduces the BSMM, a *meta model* for defining service descriptions with a business model, used to grasp services' core ideas. The following subsection shows how to develop the BSMN, a modeling notation for the meta model based on a UML Profile as a *tool*. The last subsection addresses possible applications for both, the meta model and the modeling notation in the domains of service-oriented modeling and service engineering. The business service diagram that is illustrated in figure 10.

3.1 BSMM: Business Service Meta Model

BSMM is a knowledge structure to define service descriptions on an abstract level. (Scheithauer et al. 2009a) discuss how this model has been developed using the work of the Business Model Ontology (BMO) (Osterwalder 2004) as well as the *e*³ Value ontology (Gordijn 2002). Whereas BMO's focus lies on the internal value generation processes, the *e*³ Value ontology highlights the value exchange between different actors. The resulting model selects only specific concepts that contribute to a service description, which includes: (1) value offer, (2) value object, (3) revenue model (4) distribution channel, and (5) target customer. Figure 4 shows the resulting meta model that is explained in the following paragraphs in more detail.

Value Offer is the root element and bundles the following properties: reasoning, value level, price level as well as life cycle step. Reasoning describes in which

way a service is valuable for targeted customers. (Osterwalder 2004) distinguishes three elementary characteristics: value is either created by *using* a service, reducing any kind of *risk* for targeted customers, or reducing customers' *efforts*. The value level states to what extent services distinguish themselves from other companies' offers. Osterwalder provides four possible classifications: either a value offer is a *commodity*, an *innovative imitation*, an *excellence*, or an *innovation*. The price level expresses a services' qualitative pricing strategy. Services are either offered for *free*, for an *economic* (low) price, for an appropriate *market* price, or for a *high-end* price. The life cycle step formalizes when value is created during the service life cycle. Osterwalder explains the life cycle with five steps: *value creation*, *value purchase*, *value use*, *value renewal*, and *value transfer*.

Value Object is the actual value which is exchanged by companies offering services and companies consuming services. Evidence for this element is found by Osterwalder (called 'Resource') as well as by Gordijn ('Value Object'). Its properties include the value object itself and the value object type. The type attribute tells whether the value object is *tangible* or *intangible*.

Revenue Model describes the transformation of value offerings into income. It comprises the following properties: stream type and pricing method as well as a link to the customer property bundle. The stream type property formalizes how income is generated. Possible stream types include: *selling*, *lending*, *licensing*, *transaction cut*, and *advertising*. The pricing method describes in which way a price is determined. According to Osterwalder, a price is either *fixed* and is agnostic to the environment and customer characteristics, is *differential* and depends on product as well as customer characteristics, or is *market-based* in that the price is determined dynamically between provider and customer.

Distribution Channel tells how companies deliver value to targeted customers. The element bundles the properties: reasoning, value level, price level, and customer buying cycle. The properties reasoning, value level, and price level have the same semantic as in the value offer bundle, and hence, these can be setup for each channel. The customer buying cycle tells which step the channel addresses. Osterwalder proposes four steps for the buying cycle: *awareness*, *evaluation*, *purchase*, and *after sales*.

Target Customer specifies customer segments. Segments base, for example, on geographical criteria. The relationship property depicts in detail the type of connection between companies and their target customers. The relationship element classifies target customers according to their equity goals. Osterwalder offers three classes, namely *acquisition*, *retention*, and *add-on selling*.

3.2 BSMN: Business Service Modeling Notation (a UML Profile)

Following the business service meta model introduction in the previous subsection, this subsection elaborates on a corresponding notation. BSMN intends to support business strategists and modeling experts while *documenting* and *discussing* business service models, and hence to apply the Business Service Meta Model.

The Unified Modeling Language (UML) (Object Management Group (OMG) 2007) is an accepted and well-known semi-formal graphical language. Originally it aims at object-oriented design, but is not limited to it. UML Profile is part of the UML specification and offers a standard way to customize UML diagrams to cover domain-specific semantics. Standard UML and these *profiles* form the basis for a domain-specific modeling notation. This enables practitioners, who are already familiar with UML, to model specific domains. The UML Profiles were developed and used with the Eclipse UML 2 Toolset (*Eclipse Model Development Tools (MDT)* n.d.). (Giachetti et al. 2009) provide a UML Profile generation process to transform domain-specific languages into UML Profiles, which consists of three main steps:

1. Definition of Integration Meta Model – *Transformation of DSL into a meta model with its elements mapped to UML's meta model.*
2. Meta Model Comparison – *Identification of differences between meta model and UML superstructure.*
3. Integration Meta Model Transformation – *Setup of transformation rules and generation of a valid UML profile.*

3.2.1 Step 1: Definition of Integration Meta Model

The first step is to establish a meta model, namely the *Integration Meta Model (IMM)*, from the BSMM (cf. figure 4). Meta model elements need to be mapped to UML meta model elements. This step clarifies how to represent domain-specific elements with UML elements. Three main areas for mapping exist: (1) Classes & Properties, (2) Enumerations & Literals, and (3) Associations. Figure 5 exemplifies this. For example, it shows that the element *Target Customer* corresponds to UML Class, the element *Customer Relationship* is a UML Enumeration, and that *RM_TC* relates to a UML Association. This mapping serves as input for step 2.

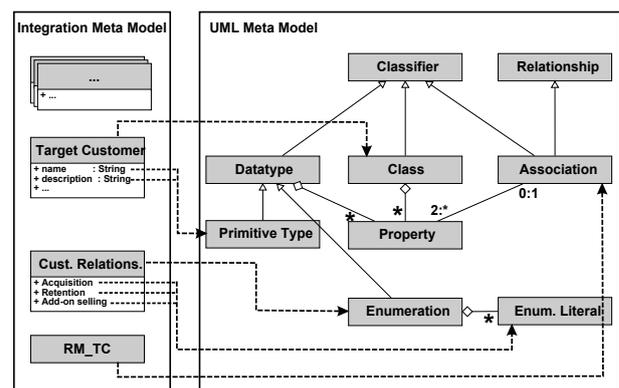


Figure 5: Excerpt of mapping between Integration Meta Model with UML Meta Model.

3.2.2 Step 2: Meta Model Comparison

With the availability of the IMM and the mapping, this step outlines differences between the IMM and the UML meta model. Each discovered discrepancy needs to be considered for the UML Profile generation. Exemplarily, table 1 shows deviances for some IMM elements that the following paragraphs explain in detail.

IMM	Differences
Target Customer name	UML Class (No Diff.)
description	UML Class :: name (No Diff.)
relationship	New Property
...	...
Cust. Rel.	UML Enumeration (No Diff.)
Acquisition	New Enumeration Literal
Retention	New Enumeration Literal
Add-on selling	New Enumeration Literal
...	...
RM_TC	UML Association (No Diff.)
endType	Different endType: IMM = TargetCustomer; UML = relatedElement
...	...

Table 1: Extract of discovered differences between IMM & UML meta model.

Classes & Properties: In step 1 identified classes and their properties were mapped to the UML meta model. For example, the IMM’s Target Customer element is mapped to UML Class. Likewise, the Target Customer’s name property is mapped to the existing UML Class property name. However, the Target Customer’s properties description and relationship may not be directly mapped and are marked with *New Property*. This Class/Property mapping is done in the same manner for the IMM’s elements: Value Offer, Distribution Channel, Revenue Model, and Value Object.

Enumerations & Literals: Likewise, enumerations and their literals are mapped to the UML meta model. E.g., the IMM’s Customer Relationship element is mapped to the UML Enumeration. The differences here are that the literals, Acquisition, Retention, and Add-on selling, are non-existent in UML’s meta model, and in consequence, marked as *New Enumeration Literals*. This mapping is similar to the other enumerations, such as, Customer Buying Cycle, Value Object Type, Life Cycle Step, Pricing Method, Stream Type, Price Level, Reasoning, and Value Level.

Associations: Lastly, associations need to be mapped to the UML meta model. The IMM outlines six associations for interconnecting classes. For example, the association RM_TC tells that a Revenue Model is valid for at least one Target Customer and is mapped with the UML element Association. However, the difference between the IMM and UML is that in case of the IMM, the Revenue Models may be only associated with Target Customer, whereas the UML Association defines its endType with any related element, and thus, the endType is marked with *Different endType*. This endType difference is similar for the remaining five associations DC_TC, VO_TC, VO_VOB, VO_RM, and VO_DC.

3.2.3 Step 3: Integration Meta Model Transformation

The last step aims at codifying the discovered differences in step 2 with *transformation rules*. Eleven rules (cf. (Giachetti et al. 2009)) are the basis for the UML Profile. This subsection goes through the rules 1, 2, 6 one by one for classes, attributes & associations, and enumerations. Other rules are skipped for they are not necessary for the BSMN.

Rule 1: (*one Stereotype for each equivalent class*) As aforementioned, the IMM shows five domain-specific classes. In coherence with rule one, each class is represented with a new Stereotype. Figure 6 exemplifies that Target Customer and Value Object are UML Classes and are represented with a Stereotype in the UML Profile definition.

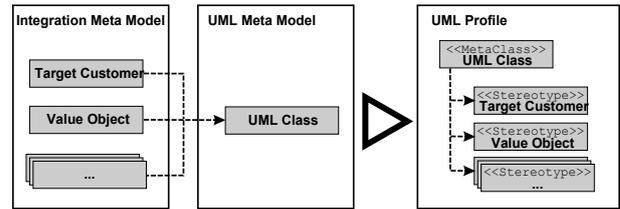


Figure 6: Application of Rule 1.

Rule 2: (*one Tagged Value for each new property*) Properties comprise *attributes* and *associations*. Tagged values consist of a name and a type. In step 2 discovered new attributes will be represented with a tagged value. For example, the class Target Customer embodies the new property description that is presented as a tagged value: description: String. Figure 7 shows rule 2’s output. It is important to note that the name attribute is not represented with a tagged value for this attribute already exists in the UML Class element.

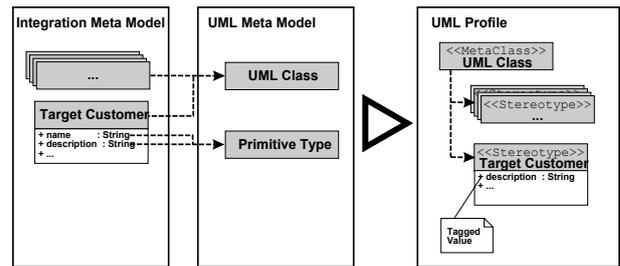


Figure 7: Application of Rule 2.

Rule 6: (*one Enumeration for each new enumeration with new literals*) Each of the IMM’s enumerations with their literals are acknowledged with a UML Enumeration. Figure 8 shows that the element Customer Relationship is an Enumeration, and that its attributes Acquisition, Retention, and Add-on selling are Enumeration Literals.

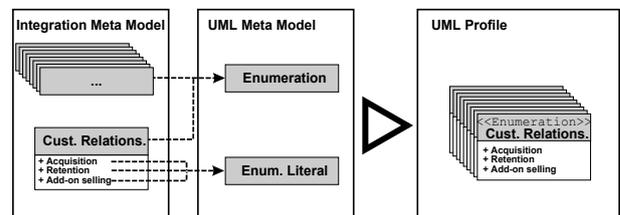
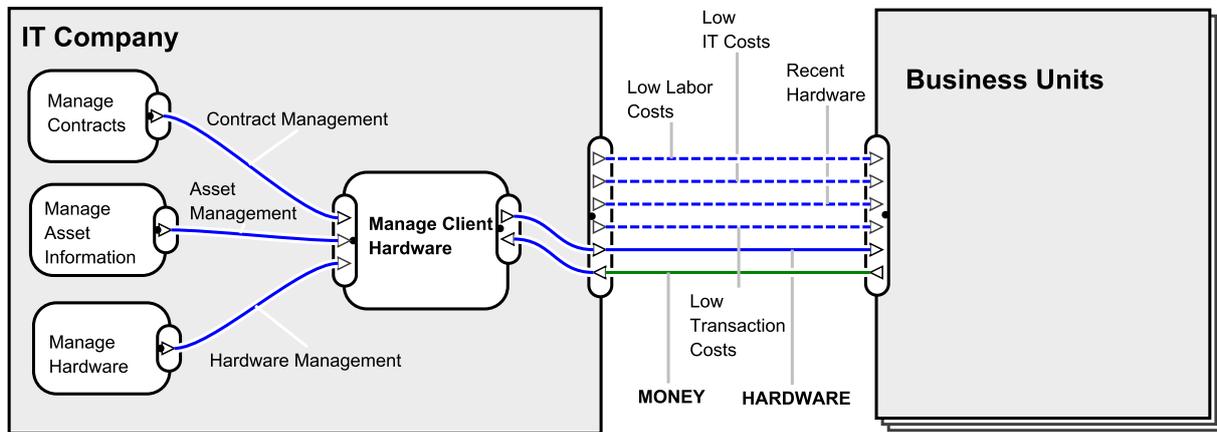


Figure 8: Application of Rule 6.

3.3 Possible Applications

Next to use BSMM and BSMN with the outlined method in section 2 and to refine service descriptions toward deployment artifacts, there exist two other promising applications for BSM: (1) modeling service-oriented architectures and (2) service engineering.

Figure 9: Manage Client Hardware Scenario (e^3 Value Diagram).

(Arsanjani et al. 2008) define Service Oriented Modeling & Architecture (SOMA) as “... an end-to-end software development method for building SOA-based solutions”. This method applies to establish a design and implementation for service-oriented architectures. It specifies a life-cycle comprising 21 steps, which are grouped into seven phases: (1) business modeling & transformation, (2) solution management, (3) identification, (4) specification, (5) realization, (6) implementation, (7) deployment, monitoring, and management. The authors recognize the first phase *business modeling and transformation* as an important first step that serves as the entry point for the phase *identification & specification*. However, Arsanjani et al. do not further describe this phase. As a suggestion, the business service meta model and the business service modeling notation may be applied to this phase.

Contrary to SOMA, (Kett et al. 2008) specify the Integrated Service Engineering (ISE) Framework for developing single business services. The ISE Framework is an orthogonal matrix and similar to the Zachman framework. The vertical axis shows four perspectives of the engineering process and is named *service perspectives*. Each perspective relates to a specific role with appropriate skills and offers different sets of tools and methods. It also implies the chronology of the framework. The horizontal axis shows five different *descriptions of a service*. Each description is valid for each perspective. Each intersection in the matrix is placeholder for a meta model, a notation, and activities, which are appropriate for the respective perspective and the modeling aspect. The Business Service Meta Model as well as the Notation fits the ISE framework’s *strategic perspective* for the *service* description.

4 Case Study

After introducing BSMM and BSMN, this section outlines a case study in the IT outsourcing domain, where a real-world business service forms the basis for evaluating the Business Service Model. The following subsections depict the case study’s scenario, the implementation of the scenario, and finally conclude with a discussion of the findings. It was necessary to modify the scenario and to disguise the company name for publication. The scenario’s scope and complexity remain the same, nevertheless.

4.1 Scenario

IT Company is a multi-national firm that offers the business service *Manage Client Hardware*. The service’s business model is to allow outsourcing of pur-

chasing and the maintaining of computer hardware e.g., a desktop PC. Figure 9 depicts the business model with the e^3 Value Ontology. The business model comprises one actor with four value activities, a market segment, and nine value exchanges. The main actor is the IT Company itself. The company possesses three internal value activities: *manage contracts*, *manage asset information*, and *manage hardware*, with value exchanges toward the main value activity *manage client hardware*, which defines the external offered service. The market segment on the figure’s right hand side pictures the company’s target customers, i.e., its own business units. Between the actor and the market segment, the figure shows six value exchanges, and their corresponding value objects. The lowest one shows the value object *Money* that goes from the business units toward the IT Company. In this case, money is exchanged for the value object *Hardware*, which is directed from the company toward the business units. Next to these tangible values which are exchanged, four other values flow from the IT Company toward the Business Units: Low Transaction Costs, Low Labor Costs, Low IT costs, and Recent Hardware. These values are so-called second-order-values that are intangible and not actually transferred between the actors (cf. (Weigand et al. 2009)). However, business units gain these values additionally to the main value objects.

4.2 Scenario Modeling

The intention of this case study is to find out BSMN’s suitability for business service modeling. It shows how to apply the UML Profile developed in section 3.2. The task includes eliciting and documenting knowledge about the *Manage Client Hardware* service for the following reasons: formalizing and communicating business ideas as well as to form a basis for service conceptualization and implementation. The scenario modeling follows the activities shown in figure 3: (1) establish value offer, (2) constitute value objects, (3) determine target customers, (4) determine relationship for each target customer, (5) determine distribution channel, and (6) setup appropriate revenue models.

Manage Client Hardware is the Value Offer. The reasoning is that it will reduce customers’ effort in that the company will provide and maintain computer hardware. The value level is set to *commodity*, for the value offer is easy to imitate by competitors. The price level is situated as *economic*. The value for customers are created while *value use* during the life cycle step. The one tangible Value Object of the service is the *hardware object*. However, next to the hardware, there exist intangible value objects

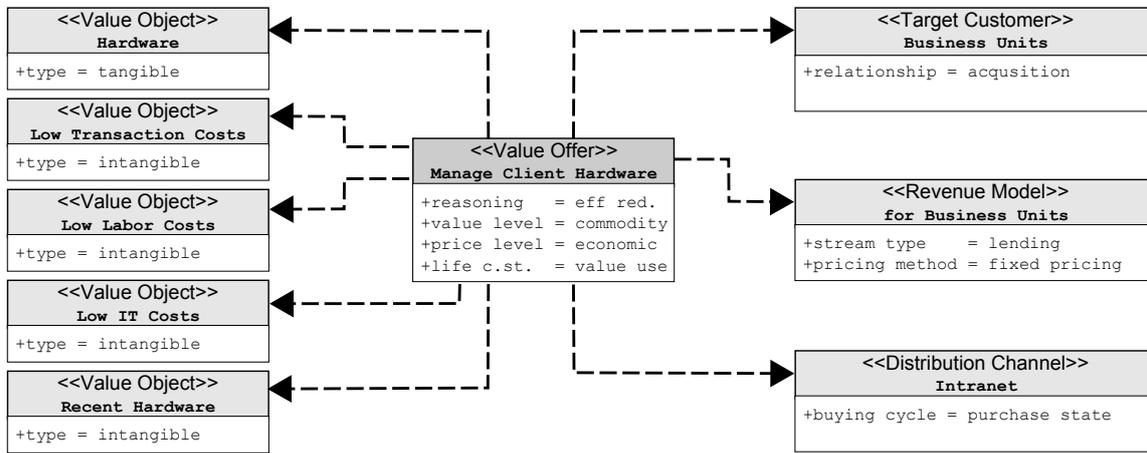


Figure 10: Business Service Diagram: Manage Client Hardware.

which also contribute to the service offering. Outsourcing hardware management to IT Company results in *lower transaction costs* for purchasing and contracting, *lower labor costs* for hardware maintenance, *lower IT costs*, and *state-of-the-art hardware*. *Business units* are the service's *Target Customers*. The Relationship to these customers is not yet established. Hence the relationship is marked as *acquisition*. The *Manage Client Hardware* service's *Distribution Channel* relies solely on the company's web online portal, which supports *purchase* state of customers' buying cycle. Likewise, the company follows one *Revenue Model*, which settles for a *fixed price* as pricing method and *lending* for the stream type.

Figure 10 depicts the final UML diagram (*resulting document*, cf. section 3) with the aforementioned Business Service Meta Model elements, which figure 4 prescribes. Additionally, listing 1 shows the corresponding XML fragment for the UML diagram that can be used for persistence and further processing such as model transformation.

4.3 Findings

The case study's intention was to figure out whether the Business Service Meta Model and Modeling Notation supports the *documentation*, the *communication*, and the *reasoning* of service descriptions on a strategic level.

The case study shows that the proposed approach is appropriate for *documenting* business service models. In particular, the developed UML Profile (BSMN) guarantees a full documentation of services' core ideas. However, business strategists were not familiar with UML or UML Profiles. This experience made it necessary to involve modeling experts who are familiar with UML in order to document business service models. One idea is to hide the notations from business strategists and rather use semi-structured interviews to elicit necessary information and use the answers to these questions to build business service diagram.

Furthermore, the case study proves that business strategists were able to *communicate* the service's main idea with involved business strategists on the basis of the business service diagram (cf. figure 10). Moreover, business strategists were in the position to *discuss* and rethink the business service model and hence to improve it.

The fact that the resulting diagrams use XML as a serialization (cf. listing 1) allows further processing of services' information, such as model transformations.

Further case studies need to detect whether the business service diagram is an appropriate starting position for the Conceptual Service Model.

Listing 1: Corresponding XML Code

```

1  <?xml version="1.0" encoding="UTF-8" ?>
2  <bm:BusinessServiceModel xmlns:bm="http:
3  //www.itcompany.com">
4  <ValueOffer Name="Manage_Client_
   Hardware" Reasoning="Effort_
   Reduction" ValueLevel="Commodity"
   PriceLevel="Economic"
   LifeCycleStep="Value_Use" VOf_DC="
   DC1" VOf_RM="RM1" VOf_TC="TC1"
   VOf_VOb="VO1_VO2_VO3_VO4_VO5" />
5  <TargetCustomer TargetCustomerID="TC1"
   Name="Business_Units">
6  <consistsOf CustomerEquity="
   Acquisition" />
7  </TargetCustomer>
8  <RevenueModel RevenueModelID="RM1"
   Name="for_Business_Units"
   StreamType="Lending" PricingMethod
   ="Fixed_Price" />
9  <DistributionChannel
   DistributionChannelID="DC1" Name="
   Intranet" Reasoning="Effort_
   Reduction" CustomerBuyingCycle="
   Purchase" ValueLevel="Commodity"
   PriceLevel="Economic" />
10 <ValueObject ValueObjectID="VO1" Name="
   Hardware" Type="Tangible" />
11 <ValueObject ValueObjectID="VO2" Name="
   Low_Transaction_Costs" Type="
   Intangible" />
12 <ValueObject ValueObjectID="VO3" Name="
   Low_Labor_Costs" Type="Intangible
   " />
13 <ValueObject ValueObjectID="VO4" Name="
   Low_IT_Costs" Type="Intangible" />
14 <ValueObject ValueObjectID="VO5" Name="
   Recent_Hardware" Type="Intangible
   " />
15 </bm:BusinessServiceModel>

```

5 Related Work

(Baida et al. 2003) argue that eCommerce is still mainly characterized by the relatively straightforward trading of commodity goods. Current challenges are advanced business scenarios, such as collaborative design over the Internet of sophisticated goods and services. Their work elaborates on further challenges in order to achieve collaborative eCommerce concerned with real-world services. Similar to the Business Service Model, Baida et al. focus on service trade and propose a knowledge structure in form of a service ontology. The differences lie in that Baida et al.'s service ontology rather targets the Conceptual Service Model than the Business Service Model, and that they neither propose a modeling notation nor a procedure model.

(Weigand et al. 2009) introduce a unified view on

services by means of a service model and a modeling method targeted to the design and analysis of services. They argue that there exists a business view on services, such as in the approaches from Gordijn and Osterwalder. Additionally, they propose a 'software view' on services, namely the Service-Oriented Modeling & Architecture (SOMA). The authors find that a gap exists between these two views and that a service model closes this gap. Following that, Weigand et al. discuss business modeling with the REA and e^3 Value Ontology, and Spohrer's service systems theory. The service model comprises of a service ontology, a service classification, and a service layer architecture. Likewise to the Business Service Model, Weigand et al. propose a knowledge structure in form of an ontology and utilize the e^3 Value ontology as a notation. Their approach differs from the Business Service Model in that it aims at service identification and classifies in the Conceptual Service Model.

(Terlouw 2008) finds the UDDI specification too technology-driven for specifying services and hence believes that it contradicts SOA promises of increased flexibility of service reuse and business-IT alignment. She finds the business component framework and particularly the task specification more suitable for doing so. Terlouw claims that for business process execution, suitable services need to be identified as well as to specified. Service registries store these specifications for identification. In consequence, she proposes the Enterprise Ontology and the business component specification for business task specification. Terlouw offers a knowledge structure on the basis of the Enterprise Ontology. Her solution focuses on services as business tasks and relates to the Conceptual Service Model and offers neither a modeling notation nor a procedure model.

(Dumas et al. 2001) identify the need for a semantic service description framework because of the Internet's global and inexpensive connectivity. Such a description aims at advertising, locating, analyzing, and comparison of services. The authors' intention is to define requirements for future service descriptions. They propose the following service characteristics: provider, availability (time & spatial), channel, pricing, payment, security, quality of service, and reputation. The authors find the UDDI's TModel appropriate as an underlying model for these service characters. Likewise to the Business Service Model, Dumas et al. aim at service proposition. However, they mainly propose requirements for service propositions that relate to the Conceptual Service Model. They neither give information about a modeling notation nor a procedure model.

6 Conclusion and Future Work

With the evolution of service-oriented architectures toward service market places in the Internet, services and their description become even more important. Service descriptions are an elementary part of service trade for they contribute to service proposition, service discovery & selection, negotiation & contracting as well as monitoring.

From the perspective of service providers, the business-oriented development of such descriptions is a crucial part of the service development process. However, until recently no conceptual formalisms do exist for doing so (Kuropka et al. 2008). There are, however, many technical specifications out there, e.g. WSDL, SA-WSDL, and WSLA, using sometimes completely different notations but describing partly overlapping aspects. Even more important, it is necessary to involve *distinct* subject-matter-experts in the service development process. Using description

formalisms primarily tailored to technical aspects will hardly be successful under these circumstances. Last, but by no means least, it is essential to align business with information technology.

Hence, service providers would benefit from a method that allows for the crucial tasks of *documenting*, *communicating*, and *reasoning about* service descriptions on the different levels of abstraction and domains of expertise that are needed during the process. An understandable description of all levels would avoid mistakes and fasten the service description process materially.

Against this background, this paper outlines a service description method that combines the Open-EDI Reference Model with method engineering, which offers a work-break-down structure in order to reduce complexity and to align business and IT. Particularly, the Business Service Model, as one part of the method, is further detailed. The paper proposes a meta model for business-oriented service descriptions and develops a corresponding modeling notation on the basis of UML Profile, which supports all three steps, i.e., documenting, communication, and reasoning about descriptions on a strategic level. The result document is a valuable input for service descriptions on a conceptual level. Two possible fields of applications have been outlined: service-oriented modeling and service engineering. The proposed approach was tested in a case study in the IT outsourcing domain to show the applicability of the Business Service Model.

The case study shows that the proposed solution is appropriate for all three tasks on a business-oriented level. The usage of UML made it necessary to involve modeling experts. Nevertheless, the modeling notation turns out to be practical for communicating and reasoning about service descriptions.

Future work includes integrating the meta model and the modeling notation with related approaches in the service description domain. Furthermore, the Conceptual Service Model needs to be detailed and integrated with the Business Service Model. This will be addressed in the next steps of the Theseus/TEXO research project (Janiesch et al. 2008).

Acknowledgements

This project was funded by means of the German Federal Ministry of Economy and Technology under the promotional reference "01MQ07012". The responsibility for the content of this publication lies with the authors.

References

- Alves, A., Arkin, A., Askary, S., Barreto, C., Bloch, B., Curbera, F., Ford, M., Golland, Y., Guzar, A., Kartha, N. & Liu, C. K. (2007), 'Specification: Business Process Execution Language for Web Services version 2.0'.
- Arsanjani, A., Ghosh, S., Allam, A., Abdollah, T., Gariapathy, S. & Holley, K. (2008), 'Soma: a method for developing service-oriented solutions', *IBM Syst. J.* 47(3), 377–396.
- Baida, Z., Akkermans, H. & Gordijn, J. (2003), 'Serviguration: towards online configurability of real-world services', in N. M. Sadeh, M. J. Dively, R. J. Kauffman, Y. Labrou, O. Shehory, R. Telang & L. F. Cranor, eds, 'ICEC', Vol. 50 of *ACM International Conference Proceeding Series*, ACM, pp. 111–118.
- Barros, A. P. & Dumas, M. (2006), 'The Rise of Web Service Ecosystems', *IT Professional* 8(5), 31–37.

- Chinnici, R., Moreau, J.-J., Ryman, A. & Weerawarana, S. (2007), 'Specification: Web Services Description Language (WSDL) Version 2.0 Part 1: Core Language', W3C Recommendation.
- Dorn, J., Grun, C., Werthner, H. & Zapletal, M. (2007), A Survey of B2B Methodologies and Technologies: From Business Models towards Deployment Artifacts, *in* 'HICSS', IEEE Computer Society, p. 143.
- Dumas, M., O'Sullivan, J., Heravizadeh, M., Edmond, D. & ter Hofstede, A. H. M. (2001), Towards A Semantic Framework for Service Description, *in* 'DS-9', pp. 277–291.
- Eclipse Model Development Tools (MDT)* (n.d.), <http://www.eclipse.org/modeling/mdt/>.
- Farrell, J. & Lausen, H. (2007), 'Specification: Semantic Annotations for WSDL and XML Schema (SA-WSDL)', <http://www.w3.org/TR/sawSDL/>.
- Giachetti, G., Marín, B. & Pastor, O. (2009), Using UML as a Domain-Specific Modeling Language: A Proposal for Automatic Generation of UML Profiles, *in* (van Eck et al. 2009), pp. 110–124.
- Gordijn, J. (2002), E^3 -value in a Nutshell, Technical report, HEC University Lausanne, Lausanne.
- Gutzwiller, T. (1994), *Das CC RIM-Referenzmodell für den Entwurf von betrieblichen, transaktionsorientierten Informationssystemen*, Physica-Verlag, Heidelberg.
- International Organization for Standardization (ISO) (2004), 'Open-edi reference model, iso standard 14662, second edition'.
- Janiesch, C., Ruggaber, R. & Sure, Y. (2008), 'Eine Infrastruktur für das Internet der Dienste', HMD - Praxis der Wirtschaftsinformatik (45:261), 2008, pp. 71-79.
- Keller, A. & Ludwig, H. (2003), 'The WSLA framework: Specifying and monitoring service level agreements for web services', *J. Network Syst. Manage* 11(1).
- Kett, H., Voigt, K., Scheithauer, G. & Cardoso, J. (2008), Service Engineering in Business Ecosystems, *in* 'Proceedings of the XVIII. International RESER Conference', Stuttgart, Germany.
- Kuropka, D., Troeger, P., Staab, S. & Weske, M., eds (2008), *Semantic Service Provisioning*, Springer Berlin Heidelberg. ISBN 978-3-540-78616-0.
- Martin, D. L., Paolucci, M., McIlraith, S. A., Burstein, M. H., McDermott, D. V., McGuinness, D. L., Parsia, B., Payne, T. R., Sabou, M., Solanki, M., Srinivasan, N. & Sycara, K. P. (2004), Bringing Semantics to Web Services: The OWL-S Approach, *in* J. Cardoso & A. P. Sheth, eds, 'SWSWPC', Vol. 3387 of *Lecture Notes in Computer Science*, Springer, pp. 26–42.
- Object Management Group (OMG) (2007), 'Specification: Unified Modeling Language (UML) version 2.0', <http://www.omg.org/docs/formal/05-07-04.pdf>.
- Organization for the Advancement of Structured Information Standards (OASIS) (2004), 'Specification: Universal Description Discovery and Integration (UDDI) Version 3.0'
- Osterwalder, A. (2004), The Business Model Ontology: A Proposition in a Design Science Approach, PhD thesis, Universite de Lausanne Ecole des Hautes Etudes Commerciales.
- Ouyang, C., van der Aalst, Dumas, M., ter Hofstede & M., A. H. (2006), 'From business process models to process-oriented software systems: The BPMN to BPEL way', ePrint: <http://eprints.qut.edu.au/archive/00005266/>.
- Papazoglou, M. P. (2003), Service-Oriented Computing: Concepts, Characteristics and Directions, *in* 'WISE', IEEE Computer Society, pp. 3–12.
- Peneder, M., Kaniovski, S. & Dachs, B. (2003), 'What Follows Tertiarisation? Structural Change and the Role of Knowledge-based Services', *The Service Industries Journal* 23 Issue 2(146), 47–66.
- Roman, D., de Bruijn, J., Mocan, A., Lausen, H., Domingue, J., Bussler, C. & Fensel, D. (2006), WWW: WSMO, WSML, and WSMX in a Nutshell, *in* 'ASWC', pp. 516–522.
- Roman, D., Keller, U., Lausen, H., de Bruijn, J., Lara, R., Stollberg, M., Polleres, A., Feier, C., Bussler, C. & Fensel, D. (2005), 'Web Service Modeling Ontology', *Applied Ontology* 1(1), 77–106.
- Scheithauer, G. (2009), Business Service Description Methodology for Service Ecosystems, *in* 'Proceedings of the 16th CAiSE-DC'09 Amsterdam, The Netherlands, June 9-10, 2009', Vol. 479 of *ceur-ws*.
- Scheithauer, G., Augustin, S. & Wirtz, G. (2008), Describing Services for Service Ecosystems, *in* G. Feuerlicht & W. Lamersdorf, eds, 'ICSOC Workshops', Vol. 5472 of *Lecture Notes in Computer Science*, Springer, Sidney, Australia, pp. 242–255.
- Scheithauer, G., Augustin, S. & Wirtz, G. (2009a), Business Modeling for Service Engineering: Toward an Integrated Procedure Model, *in* 'Proceedings of the 21st International Conference on Software Engineering & Knowledge Engineering (SEKE'2009), Boston, Massachusetts, USA, July 1-3, 2009', Boston, MA, USA, pp. 322–327.
- Scheithauer, G., Augustin, S. & Wirtz, G. (2009b), Service Value Properties for Service Ecosystems: A Reference Model and a Modeling Guideline, *in* J. Barjis, J. Kinghorn, S. Ramaswamy, E. Dubois & P. Johannesson, eds, 'Proceedings of the EOMAS Workshop', Vol. Vol-458 of *CEUR-WS*, Amsterdam, Netherlands.
- Terlouw, L. (2008), Towards a Business-Oriented Specification for Services, *in* 'Advances in Enterprise Engineering I, 4th International Workshop CIAO! and 4th International Workshop EOMAS', Vol. 10 of *Lecture Notes in Business Information Processing*, Springer, Montpellier, France, pp. 122–136.
- van Eck, P., Gordijn, J. & Wieringa, R., eds (2009), *Advanced Information Systems Engineering, 21st International Conference, CAiSE 2009, Amsterdam, The Netherlands, June 8-12, 2009. Proceedings*, Vol. 5565 of *Lecture Notes in Computer Science*, Springer.
- Weigand, H., Johannesson, P., Andersson, B. & Bergholtz, M. (2009), Value-based service modeling and design: Toward a unified view of services, *in* (van Eck et al. 2009), pp. 410–424.

Metric of Intrinsic Information Content for Measuring Semantic Similarity in an Ontology

Md. Hanif Seddiqui

Masaki Aono

Department of Electronics and Computer Engineering,
Toyohashi University of Technology,
1-1 Hibarigaoka, Tempaku, Toyohashi, Japan,
Email: hanif@kde.ics.tut.ac.jp, aono@ics.tut.ac.jp

Abstract

Measuring information content (IC) from the intrinsic information of an ontology is an important however a formidable task. IC is useful for further measurement of the semantic similarity. Although the state-of-art metrics measure IC, they deal with external knowledge base or intrinsic hyponymy relations only. A current complex form of ontology conceptualizes a class (also often called as a concept) explicitly with the help of the hyponymy classes and the asserted relations and restrictions. Therefore, we propose a modified metric for measuring IC intrinsically taking both the concept-to-concept and the concept-to-property relations. We evaluate our system theoretically and with experimental data. Our evaluation shows the effectiveness of our modified metric for extracting intrinsic information content to measure semantic similarity among concepts in an ontology.

Keywords: Concept, Ontology, Information Content, Semantic Similarity

1 Introduction

“An ontology is an explicit specification of a conceptualization” is a prominent definition by T.R. Gruber in 1995 (Gruber 1995). The definition was then extended by R. Studer et al., in 1998 as “an ontology is an explicit, formal specification of a shared conceptualization of a domain of interest” (Studer et al. 1998). Ontology is the backbone to fulfill the semantic web vision (Berners-Lee et al. 1999, Maedche & Staab 2001) and is a knowledge base to enable machines to communicate each other effectively. The knowledge captured in ontologies can be used to annotate data, to distinguish homonyms and polysemies, to drive intelligent user interfaces and even to retrieve new information.

An ontology contains core ontology, axioms or asserted rules, knowledge base and lexicon. Furthermore, core ontology is defined by a set of concepts, a set of properties, concept hierarchy, property hierarchy and functions to relate properties with concepts.

There are usually various size of ontologies, small-scale or large-scale. Large-scale ontologies often represent

distributed knowledge area within a problem domain. Ontology segmentation or alignment of large-scale ontologies often requires method of ontology partitioning. In this regard, concept to concept semantic relatedness or similarity measure is necessary. The partition is often performed by the semantic relatedness or similarity measure among concepts of ontology.

The state-of-art metrics by Resnik (Resnik 1999), Lin (Lin 1998), Jiang and Conrath (Jiang & Conrath 1997), and Seco et al. (Seco et al. 2004) used extrinsic or intrinsic information content for semantic similarity measure. Resnik, Lin and Jiang and Conrath used the external source of information content. Although Seco et al. measured the information content within ontology, they used hyponyms of concepts only. They applied their metric to the trivial taxonomy of concepts like WordNet (Miller et al. 1990). However, ontologies, such as those developed by the Web Ontology Language (OWL) (McGuinness et al. 2004), are significantly more complex in data structures than the taxonomy of concepts only.

Our proposed metric of information content extends to take concept, properties and their relations of ontology into account. Therefore, it can be applied in both cases of a simple taxonomy and a complex ontology with concept-properties relations.

The scope of the this work has a well accepted field of ontology partitioning to achieve the scalability. As ontologies grow in size they become more and more difficult to create, use, understand, maintain, transform and classify. Therefore, Stuckenschmidt et. al. (Stuckenschmidt & Klein 2004) and Grau et. al. (Grau et al. 2005a,b, 2006) focus on partitioning OWL ontologies. Seidenberg and Rector (Seidenberg & Rector 2006) suggested segmentation of gigantic large ontologies to solve the scaling problems. Hu et al. (Hu, Cheng, Zheng, Zhong & Qu 2006, Hu, Zhao & Qu 2006, Hu et al. 2008) proposed partition based block matching for aligning large ontologies. Therefore, ontology partitioning with the help of semantic similarity measurement is necessary in segmentation, aligning large ontologies or obtaining scalability in large ontologies.

This work is to integrate with our scalable and efficient algorithm of ontology alignment called Anchor-Flood algorithm (Seddiqui & Aono 2008), which performs the best running time in the OAEI-2008 campaign.

The rest of the paper is organized as follows. **Section 2** introduces the state-of-art techniques of semantic similarity metrics, while **Section 3** focuses on the ontology structure of semantic web. **Section 4** describes the limitation of the state-of-art metrics. **Section 5** includes the detailed elaboration of our proposed metric. **Section 6** includes experiments and evaluation to show the effectiveness of our proposed metric. Concluded remarks and some future

This study was supported by Global COE Program “Frontiers of Intelligent Sensing” from Japan’s Ministry of Education, Culture, Sports, Science and Technology (MEXT).

Copyright ©2010, Australian Computer Society, Inc. This paper appeared at the Seventh Asia-Pacific Conference on Conceptual Modeling (APCCM 2010), Brisbane, Australia, January 2010. Conferences in Research and Practice in Information Technology (CRPIT), Vol. 110, Sebastian Link and Aditya K. Ghose, Ed. Reproduction for academic, not-for profit purposes permitted provided this text is included.

directions of our work is described in **Section 7**.

2 State-of-art Metrics

Semantic similarity metric is an important measure for partitioning a taxonomy of an ontology. A taxonomy is a hierarchical representation of a semantic network with a partial ordering, typically given by the concept inclusion relation (ISA). The concepts in a semantic network or taxonomy of an ontology have proximities among other concepts connected by edges.

There are a number of researches to measure the similarity among the concepts in a semantic network. Of the many approaches presented in the literature (Collins & Quillian 1969, Rada et al. 1989, Knappe et al. 2007, Hirst & St-Onge 1998, Sussna 1993, Wu & Palmer 1994), we divide them into two classes: edge-relative approaches and information theoretic approaches. Edge-related approaches focus on counting the edges or pre-assigned weight of edges, while information theoretic approaches analyze the essence of taxonomy of ontology. Each of them are described below for understanding the content of this paper easily.

2.1 Edge relative Approaches

Rada et al. (Rada et al. 1989) assumes that the similarity is proportional to the number of edges separating concepts. Sussna et al. (Sussna 1993) introduces a depth-relative scaling approach, based on the observation that siblings deep in the tree are more closely than siblings higher in the tree. Wu and Palmer (Wu & Palmer 1994) define their conceptual similarity based on the principle of depth-relative scaling as:

$$sim(c_i, c_j) = \frac{2 * depth(c_{ij})}{depth(c_i) + depth(c_j)}, \quad (1)$$

where c_{ij} is the common super class of c_i and c_j , and $depth(c_k)$ gets the depth of c_k in the original class hierarchy.

In Eq. 1 each edge has the weight of unity regardless of their direction in the subsumption relation of a taxonomy. On the contrary, some proposed distance metrics use different weights of a particular edge for differentiating their direction in subsumption. A taxonomy of an ontology contains a network of a directed graph. This means that the edge between two concepts represents a relationship in the direction of the edge. In Fig. 1, a *dog* ISA *animal* and not the other way around. When we move in the direction of edge, we get generalization, whereas we obtain specialization while moving in an opposite direction of the orientation. Therefore, concept inclusion (ISA) intuitively implies strong similarity in the opposite direction from inclusion (specialization). In addition, the direction of inclusion (generalization) must contribute some degree of affinity. However, for the same reasons as in the case of specializations, transitive generalizations should contribute a decreased degree of similarity.

To make the edge influence the similarity, weight factor δ and γ is introduced for expressing similarity of immediate specialization and generalization respectively. The similarity function is then defined as:

$$sim_{wsp}(x, y) = \max_{j=1 \dots m} \left\{ \sigma^{s(P_j)} \gamma^{g(P_j)} \right\}, \quad (2)$$

where $P_1 \dots P_m$ are all paths connecting x and y . P_k is an edge, $s(P_j)$ is the number of edges toward specialization and $g(P_j)$ is the number of edges toward generalization.

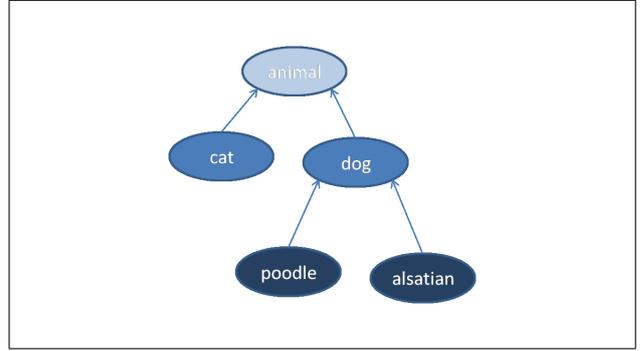


Figure 1: An example of a pet ontology

This similarity can be derived from an ontology by transforming the ontology into a directional weighted graph with σ as a downward and γ as an upward weights, and the similarity is the product of the weight on the path 2. An example ontology is displayed in Fig. 2.

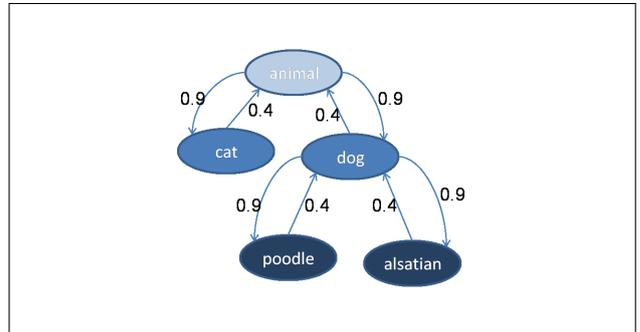


Figure 2: The ontology transformed into directed weighted graph, with the immediate specialization and generalization similarity values $\sigma = 0.9$ and $\gamma = 0.4$ respectively. Similarity is derived as the maximal (multiplicative) weighted path length, and thus $sim(poodle, alsatian) = 0.4 * 0.9 = 0.36$.

The edge between concepts in a taxonomy represents some degree of affinity. However the edge weights are defined by different factors. The deeper concepts in a taxonomy has more specification than the shallower concepts in the taxonomy. A taxonomy is organized by a directed network. The direction in a network has an influence over the affinity. Defining the weight of an edge to measure the affinity of concept pair depends on the factors. Therefore, several researchers focus on the information theoretic approaches to define the weights to measure semantic affinities between concepts by introducing IC. Measuring IC removes the complexity of assigning dynamic weights to every edges and of considering direction of edges as well. Information theoretic approaches are discussed in more details below.

2.2 Information Theoretic Approaches

Information theoretic approaches are well defined in a couple of research works by (Jiang & Conrath 1997, Lin 1998, Resnik 1995, Seco et al. 2004). They obtain their needed IC values by statistically analyzing corpora. They associate probabilities to each concept in the taxonomy based on word occurrences in a given corpus. These probabilities are cumulative as we go up the taxonomy from specific concepts to more abstract concepts. The IC value is then obtained by considering the negative log likelihood (Resnik 1995, 1999):

$$ic_{res}(c) = -\log p(c) \quad (3)$$

where c is any concept in WordNet and $p(c)$ is the probability of encountering c in a given corpus. It should be noted that this method ensures that IC is monotonically decreasing as we move from the leaves of the taxonomy to its roots. (Resnik 1995) was the first to consider the use of this formula, that stems from the work of Shannon (Shannon & Weaver 1948), for the purpose of semantic similarity judgments. The basic intuition behind the use of the negative likelihood is that the more probable a concept is of appearing then the less information it conveys, in other words, infrequent words are more informative than frequent ones. Resnik describes an implementation in using WordNet's (Miller et al. 1990) taxonomy of noun concepts. According to Resnik, semantic similarity depends on the amount of information two concepts have in common, this shared information is given by the most specific common abstraction i.e. the super-concept that subsumes both concepts. In order to find a quantitative value of shared information we must first discover the super-concept, if one does not exist then the two concepts are maximally dissimilar, otherwise the shared information is equal to the IC value of the super-concept. Formally, semantic similarity is defined as:

$$sim_{res}(c_1, c_2) = \max_{c \in S(c_1, c_2)} \{ic_{res}(c)\}, \quad (4)$$

where $S(c_1, c_2)$ are the set of concepts that subsume c_1 and c_2 .

Lin (Lin 1998) proposes a modification of measuring the semantic similarity using Resnik's equation defined in Eq. 3 and 4. He states that the similarity between concepts c_1 and c_2 is measured by the ratio between the amount of information needed to state the commonality of c_1 and c_2 and the information needed to fully describe what c_1 and c_2 are. The definition might be expressed as:

$$sim_{lin}(c_1, c_2) = \frac{2 * sim_{res}(c_1, c_2)}{ic_{res}(c_1) + ic_{res}(c_2)} \quad (5)$$

Jiang and Conrath (Jiang & Conrath 1997) proposes a combined model taking the shortest path, edge-counting methods, Resnik's information content into account and by adding decision factors. They calculate the weights between two concepts of parent and child relation.

$$wt(c_c, c_p) = \left(\beta + (1 - \beta) \frac{\bar{E}}{E(c_p)} \right) \left(\frac{d(c_p) + 1}{d(c_p)} \right)^\alpha \quad (6)$$

$$[ic_{res}(c_c) - ic_{res}(c_p)] T(c_c, c_p)$$

where $d(c_p)$ is the depth of the node (concept) c_p , $E(c_p)$ is the number of children of c_p , the local density (\bar{E}) is the average density in the entire taxonomy, and $T(c_c, c_p)$ is the link relation/type factor. The parameters α ($\alpha \geq 0$) and β ($0 \leq \beta \leq 1$) control the influence of node depth and density, respectively.

If we consider the case where node depth (as we will consider node depth indirectly by considering the number of children a particular node contains) is ignored and link type and local density both have a weight of 1. In this special case, the dissimilarity metric is:

$$dist_{jcn}(c_1, c_2) = (ic_{res}(c_1) + ic_{res}(c_2)) - 2 * sim_{res}(c_1, c_2) \quad (7)$$

2.3 Intrinsic Information Content Metric

The classical way of measuring IC of concepts combines knowledge of their hierarchical structure from an ontology with the statistics on their actual usage in text as derived from a large corpus. However, Seco et al. (Seco et al. 2004) derived a wholly intrinsic measure of IC that relies on hierarchical structure alone and applied their derivation to large taxonomy of WordNet. They report a competitive correlation value between human and machine similarity judgment on the dataset of Miller and Charles (Miller & Charles 1991) against WordNet.

Seco et al. argue that the more hyponyms a concept has the less information it expresses, otherwise there would be no need to further differentiate it. Likewise, concepts, that are leaf nodes, are the most specified in the taxonomy so the information they express is maximal. Formally they define:

$$ic_{seco}(c) = 1 - \frac{\log(hypo(c) + 1)}{\log(max_{wn})} \quad (8)$$

where the function $hypo$ returns the number of hyponyms of a given concept and max_{wn} is a constant that is set to the maximum number of concepts that exists in the taxonomy.

Seco's IC decreases monotonically as we transverse from leaf to root. The information content of the imaginary top node of WordNet would yield an IC value of 0. This metric gives the same score to all leaf nodes in the taxonomy regardless of their overall depth. All leaves have same maximum value 1.

Like Resnik's and Lin's measure, Seco's metric of semantic similarity yields result in $[0, \dots, 1]$. Seco et al. formulated their metric as:

$$sim_{seco}(c_1, c_2) = 1 - \frac{ic_{seco}(c_1) + ic_{seco}(c_2) - 2 * sim'_{res}(c_1, c_2)}{2}, \quad (9)$$

where sim'_{res} corresponds to Resnik's similarity function but accommodating Seco's IC values.

3 Ontology in Semantic Web

Apart from the general discussion about measuring IC against WordNet which contains a complete list of concepts, the definition of a domain ontology would reveal the fact about the current complex form of ontology of semantic technology. According to (Ehrig 2007), ontology contains core ontology, logical mapping, knowledge base, and lexicon. Furthermore, a core ontology, S , is defined by a tuple of five entities as follows:

$$S = (C, \leq_C, R, \sigma, \leq_R),$$

where C and R are two disjoint sets called "concepts" and "relations" respectively. A relation is known as a property of a concept, or a restriction on a property about a concept. Throughout the paper, we use the term "relation" to represent property of a concept, or a restriction on a property about a concept. A function represented by $\sigma(r) = \langle dom(r), ran(r) \rangle$ where $r \in R$, domain is $dom(r)$ and range is $ran(r)$. A partial order \leq_R represents on R , called relation hierarchy, where $r_1 \leq_R r_2$ iff $dom(r_1) \leq_C dom(r_2)$ and $ran(r_1) \leq_C ran(r_2)$. The notation \leq_C represents a partial order on C , called concept hierarchy or "taxonomy". In a taxonomy, if $c_1 <_C c_2$ for $c_1, c_2 \in C$,

then c_1 is a sub-concept of c_2 , and c_2 is a super-concept of c_1 . If $c_1 <_C c_2$ and there is no $c_3 \in C$ with $c_1 <_C c_3 <_C c_2$, then c_1 is a direct sub-concept of c_2 , and c_2 is a direct super-concept of c_1 denoted by $c_1 \prec c_2$ (Ehrig 2007).

Ontology relationship among its concepts is defined by their taxonomy, where concepts are maintained in a hierarchical or super or sub-concept organization along with the properties and restrictions. Therefore properties along with the restrictions are sometimes referred to as relations. They usually play important roles to define the relationships among concepts. Relations of an ontology are differentiating our metric from the other.

4 Limitation of the State-of-art Metrics

The metrics we describe so far are used to measure the semantic similarity among concepts where concepts are organized in a hierarchy or a taxonomy. They are only considering the concept to concept relation, i.e. the metrics consider only super-concept and sub-concept organization. However description logic (DL) based domain ontology of semantic technology usually contains properties, restrictions and other complex relations in addition to the trivial taxonomy or concept hierarchy.

WordNet does not heavily depend on the properties, rather it has a complete list of concepts to define another concept. As it is a thesaurus, we can have a large text corpora having connection with WordNet. Unlike WordNet, description logic based domain ontology only focuses on a particular domain of interest. It is seldom complete by its concepts alone as it may contain a limited number of concepts of one's interest. Different ontologies of a particular domain might widely be different and influenced by its targeted users and the knowledge of the its developers. Moreover, it has seldom large text corpora to define its concepts. On the contrary, DL ontology has an explicit specification to define a concept not only by the concept alone, but also with the help of the other concepts, its properties and restrictions and the other logical assertions available inside the ontology.

The classical metrics of measuring semantic similarity often use the available concepts, a large text corpora or a large complete hierarchy for using the hyponymy relations to measure the IC of a concept. These metrics cannot be used against domain ontologies at their current states. However, semantic relatedness measure plays an important role in ontology for partitioning or segmenting large ontologies for resolving scalability issues.

5 Proposed Modification in IC Metric

To overcome the limitation of the state-of-art metrics of computing semantic similarity among concepts within a domain ontology and to cope with the new ontologies with the introduced complex description logics, we propose a modified metric of computing intrinsic information content. The metric can be applied to a simple taxonomy and to a recent complex OWL ontology as well.

The primary source of IC in ontology is obviously concepts and concept hierarchy. However, OWL ontology also contains properties, restrictions and other logical assertions, often called as relations. Properties are used to define functionality of a concept explicitly to specify a meaning. They are related to concept by means of domain, range and restrictions.

According to Resnik, semantic similarity depends on the shared information. As Resnik introduces

the IC which represents the expressiveness of a particular concept. Classical metric of IC are based on the available concepts in a taxonomy or in a large text corpora. However, as time passes on, the definition and the content of ontology becomes more and more complex. The expressiveness of a concept is not only rely on the concept taxonomy but also on the other relations like properties and property-restrictions. Fig. 3 shows an example ontology of concepts *Biblio*, *Institution*, *Reference*, *InProceedings*, *Article*, *School* and *Publisher* with the support of 37 different properties which is playing an important role to define concepts and distinguish a concept from the others. Let us consider the concepts *Biblio* and *Institution*, where they are sharing no property and although *Institution* has only three properties to specify its meaning and is distinguish from the root. However, it does not contain other properties to specify its children more concisely. They have the only difference in their hyponym or subsumption relation. On the other hand, concepts *Biblio* and *Reference* has no common property as well. However *Reference* is expressed with 21 properties. Concept *Reference* is defined concisely and specifically with 21 properties. The fact is that *Reference* and *Institution* are subsumed by a concept *Biblio* though, *Reference* has more expressiveness than *Institution*. Therefore, information content of *Reference* is larger than that of concept *Institution*. In the figure, *Reference*, *Article* and *InProceedings* has close semantic relatedness. Likewise, *Institution*, *Publisher* and *School* has semantic relatedness. However, *Institution* and *Reference* has weak semantic relatedness because of their less sharing information. Therefore, we can consider that the information content or expressiveness of a concept is directly proportional to the number of properties it is related to by means of property functions or property restrictions.

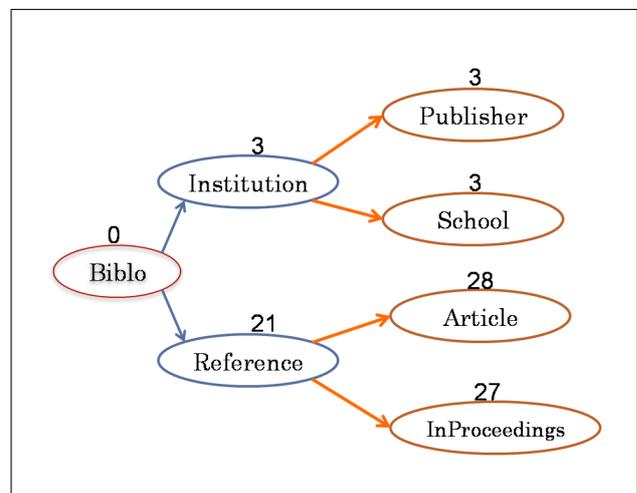


Figure 3: Illustration of the impact of properties on semantic similarity.

5.1 Proposed IC Metric

We already have discussed about the probable sources of IC or the expressiveness of semantic similarity among the concepts of ontology. We find that the IC of a concept is negatively related to the probability of a concept in an external large text corpora (Resnik 1995). We also find that the IC of a concept is inversely related to the number of hyponyms or the concepts it subsumes (Seco et al. 2004). Moreover, we observe that description logic (DL) based ontol-

ogy of semantic technology is formal and explicit in its conceptualization with the help of relations. Every concept is defined with sufficient semantic embedding with the organization, property functions, property restrictions and other logical assertions. Current ontology of semantic technology is defined as “an explicit specification of a conceptualization” (Gruber 1995). Although the most domain ontologies are not as complete as WordNet in terms of concepts and concept organization, they have well support from logical assertions to define a concept concisely. Therefore, we can obtain sufficient IC of a concept without depending on the external large text corpora heavily, required that we use intrinsic information of the concept. One of the good source of intrinsic information of a concept is its relations by means of property functions and property restrictions. Our relation based IC is defined as:

$$ic_{rel}(c) = \frac{\log(rel(c) + 1)}{\log(total_rel + 1)}, \quad (10)$$

where “*rel*” stands for the relation of properties, property function and restrictions, $rel(c)$ denotes the number of relations of a concept c and $total_rel$ represents the total number of relations available in the ontology.

As long as the information content of a concept depends both on the hyponyms or subsumption relations of a concept and the related properties of the concept, we need to integrate the $ic_{re}(c)$ with the Seco’s metric of intrinsic information content defined in Eq. 8. This integration introduces a coefficient factor ρ and the equation becomes as:

$$ic(c) = \rho \cdot ic_{rel}(c) + (1 - \rho) \cdot ic_{seco}(c), \quad (11)$$

where the coefficient factor ρ is defined by the nature of an ontology. While a small size of ontology is often incomplete by its concepts alone, the coefficient factor tends to increase to focus on relations. On the contrary, when relations are inadequate to define a concept and there are a large number of concepts in the taxonomy, ρ tends to decrease its value. However, we definitely need a trade-off to select the coefficient factor and we define it as:

$$\rho = \frac{\log(total_rel + 1)}{\log(total_rel) + \log(total_concept)},$$

where $total_rel$ is the maximum number of relations while $total_concepts$ is the maximum number of concepts available in an ontology.

Moreover, the semantic similarity is computed as described in Eq. 9 by replacing ic_{seco} with ic defined in Eq. 11 and the sim'_{res} is measured by our modified information content metric. Then the semantic similarity $sim_{proposed}$ is defined as below:

$$sim_{proposed}(c_1, c_2) = 1 - \frac{ic(c_1) + ic(c_2) - 2 * sim'_{res}(c_1, c_2)}{2}, \quad (12)$$

6 Experiments and Evaluation

For experiment with our modified metric of IC, we obtained a reference ontology of a benchmarks from the Ontology Alignment Evaluation Initiative (OAEI),

Table 1: contains IC values measured by Seco’s metric and our modified metric.

Concepts	Number of Relations	Number of Hyponyms	ic_{seco}	ic_{rel}	$ic_{modified}$
Date	3	0	1.000	0.332	0.641
PageRange	2	0	1.000	0.263	0.603
Organization	0	3	0.613	0.000	0.283
Institution	3	2	0.693	0.332	0.499
Publisher	3	0	1.000	0.332	0.641
School	3	0	1.000	0.332	0.641
List	0	1	0.807	0.000	0.373
PersonList	4	0	1.000	0.386	0.670
Journal	7	0	1.000	0.498	0.730
Address	3	0	1.000	0.332	0.641
Person	0	0	1.000	0.000	0.462
Conference	6	0	1.000	0.466	0.713
Reference	21	23	0.113	0.740	0.450
Academic	24	2	0.693	0.771	0.735
PhdThesis	26	0	1.000	0.790	0.887
MastersThesis	26	0	1.000	0.790	0.887
Misc	22	0	1.000	0.751	0.866
MotionPicture	22	0	1.000	0.751	0.866
Part	23	5	0.500	0.761	0.640
InCollection	26	0	1.000	0.790	0.887
InProceedings	27	0	1.000	0.798	0.891
Article	28	0	1.000	0.807	0.896
Chapter	26	0	1.000	0.790	0.887
InBook	27	0	1.000	0.798	0.891
Report	25	2	0.693	0.781	0.740
TechReport	25	0	1.000	0.781	0.882
Deliverable	25	0	1.000	0.781	0.882
Informal	21	4	0.551	0.740	0.653
Manual	23	0	1.000	0.761	0.871
Unpublished	23	0	1.000	0.761	0.871
Booklet	23	0	1.000	0.761	0.871
LectureNotes	22	0	1.000	0.751	0.866
Book	27	3	0.613	0.798	0.713
Collection	31	0	1.000	0.830	0.909
Monograph	30	0	1.000	0.823	0.905
Proceedings	34	0	1.000	0.852	0.920

2009¹ displayed in Fig. 4. We can download the ontology from its homepage².

As the Fig. 4 displays the concepts associated with its number of relations, we obtained the results displayed in Table 1 by using Eq. 8, 10 and 11. From the Table 1 and observing the Fig. 4, we see that hyponyms of “Reference” concepts has a close value of information content. Therefore, they are semantically close related and the other concepts are not as close as a group. However, the values of ic_{seco} in the Table 1 do not reveal a group of semantically related concepts because of their scatteredness. Moreover, Seco’s IC values of all the leaves are unity regardless of their depth and position in ontology, which is certainly misleading to a group of semantically related concepts in an ontology.

From the experiments, we also observe that the deeper concepts have more expressiveness or larger IC values. Therefore, it guarantees that our modified IC metric takes the depth of a concept implicitly and the children of a concept explicitly. However, we do not take the link type and local concept density into account unlike expressed in (Jiang & Conrath 1997). As we consider the hyponyms by incorporating the Seco’s IC metric, it consider the edges between subsumption concepts implicitly.

Using Table 1 we produce the semantic similarity between *Reference* to each of its leaves considering similarity equation proposed by Seco et al. in Eq. 9 and proposed by us in Eq. 12. The semantic similarity is displayed in Table 2. We observe that our semantic similarity is close to the real proximity. *Article*, *Chapter* and so on has close semantic relationship with *Reference* in the domain of bibliography.

Furthermore, we also compute semantic similarity for every possible pair of concepts of the ontology depicted in Fig. 4 and derive only the semantically related groups or blocks of concepts using both equation. Our proposed method produces *Reference* with its 23 children as a unique block with another small block of containing *Institution*, *Publisher* and *School*. Therefore, our proposed metric produce two blocks containing 24 and 3 elements respectively. On

¹<http://oaei.ontologymatching.org/2009/>

²<http://oaei.ontologymatching.org/2009/benchmarks/101/onto.rdf>



Figure 4: It shows a taxonomy of an ontology where each concept is associated with its number of relations.

the other hand, according to Eq. 9, we get several number of small blocks even among the children of *Reference* concept. It produces six small blocks having 4, 5, 3, 6, 3, and 4 elements. We also produce two real blocks manually by the domain experts with 24 and 4 elements. We evaluate the blocks by well-known Purity, Inverse-purity and F-measure metrics which rely on precision, recall and f-measure ideas of information retrieval.

Let B be a set of computed blocks ($|B| = n$) and R be a set of manual blocks produced by experts ($|R| = m$). b_i denotes a block in B , while r_j denotes a block in R . $|b_i|$ returns the number of entities in b_i , and $|r_j|$ is defined analogously. $b_i \cap r_j$ calculates the common entities in both b_i and r_j . N be the total number of blocked items. Then, Purity is computed by taking the weighted average of maximal precision values:

$$Purity(B) = \sum_{i=1}^n \frac{|b_i|}{N} \max_j Precision(b_i, r_j), \quad (13)$$

where the Precision of a block b_i against a real block r_j is defined as:

$$prec(b_i, r_j) = \frac{|b_i \cap r_j|}{|b_i|}$$

Inverse-purity focuses on the block with maximum recall against a real block and is defined as:

$$Inverse-purity(B) = \sum_{j=1}^m \frac{|r_j|}{N} \max_i Precision(r_j, b_i), \quad (14)$$

where, $Precision(r_j, b_i)$ is also called as $Recall(b_i, r_j)$.

However, a more robust evaluation metric can be obtained by combining the ideas of Purity and Inverse-purity, called as F-measure and defined as:

$$F-measure(B) = \sum_{j=1}^m \frac{|r_j|}{N} \max_i \{F(r_j, b_i)\}, \quad (15)$$

where

$$F(r_j, b_i) = \frac{2 \times Recall(r_j, b_i) \times Precision(r_j, b_i)}{Recall(r_j, b_i) + Precision(r_j, b_i)}$$

Now, we apply the Precision, Recall and F value against the blocks derived by both of the metrics and the results are displayed in Table 3 and Table 4 to represent the model of Seco et al. and our proposed model respectively.

Table 2: contains semantic similarity between *Reference* to each of its leaves considering Seco’s metric and our proposed metric.

e_1	e_2	sim_{seco}	$sim_{proposed}$
Reference	PhDThesis	0.113	0.782
Reference	MastersThesis	0.113	0.782
Reference	InCollection	0.113	0.782
Reference	InProceedings	0.113	0.780
Reference	Article	0.113	0.777
Reference	Chapter	0.113	0.782
Reference	InBook	0.113	0.780
Reference	TechReport	0.113	0.784
Reference	Deliverable	0.113	0.784
Reference	Manual	0.113	0.790
Reference	Unpublished	0.113	0.790
Reference	Booklet	0.113	0.790
Reference	LectureNotes	0.113	0.792
Reference	Collection	0.113	0.771
Reference	Monograph	0.113	0.773
Reference	Proceedings	0.113	0.765

Table 3: Precision, Recall and F value for the suggested blocks by Eq. 9

$Block$	$Ref.Block$	Precision	Recall	F
4	24	1.000	0.167	0.286
5	24	1.000	0.208	0.344
3	24	1.000	0.125	0.222
6	24	1.000	0.250	0.400
3	24	1.000	0.125	0.222
4	4	1.000	1.000	1.000

We summarize Purity, Inverse-purity and F-measure for both of the metrics in Table 5. The Table 5 depicts that our proposed metric outperforms in the given example. Finally, we can state that our proposed modified metric for intrinsic information content works efficiently provided that the ontology concepts are specified by their properties. Once the ontology does not contain any properties, it behaves just as a Seco’s model.

7 Conclusions and Future Works

In this paper, we describe the modified metric of information content (IC) that would be applicable to both of the domain ontologies of semantic technology and the simple however complete taxonomy like WordNet as well. Our proposed IC metric can be used to measure the semantic similarity among the concepts of an ontology regardless of its complex structure. In our modified metric we consider both the concept hyponyms or subsumptions and the relations as well with a coefficient factor of logarithmic ratio to combine them. We implemented this metric of IC to measure the semantic similarity among concepts. Ontology processing algorithms like ontology alignment and ontology segmentation can get benefits by detecting semantically related blocks of an ontology with the semantic similarity measure.

Table 4: Precision, Recall and F value for the suggested blocks by our proposed equation

$Block$	$Ref.Block$	Precision	Recall	F
24	24	1.000	1.000	1.000
3	4	1.000	0.750	0.857

Table 5: Purity, Inverse-purity and F-measure for the blocks suggested by Eq. 9 and by our proposed metric

Metric	Purity	Inverse-purity	F-measure
Seco	1.000	0.318	0.424
Proposed	1.000	0.970	0.984

We experimented with a small however representative ontology to observe the trend of our metric and we obtained more accurate semantic similarity among concepts and eventually we detected semantically related blocks of concepts. We measured the Purity, Inverse-purity and F-measure for our proposed metric and we observed that our proposed metric outperformed against Seco’s model of intrinsic information content on ontology, where concepts are specified with properties. The semantic groups of closely related concepts can be further used in ontology segmentation and in large scale ontology alignment. The experiment shows that our modified IC metric works better.

Our future plan is to experiment more with large DL ontologies. Furthermore, we have a plan to integrate our modified metric with our fastest and scalable Anchor-Flood algorithm (Seddiqui & Aono 2008) to align large scale ontologies and to retrieve segmented alignment.

References

- Berners-Lee, T., Fischetti, M. & Dertouzos, M. (1999), *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web*, Harper San Francisco.
- Collins, A. & Quillian, M. (1969), ‘Retrieval time from semantic memory’, *Journal of Verbal Learning and Verbal Behavior* **8**, 240–247.
- Ehrig, M. (2007), *Ontology Alignment: Bridging the Semantic Gap*, Springer, New York.
- Grau, B., Parsia, B. & Sirin, E. (2006), ‘Combining OWL ontologies using E-connections’, *Web Semantics: Science, Services and Agents on the World Wide Web* **4**(1), 40–59.
- Grau, B., Parsia, B., Sirin, E. & Kalyanpur, A. (2005a), ‘Automatic Partitioning of OWL Ontologies Using ϵ -Connections’, *Proceedings of the International Workshop on Description Logics (DL’05), Edinburgh, Scotland*.
- Grau, B., Parsia, B., Sirin, E. & Kalyanpur, A. (2005b), ‘Modularizing OWL ontologies’, *Proc. KCAP-2005 Workshop on Ontology Management, Banff, Canada*.
- Gruber, T. (1995), ‘Toward Principles for the Design of Ontologies Used for Knowledge Sharing’, *International Journal of Human-Computer Studies* **43**(5/6), 907–928.
- Hirst, G. & St-Onge, D. (1998), ‘Lexical chains as representations of context for the detection and correction of malapropisms’, *WordNet: An electronic lexical database* pp. 305–332.
- Hu, W., Cheng, G., Zheng, D., Zhong, X. & Qu, Y. (2006), ‘The Results of Falcon-AO in the OAEI 2006 Campaign’, *Proceedings of Ontology Matching (OM-2006), Athens, Georgia, USA* pp. 124–133.

- Hu, W., Qu, Y. & Cheng, G. (2008), 'Matching Large Ontologies: A Divide-and-Conquer Approach', *Journal of Data & Knowledge Engineering* **67**(1), 140–160.
- Hu, W., Zhao, Y. & Qu, Y. (2006), 'Partition-based Block Matching of Large Class Hierarchies', *Proceedings of the 1st Asian Semantic Web Conference (ASWC2006), Beijing, China* pp. 72–83.
- Jiang, J. & Conrath, D. (1997), 'Semantic similarity based on corpus statistics and lexical taxonomy', *Proceedings on International Conference on Research in Computational Linguistics, Taiwan* pp. 19–33.
- Knappe, R., Bulskov, H. & Andreasen, T. (2007), 'Perspectives on ontology-based querying', *International Journal of Intelligent Systems* **22**(7).
- Lin, D. (1998), 'An information-theoretic definition of similarity', pp. 296–304.
- Maedche, A. & Staab, S. (2001), 'Ontology Learning for Semantic Web', *IEEE Intelligent Systems* **16**(2), 72–79.
- McGuinness, D., Van Harmelen, F. et al. (2004), 'Owl web ontology language overview', *W3C recommendation*, <http://www.w3.org/TR/owl-features> **10**.
- Miller, G., Beckwith, R., Fellbaum, C., Gross, D. & Miller, K. (1990), 'WordNet: An on-line lexical database', *International journal of lexicography* **3**(4), 235–312.
- Miller, G. & Charles, W. (1991), 'Contextual Correlates of Semantic Similarity.', *Language and cognitive processes* **6**(1), 1–28.
- Rada, R., Mili, H., Bicknell, E. & Blettner, M. (1989), 'Development and application of a metric on semantic nets', *IEEE transactions on systems, man and cybernetics* **19**(1), 17–30.
- Resnik, P. (1995), 'Using information content to evaluate semantic similarity in a taxonomy', *Proceedings of the 14th International Joint Conference on Artificial Intelligence, Montreal, Canada* pp. 448–453.
- Resnik, P. (1999), 'Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language', *Journal of artificial intelligence* pp. 95–130.
- Seco, N., Veale, T. & Hayes, J. (2004), 'An intrinsic information content metric for semantic similarity in WordNet', *Proceedings of ECAI'2004, the 16th European Conference on Artificial Intelligence* **16**, 1089–1090.
- Seddiqui, M. H. & Aono, M. (2008), 'Alignment Results of Anchor-Flood Algorithm for OAIE-2008', *Proceedings of Ontology Matching Workshop of the 7th International Semantic Web Conference, Karlsruhe, Germany* pp. 120–127.
- Seidenberg, J. & Rector, A. (2006), 'Web Ontology Segmentation: Analysis, Classification and Use', *Proceedings of the 15th International Conference on World Wide Web (WWW2006), Edinburgh, Scotland* pp. 13–22.
- Shannon, C. & Weaver, W. (1948), 'A mathematical theory of communication', *Bell System Technical Journal* **27**, 379–423.
- Stuckenschmidt, H. & Klein, M. (2004), 'Structure-based Partitioning of Large Concept Hierarchies', *Proceedings of the 3rd International Semantic Web Conference (ISWC2004), Hiroshima, Japan* pp. 289–303.
- Studer, R., Benjamins, V. & Fensel, D. (1998), 'Knowledge Engineering: Principles and Methods', *Journal of Data & Knowledge Engineering* **25**(1-2), 161–197.
- Sussna, M. (1993), Word sense disambiguation for free-text indexing using a massive semantic network, in 'Proceedings of the second international conference on Information and knowledge management', ACM New York, NY, USA, pp. 67–74.
- Wu, Z. & Palmer, M. (1994), Verb semantics and lexical selection, in 'Proceedings of the 32nd annual meeting of the Association for Computational Linguistics', Las Cruces, New Mexico, pp. 133–138.

Author Index

- Aono, Masaki, 89
Brown, Ross A., 25
Chen, Shan, 33
Ghose, Aditya, iii
Governatori, Guido, 3
Grossmann, Georg, 43
Hartmann, Sven, 15
Koehler, Henning, 15
Link, Sebastian, iii
Mancilla-Amaya, Leonardo, 53
Neumayr, Bernd, 61
Punnarut, Ravikarn, 71
Rotolo, Antonino, 3
Sanin, Cesar, 53
Schrefl, Michael, 43, 61
Schreithauer, Gregor, 79
Seddiqi, Md. Hanif, 89
Sriharee, Gridaphat, 71
Stumptner, Markus, 43
Szczerbicki, Edward, 53
Thalheim, Bernhard, 61
Wang, Jing, 15
Williams, Mary-Anne, 33
Wirtz, Guido, 79

Recent Volumes in the CRPIT Series

ISSN 1445-1336

Listed below are some of the latest volumes published in the ACS Series *Conferences in Research and Practice in Information Technology*. The full text of most papers (in either PDF or Postscript format) is available at the series website <http://crpit.com>.

- Volume 84 - Artificial Intelligence and Data Mining 2007**
 Edited by Kok-Leong Ong, Deakin University, Australia, Wenyuan Li, University of Texas at Dallas, USA and Junbin Gao, Charles Sturt University, Australia. December, 2007. 978-1-920682-65-1. Contains the proceedings of the 2nd International Workshop on Integrating AI and Data Mining (AIDM 2007), Gold Coast, Australia. December 2007.
- Volume 85 - Advances in Ontologies 2007**
 Edited by Thomas Meyer, Meraka Institute, South Africa and Abhaya Nayak, Macquarie University, Australia. December, 2007. 978-1-920682-66-8. Contains the proceedings of the 3rd Australasian Ontology Workshop (AOW 2007), Gold Coast, Queensland, Australia.
- Volume 86 - Safety Critical Systems and Software 2007**
 Edited by Tony Cant, Defence Science and Technology Organisation, Australia. December, 2007. 978-1-920682-67-5. Contains the proceedings of the 12th Australian Conference on Safety Critical Systems and Software, August 2007, Adelaide, Australia.
- Volume 87 - Data Mining and Analytics 2008**
 Edited by John F. Roddick, Jiuyong Li, Peter Christen and Paul Kennedy. November, 2008. 978-1-920682-68-2. Contains the proceedings of the 7th Australasian Data Mining Conference (AusDM 2008), Adelaide, Australia. December 2008.
- Volume 88 - Koli Calling 2007**
 Edited by Raymond Lister University of Technology, Sydney and Simon University of Newcastle. November, 2007. 978-1-920682-69-9. Contains the proceedings of the 7th Baltic Sea Conference on Computing Education Research.
- Volume 89 - Australian Video**
 Edited by Heng Tao Shen and Michael Frater. October, 2008. 978-1-920682-70-5. Contains the proceedings of the 1st Australian Video Conference.
- Volume 90 - Advances in Ontologies**
 Edited by Thomas Meyer, Meraka Institute, South Africa and Mehmet Orgun, Macquarie University, Australia. September, 2008. 978-1-920682-71-2. Contains the proceedings of the Knowledge Representation Ontology Workshop (KROW 2008), Sydney, September 2008.
- Volume 91 - Computer Science 2009**
 Edited by Bernard Mans Macquarie University. January, 2009. 978-1-920682-72-9. Contains the proceedings of the Thirty-Second Australasian Computer Science Conference (ACSC2009), Wellington, New Zealand, January 2009.
- Volume 92 - Database Technologies 2009**
 Edited by Xuemin Lin, University of New South Wales and Athman Bouguettaya, CSIRO. January, 2009. 978-1-920682-73-6. Contains the proceedings of the Twentieth Australasian Database Conference (ADC2009), Wellington, New Zealand, January 2009.
- Volume 93 - User Interfaces 2009**
 Edited by Paul Calder Flinders University and Gerald Weber University of Auckland. January, 2009. 978-1-920682-74-3. Contains the proceedings of the Tenth Australasian User Interface Conference (AUIC2009), Wellington, New Zealand, January 2009.
- Volume 94 - Theory of Computing 2009**
 Edited by Prabhhu Manyem, University of Ballarat and Rod Downey, Victoria University of Wellington. January, 2009. 978-1-920682-75-0. Contains the proceedings of the Fifteenth Computing: The Australasian Theory Symposium (CATS2009), Wellington, New Zealand, January 2009.
- Volume 95 - Computing Education 2009**
 Edited by Margaret Hamilton, RMIT University and Tony Clear, Auckland University of Technology. January, 2009. 978-1-920682-76-7. Contains the proceedings of the Eleventh Australasian Computing Education Conference (ACE2009), Wellington, New Zealand, January 2009.
- Volume 96 - Conceptual Modelling 2009**
 Edited by Markus Kirchberg, Institute for Infocomm Research, A*STAR, Singapore and Sebastian Link, Victoria University of Wellington, New Zealand. January, 2009. 978-1-920682-77-4. Contains the proceedings of the Fifth Asia-Pacific Conference on Conceptual Modelling (APCCM2008), Wollongong, NSW, Australia, January 2008.
- Volume 97 - Health Data and Knowledge Management 2009**
 Edited by James R. Warren, University of Auckland. January, 2009. 978-1-920682-78-1. Contains the proceedings of the Third Australasian Workshop on Health Data and Knowledge Management (HDKM 2009), Wellington, New Zealand, January 2009.
- Volume 98 - Information Security 2009**
 Edited by Ljiljana Brankovic, University of Newcastle and Willy Susilo, University of Wollongong. January, 2009. 978-1-920682-79-8. Contains the proceedings of the Australasian Information Security Conference (AISC 2009), Wellington, New Zealand, January 2009.
- Volume 99 - Grid Computing and e-Research 2009**
 Edited by Paul Roe and Wayne Kelly, QUT. January, 2009. 978-1-920682-80-4. Contains the proceedings of the Australasian Workshop on Grid Computing and e-Research (AusGrid 2009), Wellington, New Zealand, January 2009.
- Volume 100 - Safety Critical Systems and Software 2007**
 Edited by Tony Cant, Defence Science and Technology Organisation, Australia. December, 2008. 978-1-920682-81-1. Contains the proceedings of the 13th Australian Conference on Safety Critical Systems and Software, Canberra Australia.
- Volume 101 - Data Mining and Analytics 2009**
 Edited by Paul J. Kennedy, University of Technology, Sydney, Kok-Leong Ong, Deakin University and Peter Christen, The Australian National University. November, 2009. 978-1-920682-82-8. Contains the proceedings of the 8th Australasian Data Mining Conference (AusDM 2009), Melbourne Australia.