BMSD 2012

Second International Symposium on

Business Modeling and Software Design

Proceedings

Geneva, Switzerland • 4 - 6 July, 2012

Organized by:

JCREST

In Cooperation with:

CTIT Centre for Telematics and Information Technology

INSTICC

TECHNICAL UNIVERSITY OF SOFIA

Hosted by:

UNIVERSITÉ DE GENÈVE

FACULTÉ DES SCIENCES ÉCONOMIQUES ET SOCIALES

ISS INSTITUTE OF SERVICES SCIENCE

BMSD 2012

Proceedings of the
Second International Symposium on
Business Modeling and Software Design

Geneva, Switzerland

July 4-6, 2012

Organized by

IICREST - Interdisciplinary Institute for Collaboration and Research on Enterprise Systems and Technology

In Cooperation with

CTIT - Center for Telematics and Information Technology

INSTICC - Institute for Systems and Technologies of Information, Control and Communication

Technical University of Sofia

Hosted by

Institute of Services Science, University of Geneva

Edited by Boris Shishkov

Graphics Production by Bozhana Yankova

Printed in Portugal

ISBN: 978-989-8565-26-6

Depósito Legal: 344848/12

http://www.is-bmsd.org

secretariat@iicrest.eu

BRIEF CONTENTS

KEYNOTE SPEAKERS

Jan L.G. Dietz

Delft University of Technology

The Netherlands

Ivan Ivanov

SUNY Empire State College

USA

Dimitri Konstantas

University of Geneva

Switzerland

Marten van Sinderen

University of Twente

The Netherlands

CHAIR AND PROGRAM COMMITTEE

CHAIR

Boris Shishkov, IICREST, Bulgaria

PROGRAM COMMITTEE

Mehmet Aksit, University of Twente, The Netherlands

Antonia Albani, University of St. Gallen, Switzerland

Ognyan Andreev, Technical University of Sofia, Bulgaria

Paulo Anita, Delft University of Technology, The Netherlands

Rumen Arnaudov, Technical University of Sofia, Bulgaria

Colin Atkinson, University of Mannheim, Germany

Csaba Boer, TBA, The Netherlands

Boyan Bontchev, Sofia University St. Kliment Ohridski, Bulgaria

Frances Brazier, Delft University of Technology, The Netherlands

Barrett Bryant, University of North Texas, USA

Cinzia Cappiello, Politecnico di Milano, Italy

Jorge Cardoso, University of Coimbra, Portugal

Kuo-Ming Chao, Coventry University, UK

Ruzanna Chitchyan, University of Leicester, UK

Samuel Chong, Capgemini, UK

Dimitar Christozov, American University in Bulgaria - Blagoevgrad, Bulgaria

Selim Ciraci, University of Twente, The Netherlands

José Cordeiro, Polytechnic Institute of Setúbal, Portugal

Dumitru Dan Burdescu, University of Craiova, Romania

Joop De Jong, Delft University of Technology, The Netherlands

Jan L. G. Dietz, Delft University of Technology, The Netherlands

Lyubka Doukovska, Bulgarian Academy of Sciences, Bulgaria

Joaquim Filipe, Polytechnic Institute of Setúbal, Portugal

Chiara Francalanci, Politecnico di Milano, Italy

Boris Fritscher, University of Lausanne, Switzerland

PROGRAM COMMITTEE (CONT.)

J. Paul Gibson, T&MSP - Telecom & Management SudParis, France

Eduardo Gonçalves da Silva, University of Twente, The Netherlands

Rafael Gonzalez, Javeriana University, Colombia

Clever Ricardo Guareis de Farias, University of Sao Paulo, Brazil

Markus Helfert, Dublin City University, Ireland

Philip Huysmans, University of Antwerp, Belgium

Ilian Ilkov, IBM, The Netherlands

Ivan Ivanov, SUNY Empire State College, USA

Dmitry Kan, AlphaSense Inc., Russia

Dimitris Karagiannis, University of Vienna, Austria

Marite Kirikova, Riga Technical University, Latvia

Samuel Kounev, Karlsruhe Institute of Technology, Germany

José Paulo Leal, University of Porto, Portugal

Kecheng Liu, University of Reading, UK

Leszek Maciaszek, Macquarie University, Australia / University of Economics, Poland

Jelena Marincic, University of Twente, The Netherlands

Michele Missikoff, Institute for Systems Analysis and Computer Science, Italy

Dimitris Mitrakos, Aristotle University of Thessaloniki, Greece

Valeri Mladenov, Technical University of Sofia, Bulgaria

Preslav Nakov, Qatar Computing Research Institute - Qatar Foundation, Qatar

Ricardo Neisse, Fraunhofer IESE, Germany

Bart Nieuwenhuis, University of Twente, The Netherlands

Selmin Nurcan, University Paris 1 Pantheon Sorbonne, France

Olga Ormandjieva, Concordia University, Canada

Robert Parhonyi, Inter Access, The Netherlands

Marcin Paprzycki, Polish Academy of Sciences, Poland

Oscar Pastor, Universidad Politécnica de Valencia, Spain

Erik Proper, Public Research Centre - Henri Tudor, Luxembourg

Ricardo Queirós, IPP, Portugal

Jolita Ralyte, University of Geneva, Switzerland

Gil Regev, EPFL / Itecor, Switzerland

Ella Roubtsova, Open University, The Netherlands

Irina Rychkova, University Paris 1 Pantheon Sorbonne, France

Shazia Sadiq, University of Queensland, Australia

PROGRAM COMMITTEE (CONT.)

Brahmananda Sapkota, University of Twente, The Netherlands

Tony Shan, Keane Inc., USA

Valery Sokolov, Yaroslavl State University, Russia

Richard Starmans, Utrecht University, The Netherlands

Cosmin Stoica Spahiu, University of Craiova, Romania

Coen Suurmond, RBK Group, The Netherlands

Bedir Tekinerdogan, Bilkent University, Turkey

Linda Terlouw, ICRIS B.V., The Netherlands

Yasar Tonta, Hacettepe University, Turkey

Roumiana Tsankova, Technical University of Sofia, Bulgaria

Marten van Sinderen, University of Twente, The Netherlands

Mladen Velev, Technical University of Sofia, Bulgaria

Kris Ven, University of Antwerp, Belgium

Maria Virvou, University of Piraeus, Greece

Martijn Warnier, Delft University of Technology, The Netherlands

Shin-Jer Yang, Soochow University, Taiwan

Benjamin Yen, University of Hong Kong, China

Fani Zlatarova, Elizabethtown College, USA

BEST PAPER SELECTIONS

The authors of around ten selected papers presented at BMSD 2012 will be invited by Springer-Verlag to submit revised and extended versions of their papers for publication in a Springer LNBIP Series book, and the authors of around five selected papers presented at BMSD 2012 will be invited to submit revised and extended versions of their papers for publication in a special issue of the international journal Enterprise Modelling and Information Systems Architectures (EMISA).

FOREWORD

This book contains the proceedings of BMSD 2012 - the Second International Symposium on Business Modeling and Software Design, held in Geneva, Switzerland (at the University of Geneva), on July 4 - 6, 2012. The proceedings consist of 18 high-quality research and experience papers that have not been published previously. These papers have undergone a detailed peer-review process and were selected based on rigorous quality standards.

Being hosted by the University of Geneva, and particularly by its Institute of Services Science (ISS), the symposium was organized and sponsored by the Interdisciplinary Institute for Collaboration and Research on Enterprise Systems and Technology (IICREST), in cooperation with the Center for Telematics and Information Technology (CTIT), the Institute for Systems and Technologies of Information, Control and Communication (INSTICC), and Technical University of Sofia.

Following its first edition in Sofia, Bulgaria, BMSD continues to be an inspiring symposium that touches upon business modeling and its relation to software design, with a high level of interaction among leading scientists, engineers, and practitioners. The gap between information systems and underlying business models continues to pose challenges to business architects, IT architects, software developers, and system analysts. The scientific areas of interest to BMSD 2012 are: (i) Business Models and Requirements; (ii) Business Models and Services; (iii) Business Models and Software; (iv) Information Systems Architectures. Each year, a special theme is chosen, for making presentations and discussions more focused. The theme of BMSD 2012 is: **From Business Modeling to Service-Oriented Solutions**.

Even though software development has evolved significantly over the previous years, it is still a challenge very often to correctly and exhaustively derive functional requirements and this is one of the reasons for numerous software project failures. Deriving requirements in turn crucially relates to building adequate business models - this is needed firstly for understanding the considered organization (and/or reengineering it on top of that, if necessary) and secondly, as a basis for specifying an automation of (part of) its processes by means of software systems. Hence, not grasping correctly and exhaustively a business system would inevitably lead to consequent software failures - business demands and technology solutions have to be aligned. BMSD 2012 has addressed these challenges, by considering a large number of research topics: from more conceptual ones, such as enterprise engineering, value modeling, normalized enterprise systems, and rules mining to more technical ones, such as software specification, database clusters, IT services, and 'e-applications', from more business-oriented ones, such as enterprise architecture management, requirements specification, and enterprise interoperability to IT architectures –related topics. We believe that the current proceedings highlight challenging technical problems and present innovative solutions relevant to the mentioned topics.

The 18 published papers (including several Invited Papers) were selected from 46 submissions and 9 of these papers were selected for a 30-minutes oral presentation (Full Papers); in addition, 9 papers were selected for a 20-minutes oral presentation (Short Papers and Special Session Papers). Hence, the full-paper acceptance ratio of 21% shows a high level of quality which we intend to maintain and reinforce in the following editions of this symposium. All presented papers will soon be available at the SciTePress digital library. Furthermore, the authors of around ten selected papers presented at BMSD 2012 will be invited by Springer-Verlag to submit revised and extended versions of their papers for publication in a Springer LNBIP (Lecture Notes in Business Information Processing) Series book, and the authors of around five selected papers presented at BMSD 2012 will be invited to submit revised and extended versions of their papers for publication in a special issue of the international journal Enterprise Modelling and Information Systems Architectures (EMISA).

The high quality of the BMSD 2012 program is enhanced by four keynote lectures, delivered by distinguished guests who are renowned experts in their fields, including (alphabetically): Jan L.G. Dietz (Delft University of Technology, The Netherlands), Ivan Ivanov (SUNY Empire State College, USA), Dimitri Konstantas (University of Geneva, Switzerland), and Marten van Sinderen (University of Twente, The Netherlands). In addition, the keynote lecturers and other BMSD'12 participants will take part in a panel discussion and also in informal discussions focused on community building and project acquisition. These high points in the symposium program would definitely contribute to positioning BMSD as a high quality event driven by a stable and motivated community - evidence for this is the fact that many of those who attended BMSD 2011 are now attending BMSD 2012.

Building an interesting and successful program for the symposium required the dedicated efforts of many people. Firstly, we must thank the authors, whose research and development achievements are recorded here. Secondly, the program committee members each deserve credit for the diligent and rigorous peer-reviewing and paper selection process. Further, we appreciate the willingness of SciTePress to publish the current proceedings, expressing also special gratitude to Vitor Pedrosa for his excellent work and cooperation regarding the proceedings preparation. We would also like to compliment the excellent organization provided by the IICREST team; it did all necessary preparations for a stimulating and productive event. Last but not least, we thank the keynote lecturers for their invaluable contribution and for taking the time to synthesize and deliver their talks.

We wish you all an inspiring symposium and an enjoyable stay in the beautiful city of Geneva. We look forward to seeing you next year in Noordwijkerhout, The Netherlands, for the Third International Symposium on Business Modeling and Software Design (BMSD 2013), details of which will be made available at http://www.is-bmsd.org.

Boris Shishkov

IICREST, Bulgaria

CONTENTS

KEYNOTE SPEAKERS

FULL PAPERS

SHORT PAPERS

SPECIAL SESSION ON BUSINESS MODELS AND SERVICE APPLICATIONS

KEYNOTE SPEAKERS

Enterprise Ontology Driven Software Generation

Jan L. G. Dietz
Delft University of Technology, Delft, The Netherlands
J.L.G.Dietz@tudelft.nl

Abstract: Model Driven Engineering has been with us for quite some time, the most well known approach being OMG's Model Driven Architecture. However, although it has brought substantial benefits compared to other software engineering approaches, Model Driven Engineering presently still suffers from two major shortages. First, it is unable to deliver domain models from which the correct functional requirements can be derived. Hence, true validation is hardly possible: the software does not meet user expectations. Second, the models to be produced during the system development process, are not formally defined. Hence, their verification remains a cumbersome task. One of the theoretical pillars of Enterprise Engineering (EE) is the Generic System Development Process. It distinguishes between the using system and the object system (the system to be built), and it states that any software development process should start from the ontological construction model of the using system. In addition, EE's systemic notion of Enterprise Ontology offers a formalized ontological model of an enterprise that satisfies the C4E quality criteria (coherent, consistent, comprehensive, concise, and essential). An operational application software generator will be presented that takes this ontological model, with some extensions, as source code input and executes the model as a professional software application. Changes in the software, as required by any agile enterprise, are brought about 'on the fly', through re-generation, based on the modified ontological model of the enterprise.

BRIEF BIOGRAPHY

Jan L.G. Dietz is emeritus full professor in Information Systems Design at Delft University of Technology, full professor in Enterprise Engineering at Delft University of Technology, and director of Sapio (www.sapio.nl). He holds a Master degree in Electrical Engineering and a Doctoral degree in Computer Science. He has published over 200 scientific and professional articles and books. His current research interests are in the emerging discipline of Enterprise Engineering, of which Enterprise Architecture, Enterprise Ontology, and Enterprise Governance are the major pillars. Before his academic career, he has practiced application software engineering for ten years in industry. Jan Dietz is the spiritual father of DEMO (Design & Engineering Methodology for Organizations), and honorary chairman of the Enterprise Engineering Institute (www.ee-institute.com). For the development of Enterprise Engineering, he chairs the international research network CIAO! (www.ciaonetwork.org). He also acts as editor-in-chief of a book series on Enterprise Engineering, published by Springer. For more information, visit http://en.wikipedia.org/wiki/Jan_Dietz.

Aligning IT Architecture to the Business Strategy

Ivan Ivanov
SUNY Empire State College, New York, U.S.A.
Ivan.Ivanov@esc.edu

Abstract: Increasingly the business success and economic opportunities steadily depend on IT-enabled capabilities and IT-driven business transformations. In today's global digital economy, the technology and business domains are colliding forcefully than ever and new business models and growing prospects emerge. The IT, and especially emerging technologies, profoundly changes how companies strategize their technology architectures and create value and business growth as a result of IT, both within specific industries, and through industry boundaries. For the IT domain it is most important to define and establish the organization's IT architecture underneath the enterprise architecture. A well-formulated IT architecture comprises content and processes and outlines the systemic effect of the mutually dependent technological components and processes of the enterprise architecture. The talk will explore the impact of emerging technologies on the process of aligning organizational IT architecture to the business strategy and will emphasize the driving forces intertwining IT domain with the business agility, growth and asset utilization.

BRIEF BIOGRAPHY

Ivan I. Ivanov is an Associate Professor of Computer Science and Information Systems at State University of New York - Empire State College. He holds a Ph.D. degree in Computers and Networking Technologies and a MS degree in Computer Engineering. He was a research fellow in leading universities in Great Britain, The Netherlands, France, and Germany. Dr. Ivanov worked in joint European IT projects with partners from France, United Kingdom, Germany, Belgium, Spain, Greece, Italy, and in cooperation with worldwide technology leaders ensuing developing advanced technological infrastructure and information services at educational establishments in Bulgaria. His scholarly work is built upon his wide range of competences within Computer Architecture, Information Systems Design, Network Services, and Information Technology Management blended with proficiency in aligning technological solutions to organizational requirements and needs. His latest professional interest is in emerging computing models and the correlation between computing and technology innovations with business and society advance. Dr. Ivanov works with students from diverse area of studies in emerging technology topics that reflect their educational plans and career opportunities. His undergraduate studies are in broad areas of Computer Architecture, Data Communications and Networks, Systems Analysis and Design, Information Security and Policy, and Information Technology for Management. He has developed and teaches graduate courses in Project Management, Management Information Systems, and Strategic IT Management for the MBA program. He is an organizer and sponsor for the Annual Technology Workshops at the Long Island Center, forums for Empire State College students to build up researching, analytical, critical thinking and presentation skills; sharing best practice in technology topics as it relates to course projects and professional development to a select group of peers, college alumni and professionals.

From Care to Prevention
A Holistic View for Future e-m/Health ICT Services

Dimitri Konstantas
University of Geneva, Geneva, Switzerland
Dimitri.Konstantas@unige.ch

Abstract: ICT based services and products are today a major element in the support of health care: e-heath and m-health are offering tools that can monitor patients 24h per day, provide valuable information to care personnel and trigger the dispatching of assistance. However, technology itself is not enough: Cure without prevention will not sustainably solve the health care problems of the next 40 years. New innovative ICT based services are needed, integrating technology and life style and social models helping educate the (future) patients to acquire healthy habits and allowing them from one side to postpone for several years the appearance of health problems and from the other side how to be as much as possible self supported in coping with their health problems.

BRIEF BIOGRAPHY

Dimitri Konstantas is Professor and Vice-Dean at the Faculty of Social Sciences and Economics, and member of the Institute of Services Sciences. He holds a Phd in Informatics from the University of Geneva, a MSc in Computer Sciences from the University of Toronto and a MSc in Electronic Engineering from the National Technical University of Athens. He was previously professor and Chair of the APS group at the University of Twente, The Netherlands and Research assistant at the Institute of Computer Sciences, FORTH, Crete, Greece. Since 1985, prof. Konstantas, is active in multidisciplinary research in the areas of Object Oriented systems, agent technologies, Multimedia applications and e-commerce services. Since 2002 his main research areas are mobile and wireless multimedia services and applications, with special interest in mobile health and location based services. He has more than 200 publications in international conferences, journals, books and book chapters and a long participation and leadership in European and national projects. Prof. Konstantas is serving as consultant and scientific expert for several international companies and governments.

Modeling Internet-based Goal-oriented Enterprise Interoperability

Marten van Sinderen
University of Twente, Enschede, The Netherlands
m.j.vansinderen@utwente.nl

Abstract: The ability of an enterprise to collaborate with other enterprises is increasingly important to stay competitive and be successful in business. Enterprises must be able to effectively and sufficiently interact with suppliers and customers, and possibly combine and coordinate efforts to satisfy the needs of a single customer. This requires that different enterprises have to devise their processes and agree on a shared universe of discourse, such that their respective collaboration goals can be fulfilled. Furthermore, it requires interoperability between enterprises, i.e. the ability of enterprises to exchange information and to use the information that has been exchanged in accordance to the collaboration goals. The enterprises' processes drive the information exchange and use the information through interpretation under the shared universe of discourse. If the collaboration is to be supported by Information and Internet Technology, underlying automated systems send and receive messages containing user data to represent the information. Much work has been done to ensure interoperability of technical systems. Communication protocols and data formats have been standardized to achieve syntactic interoperability (exchange of data), and ontology definitions and ontology languages have been developed to facilitate semantic interoperability (interpretation of data). In order to achieve enterprise interoperability, business requirements and technology solutions have to be aligned. In this talk we will explore the modeling of business requirements with respect to enterprise interoperability in relation to Internet-based technology solutions. We will briefly discuss the challenges emerging from evolving enterprises as a result of, for example, changing goals, partners or technology.

BRIEF BIOGRAPHY

Marten van Sinderen holds a MSc in Electrical Engineering and a PhD in Computer Science, both from the University of Twente (UT). He is currently Associate Professor at the Faculty of Electrical Engineering, Mathematics and Computer Science of the UT, and coordinator of the research area on Service Architectures and Health Applications at UT's Centre for Telematics and Information Technology (CTIT). His research focuses on design methods and technologies for distributed information systems. His research interests include service-oriented architectures, model-driven design, enterprise interoperability, and business-IT alignment. Marten van Sinderen is active in both national and international communities on his field of interest. He was project manager of the Dutch Freeband/A-MUSE project (BSIK 03025) on model-driven design of context-aware services. He currently leads the Dutch GenCom/U-Care project (IGC0816) on tailorable and adaptive homecare services. He is chairman of the steering committee of the International IEEE Enterprise Computing Conference (EDOC), and program co-chair of the International Conference on e-Business (ICE-B). He is also a member of the managerial board of IFIP WG5.8 on Enterprise Interoperability, and a member of the editorial boards of the Enterprise Information Systems journal published by Taylor & Francis and the Service oriented Computing and Applications journal published by Springer.

FULL PAPERS

Homeostasis
The Forgotten Enabler of Business Models

Gil Regev[1,2], Olivier Hayard[2] and Alain Wegmann[1]
[1]*Ecole Polytechnique Fédérale de Lausanne (EPFL), School of Com-puter and Communication Sciences, CH-1015 Lausanne, Switzerland*
[2]*Itecor, Av. Paul Cérésole 24, cp 568, CH-1800 Vevey 1, Switzerland*
{gil.regev, alain.wegmann}@epfl.ch, {g.regev, o.hayard}@itecor.com

Keywords: Business Modelling, Survival, Systems, Identity, Homeostasis, Entropy, Negentropy.

Abstract: Business modelling methods most often model an organization's value provision to its customers followed by the activities and structure necessary to deliver this value. These activities and structure are seen as infinitely malleable; they can be specified and engineered at will. This is hardly in line with what even laymen can observe of organizations, that they are not easy to change and that their behaviour often is not directly centred on providing value to customers. We propose an alternative view in which organizations exist by maintaining stable states that correspond to their identity. We analyse how these states are maintained through homeostasis, the maintenance of ones identity. Homeostasis helps to explain both the inability of organizations to provide maximum value to their customers and their reluctance to change. From this point of view, resistance to change is not something to fight or to ignore but an essential force behind organizational behaviour that can be built upon for creating adequate strategies.

1 INTRODUCTION

Business modelling refers to the description of organizations for the purpose of understanding their informational needs. The premise of business modelling is that IT needs to support the organization and it is therefore crucial to fully understand it (Shishkov, 2011). Business modelling often begins by modelling the business processes of the organization and proceeds down the hierarchy to the way these processes are supported by the IT. In this view, a business model is a description, as complete as possible, of some part of the organization that needs IT support.

Business modelling also refers to the modelling of organizational strategy through initiatives such as Service Science (Spohrer and Riecken, 2006) and frameworks such as e^{3}value (Gordijn and Akkermans, 2003) and the Business Model Ontology (Osterwalder and Pigneur, 2010). A business model in these later frameworks describes how a company provides and captures value (Osterwalder and Pigneur, 2010). In this paper, we will refer to business modelling as the description of the complete organization, including its business strategy.

The strategy formulation of business modelling seems to be the direct product of the two leading schools of strategic thinking (Design and positioning) as viewed by Mintzberg et al. (1998). Just like the methods inherited from these three schools, business modelling dissociates strategy formulation from implementation. Implementation, it seems, is straightforward. If the strategists can define winning business models, the company can surely implement them. Also, strategic business modelling considers that organizations exist by maximizing value to customers and capturing part of this value. This view glosses over the everyday observation that organizations define their own rules that they consider good enough for their customers. Maximizing value for customers is not apparent in even the most successful commercial companies.

When describing the operational part of Business modelling, the models mostly contain roles processes, business rules, IT systems etc. There is no description of the mechanisms that maintain the organization in place. Business modelling does not address questions such as how come the organization exists and what is its capacity for change. Business modelling is all about change in the way the organization does business today. If

there were no need to change, it would mean that whatever the organization is doing right now is working and therefore there is no need for a new business model. Implementing a new business model is not an easy change and in most organizations it either fails or is accompanied by years of upheaval. Or as Michael Hammer, who coined the term Business Process Reengineering (BPR) in the late 1980s, has subsequently said reengineering transformed (Hammer, 1996): "organizations to the point where they were scarcely recognizable." or in other words "it saved companies by destroying them."

This is because most of the times companies, just like any organization survive not through radical change but by closely controlling change.

In this paper, we take the challenge of explaining an essential ingredient of organizational survival called homeostasis. Homeostasis was developed in the field of Physiology by Walter Cannon (Weinberg and Weinberg, 1988) but has such a broad description that it readily useful for describing the way organizations survive in a changing environment. Homeostasis, at its core, is a struggle against change. It therefore provides a good basis for explaining why new business models often fail. Because it is at the core of what the organization is, taking it into consideration will create much more accurate business models.

In Section 2 we give a few examples drawn from everyday life and from published cases, such as Apple, to show the problem of strategy, organizational culture and customer value. In Section 3 we provide an overview of business modelling. In Section 4 we explain the fundamentals of maintaining identity and negative entropy. In section 5 we explain the concept of Homeostasis. In Section 6 we describe the practical aspects of thinking in terms of Homeostasis.

2 A FEW BUSINESS EXAMPLES

Whereas the focus of business modelling is to understand what value the business provides to its customers and how it provides it, there are many examples where this value is not readily apparent. These examples do not belong to failed companies but to all existing companies. We provide a few examples to illustrate this point.

Apple's Strategic Constancy

Apple is in the enviable position of having created a host of products and services that customers find very valuable, which allows it to charge a premium price.

But, as much as competitors try to imitate Apple's business model, they are often unable to replicate the same products and services. This, we believe, is explained by Katzenbach (2012) as Jobs ability in understanding the role of culture in sustained strategic capabilities. Jobs, Isaacson (2011) says, was interested mostly in creating "an enduring company." It is more difficult to create an enduring company entrenched in culture than to create business models that imitate Apple's. However, without the culture, the business models have little chance of succeeding, as can be seen in HP's recent experience with the TouchPad.

Apple's endurance can be seen in many aspects of its culture. For example, the three principles that to this day guide Apple's Marketing Philosophy were written at its very beginning by Mike Markkula who was brought in by Steve Jobs (Isaacson, 2011), These three principles are (Isaacson, 2011): (a) Understanding customer needs better than other companies by establishing an intimate relationship with their feelings. (b) Focus on what is important and eliminating what is not. (c) Presenting Apple's products professionally and creatively. Isaacson (2011) says that these principles shaped Jobs's approach to business ever since.

After he was ousted from the company in 1985, these three principles held on for a while but slowly fizzled with Apple producing an ever-larger number of products with lower appeal to customers and less attention to their presentation and packaging.

When Jobs returned to Apple in 1997, he recreated its culture by various measures, such as (Isaacson, 2011): bringing back previous employees, creating incentives for keeping trusted employees, replacing Apple's board, and removing all projects that he estimated as not focused.

The quality Jobs built into the iMac, MacBook, iPod, iPhone and iPad is in a direct line with the quality he built into the original Macintosh. Like the original Macintosh, all these products are physical sealed so that only Apple can open them. In most of these products, customers cannot even replace the batteries. Replacing batteries in Notebooks, MP3 players and smartphones is a standard and highly valuable feature, as batteries tend to fail with time. An iPhone with a dead battery can only be returned to the Apple for replacement, at a much higher cost and delay than it would be for replacing a battery in a Nokia smartphone. An evaluation of customers' desires, will most probably list the simple and low

cost change of battery as a very valuable feature. However, this is purposefully missing from Apple's products.

Steve Jobs maintained a remarkable constancy in the kind of products he envisioned. According to Isaacson (2011) as way back as the early Macintosh days he wanted to design more curvy, colourful and friendly looking computers. Although the first Macintosh was a good start, the really curvaceous and colourful computers appeared with the iMac some 15 years later. Whereas during his absence Apple designed more and more common looking computers, when Jobs took over in 1997 he re-established the design philosophy he began in the 1970s.

As strange as it may sound today, Jobs was reluctant to allow third party developers to create apps for the iPhone for fear of compromising its security and integrity (Isaacson, 2001). Only when he found a way to bring together both aspects of opening the iPhone to external developers while maintaining strict control over what they provided, did Jobs accept to change his opinion. The result was the famous Apple Store, which today provides great value to customers as well as to Apple and to developers.

A Motorcycle Manufacturer

In (Hopwood, 2002) Hopwood describes a project in a motorcycle manufacturer in the 1930s where engineering produced a magnificent motorcycle, far superior to competition but management was unwilling to invest in the new tooling that was required to produce it. The changes made to produce the motorcycle with the old tooling made it too heavy and inferior to competitors. Why would management act in such a way? Is it simply insensitivity or is it that there's something else to say about resistance to change?

Maintaining a Revenue Stream

A recent article in the New York Times (Chozick, 2012) describes attempts by General Motors to maintain its appeal for youngsters. Apparently, present day youngsters are less interested in owning a car than previous generations. GM is trying to compensate for this lack of interest by hiring MTV. A dramatic cultural change is needed for GM to be able to carry out the changes that MTV deems necessary. GM does not seem to be primarily concerned by the value it provides to these new customers but by insuring its own revenue stream in the medium to long-term future.

Prices of periodical subscription by for individuals and libraries have increased even though printing cost has largely disappeared. Where is the value for the customer? At the same time, "new" schemes such as borrowing ebooks are appearing even though the borrowing of books was invented because books were a scarce resource whereas ebooks can be reproduced ad-infinitum. Again, the value for the customer is unclear.

On-line Retailers

Many companies' websites terms of use, e.g. Apple's iTunes, explicitly state that they make no promise that their website contents will be error-free, that they will offer continuous service and the like. Apple even goes as far as saying that the sole remedy available to dissatisfied customers is to stop using their website.

On-Line retailers have strict policies applied to customers who want to return products. These rules vary with the location of the company. In Switzerland, for example, the rules are often much more strict than in the United States. These policies embody legal and cultural aspects of the company and of the country and region concerned. Return Merchandise Authorizations (RMA) are sometime imposed by retailers. International sales are frequently subject to stricter rules and exclusions than domestic sales. The value for the customer is not apparent in these policies.

Even Amazon.com that prides itself on its superior customer service cannot avoid having some rules concerning the return policies governing items purchased by customers. These rules exclude the returns of some items and defines what can be returned and it what state. Returns are accepted within 30 days, for example, but why 30 days? Why not 90 or 10? Why is there a limit at all? The return policy also excludes returns for items bought through the CDNOW Preferred Buyer's Club. Why are these items excluded? Also music items must be unopened for the return to be accepted whereas books do not. Why is this?

A Healthcare Insurance

All Swiss residents are obliged by law to have health insurance, which they pay for themselves. Health insurance premiums for a family of four)2 adults and two children under 18 years old) is about 1000 Swiss Francs for a standard plan with the lowest co-

payment. The premiums have increased regularly every year for the last 10 to 15 years. Switching from one insurer to another is possible once a year. Many Swiss residents try to switch to the insurer that offers the lowest cost each year. In 2010 one of the smaller insurers was at the top of the list of the least expensive insurers. This insurer also had a very good reputation for quality. The result was a massive flow of new customers to this insurer. About 18 months later many new customers received a surprising and important premium hike, some of more than 60%, making this insurer the most expensive. Thus, this insurer went from the least expensive to the most expensive. This new pricing scheme will probably result in a massive drain of customers to other insurers. This price increase makes no sense if the goal of the insurer, as enterprise modelling methods consider, is to attract more customers.

3 AN OVERVIEW OF BUSINESS MODELING

Business modelling, enterprise modelling, enterprise architecture and enterprise engineering are used somewhat interchangeably to mean models of how an organization functions. Business modelling has emerged from the Information Technology (IT) practice as a way for IT people to understand the business's information needs. One of the early IT frameworks that integrate some aspects of business is what came to be called the Enterprise Architecture Framework, the Information Systems Architecture Framework, or more commonly the Zachman Framework (Zachman, 1987). Zachman's framework is made of a matrix in which the rows represent entities and the columns represent questions about these entities (e.g., what, how, when, why, where). The two topmost rows of Zachman's matrix represent the entities that are important to the business, the actions (processes) that are important, and the business locations.

Sowa and Zachman define a business model as (1992): the "design of the business" that shows "the business entities and processes and how they interact." From the architecture perspective inherent in this framework, a business model is seen as (Sowa and Zachman 1992): "the architect's drawings that depict the final building from the perspective of the owner, who will have to live with it in the daily routines of business."

The Reference Model of Open Distributed Processing is an ISO/IEC standard for describing organizations, their informational needs and their IT support (ISO/IEC, 1995-98). It consists of five viewpoints on the business: Enterprise, information, computational, engineering and technology (Kilov, 1999). The enterprise viewpoint captures the purpose, scope and policies of the organization (Kilov, 1999).

ArchiMate is a more recent enterprise architecture method, which models business processes and their support by IT. ArchiMate defines business systems as dynamic systems. A dynamic system is described by active structure concepts (also called agents), passive structure concepts (also called patiens) and behavioural concepts (Lankhorst et al., 2009). ArchiMate is made of three layers called Business, Application and Technology (Lankhorst et al., 2009). The business layer describes business actors and roles performing business processes that deliver products and services to external customers. The application layer describes the support provided by software applications to the business layer. The technology level describes the infrastructure necessary to run the software applications (Lankhorst et al., 2009).

The Design & Engineering Methodology for Organizations (DEMO) is a methodology for literally engineer organizations (Dietz, 2006). Organizations are said to be "designed and engineered artifacts" much like cars and IT systems but with the exception that their "active elements are human beings in their role of social individual or subject." (Dietz, 2006). In DEMO the essence of the enterprise are transactions consisting of production acts and coordination acts between the subjects. With production acts the subjects create the "goods or services delivered too the environment." With coordination acts the subjects "subjects enter into and comply with commitments toward each other regarding performance of P-acts Examples of C- acts are "request," "promise," and "decline."" (Dietz, 2006).

The examples above are all methods that attempt to model the organization with multiple viewpoints and multiple levels, i.e. from business to IT. Other business modelling methods address only the strategic definition level. e^3value focuses on the exchange of value objects between economic actors (Gordijn and Akkermans, 2003). The organization is viewed only as a black box. e^3value has been linked with i*, a leading Goal Oriented Requirements Engineering method (Gordijn and Yu, 2006). Value and goal models are used to show the value activities that contribute to the enterprise goals (Gordijn and Yu, 2006).

The Business Model Ontology, BMO, (Osterwalder and Pigneur, 2010) provides multiple ways of defining business models. Osterwalder and Pigneur define the concept of business model as (Osterwalder and Pigneur, 2010): "the rationale of how an organization creates, delivers and captures value." BMO proposes a canvas containing 9 elements: Customer Segments, Value Propositions, Channels, Customer Relationships, Revenue Streams, Key Resources, Key Activities, Key Partnerships and Cost Structure (Osterwalder and Pigneur, 2010). Osterwalder and Pigneur (2010) describe a number of business model patterns and show how they can described in the canvas. BMO focuses on the strategy formulation level and doesn't have an architecture component. The execution of the business model stops at the definition of the key resources and partnerships. More recently, work is underway (Fritscher 2011) to couple BMO and ArchiMate in order to provide a more complete business layer for ArchiMate and to more finely define the execution of BMO business models.

Most business modelling frameworks assume that the organization's main purpose is to provide value to the customer. Hammer, for example (Hammer, 1996), asks the question "what is a company? What is it for?" The answer according to Hammer is that (Hammer, 1996): "a company exists to create customer value. Everything a company does must be directed to this end." Hammer defines a customer in quite unorthodox terms. Moving beyond the notion of (Hammer, 1996) "someone who buys what the company sells." He defines a customer as (Hammer, 1996): "people whose behavior the company wishes to influence by providing them with value." Hammer considers as customers a much larger set than is traditionally the case. He gives the following list as customers of a pharmaceutical company (Hammer, 1996):

A. "The patient
B. The physician
C. The pharmacist
D. The wholesaler
E. The Food and Drug Administration
F. The Insurance company"

Notice that some of these customers, most notably, the physician, pharmacist and wholesaler would often be seen as suppliers rather than customers, whereas the Food and Drug Administration would be seen as a regulator today. Hammer's point is that their behaviour needs to be influenced by the company so that they are all willing to do their part in the sale of the medicine sold by the company. But the value expected by each of these customers is not the homogenous. The pharmacist expects a different value than the patient and the insurance company. The Food and Drug Administration is there to impose rules that constrain the sale of the medicine.

For Hammer (Hammer, 1996): "All of a company's activities and energies must be focused on and directed to the customer, who is, after all, the source of the company's revenue." Hammer (1996) explains why he puts customers as the sole and only reason for existence of a company by arguing that (a) shareholders also provide funds to the company but employees of a company cannot be motivated by the argument that they need to create more shareholder value. (b) In a global economy, customers have the upper hand over suppliers.

As we have seen in the various examples from the business modelling literature, value creation for customers is seen as the single most important reason for a company's existence. The resources, activities, and structure of the company are subservient to this all-encompassing goal.

With respect to the business examples we gave in the previous section, we can formulate a few critiques of this view.

If customers have the upper hand then why is it that the supplier defines the sales conditions and not the customer? Can an iPhone customer define the iTunes store conditions, or Google's privacy rules? Inspecting sales conditions and contracts of all kinds, shown in the previous section, we see that they protect the supplier more than they protect the customer. We are forced to conclude that companies cannot really maximize the value proposed to any individual customer, as proposed in business modelling.

In business modelling, it is assumed that the structure of the organization is defined once the value proposition has been defined. In other words, structure follows strategy. But structure, as Mintzberg et al. put it (1998): "follows strategy like the left foot follows the right" meaning that it is structure that enables strategy and strategy that changes the structure. Hence, without a firm structure of some kind, no strategy is possible. But where does structure comes from and how is it maintained?

As we have seen, business modelling methods mostly use abstractions such as roles, agents, actors, processes, transactions, commitments, services and value. These abstractions have been carefully devised to be free of any real human element, which rarely or ever appear in these models. However, ultimately it is people and organizational

departments that must execute the business models and it is then that problems arise because they were abstracted since the beginning.

4 MAINTAINING IDENTITY

Remember that Steve Jobs wanted to create an enduring company, but what did he mean by the term enduring? What is an enduring company? Let's take a few examples? Compaq existed for some 20 years, since1982 until its acquisition by HP in 2002. During that time it could have been said to be an enduring company, since nothing ultimately lasts forever. But what made Compaq enduring and what ended this endurance? We have shown elsewhere that for an organization to exist, it needs to maintain a number of norms (states that remain stable or constant) for a set of observers (Regev and Wegmann, 2004, Regev and Wegmann, 2005, Regev et al., 2009, Regev et al., 2011, Regev and Wegmann, 2011). Based on this model, Compaq existed because it maintained a number of norms that customers, shareholders, suppliers, employees, competitors and others could see as identifying the organization called Compaq. When Compaq was acquired by HP most of its constituent elements, e.g. people, buildings, machines and even website, continued to exist but were not organized in a coherent whole that observers could identify and call Compaq. Instead, most of them were absorbed in a new structure called HP with different relationships giving them a different meaning for observers.

Drawing an analogy with biological phenomena, we can say that a company that is being acquired by another is quite similar to a mouse being eaten by a cat. The mouse maintains somewhat independent existence and as observers, we can identify it as a mouse. If it is caught and eaten by a cat, none of its constituent elements have disappeared, but the relations that they had, which made a whole that we could identify as a mouse, have been altered so that we cannot see the mouse anymore.

Whether it is a company or a mouse, from this general systems point of view, the process is the same. An organized entity that can be identified as a whole, having some integrity, is swallowed by another and cannot be identified as this whole anymore.

The concept of an open system explains the threats and opportunities posed by the environment to the organization (Regev and Wegmann, 2004, Regev and Wegmann, 2005, Regev et al., 2009, Regev et al., 2011, Regev and Wegmann, 2011). An open system draws energy from its environment in order to decrease its entropy. Negative entropy (Negentropy) is a measure of order. In a world governed by the second law of thermodynamics, any closed system will move toward positive entropy, i.e. disorder. To maintain order an open system draws energy from its environment. In terms of our discussion above, this means that organizations exchange goods, services, ideas and money with their environment in order to maintain their internal relationships in specific states so that their stakeholders identify them (Regev and Wegmann, 2004, Regev and Wegmann, 2005, Regev et al., 2009, Regev et al., 2011, Regev and Wegmann, 2011). Organizations, therefore, must establish relationships with other organizations (Regev and Wegmann, 2004, Regev and Wegmann, 2005, Regev et al., 2009, Regev et al., 2011, Regev and Wegmann, 2011). These relationships, as we have seen are necessary but also potentially harmful (Regev et al., 2005). Compaq, for example, had to have relationships with its competitors, which opened the door for its acquisition by HP.

To endure, therefore, the organization as much as the animal, must protect itself from threats to its organized whole. Not all of these threats come in the form of a cat or a buyout. The organization must protect itself from many threats, most of which may look benign (consider Amazon's threat to Barnes and Noble or Borders in 1995).

In the next section we explain Cannon's heuristic device, Homeostasis, which explains how this protection is done.

5 HOMEOSTASIS

Homeostasis is a term coined by Walter Cannon, a physiologist, to describe the way a human body and other organized entities maintain constancy in a changing world (Regev et al., 2005, Regev and Wegmann, 2005, Weinberg and Weinberg, 1988). Homeostasis literally means (Weinberg and Weinberg, 1988): "remaining the same."

Weinberg and Weinberg (1988) describe Homeostasis as a heuristic device to think about how states remain constant (i.e. how norms are maintained). They provide the following quote from Cannon (Weinberg and Weinberg, 1988):

> Proposition I In an open system, such as our bodies represent, compounded of unstable material and subjected continually to disturbing conditions, constancy is in itself evidence that

> agencies are acting or ready to act, to maintain this constancy [..]
>
> Proposition II If a state remains steady it does so because any tendency towards change is automatically met by increased effectiveness of the factor or factors which resist the change. [..]
>
> Proposition III The regulating system which determines a homeostatic state may comprise a number of cooperating factors brought into action at the same time or successively. [..]
>
> Proposition IV When a factor is known which can shift a homeostatic state in one direction it is reasonable to look for automatic control of that factor, or for a factor or factors having an opposing effect. [..]

Note that Cannon speaks in very general terms, he takes the example of a body but what he says can be applied to any enduring organization. Hence, Weinberg and Weinberg (1988) note that homeostasis is a very general and useful heuristic device.

Weinberg and Weinberg (1988), give colourful names to Cannon's proposition, arguing that they are so important that they merit memorable names. They identify 5 principles in Cannon's four propositions. The fourth proposition giving two distinct principles. They thus call them pervasiveness, perversity, plait, pilot and polarity principles.

The Pervasiveness principle is a general statement that draws our attention to the fact that in a changing environment, behind every constant state there are mechanisms that act against change. It refers to the ubiquity and never ending nature of regulatory mechanisms. It reminds us that we need to investigate how each entity we observe is maintained constant.

The perversity principle tells us to look for activities that maintain this constancy (Weinberg and Weinberg, 1988). Not only should we look for activities, but we should also expect increased effectiveness of these activities when they oppose change.

The plait principle tells us to look for multiple mechanisms and not stop when we found only one (Weinberg and Weinberg, 1988). A homeostatic system brings together multiple mechanisms, each having a specific state to maintain constant.

The pilot principle makes us look for an automatic control of each mechanism (Weinberg and Weinberg, 1988). This automatic control explains, in part, why we often do not see homeostatic mechanisms. When some state is controlled automatically, by definition, no conscious control is needed. Hence, most homeostatic mechanisms are applied without us even being aware of them. They have been internalized and made tacit (Vickers, 1987). Often, it is only when they fail that we become conscious of them, as shown in (Winograd and Flores, 1986).

The polarity principle makes us look for mechanisms that have opposite effects from one another (Weinberg and Weinberg, 1988). These are mechanisms that counter the counter of change. These opposing mechanisms can be quite confusing. They act against each other in ways that often seem to us to be at odds or to be inconsistent. Their overall effect, however, is to ensure that the state controlled by the homeostatic system does not stray outside the tolerance level, or as Weinberg and Weinberg (1988) call them, the "critical limits."

A homeostatic system does not distinguish right from wrong. It only maintains some state constant. It doesn't care whether maintaining this constancy is good or bad. This is as true for a human body as it is for an organization. Weinberg and Weinberg describe this property of homeostatic systems as (1988): "The same mechanisms that prevent us from being poisoned also prevent us from being medicated." The "right" strategy that may be able to save a company can be effectively diffused by homeostasis. When conditions change and the homeostatic system doesn't, its reaction to change may not be effective. Hence, it is up to an observer to determine whether a given constancy is good or bad, not for the homeostatic system itself.

What we often call learning, is a way to change this constant state (Regev and Wegmann 2005). This means that the homeostatic system needs to create new mechanisms for maintaining this state constant.

The perversity and polarity principles create inconsistency that is often judged by observers to be a bad situation to be corrected but is merely what it takes to maintain constancy.

Despite the complexity of a homeostatic system, it may not always be successful. If all homeostatic systems were always successful, nothing would ever change and everything would last forever. Thus, when some changes occur, the homeostatic system will adapt to them and different mechanisms will be produced to enforce a new constancy. This may result in a new identity for one or more observers.

Because homeostasis is such a ubiquitous phenomenon in enduring organizations and because business modelling is ultimately concerned with creating enduring organizations, homeostasis has a very large applicability to business modelling, In the

next section we outline some aspects that can be used in future business modelling methods.

6 HOMEOSTASIS FOR BUSINESS MODELING

We begin by asking the basic question of what does it mean to be part of an organization, or to be inside an organization. What we put inside the box of an organizational model, such as the one in Figure 1, is not necessarily what is physically contained in it but what is subject to the protection of its homeostatic system (Weinberg and Weinberg, 1988). A company is not necessarily physically contained in one area or building and even if it is its people come and go from the given building. What does it mean then to be a member of a company? As Weinberg and Weinberg (1988) explain, it means being under some protection from some threats. This protection is not absolute. It simply makes it less easy for the inside of the organization (Weinberg and Weinberg, 1988) "to come into equilibrium with the exterior." This in turn means that signals or stimuli from the environment will not be easily transmitted to within the organization. Physical containment is also offers a homeostatic protection.

The business rules, terms of use, conditions of sale, contracts etc. that organizations impose on their suppliers, members and customers are the visible part of homeostasis. They are designed to protect the inside of the organization from what it considers to be unstable relationships inside and outside the organization. Remember that Cannon refers to "unstable material and subjected continually to disturbing conditions." Enduring organizations have multiple mechanisms in place in to enforce these rules, as specified by Cannon's propositions.

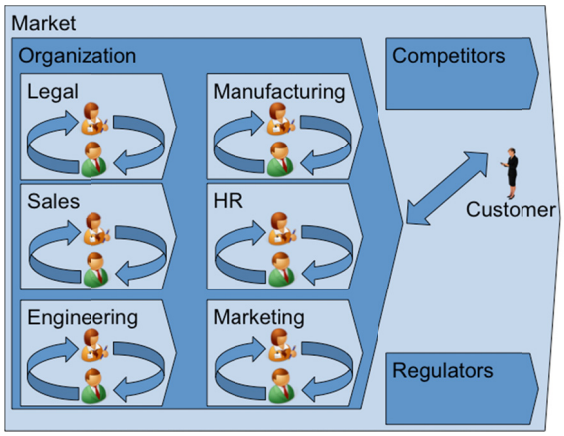


Figure 1: What makes these relationships durable?

These mechanisms will be upheld by different entities within the organization, each having its own homeostatic system. Referring to Figure 1, we can ask what makes each of the entities within the organization lead a somewhat independent existence, what makes it durable. As homeostasis is a heuristic device for understanding continuing existence, we can presume that each entity in a business model has some sort of homeostatic system that keeps it alive. This, in turn, means that some of its activities will not be aligned with the overall organization's needs. Total alignment inside or outside the organization, as sought by Enterprise Architecture methods is not even an idealized situation but a non-desired situation. Because we want our models to be as accurate as possible, it is useful to model these homeostatic mechanisms.

Business modelling is often done with some change in mind. We make a model of the organization as it is today (the as-is model) and the way we want it to be in the future (the to-be model). Modelling the homeostatic mechanisms can be useful in both cases. For the as-is model this would explain what makes the organization or some part of it durable. For the to-be model it would explain how to make it durable.

When creating strategic business models for a company. We need to pay attention to the structure of the company and the way it is maintained through homeostasis. The structure, maintained by homeostasis can either enable the envisioned strategy or defeat it, depending on whether the strategy fits the structure. In Apple's case, Jobs was fortunate that some of the culture (read structure) of Apple pre 1985 still existed. He reinforced it by removing projects and people who did not fit it (a homeostatic mechanism) and proceeded to a strategy reminiscent of 1985. This in itself is a homeostatic reaction because Jobs countered the change that Apple was subjected to from 1985 to 1997 and imposed the same vision and strategy that was there before, albeit with some minor changes. Remember that the new Apple products (e.g. iPhone, iPod, iPad) share some of the basic characteristics of the original 1984 Macintosh, for example, sealed cases, integrated hardware and software, very close attention to design and packaging. This has not happened by chance, but is due to Jobs's constancy being imposed on Apple. Prior to Jobs's return to Apple, Apple was producing computers that accepted expansion cards, which increasingly resembled IBM PCs. This strategy was brought to halt when Jobs's constancy took over.

The homeostatic system's ability to maintain

constancy in the presence of change sometimes has negative implications (from the point of view of some observer). Hence, Jobs unwillingness to allow third party developers to offer applications on the iPhone in order to maintain its integrity and despite extensive lobbying from colleagues could have resulted in serious loss of business opportunities. Again, Jobs agreed to open the platform only when he was convinced that he could control the applications, in itself a research for homeostasis.

We should not forget the polarity principle of homeostasis where homeostatic mechanisms have opposing effects. In Apple's example, Jobs slashed the majority of Apple's products and laid off thousands of people (Isaacson, 2011) in what can be seen as an attempt to defeat Apple's homeostatic system created while he was away.

The perversity principle can explain another action that saved Apple. Jobs convinced Bill Gates, Apple's main competitor, to invest $150M in Apple. Saving a competitor is a way for the homeostatic system to not damage itself by being too successful in moving a state in a given direction. If Microsoft would have been too successful in driving off competitors and Apple would have gone bankrupt, Microsoft would have been more vulnerable to the anti-trust litigation that was already beginning.

In the case of the Swiss healthcare insurer, the reversal in strategy can be explained by the homeostatic system prevailing on the change that is considered unacceptable. The insurer was overwhelmed with the new influx of customers from regions in which it was not traditionally present. It risked lowering its quality standards. By law, it has to have a certain reserve of money for each person insured and it was difficult to have this reserve with a massive influx of customers. All in all, the insurer preferred to get rid of many customers in order to maintain its quality standards and its compliance. This is a typical homeostatic reaction. Thus, the organization separates between customers that it wants to keep and those that it does not, thereby maintaining the states that it deems important (level of quality, reserves) unchanged. The value to these customers may be described as negative. We see that the homeostatic system does not necessarily maximize value for a given customer.

Likewise, insuring a revenue stream drives companies such as banks and mobile phone operators, to provide better service to customers who bring large revenues (premium customers). Just providing value to customers is not the main point. It is rather insuring a steady or steadily increasing revenue stream. Maintaining a steady revenue stream also explains why it is the supplier that usually fixes the price of a good or service. It is rarely the customer who fixes the price. If companies were truly interested in providing value to customers, they would give their products or goods for free or would allow customers to negotiate the price. Similarly, employees do not fix their own salaries so as to maintain the profit of the company. When the revenue or profit do decline below expectations (below what the homeostatic system defines as acceptable) many actions will be taken at the same time or successively, as described by the plait principle, in order to reduce cost, increase sales, increase research and development, warn shareholders to lower their expectations, freeze hiring, renegotiate credit, layoffs etc. Some of these actions may ignite other actions from other homeostatic systems, such as strikes and demonstrations by employees, intervention by political authorities, and the like.

The obliviousness of homeostatic systems for the goodness or badness of the constancy they maintain often results in frustration by change agents. For a homeostatic system, every change is a threat, not an opportunity. An opportunity is necessarily a change to a state kept constant by the homeostatic system and is therefore an unwelcome occurrence.

Finally, taking homeostasis seriously is to accept inconsistencies rather than seeking alignment. From a homeostasis perspective inconsistency can be seen from the polarity and perversity principle perspective as a necessary mechanism to insure survival.

7 CONCLUSIONS

Business modelling methods take the underlying organization that is supposed to carry out the strategy defined in the business model for granted. They assume that the organization will either follow the defined strategy or that it can be engineered to fit the strategy. In essence they consider that the organization has an infinite capacity to change. This is overlooking the everyday observation that any organization that has been in existence for even a few years has built some very strong mechanisms that resist change.

Any surviving organization has adapted to a specific environment. It has built a fit (or congruence) between its environment (customers, regulators, investors, competitors) and its internal structure. Changing this internal structure to fit a different environment is quite difficult. Without

taking this aspect into consideration, the probability of successfully implementing a new business model is very low. Business modelling must take this into account. Homeostasis is a heuristic device that provides a plausible explanation to the way organizations resist change in order to maintain their identity and therefore survive in a changing environment. We have shown that homeostasis can explain both formulation of Business Models (how to deliver and capture value) and the operational part (how the strategy is carried out).

Modelling homeostasis does not mean that we consider that change is impossible, only that change is very hard to create and maintain. To institute change, the homeostatic system first must be neutralized. This is very hard to do because of Cannon's four propositions. However hard it is, resistance to change can have very good reasons that need to be investigated.

Weinberg and Weinberg (1988) point out that Cannon doesn't speak of goals and targets but rather about constancy. A homeostatic system, therefore, has no specific goal or target. It simply maintains some constancy with whatever number of mechanisms it can bring to bear. If we want to take homeostasis seriously, being that it provides such a good explanation of organizational life (and even life in general), we need to overcome our own homeostatic system and remove the terms goals, targets, purpose, ends etc. Rather we need to search for constancy and how it is maintained. This can be a radical change in business modelling, a change that its own homeostasis may be unwilling to allow.

This work should be followed by a more humanistic view in business modelling, modelling people and their attitude toward change rather than the traditional role, business rule, business process paradigm.

REFERENCES

Chozick, A., 2012. As Young Lose Interest in Cars, G.M. Turns to MTV for Help. *New York Times, March 22, 2012*.

Dietz, J.L.G., 2006. The Deep Structure of Business Processes. *Communications of the ACM* Vol. 49, No. 5.

Fritscher, B., Pigneur, Y., 2011. Business IT Alignment from Business Model to Enterprise Architecture, In *Busital 2011, 6th International Workshop on BUSinness/IT ALignment and Interoperability, LNBIP 83*. Springer.

Gordijn, J., Akkermans, J.M., 2003. Value-based requirements engineering: exploring innovative e-commerce ideas. In *Requirement Engineering*. Vol. 8, No. 2, 114–134, Springer.

Gordijn, J., Yu, E., van der Raadt, B., 2006. e-Service Design Using i* and e3value Modeling, *IEEE Software*. Vol. 23, No. 3.

Hammer, M., 1996. *Beyond Reengineering*, HarperCollins. New York.

Hopwood, B., 2002. *Whatever Happened to the British Motorcycle Industry? The classic inside story of its rise and fall*. Haynes.

Isaacson, W., 2011. *Steve Jobs*. Simon&Schuster.

ISO/IEC 10746-1, 2, 3, 4 | 1995-98. ITU-T *Recommendation X.901, X.902, X.903, X.904. "Open Distributed Processing - Reference Model"*.

Katzenbach, J., 2012. The Steve Jobs Way, strat egy+business, http://www.strategy-business.com/ article/00109?gko=d331b&cid=20120424enews, accessed April 2012.

Kilov, H., 1999. *Business Specifications: The Key to Successful Software Engineering*, Prentice Hall PTR.

Mintzberg, H., Ahlstrand, B., Lampel J., 1998. *Strategy Safary – The complete guide through the wilds of strategic management*. Prentice Hall.

Lankhorst, M.M., Proper, H.A., Jonkers, H., 2009. The Architecture of the ArchiMate Language.In *BPMDS 2009 and EMMSAD 2009, LNBIP 29.* Springer.

Osterwalder, A., Pigneur, Y., 2010. *Business Model Generation*.

Regev, G., Wegmann, A., 2004. Defining Early IT System Requirements with Regulation Principles: the Lightswitch Approach. In *RE'04, 12th IEEE International Requirements Engineering Conference*, IEEE.

Regev, G., Wegmann, A., 2005. Where do Goals Come From: the Underlying Principles of Goal-Oriented Requirements Engineering. In *RE'05 13th IEEE International Requirements Engineering Conference*, IEEE.

Regev, G., Alexander, I. F., Wegmann, A., 2005. Modelling the regulative role of business processes with use and misuse cases, *Business Process Management*, Vol. 11 No. 6.

Regev, G., Hayard, O., Gause, D.C., Wegmann, A., 2009. Toward a Service Management Quality Model. In *REFSQ'09, 15th International Working Conference on Requirements Engineering: Foundation for Software Quality* Springer.

Regev, G., Hayard, O., Wegmann, A., 2011. Service Systems and Value Modeling from an Appreciative System Perspective. In *IESS1.1, Second International Conference on Exploring Services Sciences*. Springer.

Regev, G., Wegmann, A., 2011. The Invisible Part of the Goal Oriented Requirements Engineering Iceberg, *In BMSD 2011, 1st International Symposium on Business Modeling and Software Design.* SciTePress.

Shishkov, B., Foreword, 2011. , *In BMSD 2011, 1st International Symposium on Business Modeling and Software Design.* SciTePress.

Sowa, J. F., Zachman, J. A., 1992. Extending and formalizing the framework for information systems architecture. *IBM Systems Journal* Vol. 31. No. 3.

Spohrer, J, Riecken, D., 2006. Special issue: services science. *Communications of the ACM* Vol. 49 No. 7.

Vickers, Sir G., 1987. *Policymaking, Communication, and Social Learning*, eds, Adams, G.B., Forester, J., Catron, B.L.,Transaction Books. New Brunswick NJ.

Weinberg, G. M., Weinberg, D., 1988. *General Principles of Systems Design*, Dorset House.

Winograd, T., Flores, F., 1986. *Understanding Computers and Cognition: A New Foundation for Design*, Ablex. Norwood, NJ.

Zachman, J. A., 1987. A framework for information systems architecture. *IBM Systems Journal* Vol. 26. No. 3.

Business Process and Motivation of Objectives in One Model

Ella Roubtsova
Open University of the Netherlands, Valkenburgerweg 177, 6419AT Heerlen, The Netherlands
Ella.Roubtsova@ou.nl

Keywords: Business Process, Business Objectives, Protocol Model, Motivation Model.

Abstract: The Object Management Group predicts that the Business Process Modelling Notation will be eventually merged with the Business Motivation Model to be implemented in integrated tool suites. However, conventional modelling semantics have asynchronous semantics and therefore have difficulties to accommodate motivation of objectives specified on the basis of synchronous semantics. This paper shows how Protocol Modelling semantics can be used both for business process modelling and motivation modelling corresponding to objectives. Protocol Modelling uses synchronous composition and this synchronization gives to Protocol Modelling the expressive means needed to accommodate motivation of objectives and business processes in one model.

1 INTRODUCTION

Goals, objectives and motives are very important parts of system specification. Goals are usually formulated as non-functional requirements. They are abstract. The goals can be even unrealizable. The objectives corresponding to goals are specific and measurable. They show realisability of goals. Presentation of goals, objectives and motives in business process specification can be seen as transformation of goals into the corresponding objectives and motives expressed as elements of business processes. Such a transformation is a way to estimate realisability of goals.

The need of combining goals and business process modelling standards is emphasized by the Object Management Group and the Business Rules Group. They predict that ”eventually specifications such as the Business Process Modelling Notation (BPMN) together with the Business Motivation Model (BMM) should be merged into a single business-oriented modelling architecture, and implemented in integrated tool suites” (OMG, 2010; BRG, 2010).

In this paper we relate the BMM with business processes and show how business modelers can benefit from modelling of motivation of objectives.

The structure of the paper is the following.

Section 2 presents elements of the Business Motivation Model (BMM).

Section 3 formulates semantic problems of combining goals and business processes in one model identified in related work.

Section 4 formally presents the semantic basis for motivation modelling.

Section 5 shows how the Protocol Modelling semantics can accommodate motivation of objectives and business process in one model.

Section 6 describes applications of motivation models as future work and concludes the paper.

2 BUSINESS MOTIVATION MODEL

The BMM provides a structure for developing, communicating, and managing business plans. The structure covers four related elements:

- The *Ends* of a business plan.
  “Among the *Ends* are things the enterprise wishes to achieve, for example, *Goals* and *Objectives*” (BRG, 2010).

- The *Means* of a business plan.
  “Among the *Means* are things the enterprise will employ to achieve the *Ends*, for example, *Strategies*, *Tactics*, *Business Policies*, and *Business Rules*”.

- “ The *Influences* that shape elements of a business plan”.

- "The *Assessments* that are made about the impacts of such *Influencers* on *Ends* and *Means* i.e., *Strengths, Weaknesses, Opportunities, and Threats*."

The OMG predicts that "three types of people are expected to benefit from the BMM: developers of business plans, business modellers, and implementers of software tools and repositories".

The BMM is not a full business model and it does not prescribe in detail business processes, workflows and business vocabulary. However, business processes are key elements of business plans and the BMM does include a placeholder for Business Processes. The relations between *Goals* and other elements of BMM are left open.

3 GOAL MODELLING

Goal modelling has its roots in the well known requirements engineering approach KAOS (Knowledge Acquisition in autOmated Specification) (Dardenne et al., 1993). Goals are specified in Linear Temporal Logic and organized using the AND and OR refinement structures.

Van at al (Van et al., 2004) proposed goal-oriented requirements animation. The modelling formalism is the UML State Machines that are generated from the goal specifications and called Goal State Machines (GSMs). A GSM contains only transitions that are justified by goals. The GSMs receive events through event broadcast. A GSM that can't accept an event in its current state keeps it in a queue. These events will be submitted to GSMs internally. This means that the composition of GSMs contains extra states that cannot be composed from the states of separate GSMs. Therefore the GSMs cannot be seen as purely goal models as they also deal with the events from the queues.

The User Requirements Notation (URN) (ITU, 2008) is a standard that recommends languages for software development in telecommunication. The URN consists of the Goal-Oriented Requirements Language (GRL), based on i* modelling framework (Yu, 1995), and Use Case Maps (UCM) (Alsumait et al., 2003), a scenario modelling notation. The GRL provides a notation for modelling goals and rationales, and strategic relationships among social actors (Yu et al., 2001). It is used to explore and identify system requirements, including especially non-functional requirements. The UCM is a convenient notation to represent use cases. The use cases are selected paths in the system behaviour and they can be related to goals by developers. The goals are used to prioritize some use cases. If a use case presents alternative behaviours or cycles, then the goals prioritize alternatives. The use cases can be simulated. However, use cases do not model data and the state of the system and they present only selected traces. This means that behaviour model as well as the motivation model shown by use cases are incomplete and cannot guarantee the achievement of goals in the whole system.

Letier at al. (Letier et al., 2008) derive event-based transition systems from goal-oriented requirements models. Then the operations are derived from goals as triples of domain pre-conditions, trigger-conditions and post-conditions for each state transition. The declarative goal statements are transformed into the operational model. To produce consistent operational models, a required trigger-condition on an operation must imply the conjunction of its required preconditions. The problems of goal-oriented approaches are mostly caused by different semantics used by process modelling and goal modelling techniques. Letier at al (Letier et al., 2008) explained that the operational specification and the KAOS goal models use different formalisms. KAOS uses synchronous temporal logics that are interpreted over sequences of states observed at a fixed time rate. The operational models use asynchronous temporal logics that are interpreted over sequences of states observed after each occurrence of an event. Temporal logic operators have very different meanings in synchronous and asynchronous temporal logics. Most operational formalisms have the asynchronous semantics. Letier at al. (Letier et al., 2008) admit that in order to be semantically equivalent to the synchronous KAOS models, the derived event-based models need to refer explicitly to timing events or include elements of synchronization.

4 SEMANTICS FOR MOTIVATION MODELING

The need of synchronization is not the only one semantic need to direct business processes to objectives. Let us identify the necessary semantics in a state transition system.

We take a state transition system which is usually presented as a triple of

$$P = (S, A, T), where$$

- S is a finite set of states $\{s_1, ...s_i, ...s_j ...\}$,
- A is the alphabet of P, a finite set of environmental actions ranged over $\{a, b, ...\}$,
- T is a finite set of transitions (s_i, a, s_j).

The semantics of a transition contains two relations (Milner, 1980):

1. $C \subseteq (A \times S)$ is a binary relation, where $(a,s) \in C$ means that action a is a possible action for P when in state s. C is called the *can-model* of P because it models the actions that P "can do" in each state.
2. U is a total mapping $C \rightarrow S$ that defines for each member of C the new state that P adopts as a result of the action. $U(a;s_i) = s_j$ means that if P engages in action a when in state s_i it will then adopt state s_j. U is called the *update-model* of P because it models the update to the state of P that results from engagement in an action.

With separation of the can- and update-models a process P is a tuple:

$$P = (S;A;C;U).$$

There are always states in the process where particular goals are achieved. Let us name them the goal-states. From the goal perspective, the actions, leading to a goal-state, are the priority actions or wanted actions in the states preceding the goal-state. So, a state preceding a goal-state and the action that is on the path may lead to the goal-state, form a new binary relation:

- $W \subset (A \times S)$, $(a;s) \in W$ means that action a is a wanted action for P when in state s. We call relation W the *want-model* to show its semantic difference from the relation C.

In order to model motivation we propose to add the want-model W to the process:

$$P = (S;A;C;U;W).$$

The can- and want-models of a process are independent of each other, so when a process is in a given state, an action can have different combinations of can- and want- alternatives:

$$\{can\ happen;\ can\ not\ happen\} \times \{wanted; not\ wanted\}$$

Usually $W \subseteq C$ and W is included into the process model. However, the new goals emerging in the life cycle of the modeled system may challenge the process and may need actions that do not belong to the alphabet A.

In this paper we base the modeling of motivation on this extra relation W added to the process.

A service may have several (n) goals. In this case several want-models should be taken into account

$$P = (S;A;C;U;W_{G1},...,W_{Gn}).$$

The goals can be OR-composed or AND-composed (Pohl and Rupp, 2011).

In real systems, some goals can be conflicting. For instance, information goals may conflict with security and privacy goals. Wishes of different user roles may also conflict. Two goals are conflicting if the system has a state from which it is impossible to reach a state where both goals are satisfied simultaneously. It is important to identify any conflicting goals and corresponding motivation models as soon as possible in the software life cycle. One of the ways to do this is modelling of motivation corresponding to objectives in process models.

5 PROTOCOL MODELS WITH MOTIVATION MODELS

A Protocol Model is a synchronous CSP parallel composition of protocol machines (McNeile and Simons, 2006). This composition has its roots in the algebra Communicating Sequential Processes (CSP) proposed by Hoare (Hoare, 1985). McNeile (McNeile and Simons, 2006) extended this composition for machines with data.

Protocol Modelling semantics accumulates can- and want- semantics needed for accommodation of objectives in business processes.

We will demonstrate the use of Protocol Modelling for business process and motivation modelling on a simple case study.

An *Insert Credit Card Number* web service can be seen in many electronic booking systems. The behaviour of the service is the following. The user of the service instantiates the service. The user is asked to insert his credit card number and read the privacy conditions of the service. The user may insert the credit card number without reading the privacy conditions and after reading and accepting the privacy conditions. When the user has accepted the privacy conditions, he can rethink and read the statement again. The service can be cancelled before inserting the credit card number.

We recognize two goals for this service (Figure 1), namely,

- to get the credit card number inserted and
- to get the privacy conditions read by the user.

The possibility of service cancelation is yet another concern. It is obvious that cancelation cannot be called a goal of the service.

5.1 Business Process

After the identification of the goals the KAOS approach suggests to identify objects, agents, entities and operations.

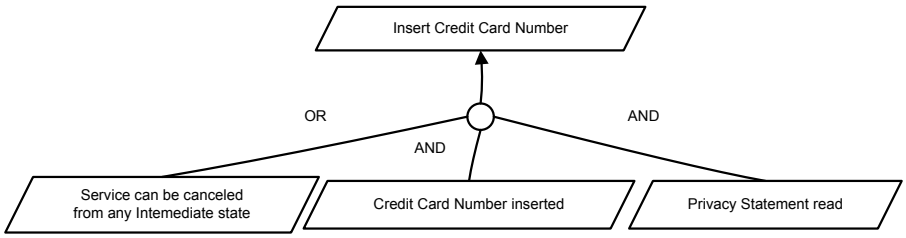


Figure 1: Goal Model.

Using Protocol Modelling we also identify entities, objects, agents and yet aspects but all of them are presented as protocol machines.

For example, we model the can-update process of the *Insert Credit Card Number* web service as a CSP composition of protocol machines `Input`, `Decision` and `Cancelation`. These protocol machines correspond to formulated goals and the cancelation requirement. Figure 2 shows the graphical presentation of our model. The executable Modelscope metacode is shown in Figure 2. The metacode is the complete artefact. As we show later the graphical form does not contain all the modelling constructs of protocol models.

It is not always the case that one requirement is mapped onto one protocol machine. However, the compositional protocol machines allows for any ways of decomposition.

The protocol machine `Input` describes behaviour of an `OBJECT` of type `Input`. Each object has its identification name.

The protocol machines `Decision` and `Cancelation` specify `BEHAVIOURS`. They do not have own identification name and included into each instance of object `Input`. The Include relations are shown between protocol machines depicted as arcs with half-dashed ends.

A human interacts with the service (and with a protocol model) by submitting events. Each protocol machine has an alphabet of recognized events. The events recognized by protocol machines are specified as types. Each type is a data structure. Each instance of an event type contains own values of specified types.

For example, each instance of event `Insert` contains own identifier `Input:Input` and `Credit Card Number: Integer` (Figure 2). All three machines are synchronously instantiated accepting event `Instantiate`. Generic `Finalize` is an alias of events `Insert` and `Cancel`.

The generic interface generated from the model by the Modelscope tool shows to the user the state and the possible events at any execution step. The advantage of using Protocol Machine with goal models is that we result in an executable model of identified objects, agents and entities and can test achievement of chosen goals. We are able to identify the goal states. Not all scenarios of system behaviour lead to the goal states. Protocol machines present all life cycle scenarios of objects, agens and entities. The execution of the protocol models allows for testing realizability of goals and completing the incomplete or imprecise requirements.

Similar to a state machine, a protocol machine has a set of states and the local storage presented with attributes. However, the semantics of a protocol machine is different.

- A transition label of a state machine presents the pre-condition and the post-condition for enabling event to run to completion. A transition from state s_1 to state s_2 is labeled by

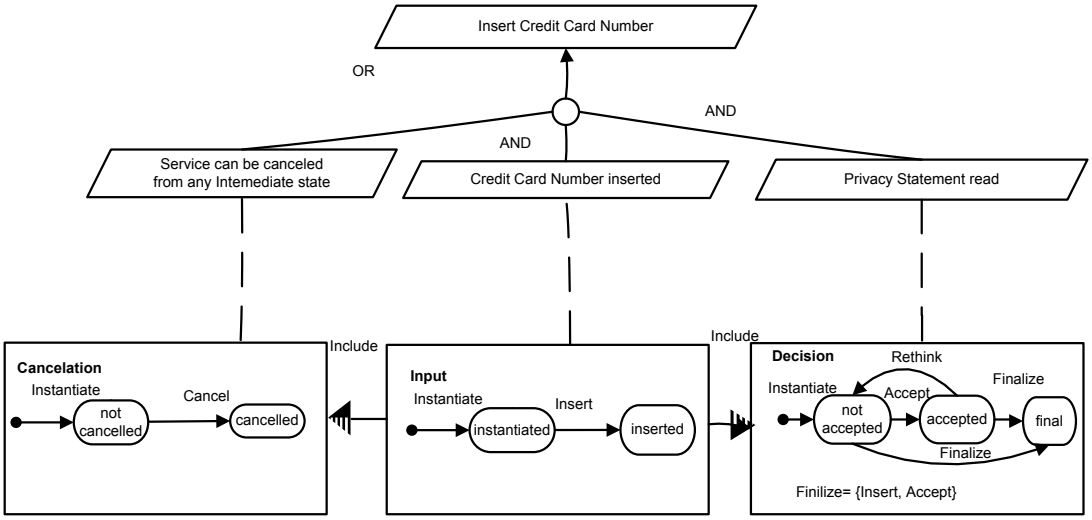
  $(s_1, [precondition]\,event\,/\,[postcondition], s_2)$ (OMG, 2003).

  The label shows that the transition in a state takes place only if the pre-condition is satisfied. If the pre-condition is not satisfied, the behaviour is defined by the semantic rules. Namely, the event is kept in a queue and waits for a state change to fire the transition.

- A transition label of a protocol machine presents an *event* that causes this transition. The storage information is localized in the state. Being in a quiescent state in which the protocol machine can accept the submitted event, the protocol machine accepts one event at a time and handles it until another quiescent state. If the protocol machine cannot accept the event in its current state, the event is *refused* (McNeile and Simons, 2006; McNeile and Roubtsova, 2009).

The default type of protocol machines is `ESSENTIAL`. Essential protocol machines are composed (synchronized) using the CSP parallel composition and these machines are used to present the can-update-model, the business process.

The CSP parallel composition means that a Protocol Model accepts an event if all the protocol machines recognizing this event accept it. Otherwise the




```
1   MODEL InsertCreditCardNumber
2   OBJECT Input
3       NAME Session
4       INCLUDES Decision, Cancelation
5           ATTRIBUTES Session: String, Card Number: Integer
6       STATES instantiated,inserted
7       TRANSITIONS @new*Instantiate=instantiated,
8                   instantiated*Insert=inserted
9
10  BEHAVIOUR Decision
11      STATES instantiated ,not accepted, accepted, final
12      TRANSITIONS @new*Instantiate=not accepted,
13                  not accepted*Accept=accepted,
14                  accepted*Rethink=not accepted,
15                  accepted*Finalize=final,
16                  not accepted*Finalize=final
17  BEHAVIOUR Cancelation
18      STATES not cancelled, cancelled
19      TRANSITIONS @new*Instantiate=not cancelled,
20                  not cancelled*Cancel=cancelled,
21
22  EVENT Instantiate
23      ATTRIBUTES Input:Input, Session:String,
24  EVENT Insert
25      ATTRIBUTES Input: Input, Credit Card Number: Integer,
26  EVENT Accept
27      ATTRIBUTES  Input:Input,
28  EVENT Rethink
29      ATTRIBUTES  Input:Input,
30  EVENT Cancel
31      ATTRIBUTES  Input:Input,
32  GENERIC Finalize
33      MATCHES Insert, Cancel
34
```

Figure 2: Goal Model and Can-Update Protocol Model.

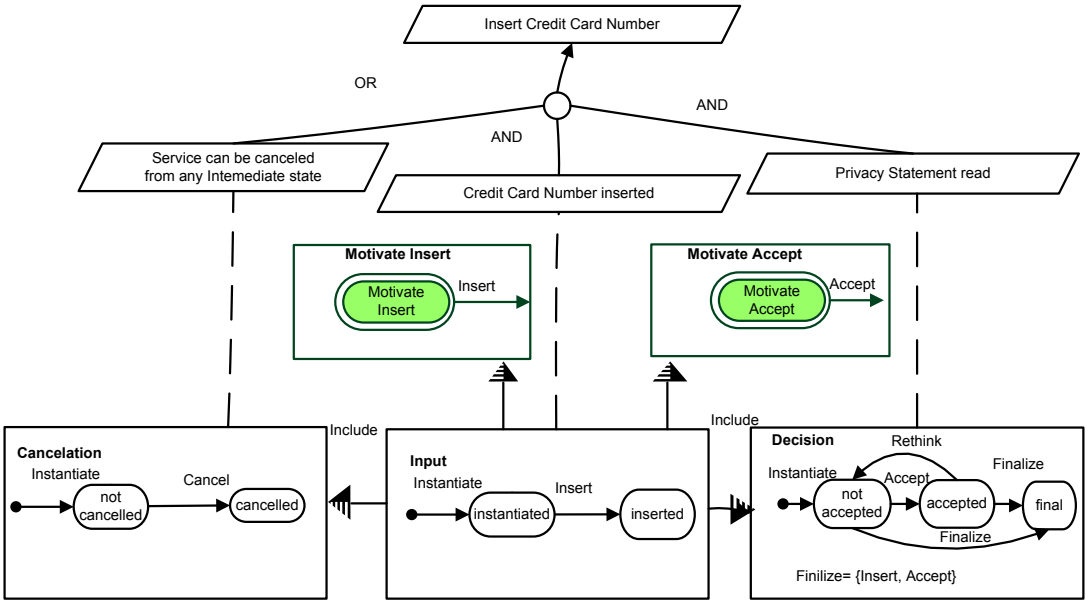


Figure 3: Goals, Protocols and Motivation.

event is refused.

A protocol model accepts one event at a time and do not accepts any other event until it achieves the quitrent state. The results of this semantics are two important distinct properties:

- the state of a protocol model at any moment is a composition of state of protocol machines;
- the behaviour of any protocol machine is preserved in the whole protocol model and it is possible to reason locally on protocol machines about behaviour of the whole model.

5.2 Semantic Elements of Protocol Modelling for Motivation Modelling

There are some other semantic properties of Protocol Modelling for modelling of objectives and separation them from the can-update-model.

1. Ability of protocol machines to read but not modify the state of other protocol machines and to have an associated state function. This property makes it possible to build protocol machines with derived states. A *derived state* is a state that is calculated from the states of other machines using the state function associated with the protocol machine.

2. Different types of protocol machines are used to changes the use of CSP composition. The protocol machines of type `ESSENTIAL` are composed (synchronized) using the CSP parallel composition technique and these machines are used to present the can-update-model of the business process. The protocol machines of type `DESIRED` are not composed using the CSP parallel composition technique. These machines can be used to model the wanted behaviour or motivation.

5.3 BMM Elements in Protocol Models

The elements of the BMM can be mapped onto protocol Models.

Ends or objectives are achieved in particular goal states.

Means or *Strategies* of the *Business Motivation Model* are events and sequences of events. Events or sequences leading to some chosen goal states form the corresponding motivation model.

Influences are presented as Protocol Machines included into the model. The influences add extra behaviour or constraints.

If an `Influence` is in the model, this means that this `Influence` is `assessed` as important.

5.4 Motivation Model

In this section we add motivation models to the can-update protocol model in order to give to the user of the model the indication of means leading to objectives.

A want-model cannot forbid any transition in the can-update-model and it does not participate in the event synchronization with the can-update-models. Therefore, the want-models are not composed using

```
34
35   BEHAVIOUR !Motivate Insert
36   TYPE DESIRED
37       STATES motivate insert, other
38       TRANSITIONS motivate insert*Insert=@any
39
40   BEHAVIOUR !Motivate Accept
41   TYPE DESIRED
42       STATES motivate accept, other
43       TRANSITIONS motivate accept*Accept=@any
44
```

```
1  package InsertCreditCardNumber;
2
3  import com.metamaxim.modelscope.callbacks.*;
4
5
6  public class MotivateInsert extends Behaviour {
7
8      public String getState() {
9
10
11         String y=this.getState("Input");
12         String x=this.getState("Decision");
13              if (y.equals("instantiated")
14              || x.equals("accepted")
15               )  return "motivate insert";
16              else return "other";
17              }
18
19 }
20
```

```
1   package InsertCreditCardNumber;
2
3   import com.metamaxim.modelscope.callbacks.*;
4
5
6   public class MotivateAccept extends Behaviour {
7
8       public String getState() {
9
10                String x=this.getState("Decision");
11              if (x.equals("not accepted")
12               )  return "motivate accept";
13              else return "other";
14              }
15
16
17  }
18
```

Figure 4: Motivation Model.

the CSP parallel composition and have type `DESIRED`.

The motivation models are depicted in Figure 3.

- Protocol machine `Motivate Insert` models motivation for the goal "to get the credit card number inserted".
- Protocol machine `Motivate Accept` models motivation for the goal " to get the privacy conditions

read by the user".

Each of those protocol machines has a derived state and an arc labeled with an event. The arc leads to any state allowed by the can-update-model. This structure is presented in the metamodel. Behaviours presented by lines $35-43$ in Figure 4) contain transitions described the arcs as transitions with the final state `@any`.

Each behaviour is labeled with an exclamation mark. The exclamation mark shows to the Modelscope tool that there is a call-back java file with the name of the marked behaviour. Each call-back function (lines $1-20, 1-18$ in Figure 4) derives state of the motivation model from the state of the objects and behaviours of the can-update-model. For example the state `Motivate Insert` is derived if `Input` is in state `instantiated` or if the `Decision` is in the state `accepted` (lines 11-15 in class `MotivateInsert`).

5.5 Motivation Model for Composition of Goals

If the goals are OR-composed then achieving any of the goals is the goal and both call-back functions shown in Figure 4 are valid.

Motivation model of the AND-combination of goals should not direct to states where at least one of goals cannot be achieved. In our case, motivation of event `Insert` when the object `Input` is in the state `instantiated` leads to the state where the goal `to get the privacy condition accepted` will never be achieved. Event `Insert` should not be motivated in state `instantiated` and lines 11 and 13 should be deleted from the call-back function in Figure 4. The motivation models will first motivate event `Accept` and then the event `Insert`. The execution steps of the protocol model with the AND combination of motivation models are shown in Figure 5. The green light is given to the motivated events.

Figure 5: Execution of the Protocol Model with Motivation Models.

There are several ways to use motivation models built into protocol models:

- `Generating user interface elements.` Motivation models can be used to generate the elements of the user interface. The wanted event and elements of user interface corresponding to them can be made of different form, color and use another order. In the generic interface of the the Modelscope tool (McNeile and Simons, 2011), the wanted events are presented in green.
- `Reuse of models.` The motivation models open another ways of reuse of business process models for systems with different goals.
- `Analysis of consistency and adequate completeness of requirements.` Motivation of goals in models stimulates analysis of realizability of goals and identification of contradicted goals and requirements. The psychological stud-

6 DISCUSSION AND FUTURE WORK

This paper has shown the expressive means of Protocol Modelling allowing combination of business processes and motivation of objectives in one model. It is the synchronous composition semantics of Protocol Modelling and the ability to derive states makes the combination possible.

ies show that people tend to think contextually. Execution of requirements presented in protocol machines is better understood by users than the result of the formal methods applied on operation models. Execution of requirements provides better chance for recognizing of inconstant or incomplete requirements.

- `Composition of business processes on the basis of matching motivation model.` Such an approach promises creating effective business processes. This is especially importance in the context of the electronic business where the motivation provided by human is gone and the motivation should be built in to services as an element of service intelligence.

In the future work we plan the projects aimed to investigate analysis of consistency and adequate completeness of requirements and composition of business processes on the basis of motivation models.

REFERENCES

Alsumait, A., Seffah, A., and Radhakrishnan, T. (2003). Use Case Maps: A Visual Notation for Scenario-Based Requirements. *10th International Conference on Human - Computer Interaction, http://wwwswt.informatik.uni-rostock.de/deutsch/Veranstaltungen/HCI2003/*.

BRG (2010). The Business Motivation Model. Business Governance in a Volatile World.

Dardenne, A., van Lamsweerde, A., and Fickas, S. (1993). Goal-directed requirements acquisition. *Sci. Comput. Program.*, 20(1-2):3–50.

Hoare, C. (1985). *Communicating Sequential Processes*. Prentice-Hall International.

ITU (2008). Formal description techniques (FDT). User Requirements Notation Recommendation Z.151 (11/08). http://www.itu.int/rec/T-REC-Z.151-200811-I/en.

Letier, E., Kramer, J., Magee, J., and Uchitel, S. (2008). Deriving event-based transition systems from goal-oriented requirements models . *Automated Software Engineering archiveD*, 15(2):1–22.

McNeile, A. and Roubtsova, E. (2009). Composition Semantics for Executable and Evolvable Behavioural Modeling in MDA. *BM-MDA'09*, pages 1–8.

McNeile, A. and Simons, N. (2006). Protocol Modelling. A Modelling Approach that Supports Reusable Behavioural Abstractions. *Software and System Modeling*, 5(1):91–107.

McNeile, A. and Simons, N. (2011). http://www.metamaxim.com/.

Milner, R. (1980). *A Calculus of Communicating Systems*. volume 92 of Lecture Notes in Computer Science. Springer.

OMG (2003). *Unified Modeling Language: Superstructure version 2.1.1 formal/2007-02-03*.

OMG (2010). Business Motivation Model. Version 1.1.formal/2010-05-01.

Pohl, K. and Rupp, C. (2011). *Requirements Engineering Fundamentals*. Rocky Nook.

Van, H. T., van Lamsweerde, A., and Philippe Massonet, C. P. (2004). Goal-oriented requirements animation. In *RE*, pages 218–228.

Yu, E. (1995). Modelling Strategic Relationships for Process Reengineering. *Ph.D. Thesis. Dept. of Computer Science, University of Toronto*.

Yu, E., Liu, L., and Li, Y. (2001). Modelling Strategic Actor Relationships to Support Intellectual Property Management. *LNCS 2224 Spring Verlag. 20th International Conference on Conceptual Modeling Yokohama, Japan*, pages 164–178.

Positioning the Normalized Systems Theory in a Design Theory Framework

Philip Huysmans, Gilles Oorts and Peter De Bruyn
Normalized Systems Institute
Department of Management Information Systems, University of Antwerp, Prinsstraat 13, 2000 Antwerp, Belgium
{*philip.huysmans, gilles.oorts, peter.debruyn*}*@ua.ac.be*

Keywords: Normalized Systems, Design Science, Design Theory, Design Theory Anatomy.

Abstract: Several frameworks have been proposed to define design science and design theory over the last decades. For this reason, positioning a research stream within both paradigms has become a difficult exercise. In this paper, the Normalized System (NS) theory is positioned within design science and design theory, in particular the design theory framework formulated by Gregor & Jones (2007). Normalized Systems theory has been proposed as a way to cope with the ever increasingly agile environment to which organizations and their software applications need to adapt. NS achieves this evolvability by articulating theorems that modular structures need to comply with in order to be evolvable. The results of positioning NS within the presented framework for design theories show that NS almost fully incorporates all components of the design theory anatomy. An application of NS theory in other fields is also discussed, which confirms the applicability of the anatomy of Gregor & Jones (2007) within other disciplines. By positioning Normalized System theory within design science and design theory, we also believe to contribute to the definition of both fields in this paper.

1 INTRODUCTION

During the last decades, the design science research approach has increasingly become of interest in information systems (IS) research. One of the first acknowledgments of this evolution was articulated by March & Smith (1995), who stated that design research (aimed at developing IT artifacts that solve relevant IS problems) could be argued to be more important within IS research than traditional (natural) science (aimed at understanding IT phenomena). As the relatively new IS research area took shape, most researchers agreed with the importance of design science approach as an alternative to the behavioral research approach. On the content of IS design science however, opinions still differ greatly (Baskerville, 2008). For this reason, giving a clear and agreed upon definition of design science is close to impossible. However, some fundamental properties of the discipline seem to exist nevertheless.

The notion of design science was first formulated in 1969 by Simon, who stated that researchers can achieve their research goals "to the degree that they can discover a science of design, a body of intellectually tough, analytic, partly formalizable, partly empirical, teachable doctrine about the design process" (Simon, 1996, p. 113). March & Smith (1995, p. 253) articulate this idea more specific for IS research, stating that "design scientists produce and apply knowledge of tasks or situations in order to create effective artifacts." The same authors state that ultimate purpose of IS design science research is the formulation of IT artifacts that provide utility in solving IS problems, as opposed to formulating and testing hypotheses in a conceptual framework as in traditional natural science.

Although most researchers agree on these principles, one of the discussion points in defining IS design science is whether or not theory should be included in the design science paradigm. Based on the literature review of Venable (2006), one can clearly detect the polemic between proponents and adversaries regarding the inclusion of theory in design science. Whereas Hevner, March, Park & Ram (2004) are ambiguous on the role of theory, March & Smith (1995) clearly state that theory should be solely reserved for natural science and should not be part of design science (Venable, 2006). Many other authors however disagree and argue theory should in fact be an essential part of design science (Nunamaker, Chen & Purdin, 1990; Walls, Widmeyer & El Saway, 1992; Venable & Travis, 1999; Markus, Majchrzak & Gasser, 2002;

Gregor, 2006; Gregor & Jones, 2007).

Recently Normalized Systems (NS) theory has been proposed as an approach to design evolvable modular structures in software in a deterministic and proven way. While the purpose of this paper is not to define design science or substantiate whether or not design theory is a part of design, we will attempt to position the state of art of the NS research stream within the design science and design theory paradigms. Indeed, as Normalized Systems theory prescribes specific constraints for building IT artifacts in a purposeful way, the research stream seems to be at least closely related to design science and theory. Positioning the stream into current design science methodologies may thus provide us the opportunity to check the approach for its theoretical completeness, while possible arising gaps might identify future research directions. This method is endorsed by Walls, Widermeyer & El Sawy (2004), who state that the notion of an Information System Design Theory can be used as guiding framework to structure design research.

This classification is a challenging exercise as both design science and design theory are not yet decisively defined. In this paper, we will argue that Normalized Systems theory is both a design theory and design science, as it is a proven and normative way of building IT modules (design theory) and it formulates IT artifacts (design science). The point of view on design science and design theory that will be used in this paper, is the distinction expressed by Walls et al. (2004). These authors defined design science different than other authors such as March & Smith (1995) and Hevner et al. (2004) by separating design science from design practice. According to this distinction, design practice is concerned with the creation of instances of artifacts, whereas design science "should create the theoretical foundations of design practice" (i.e. design theories) (Walls et al., 2004, p.48). As this paper will show, Normalized Systems is such a design theory that creates the theoretical foundation for designing evolvable systems.

Positioning Normalized Systems as a design theory will also express the different nature of NS from most of existing design science. According to March & Smith (1995, p. 253) design science should be concerned with creating effective artifacts by producing and applying apply knowledge of tasks or situations. These authors also specify the aforementioned should be the primary goal of design science, "rather than producing general theoretical knowledge" (March & Smith, 1995, p. 253). Normalized Systems theory however uses a theoretical foundation to prove that using Normalized Design Theorems leads to Normalized Systems theory that are stable with respect to the anticipated changes. This alternative approach of Normalized Systems to design science will become clear in the next sections of this paper.

The remainder of this article is structured as follows: in the following section, we discuss the origin and structure of Normalized Systems theory. In the third section, we will position Normalized system theory within the design theory anatomy suggested by Gregor & Jones (2007). In the discussion we will take a look at the results of this evaluation. The anatomy used in this paper will also be placed within its context and the applications of the evaluation will be discussed.

2 NORMALIZED SYSTEMS THEORY

In the current information-intensive and agile business environment, organizations as well as their supporting software applications need to cope with changes in their structure and functionality. The way in which an organization can assimilate these changes, determines its evolvability. On top of these changes, systems are also faced with an ever increasing size and complexity of their structure and functionality. To tackle these two challenges, *modularity* has frequently been suggested to divide complex system in easier to handle subsystems and cope with the evolvability requirement by allowing the modules to change independently (Baldwin & Clark, 2000). In fact, modularity is a core concept from systems theory and has been applied in many different application domains, including software engineering (Parnas, 1972) were multiple modular primitives have been defined for representing *data* (e.g., structures), *actions* (e.g., procedures) or both (e.g., classes) as the core of information systems. However, (hidden) coupling between those modules tends to limit the anticipated evolvability (De Bruyn & Mannaert, 2012). Thoroughly based on concepts and reasoning from systems theory and thermodynamics, Normalized Systems theory (NS) was recently proposed to strictly guide engineers in devising such *evolvable modularity*. While it was originally applied for software architectures, its relevance for other application domains (e.g., organizational design) has been demonstrated previously (Van Nuffel, 2011; Huysmans, 2011).

First, with its primary aim of enabling evolvability in software architectures, NS defines the occurrence of so-called *combinatorial effects*: changes within the modular structure of which the impact is dependent on the size of the system they are applied to (Man-

naert, Verelst & Ven, 2011, 2012). Such combinatorial effects are clearly undesirable in the context of evolvability. Indeed, as a system grows throughout time, a combinatorial effect would imply that the effort required to implement a same kind of change becomes ever more complex as time goes by. Moreover, the concept of *stability from systems theory* is highly related to this reasoning and offers clues in how to avoid combinatorial effects. In systems theory, stability is regarded as an essential property of systems and implies that a bounded input function should always result in a bounded output function, even if an unlimited time period $T \rightarrow \infty$ is considered. Applied to information systems, this means that a bounded set of changes (selected from the so-called *anticipated changes* such as "add additional data attribute" or "add additional action entity") should result in a bounded impact to the system, even for $T \rightarrow \infty$ (i.e., an unlimited systems evolution is considered). Consequently, stability reasoning confirms that the impact of changes should not be dependent on the size of the system, but only related to the kind of change performed (and hence, combinatorial effects should be avoided). As such, normalized systems are defined as systems exhibiting stability and lacking any combinatorial effects towards a defined set of anticipated changes (Mannaert et al., 2011, 2012).

In order to obtain such normalized systems, NS theory proposes four *theorems* which should be systematically adhered to (Mannaert et al., 2012):

- *Separation of Concerns*, stating that each change driver (concern) should be separated from other concerns;
- *Action Version Transparency*, stating that action entities should be updateable without impacting their calling action entities;
- *Data Version Transparency*, stating that data entities should be updateable without impacting their calling action entities ;
- *Separation of States*, stating actions in workflow should be separated by state (and called in a statefull way).

For each of these theorems it has been formally proven that any violation against them at any time will result in a combinatorial effect (Mannaert et al., 2012). Consequently, if the aim is to obtain a true normalized system, these principles for building software architectures should all be consistently applied. In reality, the systematic isolation of all concerns prescribed by the theory, result in a very fine-grained modular structure. While exhibiting a proven degree of evolvability, the structure in itself may be regarded as complex in the sense that the system becomes an aggregation of many fine-grained instantiations of the modular primitives offered by the employed programming language, many more than in commonly developed software applications.

Therefore, a set of *elements* were proposed to make the realization of normalized systems more feasible. These elements are each a structured aggregation (i.e., encapsulation) of the available software primitives and together providing the core functionality of information systems, be it in a highly generic and reusable way. As such, the internal structure of these elements could be considered to be a set of design patterns regarding higher-level modular building blocks, adhering to the above-described theorems. These five elements are (Mannaert et al., 2011):

- *action element*, a structured composition of software constructs to encapsulate an action construct into an action element;
- *data element*, a structured composition of software constructs to encapsulate a data construct into a data element;
- *workflow element*, a structured composition of software constructs to create an encapsulated workflow element (representing a sequence of action elements);
- *trigger element*, a structured composition of software constructs to create an encapsulated trigger element (controlling for and representing the activation of a workflow or action element)
- *connector element*, a structured composition of software constructs to create an encapsulated connector element (allowing the stateful connection of data elements with external systems).

Each of these elements are described in more detail in Mannaert et al. (2011) and illustrated to be able to cope with a set of anticipated changes in a stable way. Normalized systems are then typically implemented by creating a set of *n* instantiations of the five elements.

Additionally, recent research efforts have demonstrated that reasoning based on the concept of *entropy from thermodynamics* (1) supports and (2) further extends NS theory (Mannaert, De Bruyn & Verelst, 2012a). More specifically, the definition of entropy as employed in statistical thermodynamics was used for this purpose, i.e., the number of microstates consistent with the same macrostate (Boltzmann, 1995). Applied to information systems, *microstates* are operationalized as binary values expressing the correct or erroneous processing of a programming language construct, while the *macrostate* is associated with loggings or database entries representing the correct or erroneous processing of the software system.

Hence, when it is uncertain which microstate configuration brought about the observed macrostate, entropy (uncertainty) is present in the system. The larger the number of microstates consistent with the same macrostate, the higher is the amount of entropy in the system. In Mannaert et al. (2012a), it was further illustrated that the aim of minimizing or controlling entropy in modular system, highly coincides with the four theorems explained above. Moreover, a set of of two new theorems were suggested based on this entropy reasoning (Mannaert et al., 2012a):

- *Data Instance Traceability*, requiring each version and values of an instance of a data structure to be tracked;
- *Action Instance Traceability*, requiring each version and thread of an instance of a processing structure to be tracked.

Finally, several real-life implementations of NS applications have been successfully deployed up to this moment. Some of them have been already briefly discussed in Mannaert et al. (2011).

3 CLASSIFICATION OF NORMALIZED SYSTEMS THEORY

In this section, Normalized Systems theory will be classified within a design theory framework. The Information System Design Theory (ISDT) framework that will be used in this paper is the design theory anatomy proposed by Gregor & Jones (2007). This anatomy is an extension of the ISDT framework proposed by Walls et al. (1992), whose sources date back to the work of Dubin (1978) on theory of the natural science type and the work of Simon (1996) on sciences of the artificial. In their work, Walls et al. (1992) attempted to formulate a prescriptive theory that articulates how a design process can effectively be carried out. Looking back on the formulation of their ISDT, (Walls et al., 2004) conclude that, although used scarcely, it can be helpful as a guiding framework that helps in structuring the "how to" of the design progress based on a theoretical foundation. As the work of Walls et al. (1992) was just an initial attempt to define an Information Systems Design Theory, several reflections and reactions have been published on their work (Gregor & Jones, 2007; Walls et al., 2004). For example Gregor & Jones (2007) clarify that the goal of a design theory can be either a methodology or a product, which was not yet clearly defined by Walls et al. (1992) (Gregor & Jones, 2007). The remainder of this section will show that Normalized System is an example of a design theory defining the design of a product (e.g. a system). Furthermore Gregor & Jones (2007) argue that two structural components of a theory formulated by Dubin (1978) are lacking from the anatomy formulated by Walls et al. (1992), namely "units" and "system states". These constructs are defined as the building blocks of theory and the range of system states that the theory covers respectively (Gregor & Jones, 2007), constructs that will be shown to be part of the Normalized System paradigm in this paper.

Considering these missing constructs, Gregor & Jones (2007) argue that an information system design theory consists of eight components. The first six components are called the Core Components, as they are essential to determine how an artifact can/should be constructed. The other two components can have a positive influence on the credibility of the work, but can be defined later (Gregor & Jones, 2007). The results of the classification of Normalized Systems Design Theory by these components are shown in Table 1. In the next paragraphs, we will discuss the classification of the NS theory within these eight components.

The first component Gregor & Jones (2007) define is the *purpose and scope* of a design theory. This component states "what the system is for, or the set of meta-requirements or goals that specifies the type of system to which the theory applies" (Gregor & Jones, 2007, p. 325). According to the same authors, the environment in which the artifact should operate is an important factor to consider. To understand the purpose and goal of Normalized Systems, it is indeed important to keep in mind the agile environment in which modern systems should operate. As mentioned earlier, the purpose of Normalized Systems theory is the elimination of combinatorial effects (towards a set of anticipated changes), as a means to achieve evolvability of information systems. Combinatorial effects however not only appear in information systems, but can be observed in a very broad spectrum of domains. Therefore Normalized Systems Design Theory is not limited to information systems and, as will be discussed in Section 4.3 of this paper, can apply to many different natural and artificial phenomena. This is in agreement to Gregor & Jones (2007) who state that the applicability of their design theory anatomy is possibly wider than the IS discipline, as it is in itself based on sources from other disciplines.

Constructs are "the entities of interest in the theory", and "are at the most basic level in any theory" (Gregor & Jones, 2007, p. 325). Other authors, such as March & Smith (1995, p. 256), define constructs

Table 1: Classification of Normalized Systems within the anatomy of Gregor & Jones (2007) and the components of Walls et al. (1992).

Gregor & Jones (2007)	**Walls et al. (1992)**	**Normalized Systems theory**
Core Components		
1. Purpose and scope	Meta-requirements	Evolvable software architectures by elimination of combinatorial effects
2. Constructs		Combinatorial effects, modularity, action/data
3. Principles of form and function	Meta-description	Five Normalized Elements Four Normalized Design Theorems
4. Artifact mutability		Anticipated changes
5. Testable propositions	Product hypothesis Process hypothesis	When theorems are applied, no combinatorial effects occur with regard to the anticipated changes
6. Justificatory knowledge	Product kernel theories Process kernel theories	Systems theory (cf. stability) Thermodynamics (cf. entropy)
Additional Components		
7. Principles of implementations	Design method	Supporting applications (e.g., "Prime radiant")
8. Expository instantiation		Real-life NS implementations (by means of instantiations of the elements)

as "the vocabulary of a domain [. . .] used to describe problems within the domain and to specify their solutions". Although the combination of these definitions gives a clear understanding of a construct, the authors believe there is still a certain subjectivity in determining what "vocabulary" or shared knowledge should be considered a construct and what should not. Within the Normalized System theory, the authors recognize three constructs. The first and second construct are generally accepted and known within IT, namely the concept of modularity and the basic building blocks of an information systems: data and actions. The third construct of Normalized Systems theory is combinatorial effects which the NS theory is determined to eliminate, as formulated by Mannaert & Verelst (2009). These three concepts constitute the main constructs of NSDT, although due to the subjectivity of the definition of this component it does seems possible to argue there are other constructs that were not mentioned.

To avoid combinatorial effects, the NS theory states that applications should be structured using the five elements defined by Mannaert & Verelst (2009), which are based on the body of thought of the Normalized Design Theorems. As the *principles of form and function* are defined as "the principles that define the structure, organization and functioning of the design product" by (Gregor & Jones, 2007, p. 325), it is clear the Normalized Elements and Normalized Design Theorems make up this component of the design anatomy. The Normalized design Theorems and Elements specify both structural and functional properties of artifacts by providing guidelines and patterns for constructing instances of the artifacts. Whereas the theorems articulates the general principles that need to be applied, the elements are in fact a design pattern as they specify a possible way of comply with the Normalized Theorems. For the imposed internal structure of the five types of Normalized Elements make applications free of combinatorial effects, the aggregation of these instances of elements subsequently makes up a Normalized System. The Normalized Design Theorems on the other hand can also be considered as a principle of form and function, as they are the guiding principles for constructing the Normalized Elements.

The next component has to do with a special nature of an IS artifact that Gregor & Jones (2007, p. 326) recognize, stating that IS artifacts are very mutable and constantly evolving. They also believe that:

> "the way in which they [i.e. IT artifacts] emerge and evolve over time, and how they become interdependent with socio-economic contexts and practices, are key unresolved issues for our field and ones that will become even more problematic in these dynamic and

innovative times".

Evolvabilty and agility are in fact exactly the goal and purpose of Normalized Systems theory, since Normalized Systems are highly evolvable and stable systems based on structured elements that minimize combinatorial effects. For this reason the component of *artifact mutability*, which is formally defined as "the changes in state of the artifact anticipated in the theory" (Gregor & Jones, 2007, p. 322) can be cleared recognized as the anticipated changes of NS.

The next component Gregor & Jones (2007) define, is *testable propositions*. These are claims or predictions about the quality of a system or tool when the design theory is applied. As such, the testable proposition of NS theory can be formulated as the elimination of combinatorial effects when the principles of form and function are pursued consistently. Although this component seems clearly defined within NS theory, the definition of the component also requires the propositions to be testable. According to Walls et al. (1992), the assertion that applying a set of design principles will result in an artifact that achieves its goal can be verified by building (an instance of) the artifact and testing it. Applying this verification method on Normalized Systems, the proposition of NS theory (the elimination of combinatorial effects) can be verified/tested by building a Normalized system according to the principles of form and function and proving that the system is exempt of combinatorial effects.

Walls et al. (1992) formulated the idea that kernel theories should be part of an Information System Design Theory (ISDT). According to these authors, kernel theories govern both the design requirements and the design process. As Walls et al. (1992) believe, these two aspects should be separated and therefore defined both product kernel theories and process kernel theories. Walls et al. (2004) elucidate the importance of kernel theories for design science, by stating that the design science process uses these theories and combines them with existing artifacts to formulate new design theories. Gregor & Jones (2007, p. 327) however argue that the two types of kernel theories (i.e. process and product kernel theoris) are "a linking mechanism for a number, or all, of the other aspects of the design theory" and should be considered as one component, the *justificatory knowledge* that explains why a design works. This symbiosis is substantiated by the argument that the design process and design product are mostly founded by a single kernel theory (e.g. the justificatory knowledge). They define this component as "the underlying knowledge or theory from the natural or social or design sciences that gives a basis and explanation for the design" (Gregor & Jones, 2007, p. 322). According to this definition, the underlying justification for Normalized Systems theory is twofold. First the central idea of Normalized Systems theory is systems stability, as formulated in the systems stability theory which states that a bounded input should always result in a bounded output (BIBO-principle). In Normalized Systems theory, this is interpreted as the transformation of functional requirements into software primitives (Mannaert et al., 2011). Secondly, Normalized Systems theory shows compatibility with the concept of entropy, as has been discussed in Section 2. Initial research efforts largely validate the use of Normalized System principles when studying the NS theory from the point of view of entropy theory (Mannaert, De Bruyn & Verelst, 2012b).

The next component of a design theory is its *principles of implementation*. Gregor & Jones (2007) consider this and the next components as additional components that are not a no essential part of a design theory but should be formulated if the credibility of the theory is to be enhanced. Concerning this component, we can refer to the model taxonomy of Winter, Gericke & Bucher (2009). According to this taxonomy, Normalized Systems theory should be classified rather as a prescriptive model with result recommendation ("a model") than a model with activity recommendation ("a method"), referring to the clear principles of form and function of NS mentioned earlier. Although the emphasis within NS theory is on the "result view" rather than the "activity view", Winter et al. (2009) argue that both are views on the same "problem solving artifact". This similarity of methods and models is also apparent in the classification used in this paper, in the form of the similarities between the principles of form and function and the principles of implementation. The form or architecture of an artifact can in itself be used as a underlying principle and target on which a method and guidelines for construction of the artifact are based. As opposed to clearly defined principles of form and function, such a formal methodology in the form of a procedure that explicitly articulates the steps that need to be followed to construct normalized elements, does not exist. The formulation of the Normalized elements simply happens while keeping the theorems at the back of one's mind, and is helped by some supporting applications (e.g. "Prime radient"). These applications are more than a tool, as they provide guidelines for constructing Normalized elements. For this reason, they could be considered the principles of implementation of Normalized System theory.

The final component, *expository instantiation*, has two functions: it shows the applicability of a design

theory and it can be used as a way to explain a design (Gregor & Jones, 2007). Instantiated artifacts are the embodiment of this component. Normalized Systems theory has been used in the development of several several software applications. According to the definition of Gregor & Jones (2007), the expository instantiation of NS are these applications and the Normalized Elements they consist of. The seven applications that have been implemented according to the NS principles range from a system for distribution of multimedia to a system responsible for the management of power units on high-speed railroads. Mannaert et al. (2012) describe these implementations in greater detail.

4 DISCUSSION

4.1 Reviewing NS According to the Anatomy

By positioning NS theory within the design theory anatomy of Gregor & Jones (2007), it becomes clear that NS theory incorporates close to all components of the anatomy. One could argue that there still is one significant gap between the anatomy and NS theory: an explicit method to guide the construction of Normalized elements (part of the principles of implementation). However, overall the NS theory shows great similarities with the design anatomy studied in this paper. In conclusion to this strong similarity, we could argue the term Normalized System Design Theory is appropriate to describe the presented theory.

4.2 Design Science vs Design Theory

Although there was opted to compare Normalized System theory to the design theory anatomy of Gregor & Jones (2007) in this paper, the authors acknowledge several other frameworks exist that attempt to define design science and design theory. To put the used anatomy into context, an overview of the different definitions and point of views on design science and design theory will be discussed in this section.

Over the last decades, several definitions and frameworks have been published that tried to formalize and structure IS design science. These publications however show some notable differences in beliefs and definition of IS design science and theory. The work of March & Smith (1995) mainly focuses on the processes and research outputs of IS design science, and emphasizes that theory should not be part of design science. In their opinion, theories are reserved for natural sciences, and design science should "strive to create models, methods and implementations" (March & Smith, 1995, p. 254) On the other hand, authors such as Nunamaker et al. (1990), Walls et al. (1992), Gregor (2006) tried to define this very notion of an IS design theory. It should be mentioned that these authors distinguish a design theory form a theory in natural sciences. According to their definition, a design theory is as a prescriptive theory that "says how a design process can be carried out in a way which is both effective and feasible", in contrast to the "explanatory and predictive theories found in natural or physical sciences" (Walls et al., 1992, p. 37).

Another point of disagreement seems to be the outcome of design science. Some academics (March & Smith, 1995; Hevner et al., 2004; Baskerville, 2008) believe the core of design science is "directed toward understanding and improving the search among potential components in order to construct an artifact that is intended to solve a problem" (Baskerville, 2008, p.441). According to this point of view, design science should therefore be limited to defining artifacts. The work of Walls et al. (1992), Walls et al. (2004) and Gregor & Jones (2007) however suggests design science should also be concerned with building design knowledge in the form of design theories, something that is simply not mentioned by the aforementioned authors. Walls et al. (2004) clearly indicate this difference by distinguishing between design science and design practice. Whereas design practice actually creates instances of artifacts, design science "should create the theoretical foundations of design practice" (i.e. design theories) (Walls et al., 2004, p.48).

Previous differences clearly show that there is an agreement that design science can not be equated with design theory (Baskerville, 2008). To the authors knowledge and feeling, there are however no publications that explicitly and decisively specify the relationship between design theory and design science, nor has there been a discussion between the proponents of both points of view. Therefore, other than claiming NS is both design science and design theory, precisely positioning Normalized Systems theory within design science is simply impossible at this point. The nature of Normalized Systems theory also makes it very difficult to relate NS to the generally accepted conception of IS design science formulated by Hevner et al. (2004). As discussed earlier, Normalized Systems theory is based on proven theorems that deterministically define that combinatorial effects will be eliminated when the four theorems are systematically applied to the construction of elements. According to Hevner et al. (2004) however, design science is typically concerned with designing artifacts

that are evaluated and improved upon and which outcome is a final design artifact that performs better than other artifacts. Although it is clear that Normalized systems theory is in contrast with this design process, NS in its essence still is design science as it is "concerned with devising artifacts to attain goals" (March & Smith, 1995, p. 253). In the authors opinion, this also shows that the current conception of design science is too stringent to allow design theories such as Normalized Systems theory to be positioned within design science. Even the notion of "theory-based design science" (Baskerville, 2008) does not cover the basic principles of Normalized Systems theory, as this type of design science should be concerned with theory testing rather than formulating a theory (which Normalized Systems does).

4.3 Applications of NS

The positioning of NS in the presented framework is an important basis for further research in the context of the NS theory. Within the design science methodology, the framework allows the positioning of at least two research directions. First, existing gaps between the current research results and the framework components can be considered, as discussed in Section 4.1. Since NS was not developed by following a specific design science methodology, it is possible that certain components are missing or insufficiently described. Consequently, this research direction would focus on the original domain of NS, i.e., software. Second, different domains can be researched using the NS theory. As a motivation for this research approach, consider the use of the systems theoretic concept of *stability* and the thermodynamic concept of *entropy* in the justificatory knowledge component in Table 1. The interpretation of such fundamental concepts indicates that the NS theory could possibly be applied to other research fields as well: in itself, these concepts do not originate from the software domain. As discussed, the application of successful solutions from related research fields is an important goal of the design science methodology. As an example of such a research project, we mention how the NS theory has already been applied to the research field of Business Process Management (Van Nuffel, 2011). In a research project in this research direction, one first has to validate whether the purpose and scope of the NS theory can be applied to that research field as well. Consequently, it is crucial to consider the research field as a *modular structure*, where *combinatorial effects* between the modules are a relevant issue. Put differently, one has to apply the constructs defined by the NS theory in the specific research field. For the Business Process Management field, Van Nuffel (2011, p. 89) explicitly mentions: "a business process essentially denotes a *modular structure*, of which the building blocks should be loosely coupled in order to avoid the described *combinatorial business process effects*". Next, the framework prescribes that principles of form and function should be described. In the software domain, NS describes four theorems, which provide the guidelines for the design of five software elements. In the Business Process Management domain, the four principles have been applied to the design of business processes. This has resulted in the formulation of 25 guidelines which describe necessary modular structures to avoid combinatorial business process effects. An example of such a guideline is the "*Notifying Stakeholders*" guideline (Van Nuffel, 2011, p. 143): "Because notifying, or communicating a message to, stakeholders constitutes an often recurring functionality in business processes, a designated business process will perform the required notification". Indeed, omitting to separate this process would result in a combinatorial effect when changes to the notification process need to be applied. Similar to NS on the software level, this insight is not necessarily new: other authors, such as Erl (2008), Papazoglou & Van Den Heuvel (2006) and Cohen (2007) provide similar guidelines. However, these authors do not provide a theoretical framework to motivate the formulation of such guidelines. In order to mature the field towards a design *science*, the authors of this paper believe that such a theoretical framework is vital. Consequently, the proposed framework is important as a methodological guide for future research. Moreover, the framework allows not only to position current research results, but also to identify missing elements. It can be noted that these guidelines can be positioned between the four NS principles and the NS elements: they are an application of the NS principles, but are not sufficient to completely specify process elements. Therefore, additional research is required to further the insight on these guidelines, and arrive at such elements. Consistent with the approach provided in the framework in Table 1, anticipated changes will need to be formulated (as prescribed in the artifact mutability component), and the absence of combinatorial effects with regard to these anticipated changes needs to be demonstrated (as prescribed in the testable propositions component).

5 CONCLUSIONS

In this paper, we made an attempt to position Normalized System theory within design science and de-

sign theory frameworks. Although we argued that NS theory in its essence is design science, we also showed that it does not completely fit in existing design science frameworks. Positioning Normalized Systems within the design theory anatomy of Gregor & Jones (2007) however showed Normalized Systems strongly resembles a design theory. The similarities are to the extent that NS can be formulated as the Normalized Systems Design Theory.

Positioning NS in the anatomy of Gregor & Jones (2007) has shown to present several contributions in this paper.

First, it endorses the applicability of the framework by showing that a theory such Normalized Systems can indeed be soundly positioned within the framework. Furthermore the positioning ratifies the premise of Gregor & Jones (2007) who stated that the possibility of applying the anatomy in other disciplines could be a question for further research. Indeed, this paper showed that it is initially possibly to position the application of NS within Business Process Management within the anatomy. This proves that the design theory anatomy of Gregor & Jones (2007) can indeed be applied in other disciplines than IS design science.

A second contribution is that positioning NS within the anatomy gave the opportunity to check NS for its theoretical completeness, which is consistent with the view of Walls et al. (2004) who state that an ISDT framework can be used as guiding framework to structure design research. The gaps that were identified according to the application of NS within Business Process Management also indicated future research directions.

Finally the authors believe that this papers contributes to the discussion on design science and design theory. It has been shown that Normalized Systems theory is an apparent example of a design theory. As Normalized System theory should however also be considered design science, the authors believe to have contributed to both the definition of design science and the elucidation of the relationship between design science and design theory.

ACKNOWLEDGEMENTS

P.D.B. is supported by a Research Grant of the Agency for Innovation by Science and Technology in Flanders (IWT).

REFERENCES

Baldwin, C. Y. & Clark, K. B. (2000). *Design Rules: The Power of Modularity*. Cambridge, MA, USA: MIT Press.

Baskerville, R. (2008). What design science is not. *European Journal of Information Systems*, *17*, 441–443.

Boltzmann, L. (1995). *Lectures on Gas Theory*. Dover Publications.

Cohen, S. (2007). Ontology and taxonomy of services in a service-oriented architecture. *The Architecture Journal*, *11*, 10.

De Bruyn, P. & Mannaert, H. (2012). Towards applying normalized systems concepts to modularity and the systems engineering process. In *Proceedings of the Seventh International Conference on Systems (ICONS 2012)*.

Dubin, R. (1978). *Theory building, revised edition*. London: Free Press.

Erl, T. (2008). *SOA: Principles of Service Design*. Prentice Hall Service-Oriented Computer Series. Upper Saddle River, NJ: Prentice Hall.

Gregor, S. (2006). The nature of theory in information systems. *MIS Quarterly*, *30*(3), 611 – 642.

Gregor, S. & Jones, D. (2007). The anatomy of a design theory. *Journal of the Association for Information systems*, *8*(5), 312–335.

Hevner, A., March, S., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, *28*(1), 75–105.

Huysmans, P. (2011). *On the Feasibility of Normalized Enterprises: Applying Normalized Systems Theory to the High-Level Design of Enterprises*. PhD thesis, University of Antwerp.

Mannaert, H., De Bruyn, P., & Verelst, J. (2012a). Exploring entropy in software systems: Towards a precise definition and design rules. In *Proceedings of the Seventh International Conference on Systems (ICONS)*.

Mannaert, H., De Bruyn, P., & Verelst, J. (2012b). Exploring entropy in software systems: Towards a precise definition and design rules. In *Proceedings of ICONS 2012: The Seventh International Conference on Systems*, (pp. 93 – 99). IARIA.

Mannaert, H. & Verelst, J. (2009). *Normalized Systems: Re-creating Information Technology Based on Laws for Software Evolvability*. Koppa.

Mannaert, H., Verelst, J., & Ven, K. (2011). The transformation of requirements into software primitives: Studying evolvability based on systems theoretic stability. *Science of Computer Programming*, *76*(12), 1210–1222. Special Issue on Software Evolution, Adaptability and Variability.

Mannaert, H., Verelst, J., & Ven, K. (2012). Towards evolvable software architectures based on systems theoretic stability. *Software: Practice and Experience*, *42*(1), 89–116.

March, S. T. & Smith, G. F. (1995). Design and natural science research on information technology. *Decision Support Systems*, *15*(4), 251 – 266.

Markus, M. L., Majchrzak, A., & Gasser, L. (2002). A design theory for systems that support emergent knowledge processes. *MIS Quarterly*, *26*(3), 179 – 212.

Nunamaker, Jr., J. F., Chen, M., & Purdin, T. D. M. (1990). Systems development in information systems research. *J. Manage. Inf. Syst.*, *7*(3), 89–106.

Papazoglou, M. P. & Van Den Heuvel, W.-J. (2006). Service-oriented design and development methodology. *International Journal of Web Engineering and Technology*, *2*(4), 412–442.

Parnas, D. L. (1972). On the criteria to be used in decomposing systems into modules. *Communications of the ACM*, *15*(12), 1053–1058.

Simon, H. (1996). *The Sciences of the Artificial* (Third edition ed.). Cambridge, MA: MIT Press.

Van Nuffel, D. (2011). *Towards Designing Modular and Evolvable Business Processes*. PhD thesis, University of Antwerp.

Venable, J. (2006). The role of theory and theorizing in design science research. In *DESRIST Proceedings*.

Venable, J. & Travis, J. (1999). Using a group support system for the distributed application of soft systems methodology. In *Proceedings of the 10th Australasian Conference on Information Systems, Wellington, New Zealand*, (pp. pp 1105–1117).

Walls, J., Widmeyer, G., & El Saway, O. (1992). Building an information system design theory for vigilant eis. information systems research. *Information Systems Research*, *3*(1), 36–59.

Walls, J. G., Widermeyer, G. R., & El Sawy, O. A. (2004). Assessing information system design theory in perspective: How useful was our 1992 initial rendition? *Journal of Information Technology Theory and Application (JITTA)*, *6*(2), 43–58.

Winter, R., Gericke, A., & Bucher, T. (2009). Method versus model - two sides of the same coin? volume 34 of *Lecture Notes in Business Information Processing*, (pp. 1–15). 5th International Workshop on Cooperation and Interoperability, Architecture and Ontology, Amsterdam, NETHERLANDS.

The MOF Perspective on Business Modelling

Berend T. Alberts, Lucas O. Meertens, Maria-Eugenia Iacob and Lambert (Bart) J. M. Nieuwenhuis
University of Twente, PO Box 217, Enschede, The Netherlands
b.t.alberts@student.utwente.nl, {l.o.meertens, m.e.iacob, l.j.m.nieuwenhuis}@utwente.nl

Keywords: Business Modelling, Business Models, Meta-business Models, Meta-object Facility.

Abstract: The business model field of research is a young and emerging discipline that finds itself confronted with the need for a common language, lack of conceptual consolidation, and without adequate theoretical development. This not only slows down research, but also undermines business model's usefulness for research and practice. We offer a new perspective on business modelling to address these issues. It looks at business modelling from the perspective of the Meta-Object Facility, emphasising the role of models and meta-models. From this new perspective, a commonality analysis can identify the important classes in business modelling. This new perspective on business modelling helps to create a common language, achieve conceptual consolidation and supports theory development; it addresses issues that hinder business model research.

1 INTRODUCTION: A NEED FOR BUSINESS MODEL THEORY DEVELOPMENT

In general, a business model is a simple and, usually, graphic depiction of a company, often using boxes and arrows. It mostly describes a single company, a group of companies, or part of a company. In the broadest sense, a business model is an abstract (which means simplified) representation of the company, a "model of the business". The business model field of research is strongly growing and maturing over the last decade, mostly since 2000 (Osterwalder, Pigneur & Tucci, 2005; Zott, Amit & Massa, 2011). Since to this date no unified view exists regarding its conceptual foundation, this young and emerging discipline has been described (Meertens, Iacob & Nieuwenhuis 2011) as "finding itself in a state of prescientific chaos", in the sense of Kuhn (Kuhn 1970).

Practitioners using business models have a need for a common language, especially since they come from different disciplinary backgrounds: strategic management, industrial organization, and information systems (Pateli & Giaglis, 2004). In addition, links to other research domains are necessary to establish the business model field as a distinct area of investigation (Pateli & Giaglis, 2004). However, researchers still have to build more on each other's work, and research generally advances slowly and often remains superficial (Osterwalder, Pigneur & Tucci, 2005).

Currently, researchers use different terms to describe similar things, and the same term for different things. *Business model* often means "*a model of a single company*" and, specifically, of the way a company does business, creates, and captures value. However, other things are called business model as well, for example when referring to a pattern in the phrasing "*...the freemium business model...*" In addition, ontologies or frameworks such as the Business Model Ontology (BMO), e3-value, RCOV or activity system are sometimes referred to as a business model too (Osterwalder, 2004; Gordijn, 2002; Demil & Lecoq, 2010; Zott & Amit, 2010). In our research, we refer to such frameworks (BMO, e3-value, RCOV) as *meta-business models*. We define these analogous to meta-models in software or systems engineering (Van Halteren, 2003):

> A *meta-business model* is the set of concepts that is used to create business models. A business model developed from this set of concepts is an instance of the meta-business model.

For example, a meta-business model may define that "*a business model consists of a value proposition, organization, and finances.*" Thus, the

meta-business model lays out the rules for modelling a business model. Consequently, a business model is an instance of the meta-model, following those rules. An example of a meta-business model is the BMO (Osterwalder, 2004), which can serve to make a business model of any company. This business model would be an instance of the BMO. However, the BMO is itself also a model. It is a model for creating business models. As such, it is a "*business model*"-model or, in modelling terms, a meta-"*business model*".

Stimulating researchers to build more on each other's work can be achieved by developing instruments for comparing different meta-business models. This can also help the integration with horizontally related concepts such as strategy and processes (Gordijn, Osterwalder & Pigneur, 2005). "*...A conceptual framework will provide a basis for business model theory development by providing a structure from which researchers can debate, recognize points of agreement and disagreement, identify potential points of integration or linkage along with areas of future research*" (Lambert, 2008). Such a conceptual framework can help to analyse shared or distinctive features of different meta-business models (Lambert, 2008).

Consensus on the theoretical underpinnings of the business model concept has not yet been achieved (Al-Debei and Avison, 2010), which undermines its applicability in different contexts. *"...The business model remains a theoretically underdeveloped (and sometimes overloaded) concept, which may raise doubts concerning its usefulness for empirical research and theory building*" (Zott, Amit & Massa, 2011). For future research, more clarity on the theoretical foundation and conceptual consolidation is necessary (Zott, Amit & Massa, 2011).

The articles referenced above are all review articles, specifically aimed at providing an overview of the status and developments of business model research and the emergence of the discipline. In short, the most important issues are:

- the need for a common language,
- lack of conceptual consolidation, and
- theoretical development of the concept.

These issues relate strongly to the different meta-business models existing separately. Consequently, using different meta-models may result in different business models of the very same organization. This can have severe consequences. For example, if a business model is used in a requirements engineering process, the resulting requirements can vary greatly depending on which meta-business model is used. Unfortunately, because of the gaps in business model research, such problems are hard to address currently.

Another area of research, software and systems engineering, has more experience dealing with a great variety of meta-models, and already addressed the need for a generic framework to manage, manipulate, and exchange these models. This generic framework is the Meta-Object Facility (MOF), created by the Object Management Group (OMG) (1999). The MOF represents a layering of meta-models for describing and representing meta-data: data *about* other data (Van Halteren, 2003). Although it originates from an object-oriented software design domain, the MOF allows the definition of (meta-) models independent of the application domain.

In this paper, we introduce the MOF perspective on business modelling. Introducing a new perspective on business modelling helps identify differences and commonalities of business modelling languages and concepts. We use the MOF to create a *meta-meta-business model* that promotes further theory development. In doing so, we contribute to advancing the discipline of business modelling.

The structure of the paper is as follows. After having presented the background and motivation in this section, section 2 further explains the MOF. Section 3 provides our main contribution: it applies MOF to business modelling. In addition, it provides examples for each of the layers. This includes suggesting a meta-meta-business model and a graphical example of this new model's use. Section 4 discusses further research possibilities with the introduction of MOF in business modelling. Finally, section 5 shows how this addresses the presented issues of business model research.

2 THE META-OBJECT FACILITY (MOF)

The MOF was introduced above as a generic framework for working with a great variety of models and meta-models. This section clarifies the concept. The central idea of MOF is that every model is an instance of some meta-model in an abstract layer above it. Hence, a business model is an instance of a meta-business model. The other way round, every meta-model provides a vocabulary for

creating models; these models are instances in an abstract layer below it. Thus, a meta-business model provides a vocabulary for creating business models.

The account of the MOF given here strictly follows Van Halteren (2003). Modelling data in terms of meta-data can continue indefinitely, in theory, with an infinite number of meta-layers. The MOF is defined as four layers only, M0 to M3, as shown in Figure 1:

- Layer M0 – *instances*: an instance is the flat data, which can describe a running system's state. This data is an instance of elements in the M1 layer.
- Layer M1 – *models*: the model provides the vocabulary for the instance. For example, if the instance is a running system, the model is its source code. The model is itself an instance of the M2 layer.
- Layer M2 – *meta-model*: the meta-model consists of generic elements used for description of the model at the M1 layer. For example, having a system's source code at the M1 layer, the M2 layer is a programming or modelling language such as java or UML. While the M1 layer is an instance of the M2 layer, this layer is again an instance of the even more generic elements of the M3 layer.
- Layer M3 – *meta-meta-model*: the meta-meta-model consists of the elements providing the most generic vocabulary for the M2 layer. For example, the M3 MOF model, can be used to describe a language such as java or UML. While in theory an infinite number of meta-layers exists, for our purpose, we follow the M3 layer as standardized in the OMG MOF specification, also called the *MOF model*.

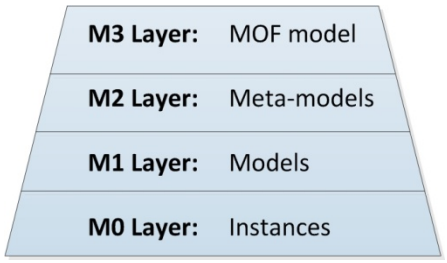


Figure 1: The MOF layers.

The MOF vocabulary comes from the context of object-oriented formalism in software engineering The MOF model itself consists of the following four concepts:

- *Classes*: classes are the primary modelling constructs. These are the central objects that interact with one another. Classes can be organized hierarchically in specializations or generalizations.
- *Associations*: associations are the relations between any two classes. Such a relationship may have a name, cardinality, and type.
- *Data types*: data types are the types used for non-class objects. For example, commonly used data types in the world of programming are integer and string.
- *Packages*: packages are groups of classes and are used to organize models and meta-models. Packages can introduce complex interactions between classes, such as nesting, inheritance, and importing.

The MOF model is a generic meta-meta-model that allows working with a diversity of meta-business models. In using the MOF, ultimately every (meta-) model is defined in terms of classes, associations, data types, and packages. In our attempt to relate business modelling to the MOF, we identify classes only.

3 THE MOF AND BUSINESS MODELLING

This section provides our main contribution: it applies the MOF to business modelling, to create a generic framework for business modelling that provides conceptual consolidation, and helps with a common language and further theory development. The most important reason for using the MOF is the perspective it provides on the practice of modelling.

First, subsection 3.1 shows how the MOF layers encompass the business modelling concepts. Second, subsection 3.2 provides general examples for each of these layers. Third, subsection 3.3 treats the M2 layer. It addresses the issue of which classes should be on this layer. Finally, subsection 3.4 shows several components at the M1 layer.

3.1 Viewing Business Modelling from the MOF Perspective

Applying the MOF layers to business modelling leads to Figure 2. It shows how the MOF layers encompass the concepts of business modelling. It is analogous to Figure 1. Every (business) model is an instance of a meta-model from the above layer. Applying this notion in terms of the MOF layers, as shown in Figure 2, leads to the following layers for business modelling:

- Layer M0 – *business model instance*: the central construct of this research area is a business

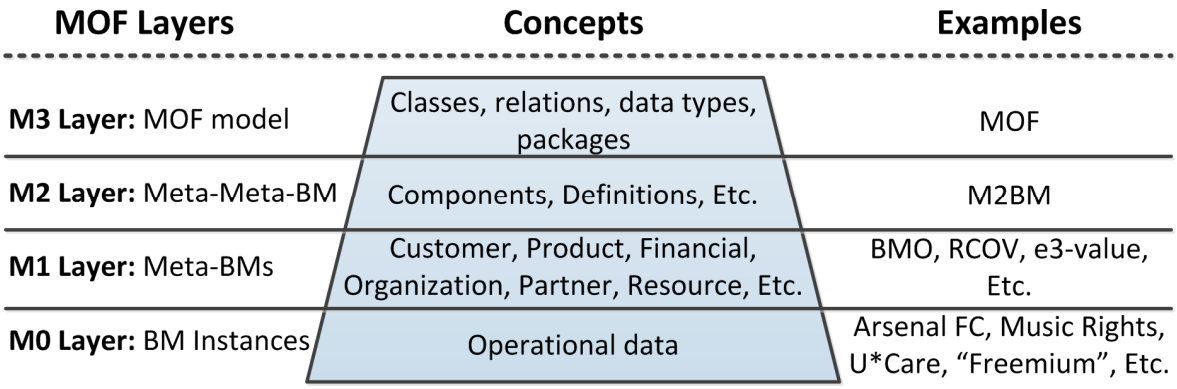


Figure 2: The MOF layers applied to business modelling.

- model instance, which can describe an organization, situation, or pattern. This business model instance is an instance of elements in the M1 layer.
- Layer M1 – *meta-business model*: the meta-business model provides the vocabulary for the business model instance. The meta-business model is itself an instance of the M2-layer. Since the instance data is a model already, the terms change compared to the MOF model. In this case, the *model* from MOF is a meta-business model.
- Layer M2 – *meta-meta-business model*: the meta-meta-business model consists of generic elements used for description of the meta-business model at the M1 layer. While the M1 layer is an instance of the M2 layer, this layer is again an instance of the even more generic elements of the M3 layer.
- Layer M3 – *MOF model*: the MOF model consists of the elements providing the most generic vocabulary for the M2 layer. This is the same model as the top layer of MOF (Figure 1). The MOF model defines every instance in terms of classes, associations, data types, and packages.

The above description shows that the concepts of business modelling and meta-business models fit effortlessly in the MOF layers. This indicates that the MOF is indeed a generic framework, which works for any form of models and meta-models.

3.2 Simple Examples for Each Layer

Starting from the bottom up, many possible examples exist at the M0 layer for business modelling. Business model instances belong in this layer, therefore, any business model that describes an organization, situation, or pattern would fit here. An example of a real life case is U*Care, a service platform for elderly care (Meertens, Iacob & Nieuwenhuis, 2011). Other examples of a business case as business model instances are two models of the clearing of music rights for internet radio stations (Gordijn, Osterwalder & Pigneur, 2005), and modelling of the development of Arsenal FC over a period of eleven years (Demil & Lecoq, 2010). A pattern, such as "freemium", also belongs on the M0 layer (Osterwalder, 2010).

At a higher level of abstraction, the M1 layer contains the meta-business models. They provide the vocabulary for the business model instances. Previously often called frameworks or even ontologies, examples of meta-business models are plentiful. For example, the music rights case is modelled in two different meta-business models, e3-value and the BMO (Gordijn, Osterwalder & Pigneur, 2005). The Arsenal FC case is modelled using the meta-business model RCOV (Demil & Lecoq, 2010). Figure 3 in subsection 3.4 provides more examples, while focussing on their components.

Since this is the first time the M2 layer is recognized in business modelling, nobody has presented examples as such at this layer yet. Following the MOF perspective, the M2 layer contains a meta-meta-business model that provides a vocabulary for meta-business models at the M1 layer. This means that such a meta-meta-business model must consist of generic elements that capture meta-business models, such as the BMO, e3-value, and RCOV. Literature that presents a review of business modelling research, such as Zott, Amit and Massa (2011), suggest those generic elements. In subsection 3.3, we propose classes for a meta-meta-business model (M2BM) that belongs on the M2 layer.

At the top of the pyramid, the M3 layer has only one example in our case. It is the MOF model itself, which we have explained in Section 2 already. It

includes classes, associations, data types, and packages.

3.3 Specifying Classes at the M2 Layer

While an interpretation of business modelling in MOF terminology provides conceptual consolidation, a meta-meta-business model at the M2 layer would provide a common language for business modelling. The meta-meta-business model would be *overarching* the meta-business models. This subsection researches what is necessary to create such an overarching meta-meta-business model.

First, 3.3.1 presents what type of elements should be in the meta-meta-business model. Second, 3.3.2 explains how to obtain these elements. Third and final, 3.3.3 suggests several of these elements in the form of classes.

3.3.1 What Should Be in the Meta-Meta-Business Model?

Business modelling is the act of creating a business model instance; this is an instance of a meta-business model. The instance is a M0 layer model, the meta-business model is a M1 layer concept. Many of these meta-business models exist already, some with a strong link to information systems, others closely related to strategic management or industrial organisation. For example, Vermolen (2010) identified nine such meta-business models published in the top 25 MIS journals. The Business Model Ontology from Osterwalder (2004) was also mentioned previously.

All meta-business models, as M1 models, must follow some sort of guidelines defined at the M2 layer. The generic rules for meta-business model should be defined at the M2 layer as a *meta-meta-business model*: M2BM (M2 both for MOF M2 layer and for meta-meta-). Such a meta-meta-business model does not exist yet; however, as the introduction shows, creating it is exactly what different researchers in the business model discipline are asking for.

The different meta-business models at the M1 layer give the first hint of what this meta-meta-business model looks like. Every model at the M1 layer must be an instance of more generic elements at the M2 layer. The meta-meta-business model must consist of such concepts that it allows the creation of any model that can be regarded as an M1 meta-business model.

The required coverage of M2 classes can be discovered with a commonality analysis amongst different meta-business models. For example, all M1 meta-business models propose some set of *components*, so one of the classes of the M2 meta-meta-business model should be *components*.

Several researchers have in fact performed such commonality analyses. We argue that the abstract meta-meta-business model that belongs on this layer should come from review literature on meta-business models. As a review synthesizes the concepts used in business modelling literature, the resulting concepts can be considered instances of classes from the M3 layer.

3.3.2 Review Literature on Business Modelling

An extensive literature survey identified five articles that can aid us in finding out what classes make up the M2 meta-meta-business model. The method we followed consisted of three steps. The first step was a search on Scopus and Web of Science for relevant articles published between 2000 and august 2011, using two queries:

- in title: “business model*”
- in title-keywords-abstract: “business model*” AND ontology OR ((framework OR e-commerce) AND (design OR analysis))

All results were checked for relevance by analysing the abstract. The second step was an analysis of the articles’ content for relevance, searching for presentation of meta-business models, or review of business model research or literature. The third step was selecting those articles usable for creating the M2 meta-meta-business model. Table 1 presents the resulting five articles.

Table 1: Overview of business model review literature.

Authors	Title	Year
Pateli and Giaglis	*A research framework for analysing eBusiness models*	2004
Gordijn, Osterwalder and Pigneur	*Comparing two Business Model Ontologies for Designing e-Business Models and Value Constellations*	2005
Lambert	*A Conceptual Framework for Business Model Research*	2008
Al-Debei and Avison	*Developing a unified framework of the business model concept*	2010
Zott, Amit and Massa	*The Business Model: Recent Developments and Future Research*	2011

Table 2: Classes for the M2 layer meta-meta-business model.

Pateli and Giaglis, 2004	Gordijn, Osterwalder and Pigneur, 2005	Lambert, 2008	Al-Debei and Avison, 2010	Zott, Amit and Massa, 2011
Definition	Definition	Definition		Definition
	Purpose of the ontology Focus of the ontology Actors using the ontology Other applications	Objective	BM reach BM Functions	Strategic marketing Value creation in networked markets Strategy Innovation
Components	Ontology content and components	Fundamentals (elements)	V4 BM dimensions	Components
Conceptual models	Representation Visualization	Fundamentals (characteristics of representations) Representations (display)		Representations
Change methodology	Change methodology			
Evaluation models	Evaluation methods for business model instances	Operational (measurement)		Firm performance
	Origins			Emergence
Design methods and tools Adoption factors	Supporting technologies Tool support		Modelling principles	
Taxonomies	Classification	Operational (recognition)		Typologies

These five articles present a number of concepts that the authors consider important in business modelling. Pateli and Giaglis (2004) identify eight streams of research in business modelling. Gordijn, Osterwalder and Pigneur (2005) compare the Business Model Ontology and e3-value on a number of criteria. Lambert (2008) creates a business modelling framework based on a conceptual framework from the domain of accounting. Al-Debei and Avison (2010) identify four facets of business modelling. Zott, Amit and Massa (2011) provide the most up to date overview of the state of art of business modelling research.

3.3.3 Classes of the M2BM: A Meta-Meta-Business Model

Table 2 identifies the important concepts in business modelling according to the review literature. It is a first indication of possible classes for the M2BM. A comforting result is that there is quite some overlap in the identified classes. For example, four of the five articles name *definition,* and all have *components*. This allows for mapping of the concepts on to each other to get to a compact list of classes. Already, Table 2 provides an attempt at this. While in some cases this mapping is obvious (as for definition and components), it remains interpretative. As section 4.2 discusses, two concepts were left out of Table 2 deliberately: *ontological role* and *ontology maturity & evaluation*. Both from Gordijn, Osterwalder and Pigneur (2005). The table suggests which classes are important to business modelling.

3.4 Example Use of M2BM: Components of Meta-Business Models

This section presents an example of the M2BM's use in business modelling. The core construct of business modelling is probably the very visible *components* of meta-business models. This example provides a comparison of ten different meta-business models based on their components. It shows how several meta-business models all have their own *instantiation* of the M2BM class *components*. Figure 3 is the result of the comparison (Alberts, 2011).

The first nine meta-business models are those identified by Vermolen (2010), published in the top 25 MIS journals. The tenth has also been

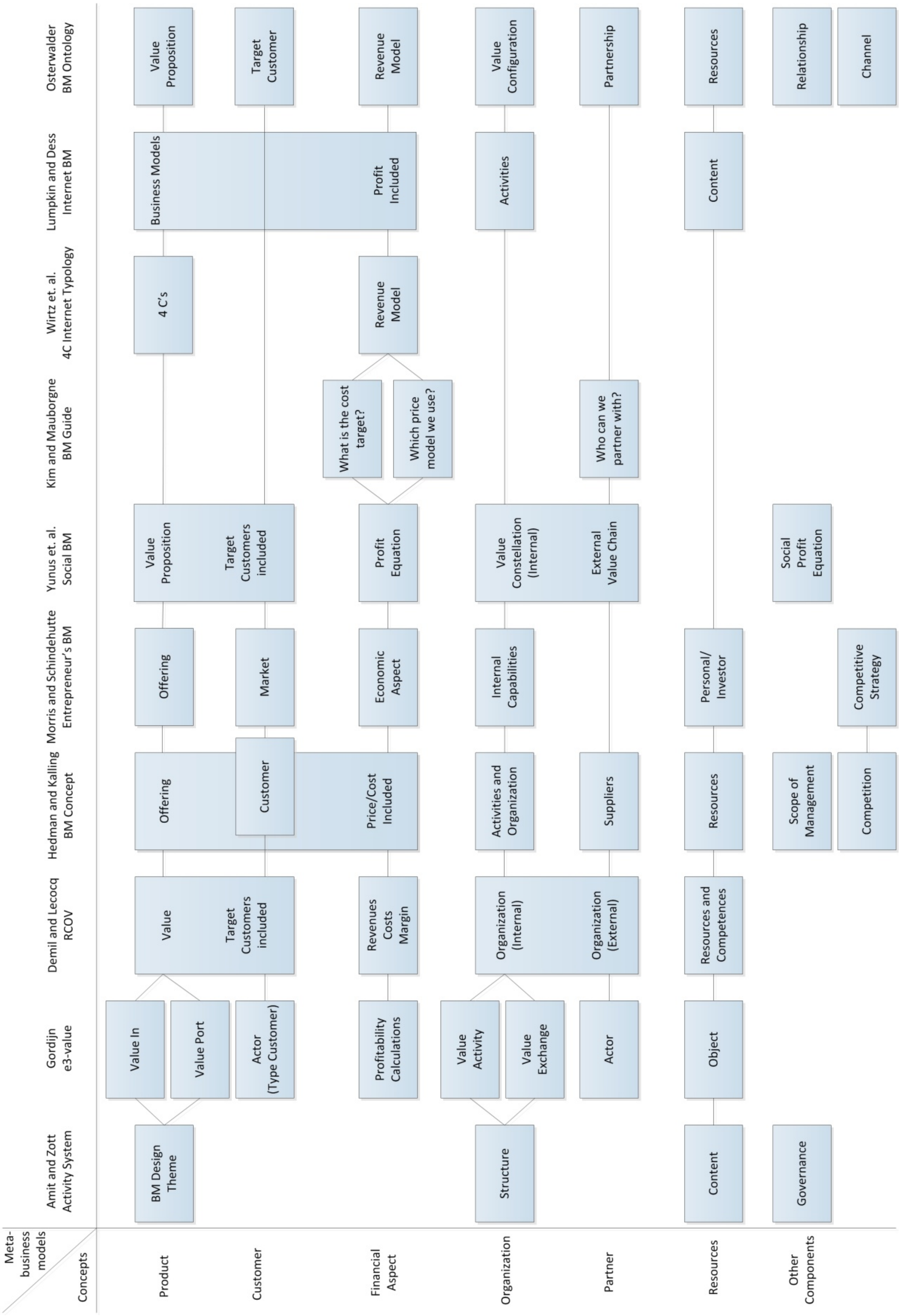


Figure 3: Comparison of M1 layer meta-business model components (adapted from Alberts (2011)).

mentioned already, Osterwalder's (2004) Business Model Ontology.

1. *Activity system* by Zott and Amit (2010).
2. *e3-value* by Gordijn (2002).
3. *RCOV* by Demil and Lecocq (2010).
4. *The BM concept* by Hedman and Kalling (2003).
5. *Entrepreneur's BM* by Morris, Schindehutte and Allen (2005).
6. *The social BM* by Yunus, Moingeon and Lehmann-Ortega (2010).
7. *The BM guide* by Kim and Mauborgne (2000).
8. *4C Wirtz*, Schilke and Ullrich (2010).
9. *Internet BM* by Lumpkin and Dess (2004).
10. *BMO* by Osterwalder (2004).

Figure 3 identifies the components used in the above articles. It is an indication of possible components for meta-business models. Quite some overlap exists in the identified components, which allows for mapping of the concepts on to each other. Already, Figure 3 provides an attempt at this. While in some cases this mapping is obvious, it remains interpretative. However, the figure still suggests which components are important to business modelling.

4 DISCUSSION

The purpose of this study is to promote theory development by viewing the concepts of business modelling in light of the Meta-Object Facility. Besides an open review of what the MOF allows, this section also comments on the classes left out of the B2BM.

4.1 Uses for the MOF Perspective on Business Modelling

Our main reason for using the MOF is the perspective it offers on the practice of modelling. As such, we have only identified classes in the M2 Layer meta-meta-business model. Still, it has become very clear that the discipline of business modelling allows for use of the MOF, and that the concept of meta-models can be of great assistance. We believe this introduction of MOF in business modelling has only scratched the surface of what is possible. Take for example an association between two classes: the scope of what is being modelled will strongly influence which components are important.

Defining the M2BM in terms of the MOF model concepts allows formalization of business modelling that promotes its use in requirements engineering and software development. In the same line of reasoning, the MOF perspective may provide a new chance to match business modelling and UML. So far, literature that uses both the terms "UML" and "business modelling" focuses on process modelling, not on business modelling. Another application of UML in this domain is creating a reference ontology (Andersson et al., 2006). This reference ontology allows model transformations between Resource-Event-Agent (REA), e3-value, and BMO. Such reference ontology may provide useful methods for the M2BM.

The MOF opens a rich new view on business modelling. So far, we have only looked at one aspect: the possible classes of the M2BM. There are still many more possibilities in using the MOF to approach business modelling.

4.2 Classes Left out of the M2BM

Two potential classes were left out of Table 2. They are ontological role, and ontology maturity & evaluation. Both concepts come from Gordijn, Osterwalder and Pigneur (2005). We argue that these two concepts are not suitable as classes for a meta-meta-business model.

The ontological role does not fit, as it is very similar to the entire concept of the MOF. As such, it has no place within one of the layers. For ontological role, Gordijn, Osterwalder and Pigneur (2005) define three levels: operational data at Level L0, ontology at Level L1, and ontology representation language at Level L2. Operational data is similar to what we call a business model instance at layer M0. Ontology is similar to what we call a meta-business model at layer M1. Finally, an ontology representation language is similar to the M2BM at layer M2.

Ontology maturity & evaluation does not fit, as it is itself not meta-data describing a meta-business model. Rather, checking maturity could be a use of the M2BM. For example, the maturity of a meta-business model could be scored based on how many of the M2BM classes it implements.

5 CONCLUSIONS

This article uses MOF to provide a new perspective on business modelling. This contributes to business modelling on three important issues:

- the need for a common language,
- lack of conceptual consolidation, and
- theoretical development of the concept.

Introducing the MOF perspective provides conceptual consolidation in business modelling. The MOF is used to take a different perspective on the meta-business models, which makes it possible to find commonalities. Identification of the M2BM classes illustrates this. In addition, existing definitions of "business model" can be positioned on the layers. This provides better options to compare definitions.

The M2BM on the M2 layer provides a common language for business modelling. In business modelling literature, many authors have their own vocabulary. In creating the M2BM, we show that different terms often refer to a single concept. Approaching the different meta-business models from a higher MOF layer addresses this issue. Doing so allows building on the strengths of the meta-business model original domains: strategic management, industrial organization, and information systems. Using the MOF, and especially the M2BM as a common language, helps overcome the differences of these domains and focus on commonalities.

Finally, theoretical development of the business model concept is promoted, as the MOF opens up a wide range of research possibilities for business modelling. Placing the concept of business model in the frame of the MOF allows for further theory development, both within the discipline and in relation to other domains. It serves as a navigational landmark for business model research when relating it to existing material. Additionally, it helps to create bridges to other research areas, especially when relating to other modelling domains.

Future research must specify a M2BM with classes, and possibly relations, data-types, and packages. This common language will define business modelling. The M2BM presented in this article is a first draft; as such, it requires more work. However, even in this rough form it shows that the MOF is a rich addition to the business modelling discipline.

ACKNOWLEDGEMENTS

This work is part of the IOP GenCom U-CARE project, which the Dutch Ministry of Economic Affairs sponsors under contract IGC0816.

REFERENCES

Al-Debei, M.M., Avison, A., 2010. Developing a unified framework of the business model concept. *European Journal of Information Systems*, 19(2010), pp.359–376.

Alberts, B. T., 2011. Comparing business modeling methods: creating and applying a comparison framework for meta-business models. In *Proceedings of the 14th Twente Student Conference on IT*. Twente Student Conference on IT. Enschede, Netherlands: University of Twente.

Andersson, B., Bergholtz, M., Edirisuriya, A., Ilayperuma, T., Johannesson, P., Gordijn, J., Gregoire, B., Schmitt, M., Dubois, E., Abels, S., Hahn, A., Wangler, B., Weigand, H., 2006. Towards a Reference Ontology for Business Models. In *Proceedings of the 25th International Conference on Conceptual Modeling*. International Conference on Conceptual Modeling.

Demil, B., Lecocq, X., 2010. Business Model Evolution: In Search of Dynamic Consistency. *Long Range Planning*, 43(2010) pp.227-246.

Gordijn, J., 2002. *Value-based Requirements Engineering: Exploring Innovative e-Commerce Ideas*. PhD Thesis. Vrije Universiteit Amsterdam.

Gordijn, J., A. Osterwalder, Pigneur, Y., 2005. Comparing two Business Model Ontologies for Designing e-Business Models and Value Constellations. In *Proceedings of the 18th Bled eConference: eIntegration in Action*. 18th Bled eConference eIntegration in Action. Bled, Slovenia.

Halteren, A. T. van, 2003. *Towards an adaptable QoS aware middleware for distributed objects*. PhD Thesis. University of Twente.

Hedman, J., Kalling, T., 2003. The business model concept: Theoretical underpinnings and empirical illustrations. *European Journal of Information Systems*, 12(1) pp.49-59.

Jasper, R., Uschold, M., 1999. A Framework for Understanding and Classifying Ontology Applications. *Proc.12th Int. Workshop on Knowledge Acquisition, Modelling, and Management (KAW'99)*, University of Calgary, Calgary, CA, SRDG Publications.

Kim, W.C., Mauborgne, R., 2000. Knowing a winning business idea when you see one. *Harvard Business Review*, 78(5) pp.129-138, 200.

Kuhn, T., 1970. The structure of scientific revolutions (2nd ed.), University of Chicago Press, Chicago.

Lambert, S., 2008. A Conceptual Framework for Business Model Research. In *Proceedings of the 21st Bled eConference: eIntegration in Action*. 21st Bled eConference eCollaboration: Overcoming Boundaries through Multi-Channel Interaction. Bled, Slovenia. pp.227-289.

Lumpkin, G.T., Dess, G.G., 2004. E-Business Strategies and Internet Business Models:: How the Internet Adds Value. *Organizational Dynamics*, 33(2) pp.161-173.

Meertens, L. O., Iacob, M. E., Nieuwenhuis, L. J. M., 2011. Developing the Business Modelling Method. In *Proceedings of the First International Symposium on*

Business Modeling and Software Design, BMSD 2011, Sofia, Bulgaria. pp.103-110.

Morris, M., Schindehutte, M., Allen, J., 2005. The entrepreneur's business model: toward a unified perspective. *Journal of Business Research*, 58(6) pp.726-735.

Object Management Group, 1999. Meta Object Facility, Version 1.3, *OMG document* ad/99-07-03.

Osterwalder, A., 2004. *The Business Model Ontology – a proposition in a design science approach*. PhD Thesis. Universite de Lausanne.

Osterwalder, A., Pigneur, Y., 2010. *Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers*, John Wiley & Sons.

Osterwalder, A., Pigneur, Y., Tucci, C. L., 2005. Clarifying Business Models: Origins, Present, and Future of the Concept. *Communications of AIS*, 15(May), pp.2-40.

Pateli, A. G., Giaglis, G. M., 2004. A research framework for analysing eBusiness models. *European Journal of Information Systems*, 13(4), pp.302-314.

Vermolen, R., 2010. Reflecting on IS Business Model Research: Current Gaps and Future Directions. In *Proceedings of the 13th Twente Student Conference on IT*. Twente Student Conference on IT. Enschede, Netherlands: University of Twente.

Wirtz, B.W., Schilke, O. and Ullrich, S. 2010. Strategic Development of Business Models. Implications of the Web 2.0 for Creating Value on the Internet. *Long Range Planning*, 43(2010) pp.272-290.

Yunus, M., Moingeon, B., Lehmann-Ortega, L., 2010. Building Social Business Models: Lessons from the Grameen Experience. *Long Range Planning* 43(2010) pp.308-325.

Zott, C., Amit, R. 2010. Business Model Design: An Activity System Perspective. *Long Range Planning*, 43(2010) pp.216-226

Zott, C., Amit, R., Massa, L., 2011. The Business Model: Recent Developments and Future Research. *Journal of Management*, 37(4), pp.1019-1042.

Engineering, Responsibilities, Sign Systems

Coen Suurmond
RBK Group, Keulenstraat 18, Deventer, the Netherlands
Csuurmond@Rbk.Nl

Keywords: Organisational Semiotics, Enterprise Engineering, Business Modelling, Sign Systems.

Abstract: This paper will address the question of how an engineering approach to enterprise modelling and system development might be combined with an approach to enterprises that does justice to its inherent social character, and where the members of the organisation are responsible for their decisions and not just operators of the systems. In the analysis of this question the concept of engineering will be discussed, along with the characteristics of different kinds of sign systems. System based sign systems (as used in ICT and engineering, and suited to the use of mathematical and logical formulas) will be contrasted with human based sign systems (natural language, appropriate for the adequate representation of values and of individual cases).

1 INTRODUCTION

The question is how we can combine the concept of the enterprise as a social system with the concept of software development as an engineering discipline. An enterprise has many different aspects: as legal entity, as economic actor, as an organisation of people with a common goal, as a community of individuals, as an actor in society. Each of those aspects has its own view on reality, with a perception of itself and of its environment, with its own language, and with norms that determine its behaviour. An enterprise has a vision, a mission and a strategy. An enterprise has processes, structures and employees. An enterprise has stakeholders, and fulfils a number of needs of those stakeholders. All of these are social concepts, invented and executed by and for people. To do so the enterprise uses a number of technical artefacts: buildings, transport of goods and people, machines for processing materials, and ICT for communication and information.

Over the past few decennia there has been a shift in the use of ICT, from EDP (electronic data processing) to information supply. While the focus used to be on the automation aspect, nowadays it is on the information aspect (while the automation of processes has meanwhile continued unabated). The computer based information systems are at the same time by their construction of a technical nature and by their use and purpose of a social nature. This intersection of technical and social system has proven to be problematic.

Firstly there is the problem of analysis and requirements: the developer tries to uncover the logic of the social system, but he is usually handicapped by being an outsider, because he does not speak the language and because he tries to uncover the logic of patterns that have evolved over time. From a systems thinking perspective and from the rationalistic/mechanistic world view of the IS-developer (and he shares this world view with many modern all-round managers) Business Process Reengineering has been presented as a remedy against the perceived irrationality of the business processes. Rationalisation first, systems development next.

Secondly there is the problem of the use of the information systems: does the newly developed system fit the social reality of the users? If that is not the case (in the subjective perception of the users) it can have a number of results. The quality of the business processes deteriorates as a consequence of the impoverishment of the available information. The users maintain the quality of the business processes by setting up and using additional information channels (free text within and outside of the systems). Or the users create their own parallel solutions for the storing and processing of information in spreadsheets and word processors.

In other words, we are dealing with two different discrepancies in the development of information

systems for enterprises: (1) the use of technical artefacts in social contexts; and (2) the mechanistic/rationalistic world view of analysts and developers in the face of the organic reality of practice. These differences can be traced back to a difference in sign systems; to the lingual reality of employees within the processes and of the external experts. In the analysis this manifests itself in the problem of capturing the extra-lingual practice of the business processes in the lingual reality of the analyst, and vice versa the challenge for the practitioners of evaluating the documents produced by the analysts. And in the implementation this manifests itself in the confrontation between the newly designed sign systems of the information systems and the established practice of the business processes.

The question is now how to best tackle these problems. It is about finding the balance. It should be clear that an enterprise is a historically and organically grown entity, embedded in social practices within the enterprise and between the enterprise and its external stakeholders. At the same time the grown practice should not be held as absolute; it should be possible to critically evaluate it. However, the backgrounds against which such an analysis is carried out are important. Is it from a mechanistic/rationalistic world view in which the enterprise is viewed as a technical artefact? Or is it from a view of the enterprise as a more organic entity, subject to a multitude of social forces and an emergent phenomenon?

2 ENGINEERING

Of old, engineering belongs to the world of the designing and building of technical artefacts. Etymologically, the word is related to the word engine. According to the OED, an engineer is nowadays (1) "one whose profession is the designing and constructing of works of public utility, such as bridges, roads, canals, railways, harbours, drainage works, gas and water works, etc"; (2) "a contriver or maker of 'engines'", (3) "one who manages an 'engine' or engines"; (4) "(with defining word, as human engineer, spiritual engineer), one who is claimed to possess specialized knowledge, esp. as regards the treating of human problems by scientific or technical means". The meaning of engineering is simply "to act as an engineer", again according to the OED (OUP, 1989). Henry Petroski cites a definition of structural engineering in his book "To Engineer is Human": "Structural engineering is the science and art of designing and making, with economy and elegance, buildings, bridges, frameworks, and other similar structures so that they can safely resist the forces to which they may be subjected" (Petroski, 1992, p40). The organisation behind this definition, the Institution of Structural Engineers, defines ""structures" as "those constructions which are subject principally to the laws of statics as opposed to those which are subject to the laws of dynamics and kinetics, such as engines and machines" (Thomas, 2012).

To put it briefly, the core task of an engineer is to design and make a technical artefact to serve some predefined function, and the artefact should be able to resist the forces applied to it without losing its usability.

The role of modelling in the work of an engineer is (1) to preview the design and (2) to study the forces that will be applied to the artefact when it is constructed and when it is in use. A beautiful example of such a model can be found in Barcelona, where Gaudi used some very simple materials (iron hoops, strings, and tiny sand bags) to study the forces for the design of his unorthodox buildings such as the Sagrada Familia and the Casa Milà.

The engineer uses a variety of scale models and prototypes, each representing one or more aspects of the intended artefact, as stepping stones to his final design. The artefact is the physical realisation of the design and the preparatory models.

In the field of information systems, the term engineering is used in different contexts. I will shortly discuss three of them. The first is Software Engineering, and in the book "The Road Map to Software Engineering" the following IEEE definition (Std 610.12) is used: "(1) The application of a systematic, disciplined, quantifiable approach to the development, operation and maintenance of software, that is, the application of engineering to software. (2) The study of approaches as in (1)". The author considers software engineering as engineering, because the process consists of related activities performed in response to a statement of *needs* and consuming *resources* to produce a *product*", in combination with systematic controlling and measurement processes (Moore, 2006, p3). Although the physical aspect of the technical artefact is lacking, all other elements of traditional engineering are present: predefined needs, a clear finished product, and a process of design and development which has a technical character. Dines Bjørner in his work about Software Engineering defines engineering as "the mathematics, the

profession, the discipline, the craft and the art of turning scientific insight and human needs into technological products" (Bjørner, 2006). The technological product is central, and its role to serve human needs. Although the context is software, the definition applies equally well to technological artefacts as bridges and engines.

The second context is Systems Engineering, and here the issues are less clear cut. The website of the International Council on Systems Engineering (INCOSE) states: "Systems Engineering is an engineering discipline whose responsibility is creating and executing an interdisciplinary process to ensure that the customer and stakeholder's needs are satisfied in a high quality, trustworthy, cost efficient and schedule compliant manner throughout a system's entire life cycle.". On another page of the website it is stated that "Systems Engineering considers both the business and the technical needs of all customers with the goal of providing a quality product that meets the user needs". This is a far cry from a well defined technical artefact. Apart from the sloppiness of the statements, there is a fundamental difference between an obligation to produce a technical artefact, and the obligation to fulfil business needs and user needs.

More recently, the term engineering is also used in the context of enterprise engineering. One of the pioneers in the field is Jan Dietz, who is editor of the book series on Enterprise Engineering at Springer Verlag and a driving force behind the CIAO network. The website of the book series mentioned above tells us: "Enterprise Engineering is an emerging discipline for coping with the challenges (agility, adaptability, etc.) and the opportunities (new markets, new technologies, etc.) faced by contemporary enterprises, including commercial, nonprofit and governmental institutions. It is based on the paradigm that such enterprises are purposefully designed systems, and thus they can be redesigned in a systematic and controlled way. Such enterprise engineering projects typically involve architecture, design, and implementation aspects" (Dietz, 2011). Key in this statement is the presumption that enterprises are purposefully designed systems. All the same, this approach recognises the enterprise as essentially social systems (postulate 2 of the Enterprise Engineering Manifesto of the CIAO network), and it recognises the ethical necessity of taking the responsibilities of the people in an enterprise seriously (postulates 2 and 5). Here, the concept of engineering is taken from the technical environment to the organisational environment. The characteristics of engineering as a discipline that designs and builds artefacts are preserved by considering the enterprise as "purposefully designed systems", and by modelling the essential structures of an enterprise in its ontological model, which captures objectively the structure of the enterprise (postulate 4). In his book about Enterprise Engineering Dietz writes "The *engineering* of a system is the process in which a number of white-box models are produced , such that every model is fully derivable from the previous one and the available specifications. ... Engineering starts from the ontological model and ends with the implementation model" (Dietz 2006, p74).- The white-box model is a model that represent the structure and workings of a system, abstracted from implementation details. As such, it is an abstract model. The interesting question is, whether this engineering approach for the enterprise considered as a designable socio-technical artefact will hold up.

3 THEORY OF THE FIRM

An enterprise has a sustainable existence if and only if it produces products for its markets in a profitable way. The business processes represent the enterprise in action; there the products are created, sold and distributed and all activities that are needed to support and manage these primary processes happen there. The formal organisation of the enterprise is designed to structure the business processes in effective and efficient ways, and in accordance with the values and norms of the enterprise towards the internal and external stakeholders. The informal organisation, the collection of actual patterns that structure the business processes, is in a continuous evolutionary process as a result of the daily interaction of people and systems in the enterprise and its environment. An important driving force of the evolutionary processes of the informal organisation is the leadership and management of the enterprise, it shapes the values in the organisation. The corporate culture influences the way people behave in the enterprise; it supplies some background norms for their decisions. For James Taylor these aspects are so important, that he hypothesises that the organisation is "constituted in the ongoing processes of its members" (the title of a presentation earlier this year in Nijmegen was "Authoring the Organization"). Others are not going so far, but John Kay emphasises that the foundation of corporate success often lie in intangible factors such as reputation and internal architecture. He calls these factors the distinctive capabilities, and they

determine the success and durability of the enterprise (Kay, 1993). The former strategic planner of the Shell Oil Company, Arie de Geus, makes a case of defining an enterprise as an organism, instead of a mechanism (De Geus 1997). When on the other hand an organisation would be considered in a mechanical way, and the behaviour of its members would be fully determined by explicit rules, we would have a perfect bureaucracy. And we know that the ultimate action weapon for a civil servant is to work to rule, making everything grind to a halt. In other words, the actual organisation, the factual business processes and the individual decisions of the members of the organisation are partly determined by explicitly defined structures and rules, partly by corporate culture, and of course partly by incidents and individual capabilities.

At the same time, the structure of an enterprise will be partly determined by objective factors, derived from the characteristics of its products and its markets. They have a kind of inherent process logic, determined by the conventions of the markets, by the products and the production and distribution processes, and by the social and legal conventions involved. That implies, that by knowing the markets and products of an enterprise, you can predict quite a lot about its internal process steps. One could consider these structures as the essential deep structures of the enterprise. What is much less predictable, however, is the way these structures are mapped unto business processes and the organisation. These mappings are the result of both conscious organisational decisions, and of evolutionary processes. Gradual changes of markets and products can drive evolutionary adaptations in the business processes, and the history of an enterprise might long be reflected in its operational ways.

For the analysis of an enterprise then, we have to deal with both process logic derived from its markets and products, and idiosyncratic characteristics derived from its specific history and evolution from the past. From the viewpoint of the enterprise, to comply with the process logic is necessary to be in business, to be unique and have distinctive capabilities is necessary to stay in business.

4 SEMIOTIC THEORY

4.1 Signs

Semiotics studies signs in use. One well known definition of a sign is: “A sign, or representamen, is something which stands to somebody for something in some respect or capacity. It addresses somebody, that is, creates in the mind of that person an equivalent sign or perhaps a more developed sign. That sign which it creates I call the interpretant of the first sign. The sign stands for something, its object. It stands for that object, not in all respects, but in reference to a sort of idea, which I have sometimes called the ground of the representamen.” (Peirce, 1998, par2.228) Fundamental in the concept of the sign is the difference between the sign and that what it stands for, and the difference between the sign and its interpretation by the sign user. The process of interpretation of a sign (semiosis) is the subject of pragmatics, semantics studies the relation between the sign and its meaning (that what it stands for), and syntax in concerned with the formal relations between signs.

For the pragmatist the concept of meaning is directly connected to its use, as Wittgenstein writes: “The meaning of a word is its use in the language” (Wittgenstein, 2009, par. 43). Knowing the meaning of a word is knowing when and how to use the word. Related to this concept of meaning is the Wittgensteinian concept of language games. A language game involves a community of users in a certain context, along with the (unwritten) rules how to use language in this context. This pragmatist (not to confuse with pragmatic!) approach of meaning emphasises the primacy of the actual use of signs, the lexical meaning is an abstraction and record of the stable kernel of the meaning in use. Even when definitions of words are explicitly written down, the actual use in the community of users will ultimately determine the meaning. Examples of this mechanism can be found in law, see for example the differences between written law and its interpretation in jurisprudence in European law, and in the workings of case law in Anglo-Saxon practices.

4.2 Sign Systems

The concept of the sign system is widely used in semiotics, but rarely defined. Intuitively it is easy to give a number of examples of clearly different sign systems. Consider the difference between natural languages (English, Japanese), formal languages (predicate logic, C#), visual languages (pictograms on airports and stations). Hereafter it quickly becomes more difficult: can different language families be considered different sign systems? Different languages within one language family? Different dialects? The language games of

Wittgenstein? Are the social media examples of new sign systems, with their own vocabulary, usage rules and expressive power? Hereafter, I will talk about "system based sign systems" for the formatted information that is recorded, processed and presented in available business information systems, and "human based information systems" for information that is processed by human beings (both linguistic and non-linguistic).

Above, more or less formal and/or structured information flows were mentioned in the context of the organisation. The ledger system in a financial administration has a prescribed formal main structure that allows external stakeholders (taxes, chartered accountants) to know in advance what information can be found where. Within this prescribed main structure an organisation can make its own choices, but the main grouping is recognisable for all stakeholders. This is an example of an engineered sign system. A very different example is the way in which all kinds of things are designated in an organisation ("office of Jean-Claude" although Jean-Claude has left 30 years ago, or volumes designated by some odd measure such as "two mills of French fresh", indicating a certain amount of a certain quality of shrimps). These are historically grown practices that refer back to field names, to an amount and a quality, and to a measure that is long abolished in trade, but still used in production planning. Both sign systems are not readily accessible by an outsider. The extent to which a sign systems is defined can differ greatly, but it is true for every sign system that it has to be learned in practice.

4.3 Characteristics of Sign System

For an analysis of the characteristics of sign systems the dimensions of the Information Space as defined by Max Boisot can be useful. He distinguishes in his analysis of information the following three dimensions: (1) degree of codification, (2) degree of abstraction, and (3) degree of diffusion (Boisot, 1998). His analysis is directed towards knowledge assets; applied to sign systems we could distinguish between the way information is codified, abstracted and diffused in a sign system. On the first sight, there is a huge difference between (1) system based sign systems that are a combination of highly codified information (structured data that might be interpreted by the computer programs) and less codified information (free text that is recorded and made available to the user) and (2) The human based sign systems that are much less codified and based on conventions (the Wittgensteinian language games).

However, the real difference might lie elsewhere, not so much in the degree of codification as well in the way of codification. Experienced employees can communicate in a very precise manner, but the communication might require a rather long initiation process. Without a professional background, the words just do not make sense to the outsider. The system based sign systems on the other hand seem easy to use but are often imprecise. The available codification in the ICT system might just not match the needs and the categories of the user. In other words, the first difference between system based sign systems and human based sign systems lies in the nature of the codification processes. In talking and writing an experienced user can code a lot of information into his utterances. The degree of imprecision can be stated in a very subtle and precise way ("the same quality as last time, but remember last week!"). The modalities and illocutionary force of speech acts are hard to codify in ICT systems ("please keep in mind that this customer might order next week, and that we will have to deliver within reason, but due to the unpredictable weather he probably will not order until Friday").

The second difference lies in the diffusion processes. It is clear that the diffusion processes are greatly influenced by ICT systems. When we restrict ourselves to the use of business information systems (rather than communication techniques such as video links, skype, et cetera), these networked systems make all information in them instantaneously available to all users (when they are granted the right authorisation). This diffusion processes create problems of their own, such as uncontrolled interpretation by inexperienced people and inconsistent information, that I will not discuss here.

Abstraction is needed whenever general rules are applied to individual cases. In system based sign systems, the abstractions are predefined in its data structures and individuals are coded as objects. The possible abstractions are predetermined by a combination of the more or less fixed data structures, software code and configuration. In the human based sign systems, the abstraction coincides with the interpretation and is contextually determined. This prototypical case for this kind of abstraction processes is the verdict of the judge. The application of the law can be seen as the subsumption of the individual case under the general rule. The perception of the individual case involves

abstraction processes, and this abstracting perceptual power is exactly where the added value of the human being is. That applies not only to professional experts, but it applies also to all kind of work where people are accountable (paid to use their own judgement, and not just 'hired hands').

4.4 Sign Systems and Business Processes

So, basically we are dealing with two differences between system-based sign systems and human based sign systems: (1) the way the signs are captured and coded, and (2) the interpretation capabilities. For practical purposes, two questions arise. The first question is about the costs and effects of information processing by ICT systems. What is the ratio between the efforts to capture the requirements and to subsequently engineer the solution, and the resulting savings and process improvements in the business processes? The second question is not about calculations of return of investment, but more fundamentally which decisions can be made by ICT systems and which decisions have to be made by the members of the organisation. The capability of the human mind to interpret information in context, to absorb new situations and to judge between norms cannot be met by ICT systems. This basic capability is fundamental to the concept of accountability.

The characteristics of the business processes in an enterprise are partly determined by its markets and products, and partly by its idiosyncratic elements like location, infrastructure, corporate culture and so on. When we bear in mind what Arie de Geus has written about the living company, and what John Kay has written about distinctive capabilities, we should be aware of the risks in considering an enterprise as a whole as an artefact that can be engineered.

At the same time, the inherent process logic, as determined by the markets and products, is much more suited to an engineering approach. One could consider this process logic as the essential structure of the enterprise, as it represents the invariant structural elements of the organisation. However, this objectified approach to the inner workings of the enterprise should not be confused with the elements that define the unique position of the enterprise. The way the process logic as an abstract model is implemented in the real enterprise systems is decisive for the operations of the enterprise, and in this area an analysis of the sign systems in use is relevant.

4.5 Representation of Elements from Business Processes in Software

Item coding of semi finished products in food processing industry is a notoriously difficult problem, because the variable characteristics of the products cannot be mapped uniquely on fixed item codes. This is quite different from item codes of engineered and standardised materials such as M3 bolt. In practice knowledge of the customer codetermines which lots are suited for a sales order. The item code is essential in the recording of the sales order, the consignment note, and the invoicing, but in itself insufficient for the preparation of the sales order. Information systems that pretend to calculate the internal processes from the sales orders will fail, the judgment of an experienced human planner is undispensable.

Another modelling challenge lies in the representation of objects. What is considered to be one object by one department, might be more objects for other departments. Some time ago, I saw a freight train consisting of pairs of freight carriages. Each pair was connected by a hooded transition in between. This creates a typical modelling problem: do we have one four-axle carriage, or two two-axle carriages? For operational logistic planning purposes it will probably be the first (each pair is a fixed capacity unit), and for the maintenance department it will probably the latter (individual carriages with each its own maintenance history).

Software often has problems with these kinds of constructs, and the requirement analyst wants to know "what it really is". This is an expression of a certain world view, where atomistic facts are presupposed. It is a world view that fits perfectly well to relational databases. Unfortunately, it is not a world view that fits to the actual world, where technical artefacts can be both one and two at the same time. However, this kind of problem is still essentially a solvable engineering problem. It may lead to complicated software, but the several interpretations can be modelled unequivocally and converted into software solutions.

Whenever there is a misfit between the sign system of the ICT solutions and the sign systems of the user and his processes, the ICT solutions will cause problems. It leads to distortion of processes, of information, or both. A habitual cause is a rationalistic engineering approach of ICT designers to user processes that are not well understood. Close observation and sometimes participation in the user processes is needed to detect the rationale behind the

patterns in the processes, or at least to understand which information matters for the user.

A second element of a successful ICT system is the design of a system of references that satisfies the needs of both the ICT system and the users. It might sound as a trivial requirement, but the solution is often difficult. It involves an understanding of the user processes, and the way the user integrates information from the ICT systems with information from other sources. At the same time, the integrity of the ICT systems must be warranted. The system engineer must understand that his system is not the centre of the world, and that it should serve its users. A sales order is an agreement between buyer and seller from the moment that they make that agreement. When and how the agreement is registered in the ICT system is relevant, but not decisive. If the order is split in the system into more sales orders for some reason, it will still be one sales order for the customer.

5 CONCLUSIONS

The engineering approach is the right approach whenever a well defined artefact is to be delivered, as it is able to deal with the well defined intended use of the artefact. An engineering approach is bound to use formalised sign systems, that allow for precise and thorough testing, based on logic and causal relations. Engineered artefacts (and their preparatory models) are subject to the rigid laws of physical forces (bridges, engines) and of logic (abstract software models and enterprise models).

However, when social factors and individual and collective decisions determine the actual use of the technological artefacts, it is not matter of engineering anymore. That applies to bridges, where traffic might be regulated by the marking of traffic in lanes, by traffic lights and by speed limits. That applies also to software systems, that must be configured for use in an organisation with a specific corporate culture, with a given composition of employees with knowledge and experience, and with a distribution of tasks and responsibilities. Decisions in this realm are partly based on goals and values, and cannot be expressed in formal sign systems.

For system development, it is a big challenge to combine an engineering approach for capturing the process logic of an enterprise, as determined by its markets and products, expressed in formal sign systems, with an implementation of this abstract process model that does justice to the idiosyncratic elements of the enterprise. The analysis and expression of these idiosyncratic elements calls for the use of human based sign systems (natural language with all its capabilities to express norms and values with all its subtleties) and might be compared to anthropological analysis.

The analysis presented in this paper should enhance the awareness of the nature of the problem. It should help to demarcate the domains where formal sign systems as used in engineering are applicable, and the domains where human based sign systems are called for. Not discussed in this paper, but most important, is the question how to reconcile the use of these different kinds of sign systems in the development of a concrete information system.

REFERENCES

Bjørner, D.. 2006. *Software Engineering 1*, Springer Verlag. Berlin.

Boisot, M.H. 1998. *Knowledge Assets*, Oxford University Press. Oxford.

De Geus, A, 1997. *The Living Company*, Nicholas Brealey Publishing. London.

Dietz, J.L.G., 2006. *Enterprise Ontology*, Springer Verlag. Berlin.

Dietz, J.L.G. (ed.), 2011. *Enterprise Engineering – The Manifesto*, CIAO Network. From: http:// www.ciaonetwork.org/publications/EEManifesto.pdf.

Kay, J., 1993. *Foundations of Corporate Success*, Oxford University Press. Oxford.

Moore, J.W., 2006. *The Road Map to Software Engineering*, John Wiley & Sons. New York.

OUP., 1989. *The Oxford English Dictionary*, Oxford University Press, Oxford.

Petroski, H.., 1992. *To Engineer is Human*, Vintage Books. New York. Smith, J., 1998. *The book*, The publishing company. London, 2nd edition.

Peirce, C.S., 1998. *Collected Papers of Charles Sanders Peirce*, Thoemmess Press. Bristol.

Thomas, R., 2012. *History of the Institution of Structural Engineers*, Institution of Structural Engineers. From: http://www.istructe.org/webtest/files/e1/e1faa2cf-e604-47d0-a888-31a9d6758fa8.pdf

Wittgenstein, L., 2009. *Philosophical Investigations*, John Wiley & Sons. New York, 4th edition.

The Capability-affordance Model
A Method for Analysis and Modelling of Capabilities and Affordances

Vaughan Michell
Informatics Research Centre, Henley Business School, University of Reading, U.K.
v.a.michell@reading.ac.uk

Keywords: Capability, Affordance, Active Resources, Driving Resources, Passive Resources, Affordance Mechanism, Affordance Path, Affordance Chain, Critical Affordance Factor, Affordance Efficiency, Affordance Effectiveness, Affordance Quality.

Abstract: This paper builds on an earlier paper modelling business capabilities as a function of resources and process to support enterprise architecture decision making. This paper develops the earlier capability model by using a realist perspective to apply the theory of affordances and the Z specification language to show how capabilities and their actualisation can be modelled as a tuple or set of resource affordance mechanisms (AM) and affordance paths AP subject to critical affordance factors. The paper identifies and develops the concept of objective and subjective affordances and shows how these relate to the capability resource model. We identify an affordance chain by which affordances work together to create a capability. We define the affordance modelling notation AMN as a method of representing the affordances in a system of interacting resources. A medical case study (based on interviews at a local hospital) is used to show how capabilities can be identified and modelled using the theory of affordances. We also propose a model of affordance efficiency, effectiveness and quality that enables the performance of a capability and its constituent affordances to be measured and modelled.

1 INTRODUCTION

The performance of a business enterprise has become increasingly important, especially in recessionary times. The term capability has been widely used in enterprise architecture and other business as a concept describing the ability of a set of business resources to perform, but it is difficult to pin down and measure (Michell, 2011; Curtis et al, 1995). Analysis of capability has been mainly at high level (Grant, 1991; Hafeez et al, 2002; Josey et al, 2009; Merrifield et al, 2008; Beimborn et al, 2005). Without a more reproducible model of capability it is difficult to consistently understand existing business capabilities, or to use the concept to measure the relative performance of the various resources and business structures. This makes it difficult for an enterprise to make critical resource investment and divestment decisions and to understand and develop core competences that are vital to maintain and grow competitive advantage (Liu et al, 2011).

1.1 Capability

An earlier paper (Michell, 2011) sought to reduce this gap by building on resource theory to identify a definition of the capability of business resources of an enterprise for use in enterprise architecture. The following section re-iterates and develops the definitions using and adapting Luck and d'Inverno's work on agent frameworks and *Z* Notation (Luck and d'Inverno, 2001). We defined the *business environment E* to comprise a set of objects we called *business resources* R_i, where i= 1-n:

Environment E = {resources R_i*}.*
Env = = $\mathbb{P}$ *Resource*

Each resource is what Luck et al call an *object*. Hence we use the term *object resource*. The set of object resources $\{R_i\}$ have what Ortman and Kuhn (Ortmann and Kuhn, 2010) call qualities *q* that can be divided into perceivable features and facts about the object e.g. size, weight, chemical composition etc., what Luck calls 'a collection of attributes'. Another class of quality is what Luck calls *capability* and Ortman calls *affordance*. We will

discuss the difference and their relationship later in the paper.

We identified *agents* as either *human intelligent* or *artificially intelligent*. We now extend this by defining a human agent by combining the artificial intelligence definition (Stoytchev, 2005) with Luck and d'Inverno's definition to give: *A human or artificially intelligent agent is a resource object that can perceive its own environment through sensors and acts on the environment according to their self-motivations through effectors*.

We identified passive resources as resources requiring other agents to realise their capability ie they are inert and not capable of their own motion or change of state which relates to Luck's lower level definition of an object (Luck and d'Inverno, 2001). We identified the concept of a driving resource in a process to refer to object resources actively enabling and driving a business transformation compared with those (objects) passively used in the transformation involved in a business process (Michell, 2011). Driving resources in a business process are autonomous agents (human/artificial) controlling a range of transformations. Our scope focuses on business activities where all the entities in a business environment are object resources to be used by a business enterprise. We defined *capability* as: *a property of a resource (tangible or intangible) that has a potential for action or interaction that produces value v for a customer via a transformation process that involves the interaction of the resource with other resources* (Michell, 2011). We use Luck's definition of action as 'a discrete event that can change the state of an environment' (Luck and d'Inverno, 2001). The tangible and intangible resources in a business environment undergo transformations and a change of state as a result of an action of one resource interacting with another. Each of these resources has a capability of adding benefit of a specific value dependent on what resource and process is used to execute the capability i.e. *Cv* (*capability of value V*) = *f (resource, process P)*. We adapt Ortman and Kuhn's definition of value, to business value is the result of any action that is of benefit to the business/client (Ortmann and Kuhn, 2002). The Capability Cv results from transformation interactions between two or more resources that increases the business value of the transformed resource (with respect to a business client), i.e. the capability process contains value transformations as defined by Weigand et al (Weigand et al, 2006).

Cv = f (resource interaction, process of interaction).

At the detail level each driving agent resource in the sequence of transformations will be driven by what Luck et al call motivations to achieve an outcome or goal state G (Luck and d'Inverno, 2001), which the transformation aims to achieve. We can say that this goal state has attributes and a value to the driving resource that is greater than the value before the transformation. Using Z notation we can say a resource R is transformed between a state a and a' such that actions are an effect on the environment that result in a partial change in the environment. We identified the process as a sequence of actions. A process is what Luck refers to as a total plan where Totalplan ==seq Action. To develop the resource capability model we need to use a structured method that enables an action level perspective of capability as the interaction between object resources.

1.2 Affordances

Gibson (Gibson, 1979) defined the term *affordance* based on the root 'phenomenon in gestalt psychology (cited in (You and Chen, 2003)). Gibson's ecological approach saw 'affordance as an invariant combination of properties of substance and surface taken with reference to an animal' and 'what the environment offers and animal or intelligent agent'. The theory of affordance provides a perspective on the interaction of objects in an environment that supports the capability approach. Stamper (Stamper & Liu, 1994) expanded the concept of affordances using behavioural norm concepts into organisational semiotics and the idea of affordance relating to the invariant behaviour of both physical objects and intangible entities such as social behaviours and concept models such as ontology. Norman (Norman, 2004) refers to the term 'perceived affordance' for the concepts to describe the interaction of objects with people and design implications of man-made objects such as handles optimised for pulling via a gripping section or plates for pushing (Gaver, 1991). Norman sees affordances as 'referring to both potential and actual properties of a designed device'. You and Chen identified the difference between the semantics of design resource objects and affordance theory (You and Chen, 2003). Ortman and Kuhn (Ortmann and Kuhn, 2010) see affordances as one part of a set of qualities of an object which separately include physical qualities or facts about the objects such as size, temperature etc. Turvey (Turvey, 1992) developed a mathematical model of affordances as a property of a pair of things e.g. object and agent where the affordance is

wholly dependent on the nature of the system of interaction. Unlike physical qualities of an object resource, an affordance is a property of the interacting object resource systems manifested only when they are interacting. Turvey's model highlights the criticality of the interaction between the two object resources. This suggests we should really refer to the term 'affordance pair' as the affordance interaction property of two systems. Using Turvey's notation and the capability notation from the previous paper an affordance between two resources R_p and R_a, where R_p is the passive resource with property p and R_a is an active agent resource with property a can be represents as Af_p. Af_{pa} is a tuple of a passive resource and a driving agent and represents the interface interaction or affordance of the pair of object resources. This interaction property is different from the properties of the separate object resource systems. The affordance Af is represented as an ordered pair. However, in this paper we use the rule that the active or driving resource should precede the passive resource undergoing transformation (in cases where this is perceptible), in this case a human agent based on Turvey's order in the stair example. We use the notation: $Af_{ap} = f(R_a,R_p)$. However, there is a dichotomy in views of affordance as Gibson (Gibson, 1979) reminds us that an affordances is at once both an objective property of an object independent of perception and a subjective property depending on the perception of the viewer. Our capability definition is a function of the resource objects and of the actions (process) by which the resources interact and affordance theory relates to the mechanism of the resource interaction. This echoes Montesano et al's view of affordances 'capture the essential world and object properties, in terms of the actions the agent is able to perform' (Montesano et al, 2007). We adopt a realist perspective with physical, cognitive and in particular geographical reality relating to Gibson's ecological affordance view (Smith and Mark, 1998). We use the theory of affordances and *Z* notation as a formal language and frame of reference to represent the capability-affordance model, focusing mainly on physical affordances. In the following we attempt to discuss and relate these two views and show the relationship between capability and affordance.

2 OBJECTIVE AND SUBJECTIVE AFFORDANCES

This section focuses on objective and subjective affordances, as already mentioned.

2.1 Objective Affordances (OA)

Affordances that exist independent of perception are inherent objective or 'passive 'structural properties of the environment or object resource. For example a solid floor objectively and passively affords support of objects placed on it vs. the perceptive view of affordance which relies on an agent identifying how an object resource could be used. We suggest these passive objective affordances (OA) are what keep object resources in an unchanging, ie motionless state of equilibrium. Also these affordances are not perceptions or possibilities of what might happen – the possibility has already happened and we are observing executed affordances. The interface relationship has been 'actualised' and the affordance is acting in a state of equilibrium. Using Luck's state approach and Ortman and Kuhn's quality approach we can say that objective affordances relate to the qualities of a passive resource in stable equilibrium at a externally perceived level, i.e. we exclude semiotics at a microscopic level as Noth's view (Noth, 1998) to avoid problems with motions of microscopic organisms etc. We therefore suggest a resource has a set of state determining qualities that are required to be balanced in order to execute a passive objective affordance. This would comprise for example the force experienced by an object, its spatial position and other quality attributes. Objective affordance then relates to actual physical qualities or properties that relate to the natural behaviour of the object resource such for example the laws of natural science such as physics (Newton's laws) chemical and biological laws etc. The passive objective affordance is a stable state where:

- Transformation forces on the object are balanced
- The object's special position is unchanging
- The object's attributes are not changing (chemical composition, dimensions etc.)

In summary the affordance transformation mechanism in objective affordances is in a state of stable passive equilibrium. An example is the weight of a cup due to gravity is balanced by the structural strength and reactive force (due to Newton's Third Law) of the surface on which the cup sits. The

equilibrium is stable as the potential transformation mechanism e.g. the forces are in balance and the cup resource object is a passive resource and therefore possesses no means of motive force or transformation mechanism capability of its own. We can say that the interface mechanism between the two resources; cup and table is stable and has no active transformation mechanism. This suggests that affordance depends on the mechanism of transformation at the interface of the affordance pair. In this case the affordance interface mechanism is the force of gravity modelled by Newtonian mechanics i.e. F=ma to stably maintain the cup in contact with the table. Objective passive affordance relates to what Norman (Norman, 2004) calls the intrinsic properties of things (Norman's 'actual properties) and innate affordances. This passive objective affordance exists without the need for perception. This relates to Turvey's (Turvey, 1992) focus on an environments capability to support an activity. Then Affordance = the disposition (of the object) i.e. force balance on table. Effectivity = the complement of the disposition i.e. cup supported as a result of the transformation mechanism being achieved ie force balance. This equilibrium state of object resources continues unless acted upon by an external force (e.g. Newton's First Law) which brings us to subjective affordances.

2.2 Subjective Affordances (SA) and Perceived Actions

Ortmann and Kuhn suggest 'Perceiving affordances generates possibilities for action.' (Ortmann and Kuhn, 2010). We suggest that subjective affordances (SA) depend on perceived actions. This uses the 'perceived design affordance' (Norman, DA14) that relates to users understanding of the possible ways (affordances) of using designed features in designed objects. Our term refers to active resources such as a driving resource i.e. an agent intelligently perceives the affordance or possibility for using a resource object to make a substantive transformation and has an intention to use it. We include the term *action* here as the intention is to do physical substantive work, and we us this term to differentiate from non-substantive actions relating to knowledge and semiosis. We also include the term *subjective* as it depends on the driving resource's goals and motivations in the intended action and also critically on the selection of the path of action or process from our earlier definition. This relates to Gibson and Uexkull seeing affordances as perciptibles (Ortmann and Kuhn, 2010). It is these perceived 'action affordances' that relate to planned actions in business processes, activities or human visions and conceptual models of possibilities for action. Subjective affordance depends on an agent's perception as to how the properties or qualities of the object can be envisaged to be used to achieve a goal. The intention or goal of the outcome must complement the affordance (Ortmann and Kuhn, 2010). The goal; also helps a) to define a set of best action paths or processes that could be used and b) to define the value or client benefit of the result expected to be achieved on completion of the action. However, the affordance depends on the interface between an affording object and the driving agent that is attempting to use it. In its simplest form active affordance is the property of interaction of two entities or systems that enable an action to achieve a goal or transformation. The affordance relates to a mechanism for a potential action achieved by a transformation mechanism that changes an object resource state from state a to a' in *Z* notation.

- This may be physical e.g. a force form an agent (active resource) sliding a cup between two positions on a table i.e. cup-table affordance pair affords sliding
- This may be chemical e.g. sugar dissolves in water i.e. sugar-water affordance pair affords dissolving due to chemical forces.

2.3 Objective Affordance – A Multiplicity of Options

Any resource in a business environment is a collection of capability options or opportunities or affordances which depend on how the resource is related to another resource and what affordance mechanism is to be employed to achieve the state transformation and meet the goal. For example a simple passive resource such as a brick can be used as a doorstop (weight affordance mechanism Af_1), as a missile (kinematic mechanism of a thrown brick affordance Af_2), as a structural support component in a wall (rigidity and bonding affordance mechanism Af_3).

i.e. $R \text{ -= } f \{Af_{ij}\}$

This type of affordance depends on identifying an object's potential for action when an active agent or driving resource can visualise the way in which the object can be used in conjunction with their end effectors and senses. Turvey (Turvey, 1992) in his nest building argument provides support for the driving and passive resource view (Michell, 2011) in

his view of disposition and complement by differentiating between the properties of an animal (or agent) affordance which seeks to identify the affordance properties of the environment i.e. how the resources can be used by the person, (i.e. via manipulation, construction, imagination etc.). This relates to Turvey's focus on the animal's capability to perform an action. So, *effectivity* equals the disposition i.e. the effectiveness of the transformation mechanism in meeting the transformation goal (to be discussed later). And *affordance* equals the capability of the transformation mechanism necessary to achieve the goal. In our capability terms the agent driving resource in a process seeks to identify the disposition of the resources (objective affordance) to achieve the transformation action required by the process.

2.4 The Relationship between Objective and Subjective Affordances

In summary, objective affordances refer to executed and operational affordances ie an on-going system interface relationship where the mechanism of transformation becomes an equilibrium mechanism and the path is completed and meets the goal eg affording structural support. In contrast the subjective affordance has to be perceived or planned and therefore the mechanism and path may change and so may the outcome or actualisation and the goal which is dependent on both of these as below.

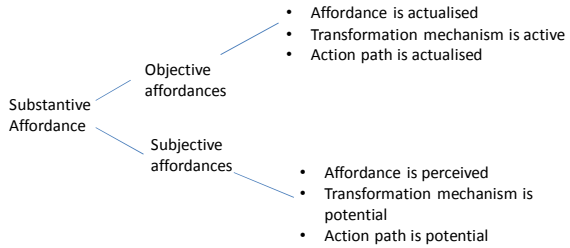


Figure 1: Objective/Subjective Affordance Types.

2.5 Subjective Affordances and Action Paths

However, a subjective affordance does not just depend on the transformation mechanism, it also depends on the way or path taken to actualise or execute the affordance. Consider the often quoted example of 'the cup affords drinking'. It does only if *a)* the cup is grasped firmly without obscuring the opening, *b)* if the cup is brought to an animal's mouth by an action that changes the position of the cup to the proximity of the mouth, *c)* it is tilted to enable gravity to act on the fluid and or the animal sucks to aid the fluid transfer. We will explore this issue of path in terms of our capability model.

3 MODELLING AFFORDANCES

3.1 Affordance Modelling Notation (AMN)

In reality, as we have seen, capability rarely comprises one affordance, but depends on multiple affordances. We will use a medical example of a capability 'to anaesthetise a patient' to explore the relationship between capability and affordance. The goal is to ensure the patient has a controlled unconscious state, the value being that a medical intervention can be performed without adverse reactions (e.g. pain, movement) from a normally conscious patient. For succinctness we will consider a sub-capability a single action of administering a medical injection. We will explore the interaction between the anaesthetist, the syringe and the patient. Using our capability terminology (Michell, 2011) the anaesthetist is the active agent driving a passive resource (the syringe) to execute the capability C_i of injecting an anaesthetic drug into a patient via a process/action P_i that results in the value of 'patient anaesthetised and can be safely operated on'. In the diagram we denote the affordance by the term A_{xy} where *A* is the affordance and *x* and *y* refer to the notation for the two elements in the affordance pair. We use an underline notation to refer to the fact that an affordance relates to a vector dynamic transforming action or a static force balance and hence has magnitude and direction. We use a line terminated by balls in the shape of a dumb bell with each ball resting of the relevant element or component to show the relationship between the two components of the affordance system. The syringe is an example of a designed device (Brown and Blessing, 2005) and can be considered as a system comprising 3 components (excluding cap) a plunger, body and needle which fit together as a passive system of affordances as in the figure below:

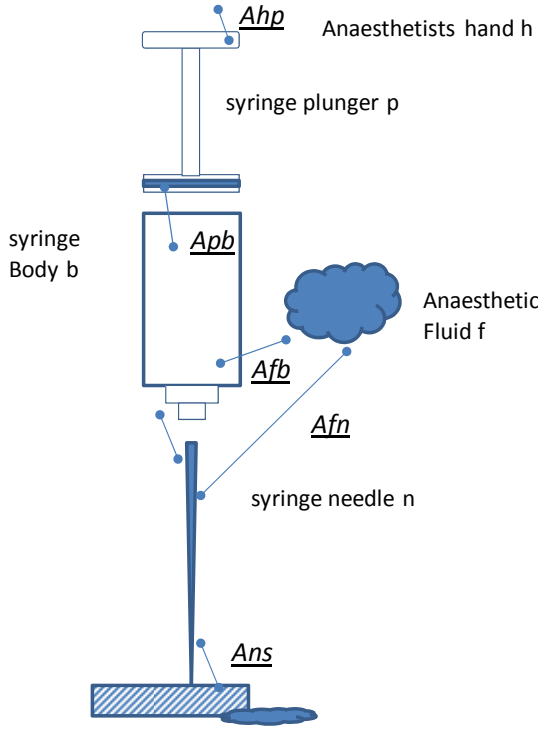


Figure 2: Syringe Affordance Pairs.

3.2 The Affordance Chain (AC) and Objective Affordances

Maier and Fadel identified that multiple affordances can be associated with a particular subsystem (Maier and Fadel, 2006). Thus the affordance of the syringe is a combination of the affordance of the component subsystems comprising the syringe i.e. Af (passive) = f{Af_s} i.e. the passive affordances of the syringe relate to the mechanical structure and engineering mechanics of the syringe in terms of physical science e.g. Newton's three laws. Reviewing the capability model we can see that there are a number of passive objective affordances. The Plunger-Body Affordance – A_{pb} ensures the plunger is retained in the barrel and slides easily up and down it by the affordance pair predicated on the fact that the plunger is designed to interface with the barrel and has only one degree of freedom in an axial direction. The affordance mechanism AM depends on friction and the affordance path is the axial plunger motion. In the Hand-Plunger affordance A_{hp}, the plunger has a tube (with a seal) and top to it which supports the affordance of 'holding' the top of plunger in order to exert downward force (A_{hp}). This downward force acts on any fluid contained by the syringe body (A_{pb}). This affordance (affords force transmission to fluid) is dependent on the properties of interaction of the plunger and body. The plunger must slide and be a secure fit vs. body, must be made out of materials not affected by the fluid to be injected etc. The act of compressing an incompressible fluid forces the fluid out through needle via an affordance Af_n i.e. the needle – fluid affordance. An affordance property of the needle (affords fluid transportation) is that its hollow tubular nature directs the anaesthetic and is chemically inert vs the fluid (a potential negative affordance). A further affordance exists between the needle tip and the patient's skin Af_{ns}. The small surface area at the point enabling a small force on the plunger to be safely and effectively transmitted (due to mechanism of mechanics i.e. pressure = force/area) as a large pressure on the patient skin surface affords penetration of the skin. However this of course is not possible without the subjective affordances of the active driving resource, the anaesthetist who is responsible for identifying the target vein and ensuring the tip of the needle enters a vein etc.

We define an *affordance chain* (AC) as *a connected and related set of affordances (and hence affordance mechanisms) acting together as a system at a point in time to achieve a specific goal of enabling a new macro affordance mechanism, i.e. the capability*; Ca = AC = f ($A_{1,2}$, $A_{2,3}$, $A_{3,4}$ …$A_{i,j}$) where j= i+1 in a sequence from driver to final resource. The idea of affordance chain allows us to link the micro level affordances to the actions that we define as part of the process referred to in our definition of capability.

3.3 Subjective Systems of Affordances

There are three affordance pairs within the capability that depend on the interaction of a passive with a driving active resource. Aph refers to affordance between the plunger and the anaesthetist's hand. By definition of the syringe structure the plunger must be depressed to work according to its design parameters. The needle in contact with the patient's skin provides another affordance pair Ans. One half of the affordance pair comprising the anaesthetist and patient can both act dynamically, deontically and independently, unlike the passive syringe which requires interaction with both these active resources, one of which – the anaesthetist, is driving the overall capability transformation. Another affordance may be the anaesthetists second hand resting on the patient to steady the needle – this is a subjective agent affordance, but is not illustrated for clarity. The third affordance pair Afp is the interaction between the anaesthetic fluid f and the

patient's body physiology p. Here the driving resource is the anaesthetic which has a sleep inducing transformation mechanism, if of the right fluid composition and if the delivery or action path – i.e. into a vein is efficacious.

4 LINKING CAPABILITIES AND AFFORDANCES

Based on the above discussion we can rewrite our original capability model:

Capability = f (resource interaction, process of interaction) i.e. any capability depends on a transformation based on the affordance mechanism and an affordance path, both of which are functions of the set of resources $\{R_i\}$ needed to deliver the capability: Capability = f (Affordance mechanism $AM(R_i)$, Affordance path $AP(R_i)$).

4.1 Affordance Mechanisms

The affordance mechanism should describe the potential transformation mechanism at the interface between the two object resource systems R_p and R_q and its properties that enable the transformation. Mathematically the affordance mechanism is a transfer function that describes the transformation in terms of input resource properties and output resource properties that defines the transfer between input and output qualities. Affordances are actualised by an affordance mechanism AM that represents the transformation. Our affordance mechanism transfer function relates to what Turvey calls *Function j* in his crystal refraction (Turvey, 1992). He shows a light ray *Z* capable of refraction (q) interacts with a crystal *X* capable of refracting the ray (p) to give transformation of light ray to a refracted or bent ray and a new position as a result of a Function j – which we call the affordance mechanism which can be represented as a mathematical function or refraction equation. However, this transformation is subject to critical affordance factors f_c being within the correct window to enable the affordance to actualise or manifest itself. Refraction is dependent on the affordance path i.e. if light ray strikes crystal to obliquely it will be reflected – dependent on the critical affordance factor – the angle of the light with respect to the normal to the crystal surface. Kornhauser's example (Turvey, 1992) of the affordance actualisation of bird flying into a branch depends on the affordance mechanism for 'fracturing'. This is the law of conservation of momentum that transfers the bird's momentum to the branch and results in an impact reaction - impacts its organs. However the actualisation also depends on the affordance path where exactly the bird hits the branch, how far it travels and what absorbs the momentum.

4.1.1 The Critical Affordance Factor

The bird's velocity (assuming a straight unimpeded flight) has a safety envelope that determines unsafe momentum transfer. This should include a lower level of speed to ensure the bird does not miss the branch. The speed is the critical affordance factor in the affordance mechanism that will determine the safe actualisation of the affordance. In Warren's stair climbing the riser height is a critical affordance factor (Warren, 1984).

4.2 Affordance Paths (AP)

The affordance path should describe how the resource systems must be acted upon to be brought together to enable the interface transformation mechanism specified by the affordance to occur. It is the path followed by actions to enable affordances.

An affordance path AP is the set of possible actions that could be taken to enable the affordance mechanisms to act and execute the capability.

A capability affordance path is a process or sequence of actions necessary to realise the capability. It refers not just to a set of closed actions (like the affordance chain), but a process of connected sequential actions that typically occur in a business i.e. linked ordered but discrete actions as part of a business activity. This will comprise physical or substantive actions e.g. the act of injecting an anaesthetic as well as non substantive i.e. semiological actions (Stamper et al, 2000) e.g. verbal communication or inspection actions by the anaesthetist as a control on the action of injecting and assurance of the resulting value/goal achievement. We can say:

- AP is set of actions within an ordered sequence;
- AP contains actions represented affordance chains;
- AP contains semiological actions.

Affordance path = $\sum$ substantive affordance chains + semiological affordance actions

The Affordance path also results in a change of the variables $AP = f \triangle a \rightarrow a'$

This relates to Luck and d'Inverno's total plan. I.e. Affordance Path AP =*Total plan* = seq *Action*

Note affordance paths P relate to Brown et al's (Brown and Blessing, 2005) 'operations' to preserve the link to capability. The anaesthetist has an almost infinite range of possible affordance paths. Consider the ways in which the substantive action of injection can be accomplished ie he can hold the syringe with one hand on the barrel and simultaneously resting on the patient to steady it and the other on the plunger. We can simply prod the patient with the syringe etc. The syringe has 1 degree of mechanical freedom – the plunger slides along the axis to afford the delivery of the fluid/anaesthetic. The passive structural affordances (A_{pb}, A_{bn}) at once constrain the motion and also effect the delivery of the capability value i.e. injecting the anaesthetic into the patient. The quality of the action path chosen will impact the efficacy of the affordance mechanism and determine how well the capability transformation is achieved. That is C = f (AM,AP) where AP is the set of substantive and semiological actions as shown in Figure 3.

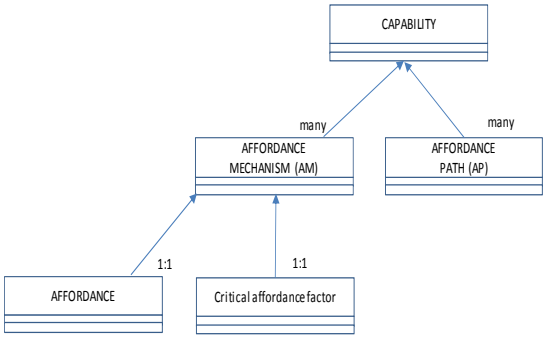


Figure 3: Affordance Mechanism and Path.

5 CAPABILITY-AFFORDANCES: A MEDICAL CASE STUDY

5.1 Drug Injection Capability

Based on structured interviews conducted at a health trust hospital (see acknowledgements) the following analysis was developed. The Capability C_i of injecting an anaesthetic drug into a patient via a process/action P_i results in the value v of 'patient anaesthetised and can be safely operated on'. Capability = f (resource affordance, process/actions of executing the affordance) = f (adding business value). But this capability results from four interacting systems, the anaesthetist, the anaesthetic fluid, the syringe and the patient. The syringe as a designed object resource can be further decomposed into its active components. The syringe is a set of interacting passive objective affordances where the laws of physics hold the assembled component affordances together in a state of equilibrium (ie objective affordance as we have seen). At a more macro or capability model level of what we can perceive, we can say; Capability of anaesthetising a patient = f {resource} = f (A-anaesthetist, S-syringe, F-anaesthetic, P-patient). The capability is dependent on a tuple or affordance chain by which the anaesthetist acts on the syringe which in turn acts on the fluid which in turn acts on the patient through a number of transformation mechanisms. It also relates to further affordance paths related to the semiological affordances involved in inspecting the patient and controlling the syringe action. The syringe affordance chain can be decomposed into its mechanisms and actions as seen in Figure 4.

5.2 Affordance Path & Mode of Action

The syringe tool has a designed way of being used – ie a set of preferred potential affordance paths. The syringe is a passive, but designed resource or a device Ds and must be guided by the surgeon in a specific designed way, at a certain angle to achieve results to meet the desired goal. This relates to refers to Chandrasekaran and Josephson's the mode M(Ds,E) (Chandrasekaran and Josephson, 2000) of deployment of the syringe device. The mode may come from tacit best practice knowledge or following procedures or documents defining usage procedures. For example, the mode of holding the syringe and injecting, the mode of the anaesthetic etc. Hence the mode refers to an optimum affordance path for the affordance chain ACs of the syringe: Mode *M* (Ds, E)= optimum f(affordance paths P_1-P_n OR actions mentioned earlier). But the optimum mode is achieved by the Anaesthetist's sensor affordances (sight/touch etc) and his tacit knowledge/conceptual model of the environment. The mode defines certain action behaviours to ensure the affordance mechanism is optimised eg: ensure syringe is in the correct location to meet a vein, it doesn't slip (negative affordance) and it reaches the correct depth to interact with the vein.

5.3 The Affordance Diagram - AMN

As affordances are pervasive we propose a simple affordance modelling notation AMN to provide

Affordance type	Affordance Ref.	Affordance Description	Affordance Mechanism (AM)	Affordance Path (AP)	Variables changed	Critical affordance factor fa
dynamic	*Ahp*	plunger affords pushing and grasping	apply force Fhp on plunger acting along axis of plunger	move plunger to end of travel (eg1 cm)	plunger location relative to body	Fhp, f1 <Fhp<f2 sufficient to smoothly move plunger& overcome friction
dynamic	*Apb*	Affords sliding and force transmission	frictional sliding force of plunger seal on syringe body	slider slides 1cm	bottom of body axially to end of travel	plunger travel to limit of body l
dynamic	*Afb*	Affords containment and force transmission	fluid force containment and expansion towards needle tube	fluid moves down body	fluid moves from body to needle	bursting pressure of plunger and tube
static	*Afn*	Affords fluid path transmission	fluid force expansion towards needle tip/skin	fluid moves down needle tube	pressure on fluid	bursting pressure of needle tube
static	*Ans*	affords skin penetration	pressure = force/area at the tip, where force Fns achieves steady penetration	needle penetrates skin to a depth d	skin penetrated	force magnitude Fns, f1 <Fns<f2
dynamic	*Afp*	Affords anaesthesia (anaesthetic body interface)	induced sleep via chemical interaction between anaesthetic and nerve/brain centres	anaesthetic induced sleep	conscious => unconscious	anaesthetic reaction kinetic factor tba

Figure 4: Anaesthetic Affordance Table.

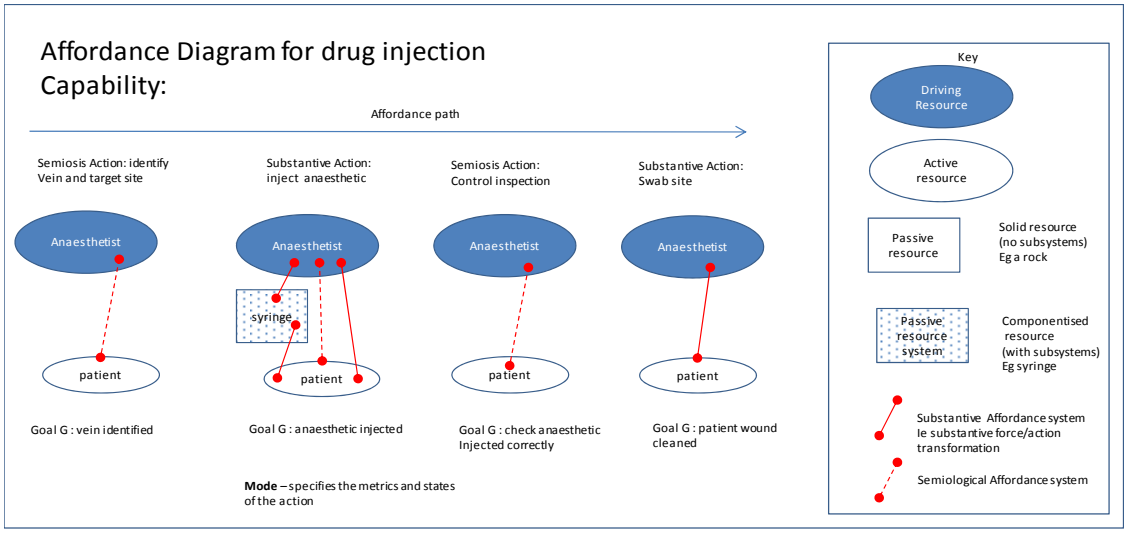


Figure 5: Anaesthetic Affordance Diagram.

succinct visibility of the affordance and system interactions. We use ellipses for active resources and differentiate between driving agent resources as a solid colour. Passive resources are represented as boxes with designed passive resources made from components identified by a shaded box. We also distinguish between action affordances modelled using the solid dumb-bell notation and intangible semiosis driven affordances involving perception e.g. anaesthetist identification of vein and control actions via a dotted line as on Figure 5.

5.4 Mode Representation

The mode specifies the ideal and optimised metrics and states of the action involving a designed object resource, e.g. the syringe as represented by procedures for using the syringe. Similar procedures are expected to exist for the identification of the vein and control inspection of the injection process, although these may be in the form of tacit knowledge and experience. The sequence of actions and the mode actually used vs the optimised mode in the procedures may vary greatly (and does in practice between anaesthetists). This corresponds to the anaesthetist using the syringe in a different way and perhaps using different resource affordances, which may not deliver the desired transformation goal of anaesthetised patient and the value they require to perform the medical intervention.

6 AFFORDANCE QUALITY, EFFICIENCY AND EFFECTIVENESS

This section explores how we can measure the performance of the transformation and the execution of the capability. The perceived benefit from the

executed affordance and the value added by the executed capability, depend on the way the affordance is executed by the driving agent, i.e. in the process or action of the agent in following the prescribed mode. In each case different parts of the syringe and potentially different designed and non-designed syringe properties may take part in the action, even though the overall capability (to inject an anaesthetic) remains the same. In the interests of medical safety there are appropriate mode guidelines to follow that describe the optimum mode. However the decision on how the substantive action of injection is executed and the affordance path that makes up the capability is deontic – an obligation. The set of actions are subject to variation depending on the behaviour of the anaesthetist. This leads to variation in the efficiency and effectiveness of the capability of the anaesthetist which we will review later.

6.1 Efficiency: Ey

An *efficient action* is *an action (transformation) that uses minimum energy compared with other actions* (Natarajan et al, 1996). E.g., if the drug is over diluted it will be inefficient i.e. drug property has insufficient strength to cause anaesthesia. We define efficiency as the ratio of energy usefully employed vs the total energy employed in an action (or sequence of actions). Affordances which utilise the designed mode M of the device, or that of a specified action or behaviour would be expected to be optimally efficient. In terms of affordances: Negative affordances (e.g. errors) are ineffective i.e. unplanned/unwanted . We can attribute a number (0-1) as a measure of affordance efficiency to represent how close to the design mode of the device or procedure the affordance is. Then the efficiency of an affordance chain is simply the scalar product of the affordance efficiencies of the affordance pairs of the parts of the chain. For the syringe example the overall efficiency for any affordance chain 1-n is Ey of Af (1-n) = E Af12* EAf23*EAf3n.

For the syringe example the efficiency of the injection process depends on the efficiency of the affordance chain between the anaesthetist and the patient receiving the anaesthetic fluid. This involves the respective affordance efficiency of the hand-plunger, the plunger-body, the fluid-body, the fluid needle, the needle-skin and the fluid- patient affordance pairs. The overall efficiency of the chain is the capability efficiency= ∑efficiency of affordance chain actions + semiological action
Eci = EAhp*EApb*EAfb*EAfn*EAns*EAfp.

6.2 Effectiveness: Es

We can say that actions that meet requirements e.g. process goals are effective (Muchiri et al, 2010). *Actions that don't meet process goals are ineffective*. For example, if the wrong drug is used the injection is ineffective. As we have seen, actions depend on the affordance mechanism and the quality of the affordance interface that enables the mechanism to be executed. We can say: a) negative affordances are ineffective (unplanned/unwanted); b) positive affordances not required for the goal are ineffective. Effectiveness therefore refers to the proportion of positive affordances vs total number of affordances experienced. We define Effectiveness Esb as:
Effectiveness = ratio of effective actions/ (effective + ineffective actions)

To calculate the effectiveness of an affordance chain we need to: a) identify effective affordances Afe and b) identify ineffective affordances Afi.

The Effectiveness of an affordance chain is:
Capability effectiveness = sum(Afe)/(sum Afe = Afi).

6.3 Quality: Qas

The overall performance of any system *s* can be defined as its qualify Qs. Quality relates to performance to specifications. Optimal action specifications relate to the mode of operation as we have seen earlier. Quality also relates to performing actions that meet the overall action or affordance chain goals i.e. effectiveness

We therefore define quality of the affordance system as as the scalar product of the efficiency and effectiveness ratios ie Qas= Ey *Es. Hence the quality of an affordance chain of i elements is the Qc = ∑ Eyi*Esi.

6.4 Capability Quality, Efficiency, Effectiveness

We note that the above discussion shows the quality, efficiency, effectiveness of an affordance chain. Affordance chains relate to composite actions where all the resource objects are interacting. However, our original process level definition of capability refers to a sequence of actions. For example the capability of the process anaesthetise patient will include the inject patient action (the composite affordance chain), but also other actions eg communication and message passing between agents and waiting periods where affordances are not active. In this case The capability refers to the

affordance path and hence the overall capability quality, efficiency and effectiveness will relate to the affordance chain efficiencies.

7 SUMMARY AND CONCLUSIONS

In this paper we have analysed the original resource capability model in terms of affordance theory of Gibson et al based on a realist approach and the use of *Z* notation. We have shown that capability can be modelled as a tuple of a set of resource affordance mechanism and actions that are dependent on one or more critical affordance factors and how these relate to the work of Turvey and others. The paper has identified and developed the concept of objective and subjective affordances and showed how these relate to the capability resource model and existing theories. We identified an affordance chain of subjective affordances by which affordances work together to enable an action and an affordance path that links action affordances to create a capability. We introduced the affordance modelling notation as a visual method of representing the affordances in a system of interacting resources. A medical case study was used to show how capabilities can be identified and modelled using the theory of affordances. We also proposed a model of affordance efficiency, effectiveness and quality that enables the performance of a capability and its constituent affordances to be measured and modelled.

7.1 Further Work

Further work is now needed to identify the practicality of the approach and to test the method in detail with larger capability sets and including more semiological affordances. Also to provide a complete worked example of affordance mechanism and path and a measure of capability from the system of affordances that constitute it.

ACKNOWLEDGEMENTS

The author would like to acknowledge the support of the Royal Berkshire Foundation Trust Simulation Centre for his Honorary Senior Lectureship in Health Informatics. Also to acknowledge his colleague Dr Debbie Rosenorn-Lanng Director of the Simulation Centre for the opportunity to witness and analyse medical procedures.

REFERENCES

Beimborn, D., Martin S.F., Homann U. 2005. Capability Oriented Modelling of the Firm. IPSI Conference .

Brown D.C., Blessing L. The relationship between function and affordance. Proceedings of IDETC CIE ASME 2005 2005

Chandrasekaran, B. and J.R. Josephson (2000) "Function in Device Representation." Engineering with Computers 16: 162-177.

Curtis, B., Fefley, W. E., Miller, S., 1995. Overview of the people capability maturity model. CMUSEI 95 MM01.

Gaver.W.W Technology affordances. In Proceedings of the SIGCHI conference on Human factors incomputing systems: Reaching through technology, pages 79–84. ACM New York, NY, USA, 1991

Grant, R.M., 1991. The Resource-Based Theory of Competitive Advantage- Implications for Strategy Formulation. California Management Review

Hafeez, K., Zhang, Y., Malak, N., 2002. Determining key capabilities of a firm using analytic hierarchy process. Int. J. Production Economics 76 39–51CS9 (Josey et al 2009)

Luck M, and d'Inverno 2001 M., A Conceptual Framework fir Agent Definition and Development. The Computer Journal.

Liu, K., Sun, L., Jambari, D., Michell, V., Chong S., 2011. A Design of Business-Technology Alignment Consulting Framework. CAiSE'11.

Maier J.R.A. Fadel G.M 2006 Affordance Based Design-Status & Promise

Merrifield, R., Calhoun J., Stevens D., 2008. The next revolution in productivity. Harvard Business Review.

Michell, V. 2011 A focussed approach to business capability. BMSD 2011 Bulgaria.

Montesano, L.; Lopes, M.; Bernardino, A.; Santos-Victor, J.; , "Affordances, development and imitation," Development and Learning, 2007. ICDL 2007. IEEE 6th International Conference on , vol., no., pp.270-275, 11-13 July 2007

Muchiri, P. et al. Development of maintenance function,performance indicators and measurement frame work.International Journal of Production Economics,v. 131, n. 1, p. 295-302, 2011. http:// dx.doi.org/ 10.1016/j.ijpe.2010.04.039

Natarajan, V Cleaveland R. An Algebraic Theory of Process Efficiency. Proceedingsof LICS'96, pages 63-72, IEEE, 1996.

Norman, D. (2004). Affordances and design. Retrieved 15 May2012from http://www.liacs.nl/~fverbeek/courses/ hci/AffordancesandDesign.pdf

Nöth, W. (1998). Ecosemiotics. Sign Systems Studies, 26:332–343.

Ortmann, J. and Kuhn, W. (2010). Affordances as Qualities. In Galton, A. and Mizoguchi,R., editor, Formal Ontology in Information Systems Proceedings of theSixth International Conference (FOIS 2010), volume 209 of Frontiers in ArtificialIntelligence and Applications, pages 117–130, Amsterdam Berlin Tokyo Washington, DC. IOS Press.

Ortmann J. and Kuhn, W. (2002) Afforded Actions http://ifgi.unimuenster.de/~j_ortm02/publications/Afforded_Actions_Manuscript.pdf.

Smith, B. and Mark, D. M. (1998). Ontology and geographic kinds. In Proceedings International Symposium on Spatial Data Handling, Vancouver, Canada. 12-15 July.

Stamper, R.K. and Liu, K. 1994, Organisational dynamics, social norms and informationsystems, in Proc. HICSS-27, Los Alamitos (IEEE Computer Society Press), VI: 645-654.

Strong D.M., Volkoff O., Johnson S.A., . On, I., Pelletier L.R., On I., Tulu, B., Kashyap, N., Tridel J., Garber L., EHR Affordance Actualisation Theory

A. Stoytchev, "Behavior-grounded representation of tool affordances, "presented at the Int. Conf. Robot. Autom., Barcelona, Spain, 2005.

Weigand, H., P. Johannesson, et al. (2006). On the Notion of Value Object Advanced Information Systems Engineering. E. Dubois and K. Pohl, Springer Berlin / Heidelberg. 4001: 321-335

You H & Chen K. (2003)A Comparison of Affordance Concepts and product semantics Asian Design Conference.

Discovering Data Quality Issues in Service-oriented Architectures
A Framework and Case Study

Plamen Petkov, Markus Helfert and Thoa Pham
School of Computing, Dublin City University,
Glasnevin, Dublin 9, Ireland
{ppetkov, markus.helfert, thoa.pham}@computing.dcu.ie

Keywords: Service Oriented Architectures, Service Composition, Data Quality, Data Monitor, Business Rules, Business Process, Quality of Service.

Abstract: In this paper we examine web services from a data quality perspective. Based on a data quality management approach, we propose a framework for analysing data produced by the composite service execution. Apart from other monitoring tools available which targeting the technical aspect of service composition – service time response, service throughput and etc., we propose data consistency and accuracy monitoring, focusing on the business value of the data produces by the services. We developed framework that will store business rules into a rules repository. By analysing service data against the rules we will be able to identify problems in service composition and execution. Moreover taking into account the Quality of Service (QoS) we are able to provide an approximate location of the error.

1 INTRODUCTION

As a result of the global networks, information systems progress, and the need for flexibility, traditional closed, static and centralized architectures have evolved to dynamic and heterogeneous. The tasks of developing completely new applications, making certain adaptors for legacy systems, or rewriting present applications are now outdated. Principally boosted by Web services connectivity, service-oriented architectures (SOA) are now considered the preferred way to designing an information system. SOA endeavours to provide existing functions of an information system as "services" that can be accessed in a loosely coupled way (Papazoglou, 2007), independently from the technical platform. The architecture is seen as an orchestration of requests for those services. Generally, In SOA, workflow or orchestration processes are fundamental.

However in more complex architectures orchestrating the services can be difficult to handle. There is no efficient way to managing such architectures without having the awareness of the data, processes and events running within the enterprise environment. To support the process of orchestrating, as well as development and evolving progress, a monitor tool(s) must be integrated. These tools, of course, must comply with business requirements, in order to achieve adequate surveillance result.

In other words, management tools and techniques are inadequate without using an appropriate monitoring. Thus why, crucial assistant for proficient and effective deployment and operation of an SOA-based net-centric system is a comprehensive monitoring capability. Nevertheless, present monitoring solutions fall short with respect to such systems because they do not hold the capabilities to implicitly aggregate metrics, effectively detect inconsistent or inaccurate data, and so to provide comprehensive shared situational perception.

In this paper we propose data quality monitoring approach by developing framework that will be able to identify data quality problems.

The remainder of the paper will be structured as follows: Section 2 describes the importance of Quality of service (QoS) for service selection. In this section we also present some data quality issues related with service composition. In Section 3 we propose framework that will identify and localize the data related problems. In section 4 deals with simple case study and discussion of our framework. Finally, conclusions are presented along with suggestions for future work.

2 TOWARD TO MONITORING UTILIY IN SOA

2.1 Importance of Service Selection

Web services are the key technology in SOA, in which services are considered as "autonomous, platform-independent entities that can be described, published, discovered, and loosely couple in novel ways" (Papazoglou, 2007). A service oriented application includes a service provider and a service requester. A service discovery agency (e.g. Universal Description Discovery and Integration UDDI)) may act as intermediate between provider and requester and provides functionality to promote available services. The service provider defines a service description and publishes it (to the agency). After retrieving a suitable service, the service requester is able to invoke that service (The SoA Open Group). In this regard, service composition encompasses the process of searching and discovering relevant services, selecting suitable web services of best quality and finally composing these services to achieve an overall goal that usually in a business context aims to support an underlying business process.

The Publish – Discover – Binding schema is depicted below.

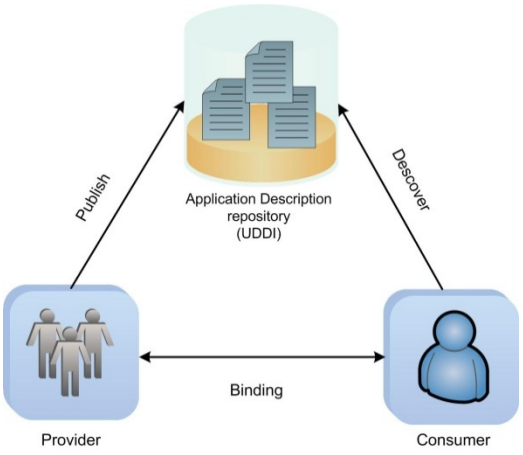


Figure 1: Publish – Discover – Binding.

On reviewing prominent approaches for service discovery, it appears they mainly involve functional attributes of service advertised in the service description. These include service type, operation name, input/output data format and semantics (Zeng, 2003). In order to select suitable services, quality of service (QoS) evaluation is usually used as approach for service selection among many services of similar functionality. In literature, many approaches have been proposed to measure QoS with non-functional quality criteria. QoS dimensions often refer to non-functional criteria that include execution price, execution duration, reputation, reliability and availability (Jeong, 2009).

Meanwhile the functional quality of a set of composed services receives little attention. Often, it is assumed that the service functions execute according to the stated and published service description. However, as with any execution and operation of applications, this may not be the case. Indeed, discussions with practitioners show, web services often do not fulfil the functional quality and thus the expected output is not achieved. In contrast to other research, we consider this problem and provide a framework that can help to detect some of the problems during the execution of the services.

Service selection is crucial stage in service composition and QoS act as blueprint of SoA reliability. QoS could also play significant role into service composition monitoring as we will show later in this paper.

2.2 Data Quality Issues in Web Service Composition

Data can be considered as output/input product of service orientated composition and execution. The product quality is key factor for entrepreneurs and customers. Therefore the quality of data component is the most important in a SoA project. Moreover, data can be manipulated by different parties like portals, devices or even orchestration engine. This means that data quality is mirror reflection of the quality of overall SoA implementation. Many researchers have been done investigations about defining Data Dimensions (Wand 1996), Data Quality Requirements Analysis and Modelling (Wang, 1998). However most of these studies apply to monolithic information systems. Since the introduction of the composite architecture few researcher take into account data quality issues (Thoa and Helfert, 2011).

Incorporating quality issues into web service composition and execution (WSCE), is a major problem regarding the application integrity. (Fishman, 2009) represented the problem with bad data quality in SoA by comparing the different services with animals and human – they are different by nature but share the same diseases. In other words, integrating an application into SoA, which does not comply with the business rules and thus provides inaccurate or inconsistent data, can affect

all other applications and cause cumulative effect of errors typical for system development lifecycle. Figure 2 illustrates the service composition and service incompliance.

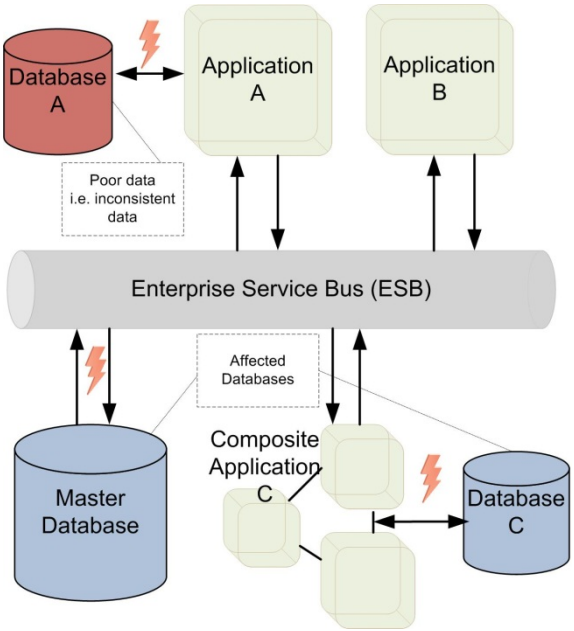


Figure 2: Data incorporation in SoA framework.

As it can be seen from the diagram, incompliance at service level (Application A) can 'spread' all irrelevant data through the Service Enterprise Bus to all other application and its databases (Master enterprise data, Composite Application C and its database D). Note that Application B has not its own database but it supplies the whole infrastructure with functionalities and thus it has been involved indirectly in the process of distributing poor data – e.g. affecting master enterprise database.

The data quality literature explored the most four dimensions of data quality which are accuracy, completeness, consistency, and timeliness (Wand, 1996).Corresponding to the functional quality of WSCE, in this paper we will explore these dimensions of data quality (Table 1). Analysing those dimensions help to clarify the causes of poor data quality in WSCE.

Taking into account quality issues in WSCE, many approaches for static or dynamic web service composition and execution have been developed (Agarwal, Narendra, Silva, Zhang). However, quality issues of mapping functional requirements to service composition and execution are current.

Table 1: Quality Dimensions and problem description.

Quality Dimension	Problem description
Accuracy	The data value does not correctly reflect the real-world condition.
Consistency	The data values persist from a particular data element of the data source to another data element in a second data source. Consistency can also reflect the regular use of standardized values, particularly in descriptive elements
Completeness	Data element is always required to be populated and not defaulted Data element required based on the condition of another data element
Timeliness	When two source data systems are synchronized outside of a shared transaction boundary, there exists a timing window during which a service accessing both systems may encounter mismatched data. The entity represents the most current information resulting from the output of a business event.

3 FRAMEWORK FOR ANALYSING DATA QUALITY

The data quality analysing process will be separated in two stages – problem discovery stage (1) and problem localization stage (2).

In order to execute the process above, our proposed framework is spitted into two modules. The first module will be able to detect a data quality issues. It follows a business rules-based approach to data quality. Based on detected problem by the first module and the QoS properties of the services, the second module will be able to recommend a particular service where the problem stems from. A simple block schema along with the analysing process is given on the Figure 3.

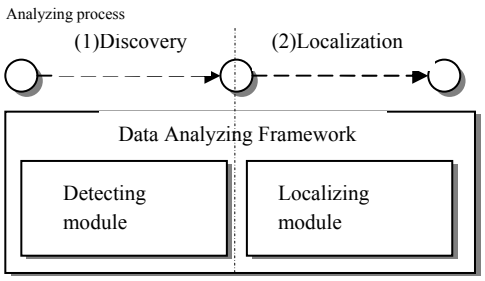


Figure 3: Data Analysing Framework.

3.1 Data Quality Issue Detecting Framework

The proposed framework follows a business rules-based approach to data quality. Business rules are "statements that defines or constrains some aspects of a business" A business rule is normally described by the natural language, which can be reconstructed in form of Event-Condition-Action (Bubenko and Herbst), in form of If-Then expression, or in form of <Subject> Must <constraints> (Morgan, 2002). Based on our earlier work (Thoa and Helfert, 2011) and the approach above, we developed a framework for service composition that is presented in Figure 4.

In the next few paragraphs we will give more detailed overview of our detecting module.

Business Process:
This block represents the conceptual business process (BP) model that a service composition must comply with. A BP model can be described with BPMN, UML or EPC model. In our framework, we are interested in the input and output of data of activities/tasks in the BP. A part of the meta-data of the BP model is stored in the Service Mapping and Rule Repository (Figure 3).

Business Rules Specification:
This component concerns specifying business rules. A business rule is related to one or many activities/tasks in a business process and/or data objects (1 and 2). The specified rules then are stored in a Rule repository which could be relational databases or XML files (3). The rule is usually in form of If <Boolean logic expression>* Then <Boolean logic expression>*, or an assertion of aforementioned. <Boolean logic expression>* can be composed of one or many Boolean logic expressions combined together with logical operators i.e. <Boolean logic expression> = <left expression> <comparison operator> <Right expression>. The left and right expression can be mathematical formula, including data values, data attributes, mathematical operations or aggregation functions. Moreover logic expression consisting of more logic expressions can be presented as binary tree.

Service Mapping Repository:
This repository captures the mapping between services to be composed to tasks/activities defined in the underlying business process. A task corresponds usually to a service, and a service can correspond to one or many tasks. However in the case that a task corresponds to many services, the service composition is significantly more complex and is currently not subject of this study. The information is usually stored within the service composition phase (8).

Service Log:
The service log captures specific events occurred during the service execution. In our framework we are particularly interested in events related to data updates and changes. It is necessary to detect what service instance in what composite service instance writes what data to the database. Since in the most service compositions the access to the service

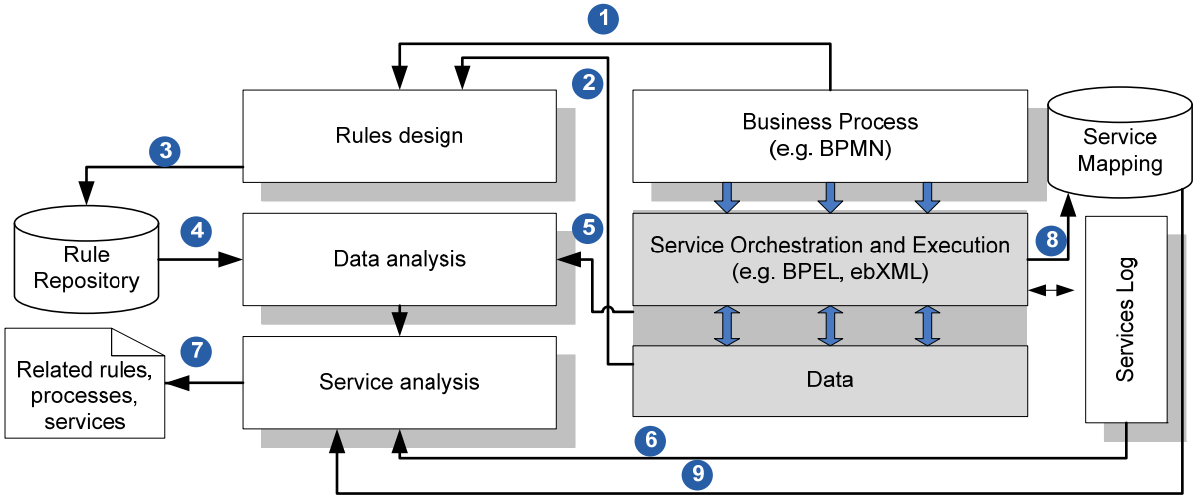


Figure 4: Data problem discovery framework.

database is indirect, we cannot mine the data directly Therefore our framework will work with the already 'digested; data from service interface described with WSDL. All these information should be stored in the service log (9).

Although there are approaches to specify log formats (Gaaloul, 2008), current approaches such as the Common Log Format and Combined Log Format of W3C are not sufficient for our approach as they are not directly able to represent the required information. Therefore, we propose a practical oriented log file. The log file entries contain following information:

```
<Entry>
<srv_location> …</Srv_location>
<Service_ID>… </Service_ID>
<instance_ID>… </ instance_ID>
<endPointName>… </endPointName>
<operationName>… </operationName>
<dataEntityName>… </dataEntityName>
<Data Value>...</Data Value>
</Entry>
```

The XML template can handle all data needed for recording a problem. `<endPointName>` tag will store the interface name of the service, while `<operationName>` tag will store the function delivering the error.

Data and Service Analysing

This component analyses the data produces by the endpoint of the service against the rule repository. Information about the service and data content produced is reported using Service log template.

3.2 Problem Localization Framework

Problem localization process (figure 3 (2)) is a part of overall data quality analysing process. This process aims to provide approximate problem location, since the problem can involve two or more services. In fact, in service orientation environments it is very likely that the problem is either in one or another service.

In order to localize the potential error giving service, we will propose localization framework will provides us with estimate solution. To do so, we will use the service log delivered by the problem detection framework during the data issues discoverer stage (figure 3 (1)). Moreover we will use the Quality of Service (QoS) properties of the 'infected' services to decide which one is more likely to be the source of the problem.

Base on approach above, we developed a framework that is represented in Figure 5.

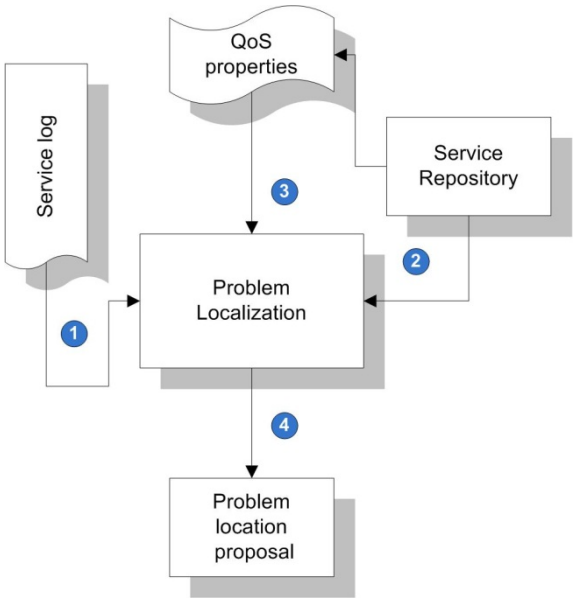


Figure 5: Framework for data problem localization.

The following paragraphs describe the main elements of our framework.

Service Log

In section 3.1 we proposed a framework for detecting quality issues and notating the services that produce data into a XML log file. We also proposed exhaustive log format that provides our localization module with needed information.

The service log is been generated during the first stage (discovery stage) of analysing process and acts as a starting point for this framework. Logged data then is passed to the 'Problem localization' element (1) to be processed.

Service Repository and QoS Properties

Service Repository in a place where services descriptions are store. Usually this repository is called Universal Description Discovery and Integration (UDDI) and supplies the 'Problem localization' with service information (2).

Quality of Service block is vital in this framework. This element contains all list with all QoS properties concerning given Service(s). QoS dimensions often refer different criteria but most of them include execution price, execution duration, availability and reliability. Reliability criteria are maybe one of the most important one, in order to localize objectively the problem. By reliability we mean, least error productivity and least time that

service is out-of-order. For the sake of brevity, we will not discuss QoS criteria further.

The QoS criteria are granted (3) to the localization element for further treatment.

Problem Localization

Problem localization is the core element in our structure. It provides the framework with the following functionalities:

- collects the necessary data from the *Service log and Service repository* along with appropriate *Quality of Service properties* i.e. out-of-order delivery
- performing a comparison (based on QoS criteria) of the problematic services.
- prioritise the services using '*the least reliable, the more likely to be problematic*' method.
- ability to tracing a problem
- working up a detailed report and proposing approximate problem location(4).

As you can perceive, because of the duality nature of the problems, our methodology can provide only rough solution of the problems. Therefore we strongly recommend manually examination on the generated by localisation framework report and further investigation of problems.

In the next chapter we will apply our data analysing framework to given simple 'package booking' study case.

4 CASE STUDY AND DISCUSSION

The following case study illustrates the developed framework with a common service that provides the booking of travelling packages. The case study is motivated by a real case study. Initially, customers search information about available travel packages, and then subsequently may book a flight and a hotel. Once the booking is completed, the user pays the total amount and confirms the booking. Alternatively the user may cancel the booking. We developed a service oriented application for Travel package booking and analyzing data quality of the application along the two phases: preparation and analyzing.

Initially, a conceptual business process model for the booking travelling package is modelled. Next, the service composition process is realized; the mapping information between the tasks of the BP and individual services are stored. We suppose this is a design time service composition. The service composition can be described with BPEL based on the orchestration depicted in Figure 6.

The composite service *Booking package* is composed of a set of available services: *Book Flight* service, *Book Hotel* service, and *Payment* service. The flows of data/message between services are also described in the orchestration.

Mapping services and tasks in BP are as following table:

Table 2: Service mapping.

7 ° ¹Œ#	6ˇ ß ó Ł ″ ˇ #
6ˇ°ß″ª # ŒÆª -#	%ł ł Œ # ŒÆª -#
%ł ł Œ # ŒÆª -#	#
6ˇ°ß″ª # ł –ˇ ‘#	6ˇ°ß″ª # ł –ˇ ‘#
%ł ł Œ # ł –ˇ ‘#	#
3 °Êæ ˇ ı -#	3 °Êæ ˇ ı -#

4.1 Analysing: Problem Discovery Stage

Once the study case framework is deployed and all service contacts are made we are ready to move to analysing stage and problem discovery stage. Mind that business rules sored into rule repository are design according to the business model,

For our case scenario we propose the following rules:

R1: If the booking is confirmed then the payment must be fully paid. This rule relates to Payment task and Confirm booking task (see Figure 6) and is described with pseudo logic predicate language as follows:

```
If PackageBooking.Status =
'confirmed' then
Payment.Status='Full_Paid'
```

R2 and **R3** relate to the Booking task and Payment task. These rules state that if the Flight or Hotels is booked, it must be paid:

R2: `If Flight.Status = 'booked' then Payment.Status='Paid'`

R3: `If Hotel.Status = 'booked' then Payment.Status='Paid'`

Once the rules are set up the needed data is ready to be retrieved through services endpoints. Then this data is analysed against the rules we composed.

If there is any data that violates a rule, then the related operation will be identified based on information stored in the rule repository. For example, we will focus on **R1** and suppose there are incorrect data produced of the services it scope.

Table 3: Service Log Report.

Service location	Service ID	Service instance ID	Service Endpoint	Service Operation	Data value	Record
url://	BookingPackage	3	BookingEndpoint	getBookingStatus()	'confirmed'	1055
url://	Payment	1	PaymenBookngEndpoint	getPaymentStatus()	'unpaid'	300014

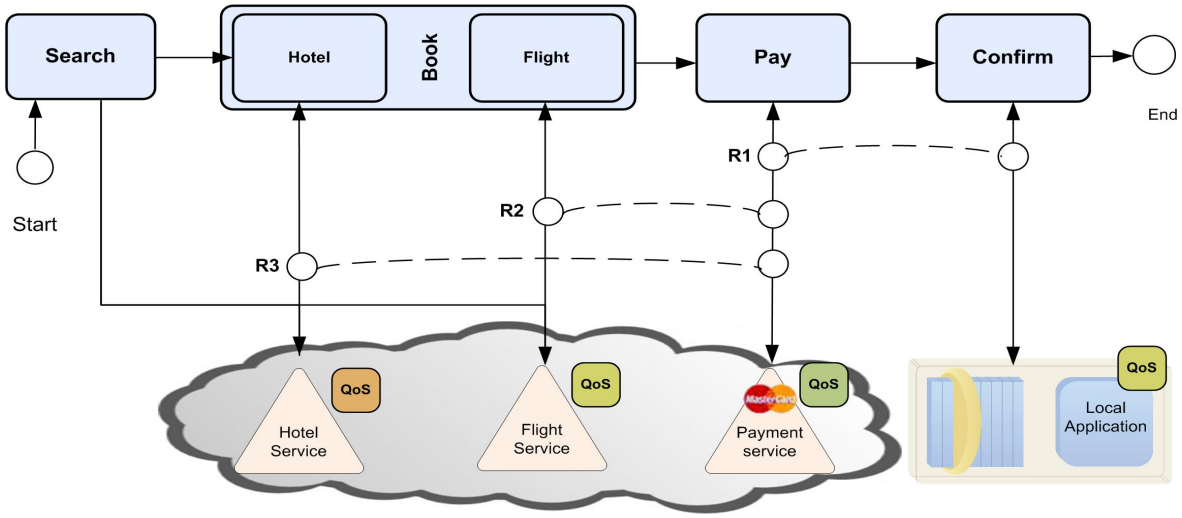


Figure 6: Booking package service composition.

Assume there is a confirmed Booking which is not fully paid. This relates to two service operation; one related to the Booking (the local one) service and the other to the Payment service. The rule R1 is related to the tasks Payment and Confirm booking. A log extraction of the occurred problem discovery is displayed in Table 3.

The service log file records that during the discovery stage the service *BookingPackage* returns through its endpoint booking status of *'confirmed'* while Payment service returns value of '*unpaid'*.

From the above information, we can identify that the cause of the incorrect data is related to the *Payment* service and the *BookingPackage* composite service.

Although this is not enough to identify which of the listed services is the problematic one. That is why we will put into further investigation our generated log file in next stage – 'Problem localization'.

4.2 Analysing: Problem Localisation Stage

In this stage we will examine produced XML log in the discovery stage, while taking into account the QoS criteria of involved services. Based on detected problem by the first module and the QoS properties of the services, the second module will be able to recommend a particular service where the problem stems from.

It is very important that Quality of Service criteria apply to all services which are part of the problem. It is also essential that QoS criteria are chosen in the context of the services and the business process. Failing to fulfil the latter requirements will result in invalid estimation by the error localization module.

In our framework we will adhere to service reliability QoS criteria. Reliability property of a service can be number of errors generated for certain period of time or time that service is out-of-order.

In this study case scenario we will take into account *number of erroneous transactions per 10000 units.*

We want to remark that this is only example criteria and there are many other criteria that services can be compared. This is very important in cases where two services have similar indexes. Bearing in mind the aforementioned QoS property and the affected services we apply our problem localization framework. Problem localization then

performs a comparison based on the selected criteria and prioritise the services using '*the least reliable, the more likely to be problematic'* method. Applying to 'booking package' scenario, it will generate a report which is given in Table 4.

Table 4: QoS Report Chart.

Service name	QoS*	Priority
Booking Package	107/10000	1
Payment	12/10000	2

* incorrect transactions per 10000 units

As the QoS dimensions may differs, for every QoS criteria a new chat will be generated. The table above shows that QoS measured for booking package service is 107 incorrect transactions per 10000 committed while payment service gives only 12 per 10000. This comparison makes payment service more trustworthy than the booking package service. Therefore the source of the problem is more likely to be the Booking package Service.

Despite of our efforts to deliver approximate location of the problem, we do not disregard the chance that the error may occur in the more trustworthy service. That is why we encourage for manual investigation by the system analytic.

5 CONCLUSIONS

Inspired from research in the area of data quality and service oriented architectures, in this paper we have presented a framework for monitoring web service composition and execution. More specifically, we have separated the monitoring process into two sub processes, namely 'problem discovery' and 'problem localization'.

We have proposed a framework and illustrated relevantly every sub-process. The framework was demonstrated using a case study.

Our problem discovery framework follows a data quality management approach and incorporates business rules concept. The core of latter is based on the comparison of the business rules and the data output of the services. Problem localization framework, on the other hand, uses the output of the first framework and the functional Quality of Service criteria to provide system analytic with approximately location of the problem.

Our approach differs from others well known approaches by inspecting data delivered by the services and Quality of Service properties.

In future we aim to improve the service log technique in the discovery module as well as expand some quality of service criteria used in localization stage. We also aim to apply the concept in further case studies.

ACKNOWLEDGMENTS

This work was supported by the Irish Research Council for Science, Engineering and Technology (IRCSET) under the Postgraduate Scholarship Scheme.

REFERENCES

Agarwal V, Chafle G, Mittal S, Sribastava B (2008) *Understanding Approaches for Web Service Composition and Execution*, COMPUTE '08 Proceedings of the 1st Bangalore Annual Compute Conference, ACM

Bubenko J, Jr & Wangler B (1993) *Objectives driven capture of business rules and of information systems requirements*. Proceedings of the International Conference on Systems, Man and Cybernetics, 670

Fishman, Neal A. *Viral Data in SOA: An Enterprise Pandemic*. IBM Press, 2009.

Gaaloul W, Baïna K, Godart C (2008) *Log-based mining technique applied to web service composition reengineering, Service Oriented Computing and Applications 2*, pp.93-110.

Herbst H (1995) *A meta-model for specifying business rules in system analysis*. Proceedings of CaiSE'95, 186–199.

Jeong B, Cho H, Lee C (2009) *On the functionality quality of service (FQoS) to discover and compose interoperable web services, Expert Systems with Applications* 36, pp.5411-5418.

Morgan T (2002) *Business Rules and Information Systems: Aligning IT with Business Goals*, Addison-Wesley, Boston, MA.

Narendra NC, Orriens B (2007) *Modelling Web Service Composition and Execution via a Requirement-Driven Approach*, ACM SAC'07, Korea.

Papazoglou M.P, Traverso P, Dustdar S, Leymann F (2007) *Service-Oriented Computing: State of the Art and Research Challenges*. IEEE Computer, November, 64-71.

Silva E, Pires LF, Sinderen MV (2009) *On the support of Dynamic Service Composition at Runtime*, Springer-Verlag, ICSOC'09.

The Business Rules Group: www.thebusinessrules group.org

The Open Group: Service Oriented Architecture, http:// www.opengroup.org/projects/soa/

Thoa P., Helfert M. Monitoring Information Quality within Web Service Composition and Execution. Dublin City University, ISD 2011.

Zhang D (2004) *Web Service Composition for Process Management in E-Business*. Journal of Computer Information Systems pp.16-18.

Zeng L, Benatallah B, Dumas M, Kalagnanam J, Sheng QZ (2003) *Quality Driven Web Service Composition*, ACM WWW.

Wand Y, Wang R. (1996) *Anchoring Data Quality Dimensions in Ontological Foundations,* Communications of the ACM, November 1996. pp. 86–95

Wang R (1998*) A product perspective on total data quality management*. Communication of ACM 41

Modeling the Design of Value in Service-oriented Business Models

Arash Golnam[1], Paavo Ritala [2], Vijay Viswanathan[1], Valerian Hanser[1] and Alain Wegmann[1]
[1]*Ecole Polytechnique Fédérale de Lausanne, School of Computer and Communication, Sciences (I&C), Systemic Modeling Laboratory (LAMS), Station 14, CH-1015 Lausanne, Switzerland*
[2]*Lappeenranta University of Technology, School of Business PO Box 20, FI-53851 Lappeenranta, Finland*
{arash.golnam, vijay.viswanathan, valerian.hanser, alain.wegmann}@epfl.ch, ritala@lut.fi

Keywords: Amazon.com, Modeling, Service-Oriented Business Models, Value Capture, Value Creation.

Abstract: Many firms redesign their business models to be service-oriented in light of the increasingly central role that services play in their business models. Two fundamental questions should be addressed in designing service-oriented business models: *how is value created for and with the customers by the service provider?* and, *how is the value captured by the service provider?*. The first question deals with "value creation" while the second addresses "value capture" in the "service value equation". A service-oriented business model that addresses these two questions can sustain the viability and competitiveness of the firm as a service provider. The extant research mainly focuses on the service design from the value creation perspective. Thereby, there has been little discussion about service providers' value capture and its trade off with value created for and with service customers. In this paper, adapting a holistic perspective, we introduce a modeling framework that can assist in understanding, analysis and design of value (i.e. value creation and capture and their interplay) in service-oriented business models. Our modeling framework is grounded in insights and conceptualizations of the extant theories, constructs and frameworks on value creation and capture in business and service systems. We illustrate the applicability of our framework by conducting a descriptive case study of the value creation and capture in Amazon service system in the period between 1997 and 2001.

1 INTRODUCTION

A business model is defined as a generic platform between strategy and practice, describing the design or architecture of the value creation, delivery, and capture mechanisms the firm employs (e.g. Teece, 2010). Due to the increasing and even focal role of services in their businesses and strategy, many firms have been forced to completely re-think their business models (Teece, 2010). In fact, this recent tendency of business model redesign has led to the emergence of "service-oriented business models". This development can be explained from the perspective of "service-dominant (S-D) logic" (Vargo and Lusch, 2004 and 2008), that attempts to view and extend the concept of service beyond a "particular" kind of intangible good as traditionally viewed in the "goods-dominant (G-D) logic". S-D perspective conceptualizes a firm's offerings not as an output, but as an input for the customer's value-creation process.

Central to the service-oriented business models are the concepts of value creation and capture. In order to understand how a service-oriented business model remains viable and competitive, two fundamental questions should be addressed: "how is value created for and with the customers by the service provider?" and, "how is the value captured by the service provider" (for discussion, see e.g. Grönroos and Ravald, 2011; Bowman and Ambrosini, 2011; Pitelis, 2009; Ritala et al., 2011). In the search for understanding such questions, the extant research has developed value modeling frameworks such as (Gordijn and Akkermans, 2003; Weigand et al. 2009; Pijpers and Gordijn, 2007; Yu, 1997; Weigand, 2009; Osterwalder and Pigneur, 2010) that provide conceptual tools to support the design of service offerings. However, such tools and framework mainly address the service design from the service customers' perspective and do not sufficiently address suppliers' value capture in the "service value equation". The same gap can be broadly identified in the service literature in general,

where value creation and co-creation issues have been emphasized over value capture. In addition, the interplay between value creation for and with customers and value capture by the suppliers has not been explicitly investigated in the design and analysis of service offering in service-oriented business models.

In this study, we propose a holistic approach that takes into account both value creation (for and with customers) and value capture (by service providers) in order to fully understand and model the new logic of service provisioning process in service-oriented business models. To this end, our research aims to provide a modeling framework that can assist in understanding, analysis and design of value (i.e. value creation and capture and their interplay) in service-oriented business models. We illustrate the applicability of our framework by means of a descriptive case study of the value creation and capture in Amazon.com service system. In a descriptive case study, the researcher pursues to describe a phenomenon of interest that occurs within the data. This type of research begins with an a priori theoretical perspective. Then, a pattern matching is conducted to describe the phenomenon in the data in a rigorous way (Yin, 2009). More specifically, the descriptive case we conduct can be labelled as an instrumental case study (Stake, 1995), where we aim to illustrate the applicability of the suggested framework. The case study focuses on one of the services offered by Amazon.com, more particularly; the sales of used and new books in Amazon.com over the period 1997-2001.

We have used data triangulation in order to gather rich evidence on Amazon.com, various aspects of its business model and its service offerings over time. We began the data gathering process in January 2009. Since then, a variety of secondary data sources have been accessed, analyzed and synthesized in order to gain an accurate understanding of diverse facets of Amazon.com's service offerings and implementation in Amazon Marketplace. Such sources include:

- Amazon.com annual reports between 1997–2010 (Amazon, 2011a); presentations and news releases (Amazon, 2011b).
- Books published on Amazon.com such as; Afuah and Tucci, 2002; Spector, 2002; Kalpanik and Zheng, 2011), etc.
- Harvard Business Review (HBR) cases published between 2000 and 2010 such as (Applegate 2002 and 2008)
- Journal articles such as (Heck and Vervest, 2007), etc.

There are several advantages in using secondary sources. For instance, in Ambrosini et al. (2010) suggest that teaching cases are an unexploited and rich source of data that should be used when primary data is not available. They also suggested using reputable sources for teaching cases (we mainly use Harvard Business Review cases here) and combine it with other sources to attain data triangulation. Analyzing multiple sources of objective and subjective evidence has enabled us to combine evidence in a way that gives an overall understanding of the research topic.

The paper is structured as follows. In Section 2 we present a conceptual model that summarizes the theoretical insights and perspectives on value creation and capture. In Section 3, after a brief introduction to Amazon.com we represent the design of value Amazon.com's business model applying our value modeling framework. Section 4 includes the related work and in section 5, we present the conclusion and the future work.

2 THE CONCEPTUAL MODEL

In this section, we develop a theoretical framework examining value creation and capture in service systems. The theoretical insights are presented in form of a conceptual model illustrated in Figure 1. In the following, we first discuss the tenets of customer value creation and then we proceed to examine how the service provider eventually captures value.

2.1 Customer Value Creation Conceptualizations

Creating value for the customers is the fundamental reason why any company exists and thrives in competition (e.g. Bowman and Ambrosini, 2000), and customer value creation is most pronounced in service-oriented companies (Chesbrough and Spohrer, 2006). Customer value creation is a process, where the service provider delivers the customer a service offering that creates value when the customer uses the service, i.e. the use value (see e.g. Grönroos and Ravald, 2011). In this setting, the service provider (and its value network) is responsible in producing the service, and actual customer value (co-)creation takes place when customer receives/uses the service (ibid.). The main interface where the service provider can affect customer value creation (e.g. time saving convenience) is through a concrete service offering

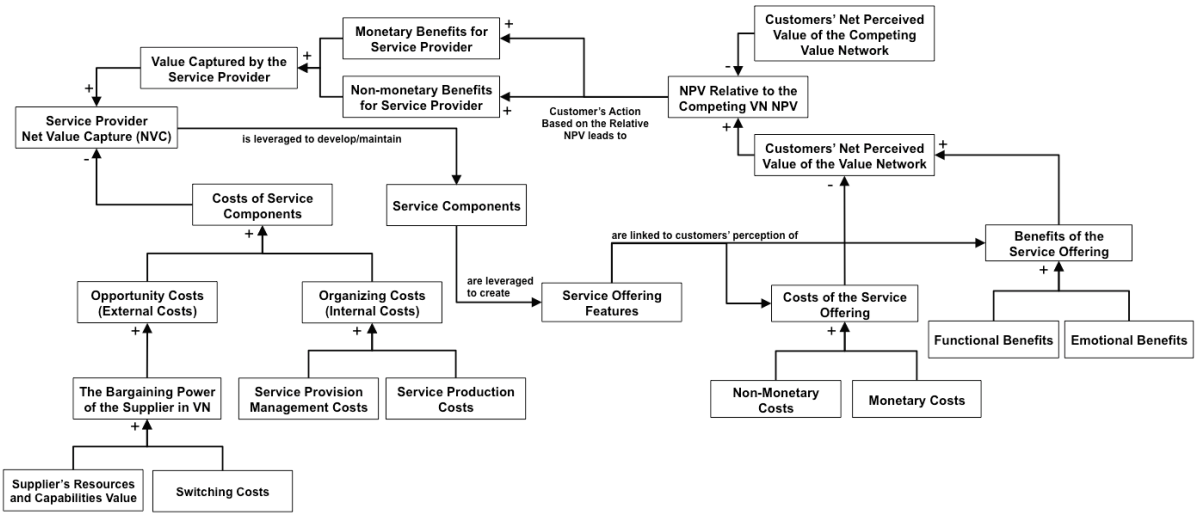


Figure 1: The conceptual model.

(e.g. transportation, entertainment). In order to provide any type service offering, the provider has a set of service components that are leveraged to create the service offering (Golnam et al., 2012). These components are created by the service provider and its value network, and they reflect the underlying resources and capabilities that are put to use to provide a certain set of service components. Thus, service components can be seen as a way of organizing the service, while service offering features are those that are linked to the actual customer's perceptions of service value.

2.1.1 Net Perceived Value (NPV)

In understanding customer value creation from the customer perspective, the net perceived value has been seen as a key concept, which is related to the overall benefits minus the costs of receiving the service (e.g. Kotler, 2000; Day, 1990; Huber, 2001). A related concept is the consumer surplus, in layman's terms often expressed as "value for money", that Bowman and Ambrosini (2000) define more precisely as the difference between the monetary amount the consumer is willing to pay and the actual price paid. It is important to recognize that consumer surplus or net perceived value is assessed ex-ante, i.e. prior to the transaction. This is precisely the reason why benefits and costs are assessed as "perceived"; this is in contrast to complementary concepts such as consumer satisfaction, which are ex-post. For instance, if a service offering consists of entertainment services, the customer perceives a certain value for being entertained, while costs of receiving it are linked to e.g. to time spent to going to the venue, as well as the monetary costs involved.

Thus, in any situation where a transaction actually occurs it is expected that net perceived value will be positive. That is, the customer is willing to pay an amount in excess of the costs (including monetary and non-monetary costs), and thus made the purchase, pocketing the "surplus". The larger this surplus is the more eager the consumer will be to make the purchase; the converse is also true, the smaller this surplus becomes, the less eager the consumer is in willing to engage in the transaction. The borderline situation is that of the monopoly supplier, where the firm is able to charge exactly the maximum amount the consumer is willing to pay, thus netting zero surplus for the consumer. Therefore net perceived value can only increase through one of the following situations: (1) an increase in perceived benefits while maintaining perceived costs unchanged; (2) a decrease in perceived costs while maintaining perceived benefits unchanged; or (3) a simultaneous increase in perceived benefits with a decrease in perceived costs.

2.1.2 Customers' Perception of Service Offering's Benefits

Customer's perceptions of the benefits are related to the use value of the service for the customer (e.g. Grönroos and Ravald, 2011). Use value covers the specific characteristics of the product or service perceived by the customer as potentially serving their needs. Bowman and Ambrosini (2000) emphasize the subjective nature of use value - it maps uniquely to each customer. Use value itself can

be further categorized into two sub-components, namely functional and emotional benefits.

Functional benefits represent the tangible benefits of the product or service that fulfill the primary needs the consumer had in seeking the solution, and Grönroos (2000) calls this the "core value" of the service. Kotler's (2000) pinpoints that these benefits – although functional – are expressed in customer terms, further reinforcing their subjective nature. Furthermore, as discussed by Amabile (1996), customers make their subjective assessment of appropriateness of the functional benefit of the service. In the majority of cases, where the product and context are well understood and established, the process is straightforward. However, in cases of innovation and disruptive products, or change in social and cultural context, buyers might not be able to properly make their assessment, resulting in a net negative impact on functional benefits.

Emotional benefits are made up of the intangible extras that the firm is able to offer that go above and beyond meeting primary needs; the analogous terminology of Grönroos (2000) is added value. Kotler (2000) highlights various specific strands of these types of benefits, such as personal interaction value and image value. Groth (1994) also suggests that customers buy products and services for other than just "pure [i.e. functional] utilitarian reasons". He provides the example of consumers not assigning significant value to near-perfect replications of famous art work as a case-in-point. Groth terms this kind of utility, serving the psychic needs of people, as an exclusive value premium (EVP).

2.1.3 Customers' Perceptions of Service Offering's Costs

In addition to various types of benefits, there are always costs incurring to the customers of receiving a service. The extant literature details the many types of such costs. The most obvious is the actual monetary cost (i.e. exchange value, Bowman and Ambrosini., 2000) In addition, it is also important to take into account the non-monetary costs. Regarding these, Kotler (2000) identifies three other varieties: time, energy, and psychic costs. Time cost is made up by the sum of durations the consumer has to spend in acquiring and acquainting oneself with the product or service. Energy cost is the net of energy that needs to be expended by the customer. Finally psychic costs form a complement to psychic utility - the cognitive stress experienced by the customer in purchasing and using the product.

2.1.4 Competing Value Networks and the Relative Net Perceived Value

In addition to the value created by the focal firm and its value network, the net perceived value created by competing value networks' offerings should also be taken into account. In analyzing this, we refer to relative net perceived value, which is the net perceived value created by the focal firm's offering in relation to the competing offerings. The higher the relative net perceived value is, the higher is the competitiveness of the focal firm in the eyes of the customers.

2.2 Service Provider Value Capture Conceptualizations

Value capture (also termed as value appropriation or retention in some sources) by the focal firm is an issue of much interest in management research and even more so in organizations themselves. Value capture is related to the actualized profit-making of a certain party. Regarding this, an in-depth discussion is provided by Bowman and Ambrosini. (2000) where they address the importance of analytical distinction between value creation and capture. Lepak et al. (2007) also makes a point of mentioning that "the process of value creation is often confused or confounded with the process of value capture or value retention" and that the two should be understood as distinct processes.

While there is certainly a strong correlation between the two, it is essential to recognize the former neither automatically nor fully translates into the latter. Bowman and Ambrosini (2000) argue that while value is created for the customer by organizational members (i.e. the value network), value capture has a different set of determinants, including "perceived power relationships between economic actors" (in other words, the bargaining power between the firm and other entities, which is explored at depth below). Lepak et al. (2007) and Ritala and Hurmelinna-Laukkanen (2009) follow a similar line of argumentation, suggesting that only through the use of specific mechanisms is the creator of value able to capture it, and that value creation and capture may have sometimes have completely different determinants and timeframes.

2.2.1 Net Captured Value (NCV)

In our model, value capture by the service provided is determined by the benefits/compensation it can extract from the markets. Furthermore, the net

captured value (NCV) of the service provider consists of two factors: the benefits for the service provider minus the costs of service components. It is notable that this view is symmetrical to the customer side where the net perceived value of the customer is also dependent on benefits and costs. However, the perspective is different in that the service provider is the producer of the value (which incurs costs), and is receiving various types of compensation for doing that.

In our model, the benefit side of net value capture is fundamentally affected by customer's action, which are based on the net value of the service as perceived by the customer. Customer's action means the activities that result in generating more or less tangible (e.g. annual subscription fee) and intangible (e.g. referrals, word-of-mouth, loyalty) contributions by the customer for the service provider as a compensation for the net perceived value of the service offering. From the service provider's perspective, these actions then lead to actual monetary and non-monetary benefits, which are discussed next, and are followed by the discussion on the costs of providing the service.

2.2.2 Benefits for the Service Provider

Benefits from the service provider range from direct monetary benefits (i.e. revenue streams) to non-monetary benefits (e.g. customer loyalty, learning). While monetary benefits for the service provider are quite straightforward to interpret (e.g. bulk price, subscription fees etc.), the non-monetary benefits are more varied and ambiguous. This is partly because non-monetary benefits consist of non-negotiable value, which means that these types of compensation cannot be clearly agreed on between the parties. Ulaga (2003) proposes various types of non-negotiable value coming from the customers, such as commitment, trust, satisfaction, and loyalty. Of course non-negotiable values in and of themselves are not the ultimate end for profit-seeking firms. Thus arises the discussion as to conversion mechanisms for non-negotiable value into negotiable forms. Allee (2008) offers significant insight in this regard, offering two pathways that this conversion can take: (a) direct conversion into monetary value, and (b) an intermediate conversion into a negotiable form that can be bartered. For instance, customer loyalty involves major (non-monetary) benefits for the service provider, which may also contribute to the monetary benefits in both short and long term. In fact, customer loyalty manifests itself in the form of repeat purchases and is thus strongly linked with superior profits: Reicheld (1994) found out that "a small increase in customer retention leads to a major increase in net present value profits."

In addition, organizations learn by doing and thus constantly evolve themselves (e.g. Nelson and Winter, 1982). Thus, one type of non-monetary benefit is also linked to the organizational learning in the form of trial-and-error, customer feedback, and therefore improved service offerings. This type of value is highly non-negotiable, but it may translate into improved service offerings and value creation in the future.

In assessing the received benefits for the provider, the offerings of competing value networks should also be taken into account. Receiving both monetary and non-monetary benefits are linked to the customer's net perceived value relative to the competing offerings, since this determines the compensation customers are willing to pay (here: customer's actions) and contribute to compensate a particular service provider. Thus, issues concerning competitive pressure and competitors' offerings are important determinants on the eventual value capture. Regarding this, Lepak et al. (2007) explains that a consequence of competition is increased supply, which following fundamental economic principles, results in decrease in exchange value (i.e. price). We suggest competition also decreases the possibility of achieving non-monetary benefits, since the potential places of customer loyalty, learning and other benefits may be decreased if competitors are too attractive.

2.2.3 Costs of the Service Components

In addition to the monetary and non-monetary benefits for the service provider, the costs of providing the service components affects value capture. We divide these costs into two broad categories: the internal organizing costs, and the external opportunity costs. Combined, these costs decrease the value captured by the service provider.

First, the *organizing costs* refer to the internal costs of the service provider related to producing the service components. These costs are comprised of the production costs, related to producing firm's offerings, and the management costs, related to administration, control, monitoring, and incentives in organizing firm's operations (e.g. Masten et al., 1991; Blomqvist et al., 2002).

Second, in addition to the organizing costs, the value network includes costs dependent on the suppliers of various independent or jointly provided service components. Following Brandenburger and

Stuart (1996), we refer to the costs of the service components provided by the suppliers/partners in the service provider value network as *opportunity costs*. Opportunity cost is defined as the financial compensation provided to the suppliers in exchange to the service components they provide to the offering, also taking into account the highest alternative compensation that they could receive from utilizing their resources in other context (ibid.). Thus, the economic rationale of the suppliers' involvement in the service system is tied to the opportunity costs of the suppliers in providing certain service components. Opportunity cost is a widely-recognized economic concept that is a measurement of the best alternative passed up on. In this analytic context opportunity cost is the option a supplier foregoes in choosing instead to deal with the focal firm – and in effect this determines the eventual cost burden that needs to be taken into account when analyzing the costs related to maintaining the external network of suppliers and partners.

The key issue affecting the opportunity costs of the suppliers is the relative bargaining power. Indeed, Bowman and Ambrosini (2000) argue that a firm's ability to bargain with suppliers and buyers from a position of strength positively influences the value it is able to capture. Macdonald et al. (2004) reinforce this line of argumentation, with a formal model, stating that bargaining is what determines a firm's "precise" level of capture. Similarly Brandenburger and Stuart (1996) state that it is bargaining power between the "players" that determines the division of value, and further that bargaining power is what determines the price of exchange between supplier and firm. Simply put, the higher the bargaining power of a focal firm relative to its suppliers and partners, the lower are the eventual costs that it has to pay for suppliers to be involved in the value network, and vice versa. There are several issues that affect the relative bargaining power of actors. At its most basic level bargaining power is garnered by the relative value of resources and capabilities of different actors, determined by e.g. rarity, inimitability, and non-substitutability of those (see e.g. Barney, 1991). The relative bargaining power is also affected by the switching costs of the supplier. Switching costs is a general microeconomics concept identifying the redundant investment (monetary and otherwise) that a supplier needs to make when switching customers. Porter (1980) highlights the proportional relationship between high switching costs and high bargaining power. This means that the higher are the switching costs of suppliers, the higher is the relative bargaining power of the focal firm.

2.3 Net Captured Value and Future Value Creation

The cyclical feedback that net captured value offers to future value creation activity remains a relatively unexplored domain in the literature. However, we suggest that this should be taken into account when building a practically oriented model of value creation and capture. As the most evident issue, the actual monetary value and related resources (i.e. the revenue streams coming to service provider) directly help to maintain service providing activities in that they provide funding for the on-going operations.

In addition, and more important in longer term, is the development of value creation activities that take place over time. Lepak et al. (2007) touches on this point in his conclusion, suggesting that "a key question is whether actors learn from past value creation efforts in terms of the amount of value they capture and use this knowledge for decisions regarding future value creation activities." In other words organizational learning accumulated by value capture over time can guide a firm to better structure its value creation efforts.

Thus, we suggest that over time, there is a feedback loop from value capture to developing and maintaining service components. In terms of development of service components, the feedback loop is a result of organizational learning, leading to improved capabilities and resources related to service production. This can lead to either increasing the customers' value, or cost reduction on the service provider's side, or both. In general, these improvements can be linked to Porter's (1980) generic strategies of cost leadership and the second as differentiation.

3 MODELING THE DESIGN OF VALUE IN AMAZON.COM BUSINESS MODEL

In July 1995 Amazon.com began as an online bookseller and by September 1995, the company was selling $20,000 per week. After nearly three years as an online bookseller, the company began aggressively diversifying its offerings to include other product categories beyond books, initially adding music, videos, toys, and electronics (Afuah and Tucci, 2002). Such diversifications were followed

by the launch of several other stores such as home improvement software and etc. In parallel with such product diversifications, in October 1998, Amazon.com expanded geographically by launching its first international sites Amazon.co.uk and Amazon.de through the acquisition of UK-based online bookstore Bookpages and German-owned Telebook (Applegate, 2002). The rationale behind such diversifications was Amazon.com's strategy of "get big fast" to turn Amazon into the biggest mass merchandiser or E-mall in the online world (Spector, 2002).

Following its evolution from an online bookseller or to an e-tailer by diversifying its product offering through new store openings, Amazon.com extended its business model to become a third-party market place by launching Amazon Marketplace in November 2000. Marketplace idea was then implemented in Amazon.com's international websites, UK and Germany in 2002, and France, Canada and Japan in 2003.

In the case study analyzed in this paper, we focus on the Amazon.com's evolution from an online bookseller to a third-party Marketplace in the online bookselling segment. From a service perspective, we model the value creation and capture in Amazon.com's transition from selling new books to establishing a partnership with other booksellers to sell used and new books. To this end, we develop a value model representing the design of value creation and capture in Amazon.com business model circa 1997. In order to map our modeling framework to the theoretical discussions in the previous section and to gain a better understanding of the modeling constructs and notations, we present the model in three parts (i.e. customer value; customer value creation; and provider value capture) and explain each part step by step. Finally, we discuss the rationale behind changes in the business model of Amazon.com in 2001 in light of the theoretical insights embodied in our modeling framework.

3.1 Modeling Customer Value in Amazon.com's Business Model

The first part of our modeling framework deals with the service value attributes as perceived by Amazon.com's customers circa 1997. In Figure 2, we have listed a number of value attributes to reflect the perceptions of customers about the benefits and costs of Amazon.com's online book selling service. In Section 2.1, we discussed that a customer assesses a service based on its net perceived. The next step is to understand the relative importance of value

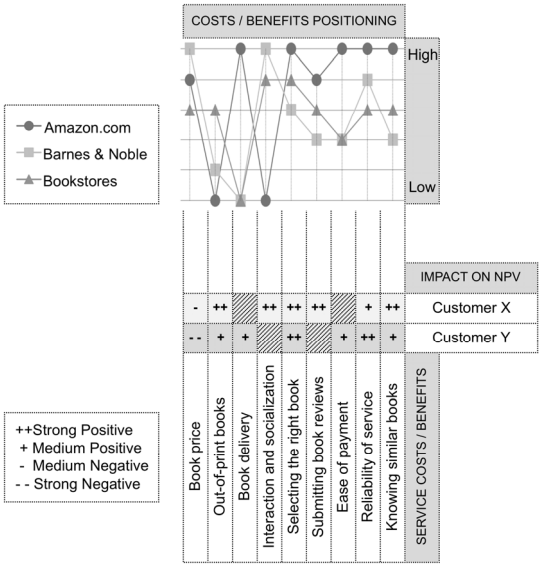


Figure 2: Modeling customer value.

attributes in terms of their impact on the net perceived value. As illustrated in Figure 2, we use minuses and pluses to represent the nature of impact (i.e. negative or positive) and its intensity (medium or strong).

Information on customers' perception and their relative importance can be gathered through direct interaction with customers or customer surveys. Revealed preference methodologies (Carson et al., 1996) can also be used to understand customer's needs and preferences based on their behavior. In this paper, the information provided on the value attributes the Amazon.com customers perceive and their relative importance has been gathered through the secondary sources outlined in Section 1.

As illustrated in Figure 2, different customers can perceive different value attributes of the service offered by the service provider. Similarly a value attribute can have different impacts on different customers. For instance as shown in Figure 2, the value attributes "Ease of payment" and "Book delivery" do not have any impact on Customer X's perception of Amazon.com's service offering. By the same token, "Submitting reviews" and "Interaction and socialization" do not influence Customer Y's perception of service value. Moreover, "Book price" and the "Reliability of service" are more important for Customer Y. Whereas, Customer X cares more about value attributes such as (availability of) "Out-of-print books" and "Knowing about similar books".

Finally, as already discussed in Section 2, it is important to identify the strategic positioning of the

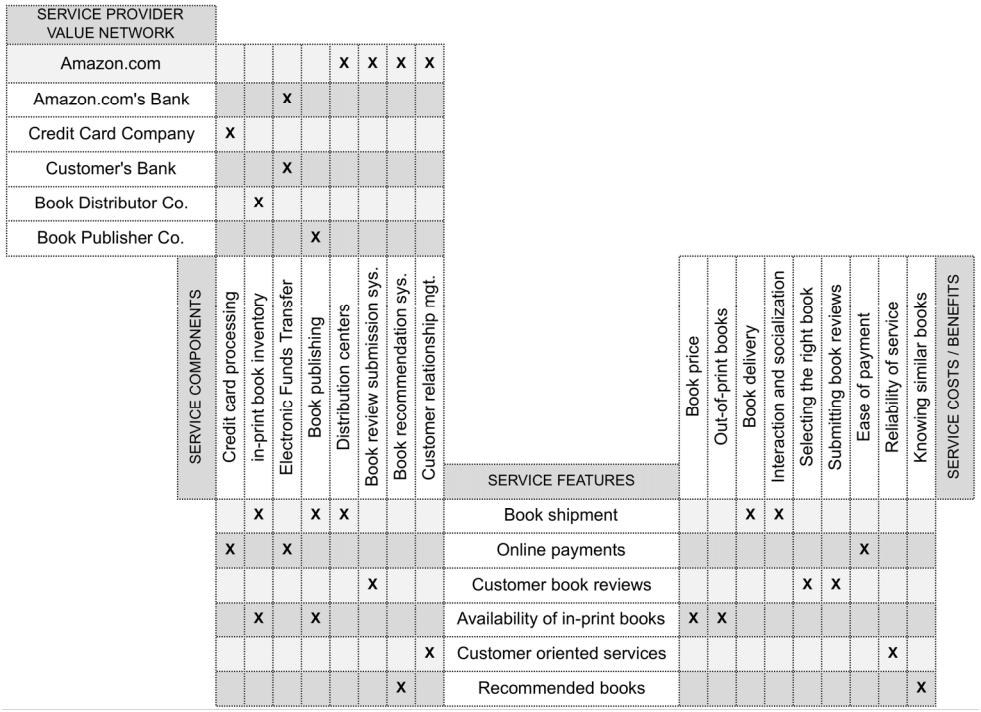


Figure 3: Modeling customer value creation.

service provider by understanding where the provider is standing relative to the competing value networks in terms of the value attributes. This assists the service provider in identifying the service improvement opportunities as well as analyzing whether delivering the perceived value attributes results in a competitive advantage. In our example we compare Amazon.com, Barnes & Noble and the Bookstores with respect to the value attributes listed in the model. By Bookstores we refer to small and independent bookstores that were not a part of the book superstores or chains such as Barnes and Noble or Borders. As illustrated, Bookstores were doing better in the price and availability of out-of-print books. Bookstores superiority in these two value dimensions was mainly due to selling used books.

3.2 Modeling Customer Value Creation in Amazon.com's Business Model

The previous section focused on the analysis of value attributes and their impact on net perceived value as well as the strategic positioning of the service relative to the competition. In this section we present the design of the value creation process.

Figure 3 illustrates the value creation process in Amazon.com business model wherein we model the service features created by the service components that are provided by the service provider and its value network and their corresponding value attributes. In the model, we put an X to map the service components to service features and service features to the value attributes. More concretely, we can see that for instance, Amazon.com provides the service component "Book recommendation system" which creates the service feature "Recommended books" that is linked to the value attribute "knowing similar books". Similarly, the Distributor Co. holds an "In-print book inventory" that creates the feature "Availability of in-print books" which pertains to the value attribute "Book delivery".

3.3 Modeling Service Provider Value Capture in Amazon.Com's Business Model

In Section 2 we explored different choices available to service providers to increase their net captured value (NCV). As discussed in section 2.2.2 the monetary and non-monetary benefits created by the customers determine the value captured by the service provider that increases the net value captured by the service supplier. The costs of the service components (i.e. organizing cost of service provider and the opportunity cost of the suppliers in the value network) reduce the net captured value by the service supplier.

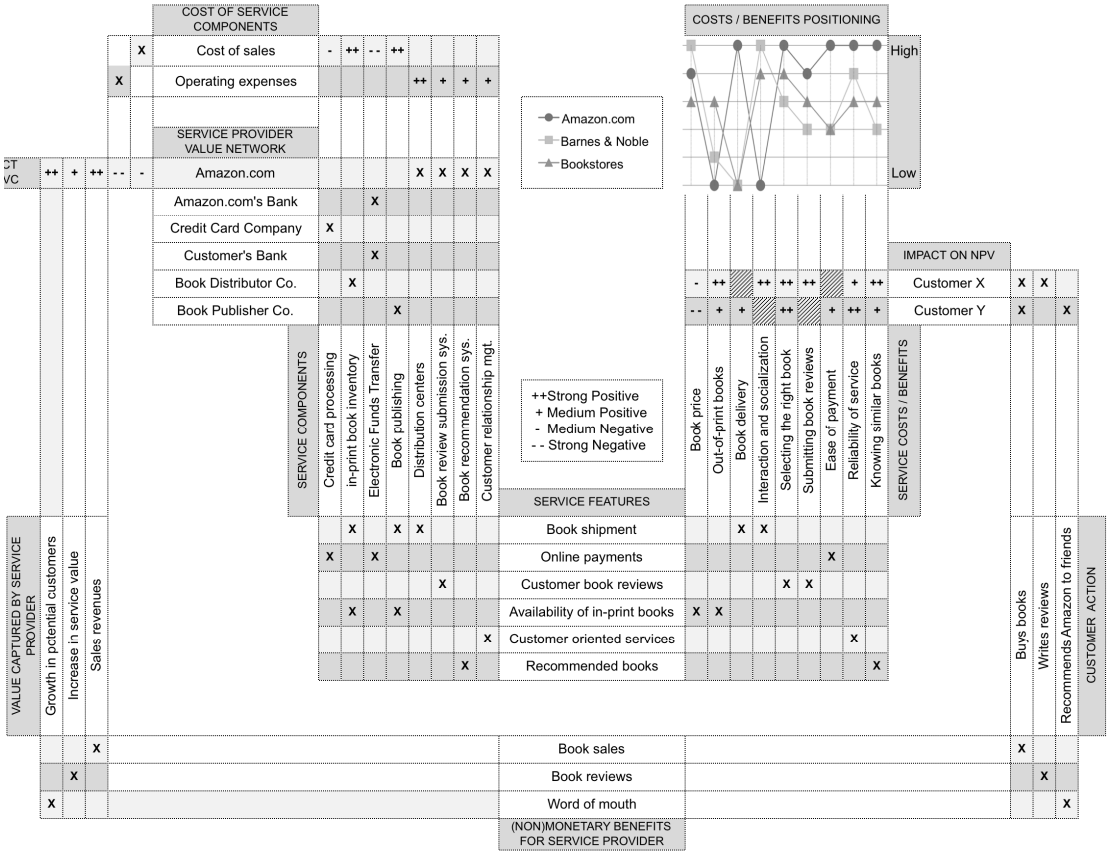


Figure 4: The overall modeling framework, including service provider value capture.

To model the service provider's net captured value we start our analysis from the customer side. As discussed in Section 2.2, the customers of the service offering take actions based on their perceptions of the net perceived value of the service provider relative to the competing offerings. As illustrated in Figure 4, both Customer X and Y buy books on Amazon.com. This action generates the monetary benefit of "Book sales" which leads to "Sales revenues" as the value captured by Amazon.com. As illustrated, revenues have a strong positive impact on Amazon.com's net captured value. Similarly, Customer X "Writes reviews" and generates the non-monetary benefit of "Book reviews" which results in an "Increase in the service value" of Amazon.com and thereby a higher net perceived value that can lead to more sales and revenues. Thereby, a non-monetary benefit can lead to the generation of monetary benefit by the passing of time. As shown in the model "Increase in the service value" has a medium positive impact on the net value captured by Amazon.com. Finally, Customer Y "Recommends Amazon.com to friends". The non-monetary benefit of "word of mouth" results in "Growth in potential customers" and a strong positive impact on Amazon.com net captured value by the passing of time. The gray background denotes that this impact will not occur immediately.

To model the cost of service components, we represent the "opportunity cost" and "organizing cost" *concepts* as elaborated in Section 2.2.3, by "cost of sales" and "operating expenses" *constructs*. We define these two indicators based on the definitions in the Amazon.com's annual reports 1997 – 2010 (Amazon, 2011b). As our study focuses on the book segment of Amazon.com's business, we modify these definitions to match the scope of our analysis.

- *Cost of sales* consists of the purchase price of the books sold by Amazon.com, inbound and outbound shipping charges to Amazon.com, packaging supplies, etc.
- *Operating expenses* comprise; marketing and sales expenses (i.e. advertising, promotional and public relations expenditures including the related expenses for personnel engaged in marketing, selling and fulfillment activities.

Product development expenses, and general and administrative expenses (i.e. payroll and related expenses).

As illustrated in Figure 4, we link the service components to the cost of sales and the operating expenses. More specifically, to represent the organizing and opportunity costs, the service components provided by Amazon.com are linked to the "Operating expenses" and the service components provided by the suppliers in Amazon.com value network are connected to the "Cost of sales".

In 1997, books could be acquired from publishers or from a network of distributors. Both the publishers and the distributors had very high opportunity costs. Months before publishing a book, the publishers should determine the number of copies they intend to print. Publishers could not come up with an estimate before negotiating a deal with the booksellers that grant the booksellers the permission to return the unsold books. In 1994 for instance, 35% of the 460 million books shipped by the publishers were returned to them. The distributors, on the other hand carried around 500,000 titles in their inventories to ensure they met the demand (Spector, 2002). Moreover, Amazon.com was also suffering from its high organizing costs that were mainly related to managing its huge distribution centers. In November of 1997 Amazon.com opened up its second distribution center. The 200,000-square-foot state-of-the-art Delaware distribution center, the length of three football fields, together with the expansion of its Seattle distribution center, drastically increased the operating expenses.

In the late 1990s, Amazon.com's net captured value capture had decreased, mainly due to: high opportunity costs of publishers and distributors; high operating expenses of its operations, and the attributes reducing the net value perceived by its customers (see Figure 4.). This reduction in the net captured value had placed Amazon.com on the brink of bankruptcy. As a matter of fact, by the summer of 2000, Amazon's stock price had dropped by more than two-thirds and by the end of 2000, was down more than 80% of the beginning of 2000. Wall Street speculated that Amazon would file for bankruptcy or that another company would buy it. Analysts assert that if Amazon had not been able to borrow $680 million in February of 2000, it would have run out of cash and gone bankrupt (Applegate, 2002 and 2008).

3.4 Value Redesign in Amazon.com Business Model Circa 2001

In November 2000, Amazon.com introduced its new service offering, Amazon Marketplace. In the online book value segment, Marketplace allows bookstores to sell new, used (including out-of-print books) on the same page that Amazon.com sells its new books. This side-by-side placement dramatically expanded the book selection available to the book buyers by enabling them to choose between new and used books from multiple booksellers including Amazon.com on one single store (Spector, 2002) and thereby, led to an increase in the value perceived by the customers by expanding the titles available.

By launching the Marketplace services, Amazon.com put itself in a head-on price competition with the bookstores to win over customer orders.

Amazon Marketplace increased customer's net perceived value by reducing the book prices and the availability of out-of-print books. Amazon.com and the bookstores had to think out ways to decrease their organizing costs so that they could offer the book at the lowest price possible in a reverse bidding process in order to win customer orders. This competition resulted in a reduction in book prices on Amazon Marketplace. In addition, the presence of the Bookstores in Amazon Marketplace led to the sales of used books on Amazon Marketplace that could once more result in a lower prices and availability of out-of-print books

Amazon Marketplace enables sellers to utilize the e-commerce services and tools to present their products alongside Amazon.com's on the same product detail page on Amazon.com's website pursuing what Bezos phrased as "single store strategy". To realize this single-store strategy, by adapting a coopetitive (simultaneously competitive and cooperative) strategy, Amazon.com provided third-part sellers with automated tools to migrate their catalogs of millions of used and out-of-print books onto the new single product pages inside the Amazon books tab and thereby, reducing the bookstores' opportunity cost by decreasing their costs of doing business with Amazon.com. More importantly, the Marketplace created the opportunity for the bookstores to merchandise their products on the highly trafficked web pages that historically had sold only Amazon products. This, in effect, would mean higher volume of orders and thus lower opportunity costs for bookstores.

The Marketplace led to the generation of significant business and thereby considerable increase in net sales and gross profit helping Amazon.com to offset operating expenses and sales costs and achieve profitability in 2003 for the first time after its establishment. The Marketplace was the major factor behind Amazon.com's profitability. Amazon reported that third-party transactions accounted for 20% of its North American units sold in the second quarter of 2002 (Applegate, 2008).

4 RELATED WORK

e3Service (Kindern and Gordijn, 2008) is a method for semi-automatically reasoning about matching service offerings with service adopter needs. In order to make this semi automatic reasoning possible, e3Service assumes that the service adopter and service supplier share the same ontology, that the service adopter specifies her needs in the same vocabulary as the service supplier specifies its offering. We precisely avoid making this simplifying assumption. This comes at the cost of enormously complicating automatic or event semi-automatic reasoning with the benefit of models that more accurately reflect reality. Also, e3Service defines the value of a service only from the point of view of the service adopter.

House of Quality (Clausing and Hauser, 1988) is an improvement method, in which the main modeling artefact is very similar to the modeling framework presented in this paper. The House of Quality was derived from Quality Function Deployment (QFD), a method that was developed by Japanese companies to improve manufacturing processes for greater service adopter satisfaction. House of Quality is, therefore, more geared toward manufacturing processes.

Strategy canvas (Kim and Mauborgne, 2004 and 2005) is a diagnostic framework for strategy development. It allows an organization to visualize the competitive factors and the current state of play of those factors within a market place and to compare the organization's offering with those of the industry in general.

The Business Model Canvas (Osterwalder and Pigneur, 2010) is a strategic management tool, which assists in the development of new and improvement of existing business models. The canvas includes the nine blocks of a business model: key partners; key activities; key resources; value propositions; customer relationships; channels, and customer segments. While Business Model Canvas presents all the building blocks of a business model it does not provide a holistic view where the interplay and the linkages between the building blocks are modeled.

Value model in this paper is an extension to the SAR (Supplier Adopter Relationship) diagram in (Golnam et. al, 2010 and 2011). The SAR is a part of the Systemic Enterprise Architecture Methodology (SEAM) (Wegmann, 2003).

5 CONCLUSIONS AND FUTURE WORK

In this paper we proposed a modeling framework to conceptualize and represent the design of value in service-oriented business models. Our framework is theoretically grounded in the theoretical insights from management science and economics, drawing principally upon work from the past two decades on value creation and capture including theories, frameworks, constructs, and other models. Thanks to the theoretical rigour embedded our framework the modeling artefact is generic enough to be applicable in the representation of value design in service-oriented business models.

We illustrated the usability and applicability of our framework by modeling value creation and capture in Amazon.com service system circa 1997 and gained insights into the changes that occurred in Amazon.com's business model circa 2001.

Future work will seek to validate and refine the proposed model by way of applying it to an actual firm and it's ecosystem. In this way this research, which has already drawn on the knowledge base for its foundations, will draw upon the environment (composed of people, organizations, and technology) through its real business needs for feedback and validation. Following this a justification and evaluation process should take place, eventually leading to the next iteration of the model.

REFERENCES

Afuah, A. , Tucci, C. L. 2002. *Internet business models and strategies: Text and cases*, McGraw-Hill Higher Education.

Amazon 2011a. Amazon Media Room: News Releases http://phx.corporate-ir.net/phoenix.zhtml?c=176060, p=irol-news.

Amazon 2011b. Amazon.com Investor Relations: Annual Reports and Proxies http://phx.corporate-ir.net/ phoenix.zhtml?c=97664,p=irol-reportsannual.

Ambrosini, V., Bowman, C. , Collier, N. 2010. Using teaching case studies for management research. *Strategic Organization,* 8**,** 206.

Applegate, L. 2002. Amazon. com: 1994-2000. *Case#: 9-801-194, Harvard Business School*.

Applegate, L. M. 2008. Amazon. com: The Brink of Bankruptcy. *Harvard Business School Case***,** 9-809.

Barney, J. 1991. Firm resources and sustained competitive advantage. *Journal of management,* 17**,** 99.

Blomqvist, K., Kylaheiko, K. , Virolainen, V. M. 2002. Filling a gap in traditional transaction cost economics::: Towards transaction benefits-based analysis. *International Journal of Production Economics,* 79**,** 1-14.

Bowman, C. , Ambrosini, V. 2000. Value creation versus value capture: towards a coherent definition of value in strategy. *British Journal of Management,* 11**,** 1,Äì15.

Brandenburger, A. M. , Stuart Jr, H. W. 1996. Value based Business Strategy. *Journal of Economics and Management Strategy,* 5**,** 5-24.

Carson, R. T., Flores, N. E., Martin, K. M. , Wright, J. L. 1996. Contingent valuation and revealed preference methodologies: comparing the estimates for quasi-public goods. *Land Economics***,** 80-99.

Chesbrough, H., Spohrer, J., 2006. "A research manifesto for services science," *Communications of the ACM*, Vol. 49, No. 7, pp. 35-40.

Clausing, D. , Hauser, J. R. 1988. The house of quality. *Harvard Business Review,* 66**,** 63-73.

Clulow, V., Barry, C. , Gerstman, J. 2007. The resource-based view and value: the customer-based view of the firm. *Journal of European Industrial Training,* 31**,** 19-35.

Cousins, P. D. , Spekman, R. 2003. Strategic supply and the management of inter-and intra-organisational relationships. *Journal of Purchasing and Supply Management,* 9**,** 19-29.

Daum, J.H., 2003. "Intangible assets and value creation". John Wiley & Sons.

Day, G.S., 1990. "Market Driven Strategy: Processes for Creating Value" *Journal of Marketing*, vol. 55, no. 4, pp. 116.

De Kinderen, S. , Gordijn, J. E 3 Service: A Model-Based Approach for Generating Needs-driven E-Service Bundles in a Networked Enterprise. *Proceedings of the 16th European Conference on Information Systems (ECIS), Galway, Ireland, 2008.*

Dierickx, I., Cool, K., 1989. "Asset stock accumulation and sustainability of competitive advantage," *Management science*, pp. 1504–1511.

Eggert, A., Ulaga, W., 2002. "Customer perceived value: a substitute for satisfaction in business markets?," *Journal of Business & Industrial Marketing*, vol. 17, no. 2/3, pp. 107-118.

Freel, M. 2006. Patterns of technological innovation in knowledge intensive business services. *Industry and Innovation,* 13**,** 335-358.

Golnam, A., Regev, G., Ramboz, J., Laparde, P. , Wegmann, A, 2010. Systemic Service Design: Alaigning Value with Implementation. *In:* Snene, M., Ralyté, J. & Morin, J.-H., eds. 1st International Conference on Exploring Services Sciences, Geneva, Switzerland. Springer, 150-164.

Golnam, A., Regev, G., Ramboz, J., Laprade, P. , Wegmann, A, 2011. Aligning Value and Implementation in Service Design - A Systemic Approach. *International Journal of Service Science, Management, Engineering, and Technology (IJSSMET),* In Press.

Golnam, A., Ritala, P., Viswanathan, V., Wegmann, A, 2012. Modeling Value Creation and Capture in Service Systems. *In:* SNENE, M., ed. Third International Conference on Exploring Services Sciences, February 2012 Geneva, Switzerland. Springer, 155-169.

Gordijn, J. , Akkermans, J. 2003. Value-based requirements engineering: Exploring innovative e-commerce ideas. *Requirements Engineering,* 8**,** 114-134.

Gordijn, J., Yu, E. , Van Der Raadt, B. 2006. E-service design using i* and e 3 value modeling. *IEEE software,* 23**,** 26-33.

Grant, R. M. 1996. Toward a knowledge-based theory of the firm. *Strategic Management Journal,* 17**,** 109-122.

Grönroos, C., 2000. "Service Management and Marketing" *European Journal of Marketing*, vol. 15, no. 2, pp. 3-31.

Grönroos, C. , Ravald, A. 2011. Service as business logic: implications for value creation and marketing. *Journal of Service Management,* 22**,** 5-22.

Heck, E. , Vervest, P. 2007. Smart business networks: how the network wins. *Communications of the ACM,* 50**,** 28-37.

Jacobides, M.G., Knudsen, T., Augier, M., 2006. "Benefiting from innovation: Value creation, value appropriation and the role of industry architectures," *Research Policy*, vol. 35, no. 8, pp. 1200-1221.

Johannesson, P., Andersson, B., Bergholtz, M. , Weigand, H. 2009. Enterprise modelling for value based service analysis. *The Practice of Enterprise Modeling***,** 153-167.

Kalpanik, S. , Zheng, C. 2011. *Inside the Giant Machine - An Amazon.com Story*, Center of Artificial Imagination, Inc. 2nd Edition.

Khalifa, A.S., 2004. "Customer value: a review of recent literature and an integrative configuration," *Management Decision*, vol. 42, no. 5, pp. 645-666.

Kim, W. C. , Mauborgne, R. 2004. Blue ocean strategy. *If you read nothing else on strategy, read thesebest-selling articles.***,** 71.

Kim, W. C. , Mauborgne, R. E. 2005. *Blue Ocean Strategy: How to Create Uncontested Market Space and Make Competition Irrelevant*, Harvard Business Press.

Kothandaraman, P. , Wilson, D. T. 2001. The Future of Competition:: Value-Creating Networks. *Industrial Marketing Management,* 30**,** 379-389.

Kotler, P. 2000. *Marketing management, millennium ed*.

Lepak, D., Smith, K., Taylor, M., 2007. “Introduction to Special Topic Forum: Value Creation and Value Capture: A Multilevel Perspective,” *The Academy of Management Review ARCHIVE*, vol. 32, no. 1, pp. 180-194.

Masten, S. E., Meehan, J. W. , Snyder, E. A. 1991. The costs of organization. *Journal of Law, Economics, and Organization,* 7**,** 1.

Moller, K. , Svahn, S. 2006. Role of Knowledge in Value Creation in Business Nets*. *Journal of Management Studies,* 43**,** 985-1007.

Norman, R. , Ramirez, R. 1993. From Value Chain to Value Constellation. *Harvard Business Review,* 71**,** 65-77.

Osterwalder, A. , Pigneur, Y. 2010. *Business model generation: A handbook for visionaries, game changers, and challengers*, Wiley.

Peppard, J., Rylander, A., 2006. “From Value Chain to Value Network:: Insights for Mobile Operators,” *European Management Journal*, vol. 24, no. 2-3, pp. 128-141.

Pijpers, V. , Gordijn, J. 2007. e 3 forces : Understanding Strategies of Networked e 3 value Constellations by Analyzing Environmental Forces. *Advanced Information Systems Engineering.*

Pitelis, C. N. 2009. The co-evolution of organizational value capture, value creation and sustainable advantage. *Organization Studies,* 30**,** 1115.

Ritala, P., Andreeva, T., Kosonen, M. , Blomqvist, K. 2011. A Problem-Solving Typology of Service Business. *Electronic Journal of Knowledge Management,* 9, 37-45.

Ritala, P., Hurmelinna-Laukkanen, P. 2009. What's in it for me? Creating and capturing value in innovation-related coopetion. *Technovation*, 29, 819-828.

Spector, R. 2002. *Amazon. com: Get big fast*, Harper Paperbacks.

Stake, R. E. 1995. *The art of case study research*, Sage Publications, Inc.

Teece, D. J. 2009. *Dynamic capabilities and strategic management: organizing for innovation and growth*, Oxford University Press, USA.

Tether, B. S. , Hipp, C. 2002. Knowledge intensive, technical and other services: patterns of competitiveness and innovation compared. *Technology Analysis and Strategic Management,* 14**,** 163-182.

Vargo, S. L. , Lusch, R. F. 2004. Evolving to a new dominant logic for marketing. *Journal of marketing,* 68**,** 1,17.

Vargo, S. L. , Lusch, R. F. 2008. Service-dominant logic: continuing the evolution. *Journal of the Academy of Marketing Science,* 36**,** 1.

Vargo, S. L., Maglio, P. P. , Akaka, M. A. 2008. On value and value co-creation: A service systems and service logic perspective. *European Management Journal,* 26**,** 145,Äì152.

Weigand, H. 2009. Value Encounters: Modeling and Analyzing Co-creation of Value. *Software Services for e-Business and e-Society***,** 51-64.

Weigand, H., Johannesson, P., Andersson, B. , Bergholtz, M. Value-based service modeling and design: Toward a unified view of services. 2009. Springer, 410-424.

Yin, R. K. 2009. *Case study research: Design and methods*, Sage Publications, Inc.

Yu, E. 1997a. Proceedings of the 3rd IEEE International Symposium on Requirements Engineering. Citeseer.

Yu, E. S. K. 1997b. Towards modelling and reasoning support for early-phase requirements engineering. *Requirements Engineering, 1997., Proceedings of the Third IEEE International Symposium on.*

From Business Services to IT Services by Capturing Design Decisions

Biljana Bajić[1], Claude Petitpierre[1], Alain Wegmann[1] and Do Qaung Tri[2]
[1]*Ecole Polytechnique Fédérale Lausanne (EPFL), CH-1015 Lausanne, Switzerland*
[2]*Ho Chi Ming City University (HCMUT), District 10, Ho Cho Minh City, Vietnam*
biljana.bajic@epfl.ch, claude.petitpierre@epfl.ch, alain.wegmann@epfl.ch,dotri84@gmail.com

Keywords: Service Design, Service Science, Business-driven Development, Model Transformation.

Abstract: The main goals of any service-oriented design include flexible support and adaptability of business services and improved business-IT alignment. The existing approaches, however, have failed to fully meet these goals. One of the major reasons for this deficiency is the gap that exists between how the computer science and management science communities perceive the concept of service. We present a flexible, semi-automatic, model-driven approach to designing IT services that directly satisfy business needs and requirements. We begin with the design of business services and the capture of the design decisions that transform the business design through multiple model layers to the IT service design. All layers can be simulated using the Alloy Analyzer tool. The last layer can be run on a given target platform. This approach is demonstrated on the running example based on the consulting project conducted at the company General Ressort. The central aspect of our approach is separating the design decisions from anything that can be automated. It provides the multi-perspective view of the system, by making the modeling process faster, leaving the designer the space to focus on the design decisions and not on drawing the models.

1 INTRODUCTION

As there are many different definitions of services, we give the one used in this paper. Based on (Blecher and Sholler, 2009, p. 1), "Business service is a business-related work activity or duty performed for others to produce a business outcome. It is the expectation of the business person that the service will accomplish this outcome. The person generally does not care how it is accomplished, as long as it is done in an effective manner from a business perspective."

A business service may be supported by one or more IT Service(s), and may consist almost entirely of IT services, especially where these service are directly used by customer. Examples include online banking and online shopping.

ITIL v.3 defines a IT service as "a service provided to one or more customers, by an IT service provider. An IT Service is based on the use of information technology and supports the customer's business process. An IT Service is made up from a combination of people, processes and technology." (OGC, 2007)

Based on these definitions, we will explain our approach for transforming business services to IT services.

The main goals of any service-oriented design include flexible support and adaptability of business services and improved business-IT alignment, i.e. orchestration of the lower level IT infrastructure services to deliver the desired business-level customer services. The existing approaches, however, have failed to fully meet these goals. One of the major reasons for this deficiency is the gap that exists between how the computer science and management science communities perceive the services. In practice, the business and technology perspectives of services have to be considered separately. Even simple changes to one perspective (e.g. due to new regulations or organizational change) require error-prone, manual re-editing of the other one (Buchwals et al., 2011). Over time, this leads to the degeneration and divergence of the respective models and specifications; this thereby aggravates maintenance and makes expensive refactoring inevitable.

Our approach for aligning business services with IT services is flexible, semi-automatic, and model-driven, enabling the implementation design of business services. In the design process, the designer begins by identifying the services required by the customers, then follows by capturing the design decisions. Based on these decisions, intermediate model

layers and finally IT services are generated. These services are necessary for the implementation of the application supporting the customer's requirements. This process allows business analysts to represent services from a business point of view, while facilitating the design and development of IT services.

The details embedded in an IT service design model-layer enables the execution of the model on the given target platform, such as JEE (Java Enterprise Edition). All the model layers can be translated and simulated with the Alloy Analyzer tool (Jackson, 2011), so that the designer, by viewing a few instances of the model, can see how each of the model layers behave.

A central aspect of our method is that, in the service design process design decisions are captured in each step. This way, they are clearly separated from the automatic part of the transformation. Thus, the design process is done semi-automatically. In these steps, the designer can independently make the decisions about different aspects, influencing the service design.

We illustrate our approach by the running example based on a consulting project conducted at a company that sells parts for watches in Switzerland, General Ressort (GR).

We organize the paper as follows. In Section 2, we explain our modeling method and outline the design process. In Section 3, we discuss the simulation and prototyping of the model layers. We present related work in Section 4. The final section concludes the study and discusses the future work.

2 MODELING THE IMPLEMENTATION DESIGN OF BUSINESS SERVICES AT GENERAL RESSORT

For a better understanding of the design process, we illustrate each design step by applying it to the example of company GR. For the purpose of this paper, we focus only on a simplified business service of order processing. We illustrate the design steps in our approach based on this example. By convention, information in italics are the corresponding names of the elements in the model.

The simplified business service is executed as follows: "GR gets order (*OrderInitial*) from the customer that contains a unique customer name and unique customer part id. The person dealing with orders (*OrderEntryPerson*) receives the information about the order (*OrderInitial*) and finds the customer and the part by unique information in the enterprise resource planning system (*ERP*). Finally, he creates the confirmed order (*OrderConfirmed*) in the *ERP*."

Notice that in the a real service, in case the customer or the part is missing in the system, they are created. However, as it does not show any new aspects of our approach, it is not shown in this paper.

2.1 Modeling Method

In order to understand the steps of our design process and the example, we will explain the main principles of the proposed modeling approach, mostly based on Catalysis approach (D'Souza and Wills, 2001).

The central aspect of our approach is a system and its two main aspects: organizational and functional (Wegmann, 2003). For both aspects, we define the black-box and the white-box view of the system. The organizational black-box view of the system is called 'system as a whole', and it hides the organizational aspects of the system; unlike the organizational white-box view of the system, called 'system as a composite', which reveals a system's construction. Similarly, the functional white-box view of the system is called 'action as composite' and it provides insight into system's functionality, unlike the functional black-box view ('action as a whole') that hides them. This can be seen in Figure 1.

There is also a special view of a system and a type of the action, called 'action as n-ary relationship', where one action is distributed among many systems connected with one action binding in between (Figure 2). In this way, it is specified what part of action is in which system. However, the action parts are still dependent on each other and cannot be treated separately; only together can they be seen as one action.

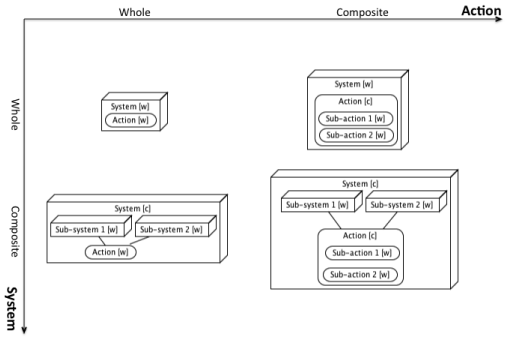


Figure 1: Organizational and functional hierarchy.

Another important characteristic of our approach is that it places the action on an equal footing with the object, because good decoupled design requires careful thought about what actions occur and what they

achieve. Therefore, behaviour and data are equally important in the proposed method and **each model layer contains both the behaviour and data part of the services**.

2.1.1 Meta-model

In order to understand the models given in this paper, we show the meta-model with relevant elements in Figure 3. The full lines in the meta-model correspond to the 'contain' relationship, where one element is inside the other. The dashed lines correspond to the 'has link to' relationship, where one element is related to the other with a line. The concepts used in the meta-model are based on Catalysis terms. The table with the corresponding business terms can be seen in Figure 4.

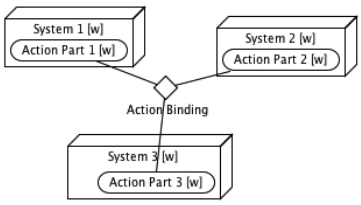


Figure 2: Action as n-ary relationship.

The root element of any model is *WOC* (working object as composite), representing the system of interest, in this case the market segment. It is composite, because it contains the main stakeholders, such as the service provider (company providing the service) and service consumer (customer company). *WOC* reveals the system structure, therefore it can contain other *WO*s (whole and composite). It can also contain actions shared among different systems (*JA* (joint action) or *SJAB* (split joint action binding)). *SJAB* corresponds to the action binding in 'action as n-ary relationship', i.e. it connects several distributed actions in different systems, thus making one action. *JA* is the whole action with all its elements between many systems. There are no action parts in the other system. *SJAB* has links to *SJA*s (split joint actions). They correspond to the action parts in 'action as n-ary relationship'. One of them contains a link to the event, showing who is initiating the action SJA_e, while the others have no event related to it SJA_{ne}.

WOW does not reveal its structure. Therefore, it does not contain other *WO*s. It can contain actions or data elements (properties (*LP*), inputs (*INP*), outputs (*OUT*) and *EVENT*s). These actions can be joint actions (*SJA* and *JA*) or localized (*LA*), meaning they are inside just one *WO*, and are not split between many *WO*s. As with all other whole-composite relations, *LAC* (localized action composite) can have many *LAW*s (localized action whole).

As a service is a duty performed for others producing outcome, it always has some input and output parameters. Therefore, all actions, i.e. services (*LA*, *JA*, *SJA*) contain inputs and outputs. Also, they have information about who is initiating the service captured in the event. In the case of *SJA*, it applies to only one action part related to the action.

In addition, in our approach service is defined with functional units (*FU*) and properties (*LP*), representing the behavioural and data part of service, respectively. This does not apply to *LAC*, because it represents the grouping of objects for many *LAW*s (services).

There are four different types of services, one for each model layer of our service design process. The top-level layer is business services, as it is defined in the introduction. Thus, it represents the service that the customer needs. This service is transformed to the joint business service, joint IT service and finally independent localized IT services for each system of interest (in this case roles in the company).

2.2 Service Design Process

As one of the characteristics of our method is that the data are on an equal footing with behaviour, services are described with behavioural and data parts, i.e. with functional units and properties. Therefore, we add **two intermediate model layers** in order to maintain dependency on high-level business services and low-level IT services: one for data details, the other for behaviour details. These layers show the constructional and functional design, as described in (Dietz and Albani, 2005). Through the process of transforming these layers, business services are extended with the details necessary for IT services. This emphasizes the two main aspects of our approach: behaviour (functional units) and data (properties). Hence, the designer can make the decisions about the data and the behaviour independently.

These model layers are then related with three intermediate steps in which the designer makes the decisions about the data and behaviour responsibility.

The designer captures the decisions in **specially formatted matrices** by using **'define and distribute' pattern**. This means, in each step, the designer defines new elements in the system, which becomes column of the matrix. Also, the designer distributes some existing elements shown in rows of the matrix to the new elements.

To sum up, there are four model layers in our service design process and three in-between steps that

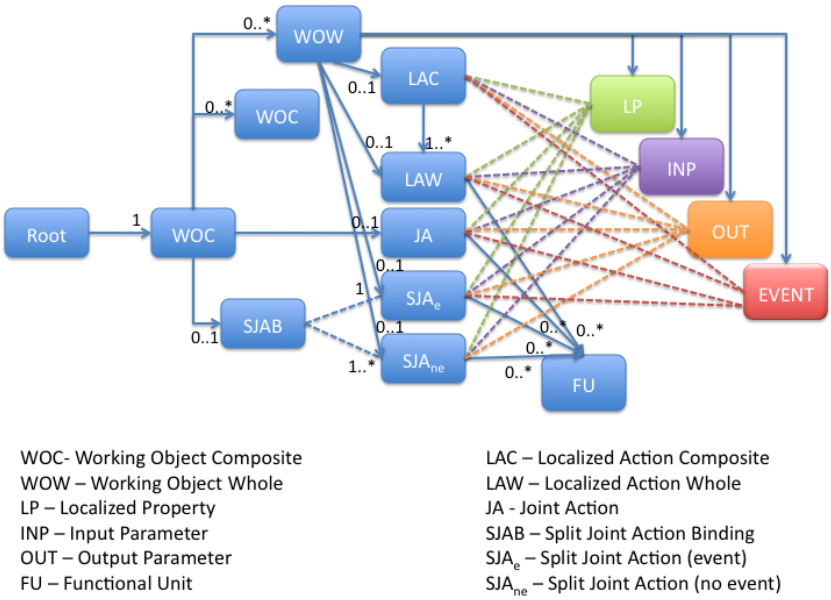


Figure 3: Meta Model.

Catalysis Term	Business Term for This Example
WOC (Working Object Composite)	Market Segment/Company with internal structure (stakeholders/roles)
WOW (Working Object Whole)	Company/Roles without internal structure
Root	Market Segment
Action	Service
JA (Joint Action)	Joint Business Service
SJA (Split Joint Action)	Joint IT Service
LA (Localized Action)	Localized Business/IT Service

Figure 4: Catalysis and Business Corresponding Terms.

capture the design decisions in specially formatted matrices. In Figure 5, we show the whole process of service design and implementation. The two orange phases represent the service implementation part and the black phases are service design part. As it can be seen, after service design process, the last layer of the IT service design can be transformed to the intermediate project containing data needed by BUD tool (Petitpierre, 2011) to generate the application. Due to the lack of space, we will not describe the service implementation in this paper. It is based on the BUD tool that is based on JSON templates (JSON, 2009). More information can be seen in (Petitpierre, 2011).

The process is shown as a spiral process (Boehm, 1986), because it combines the prototyping with the steps of the proposed process. In this way, the designer can analyse and validate how the design decisions can influence the design and implementation of the services.

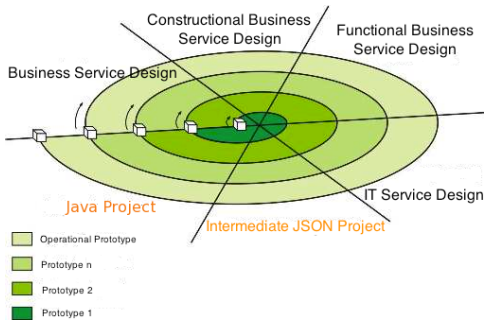


Figure 5: Service Design and Service Implementation Process.

Finally, here is the description of the model layers and steps of the proposed design process. By convention, names from the model will be marked in italics.

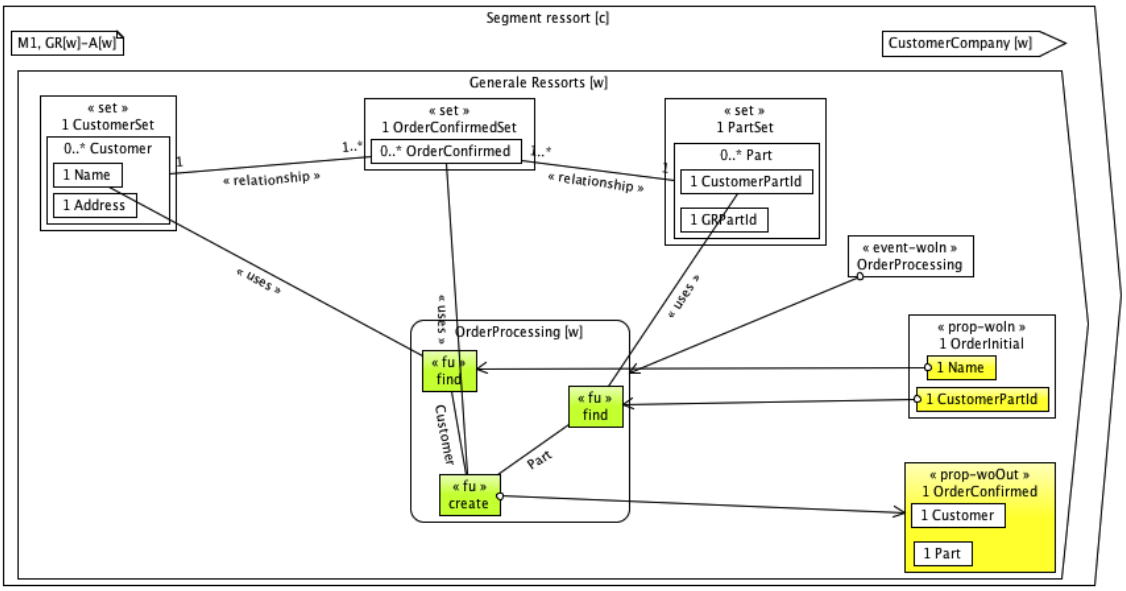


Figure 6: Business Service Design.

2.2.1 Business Service Design

In the first layer, the designer specify the business services as in Figure 6. As it can be seen, we show a segment with company *GeneralRessort* and *CustomerCompany*. There is one main business service that is modeled: *OrderProcessing*. We do not show either the organization of the company or the sub-services (sub-actions). Therefore, this model layer represents the system as a whole (*GeneralRessort[w]*), action as a whole (*OrderProcessing[w]*).

As we have mentioned in the section Modeling Method, both the behavioural and data part of the services are shown. The behavioural part is shown by functional units (*fu*) marked in green colour. They represent atomic operations, such as *find*, *create*, etc. They can be parametrized by the properties related to them (by link *uses*), such as *CustomerSet, PartSet, OrderConfirmedSet*. *Set* properties represent the set of elements of one kind, e.g. *Customer*. The relation between these elements is shown by *relationship*. For each property cardinality and name can be seen. Depending on which attribute of the *Customer* element *fu find* is related, we can specify different operations, such as 'find customer in the set by its name' in Figure 6.

The inputs and outputs are marked as yellow properties. The business service order processing has two input parameters, *Name* and *CustomerPartId* and one output parameter, *OrderConfirmed*. *OrderInitial* and *OrderConfirmed* are marked with *prop-woIn* and *prop-woOut*. *woIn* and *woOut* mean that they come into and go out from the system *GeneralRessort* from and to outside (*CustomerCompany*), respectively.

Also, each service has one event (in this case *event-woIn*) associated to it, showing who is initiating the service. In this step, there are no roles, therefore the event is shown inside the whole system *GeneralRessort*.

Functional units can be connected with lines that can contain the name of the data they share, such as *Customer*, meaning that *fu find* and *create* share one data of the type *Customer*.

2.2.2 Joint Business Service Design

In the next model layer, the company construction is revealed and joint business services are defined by providing details about the business service-related data responsibilities within the company's roles. Therefore, the layer corresponds to the system as a composite, action as a whole.

The designer defines the roles (organizational units) in the system and distributes the service-related data to these roles, according to their responsibilities. This can be seen in matrices in Figure 7.

The designer defines roles: *OrderEntryPerson* and *ERP*, marked in green in the matrices and in the next model layer.

Then all data from the model layer in Figure 6, shown in the rows of the matrix, are distributed to the newly defined roles (Figure 8).

As we can see, joint business service design contains defined business services without changes of the

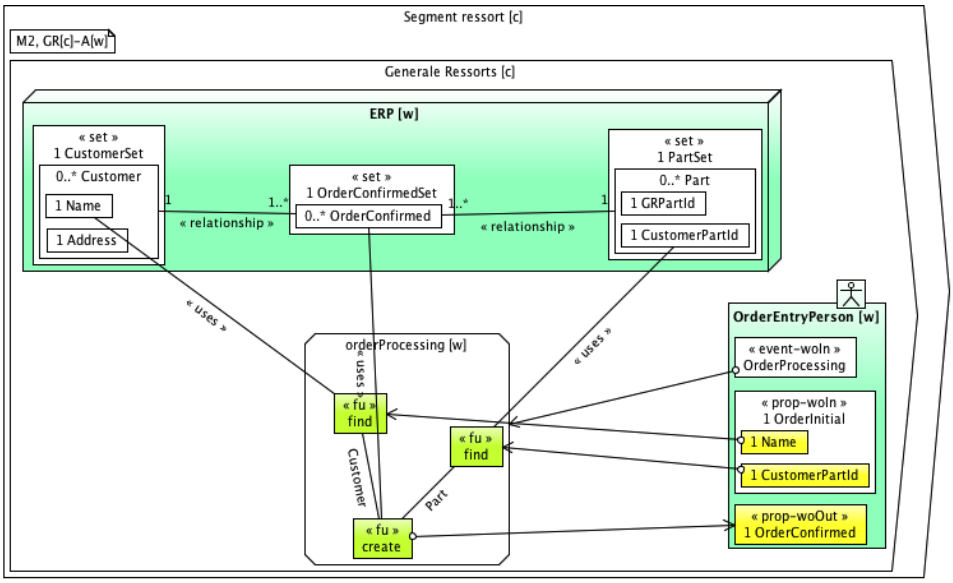


Figure 8: Joint Business Service Design.

	Roles
1	OrderEntryPerson
2	ERP

(a) Role Definition

	OrderEntryPerson	ERP
(<<prop-woIn>> OrderInitial) Name		
(<<prop-woIn>> OrderInitial) CustomerPartId	X	
<<prop-woOut>> OrderConfirmed	X	
<<set>> CustomerSet		X
<<set>> PartSet		X
<<set>> OrderSet		X
<<event-woIn>> OrderProcessing	X	

(b) Data Responsibility

Figure 7: Step 1 - Design Decisions from Figure 6 to Figure 8.

functional units. However, the properties related to service, as well as the inputs and outputs are distributed to the newly defined roles. Notice that there is still only one service defined between many roles, it is still unknown which role is responsible for which part of the service performance.

2.2.3 Joint IT Service Design

The next model layer defines which role performs which part of the service. This provides insight into the functional decomposition of the system, without complete split of services. Therefore, this layer corresponds to system as a composite, action as a n-ary relationship.

The designer defines new services of the roles and distributes existing functional units to these service.

The designer defines two services: *OrderEntryPerson* service and *ERP* service, marked in blue in the matrices in Figure 9 and in Figure 10.

Based on the design decisions, functional units are distributed automatically to the role's services marked with 'X'. Based on the arrow lines connected to the functional units, special functional units are added to the roles where the origin and ending of line is: *enter* and *get*, respectively. *enter* is added when the role initiates the *fu* (the line going from the role), and *get* when the role obtains the result from *fu* (the line directed to the role). This is based on the 'send-respond-reply' pattern described in (Beach et al., 1982). On the lines connecting these *fu*, the names of the data are written, *Name, CustomerPartId, OrderConfirmed*.

We show the result of added design decisions in Figure 10. As we can see, joint IT service design contains services for each role in the company, containing some existing functional units and some newly added ones. The properties are not changed in this step. Notice that there are no intermediate results in the services, because as this is action as a n-ary relationship, all these services represent together one service, they are still not completely independent.

2.2.4 Localized IT Service Design

In this step, new sub-services (and implicitly their events) are defined, and functional units are distributed to these services. Therefore, this layer represents system as a composite, action as a composite.

The designer defines new sub-services and dis-

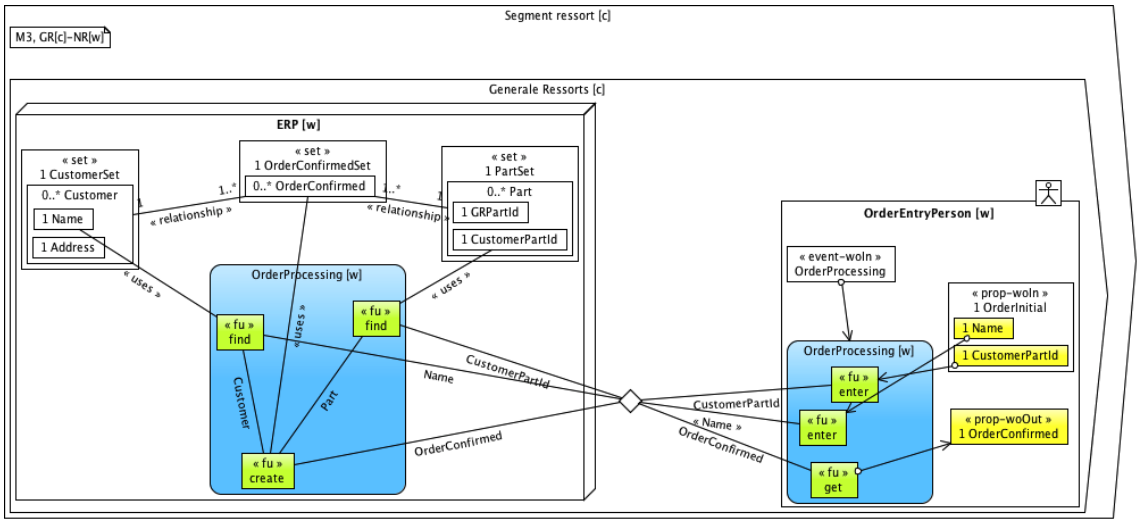


Figure 10: Joint IT Service Design.

	Role's Services
1	OrderEntryPerson Service
2	ERP Service

(a) Role's Service Definition

	OrderEntryPerson Service	ERP Service
<<fu>> find Customer Name		X
<<fu>> find Part CustomerPartId		X
<<fu>> create OrderConfirmed		X

(b) Behaviour Responsibility

Figure 9: Step 2 - Design Decisions from Figure 8 to Figure 10.

tribute existing functional units to these services. This is repeated for each of the roles. The events for new sub-services are implicitly specified. Additionally, the designer can specify that events are shared between different roles.

For example, the designer defines new sub-services for *ERP*: *FindCustomer*, *FindPart* and *CreateOrderConfirmed*. They are marked in red in the matrices in Figure 11 and in the resulting layer in Figure 12. Then, the designer distributes the existing functional units of *ERP* to defined sub-services. He does the similar for *OrderEntryPerson*, for which he defines six sub-services. Finally, he specifies that some events are shared, such as *EnterName* service of *OrderEntryPerson* and *FindCustomer* service of *ERP*, showing it is transmitted from one role to the other.

Based on the design decisions, new services are created inside existing services containing defined functional units as in Figure 12. In addition, some other elements are automatically added to the model. The lines between roles are replaced by the corresponding properties in them, such as *prop-woOut Name* and *prop-woIn Name*. Also, as we show the sub-services, we also add the intermediate data (such as *Customer*) and the corresponding *fu*s (such as *get*). These *fu*s are also included in the distribution matrices of the designer. Finally, the default IT sub-service is added (*CreateOrderProcessing*), which is responsible for the basic initialization of the services in the IT system.

All necessary data for the *ERP* service now appear in the *ERP* system, because services of all roles are now separated and their systems contain all necessary data for their services. Thus, if we would cover completely the other roles in the model, we would be able to see everything that is necessary for one visible role.

This model contains IT services that are platform independent and ready to be executed in any target language. In addition, it also contains the human services and human-human interaction, which are very often very important to show in one consulting project.

As mentioned in the service design and implementation process, by using this model layer it is possible to generate the running application for the corresponding business service and its supporting IT services as defined in this model layer.

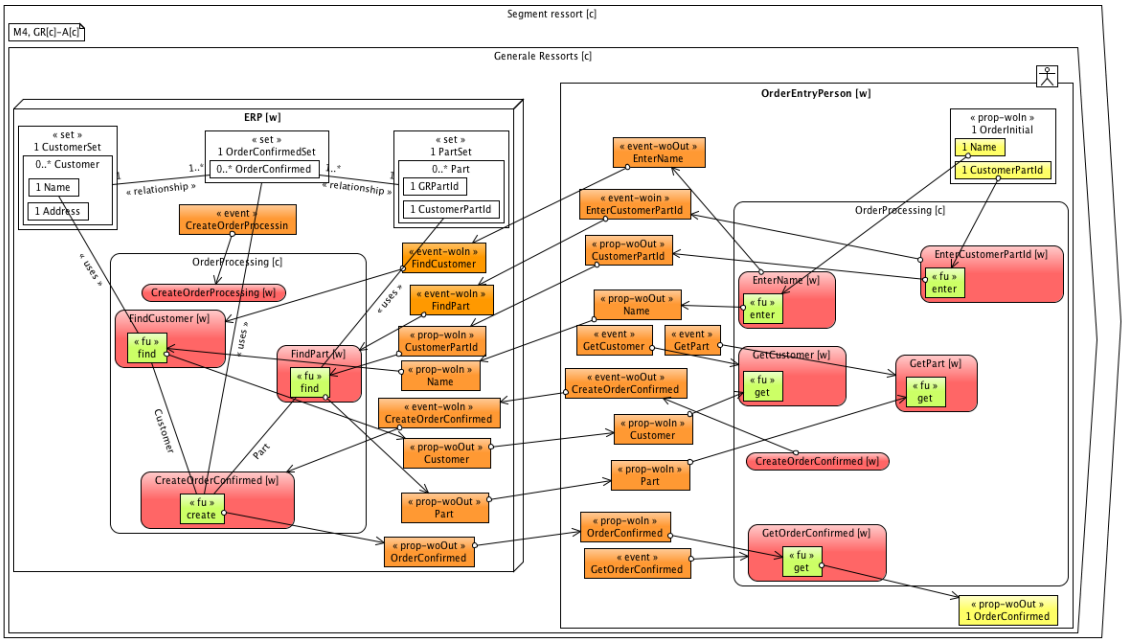


Figure 12: Localized IT Service Design.

	IT Services
1	**FindCustomer Service**
2	**FindPart Service**
3	**CreateOrderConfirmed Service**

(a) IT Service Definition

	FindCustomer Service	FindPart Service	CreateOrderConfirmed Service
<<fu>> find Customer Name	X		
<<fu>> find Part CustomerPartId		X	
<<fu>> create OrderConfirmed			X

(b) Business Service Support

Figure 11: Step 3 - Design Decisions from Figure 10 to Figure 12.

3 MODEL SIMULATION AND PROTOTYPING

One of the main challenges in service design is ”how to prototype services (to generate, develop, test and evaluate ideas) throughout the design process?” (Vaajakallio et al., 2009). In this section, we will briefly explain how the prototyping is done in the proposed approach.

In order to evaluate if the model corresponds to the customer's needs and requirements, this approach enables us to prototype each of the model layers, thus enabling the designer to simulate the behaviour of the model layer and to find design mistakes in the early phase. In addition, the last model can be executed in the given target platform, which also provides one way of validation.

In order to get the prototypes, we first formalize the models using declarative language Alloy (Jackson et al., 2000), and then we run and simulate them using the Alloy Analyzer tool (Jackson, 2011). We use Alloy, because it can be also used to check the refinement between different model layers, as it is explained in (Rychkova, 2008).

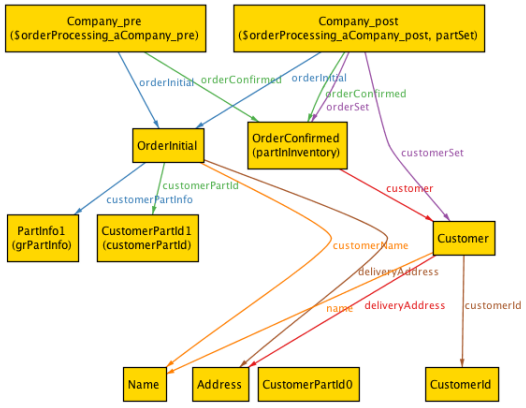


Figure 13: Result of Alloy model simulation.

We show in Figure 13 the result of one of the simulations of the GR case, where the customer and part

are created in case they do not exist. *Company_pre* and *Company_post* are the states of the company General Ressort, before and after the order is processed, respectively. As we can see, they both have the same *OrderInitial* that is the input to the service and *OrderConfirmed* that is output of the service. Before the order processing, there was no customer with the name *Name* given in *OrderInitial*, so the new customer *Customer* with this name is created in *customerSet* that is in *Company_post*. And the *OrderConfirmed* contains information about that customer and becomes member of *OrderSet*. We do not provide the Alloy code here, due to the lack of space.

So far, we have transformed manually the model to the Alloy code, which can be run using the Alloy Analyzer tool. The goal in the future is to automate this simulation process. Also, the goal is to provide simulation results in the a more business user-friendly form.

4 RELATED WORK

We take the basic principles of our modeling technique from the Catalysis (D'Souza and Wills, 2001) approach. Therefore, unlike some object-oriented methods, our approach does not always begin by assigning responsibilities for services to specific roles. We believe in not taking decisions all at once. We first state what happens, then we state which role is responsible for doing it and which one is responsible for initiating it; and finally we state how it is done.

Another specific aspect of Catalysis overtaken in our approach is that it places the behaviour on an equal footing with the data. Therefore, unlike other modeling techniques, there is only one diagram type and each model layer contains both the objects and actions. Also, many other approaches for business-IT alignment of services, like (Kochler et al., 2008) and (Buchwals et al., 2011) are process oriented, whereas in our approach each layer contains both the behaviour and the data.

In addition, we believe in using declarative business process as long as possible. From our experience, very often in the projects the sequence of services is not known. Also, in this way, the process is more configurable, and the designer can decide in a separate step from many possible execution paths; or it can be concluded from the data dependency in the model. However, in most service design approaches (Vaajakallio et al., 2009) this is not possible.

The central aspect of our approach is the capture of the design decisions. In this way, the designer creates the business service design and enters the design decisions that need to be made, and the rest is done automatically. This clearly separates the design decisions of the automatic part of transformation, thus enabling the designers to have a multi-perspective view of the system and to zoom in and out the models in order to see the system with as much detail as they need. In this way, they can quickly prototype business requirements and evaluate several architectures. This is something that, to the best of our knowledge, does not exist in the other techniques.

Also, one of the challenges of the service design, not covered very well in the techniques, is the prototyping of the models (Vaajakallio et al., 2009). We also provide a simulation of the models using the Alloy Analyzer tool.

Besides simulations, our approach also provides the service implementation. The whole service design and implementation process is MDA (model driven architecture)-based (OMG, 2001): it proposes a set of models extending from the CIM (computation-independent model) level, the highest level of abstraction of the MDA, to the PIM (platform-independent model) and PSM (platform-specific model) levels. Business service and joint business service design correspond to the CIM level, because they represent the context and purpose of the model without any computational complexities. Joint IT service design and localized IT service design correspond to PIM level. It describes which part is done by software application and gives its behaviour and structure regardless of the implementation platform. In the service implementation part of the process, the intermediate project containing the templates and specification objects correspond to the PSM level, because they are strictly related to the specific application platform. Also, in our service design and implementation cycle, the mapping between these different levels is clearly and systematically given.

To conclude, we provide the flexible, coherent service design and implementation approach that follows the standard levels of the MDA. This approach enables us to clearly and systematically map between business services and IT services, as well as to prototype the different model layers and execute the IT service layer.

5 CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a flexible, semi-automatic, model-driven approach for aligning business services with IT services, thus enabling the im-

plementation design of business services. We have briefly presented the whole process, containing the service design and service implementation. Then, we have explained the service design part in more details.

We have illustrated the design process by the example based on the consulting project conducted in the company General Ressort based in Switzerland, which sells parts for watches. In order to be able to understand the example, we have explained some of the basic characteristics of the proposed method, including the meta-model. In the meta-model, it can be seen that the service is characterized by: inputs, outputs, event, functional units and properties. Inputs and outputs are the input and output parameters of the service, event contains information about who is initiating the service and functional units and properties correspond to behaviour and data related to the service.

Another important characteristic of our modeling method is that data and behaviour are equally important. Therefore, unlike many other modeling methods, there is only one diagram type that contains both of them.

The proposed service design process includes four model layers containing service design and three in-between steps, in which the design decisions are captured. Capturing the design decisions is the central aspect of our approach. It enables clear separation of the decisions that need to be made by the designer and the automatic part of transformation.

The first model layer is business service design and the last contains IT service design, so that both the business experts and IT experts have the perspective of the system necessary for them. Two more layers are added in-between, for which the user decides on data and behaviour responsibility as two main parts of any service design.

The layers are connected based on the design decisions captured in the specially formatted matrices. To sum up, the designer defines the business service design, inserts necessary design decisions following the strict rules of the proposed method. In this way, he transforms the business service design into the IT service design through revealing the service construction and functionality. We also provide the tool for this transformation.

In our approach, IT service design includes human services and human-human interactions, as from our experience, it is very important in many consulting projects.

Each of the model layers can be transformed to Alloy code and simulated with the Alloy Analyzer tool. We have also shown the example of such a simulation. In this way, it can be validated on early stage if the models satisfy the customer needs and requirements and errors can be detected.

Also, the last layer has enough technical details and can be executed on the given target platform, such as JEE. We also provide the tool for this. However, as it is not the main topic of this paper, we have not given many details about it.

So far, we have tested the approach iteratively on the laboratory examples based on the consulting projects, specifically designed to investigate the ideas of the proposed service design process. In the future, we will validate the approach on real case studies, i.e. designing in real situations (Castro et al., 2008). Also, we will automate the transformation to Alloy language and provide more user-friendly representations of the results of simulation.

REFERENCES

Beach, R., Beatty, J., Booth, K., Plebon, D., and Fiume, E. (1982). The Message is the Medium: Multiprocess Structuring of an Interactive Paint Program. *Computer Graphics Journal*, 18(3).

Blecher, M. and Sholler, D. (2009). Defining Business and SOA Services. http://www.gartner.com/id=1002314.

Boehm, B. (1986). A Spiral Model of Software Development and Enhancement. *ACM SIGSOFT Software Engineering Notes*, 11(4).

Buchwals, S., Bauer, T., and Reichert, M. (2011). *Service Life Cycle Tools and Technologies: Methods, Trends and Advances*, chapter Bridging the Gap Between Business Process Models and Service Composition Specifications, pages 124–153. Idea Group Reference.

Castro, V., Marcos, E., and Wieringa, R. (2008). Towards a service-oriented MDA-based approach to the alignment of business processes with IT systems: from the business model to a web service composition model. *International Journal of Cooperative Information Systems*, 18(2):225–260.

Chen, H. M. (2008). Towards Service Engineering: Service Orientation and Business-IT Alignment. In *Proceedings of the 41st Hawaii International Conference on System Sciences*.

Crawford, C., Bate, P., Cherbakov, L., Holley, K., and Tsocanos, C. (2005). Toward an on demand service-oriented architecture. *IBM Systems Journal*, 44(1):81–107.

Dietz, J. and Albani, A. (2005). Basic notions regarding business processes and supporting information systems. *Requirements Engineering Journal*.

D'Souza, D. and Wills, A. (2001). *Objects, components, and frameworks with UML - The Catalysis approach*. Addison-Wesley, 4th edition.

Jackson, D. (2011). Alloy Analyzer tool. http://alloy.mit.edu/alloy/.

Jackson, D., Schechter, I., and Shlyakhter, I. (2000). ALCOA: The Alloy constraint analyzer. In *Proceedings of the 22nd International Conference on Software Engineering (ICSE)*, Limerick, Ireland.

JSON (2009). http://json-template.googlecode.com/svn/trunk/doc/Introducing-JSON-Template.html.

Kochler, J., Hauser, R., Kuster, J., Ryndina, K., Vanhatalo, J., and Wahler, M. (2008). The Role of Visual Modeling and Model Transformations in Business-driven Development. In *Electronic Notes in Theoretical Computer Science, URL: http://www.elsevier.nl/locate/entcs*.

OGC (2007). ITIL v3, Glossary of Terms, Definitions and Acronyms. http://www.itilfoundations.com.

OMG (2001). Model driven architecture. http://www.omg.org/mda/. Document number ormsc/2001-07-01.

Petitpierre, C. (2011). Bottom Up Creation of a DSL Using Templates and JSON. In *SPLASH'11*.

Rychkova, I. (2008). *Formal Semantics for Refinement Verification of Enterprise Models*. PhD thesis, EPFL.

Vaajakallio, K., Mattelmaki, T., Lehtinen, V., Kantola, V., and Kuikkaniemi, K. (2009). Literature Review on Service Design, extreme-design project. Technical report, University of Art and Design Helsinki Helsinki University of Technology.

Wegmann, A. (2003). On the Systemic Enterprise Architecture Methodology (SEAM). In *ICEIS, International Conference on Enterprise Information Systems*.

SHORT PAPERS

Investigating on the Role of EA Management in Mergers and Acquisitions

Initial Findings from a Literature Analysis and an Expert Survey

Andreas Freitag and Christopher Schulz
Technische Universität München, Boltzmannstraße 3, 85748 Garching b. München, Germany
Andreas.Freitag@mytum.de, Christopher.Schulz@mytum.de

Keywords: Enterprise Architecture Management, Mergers and Acquisitions, Literature Analysis, Expert Survey.

Abstract: Mergers and Acquisitions (M&A) lead to substantial changes for the companies involved. The resulting enterprise transformation is challenging as it includes, among others, elements like business processes, organizational units, applications, data, and infrastructure components. Enabling the alignment of business and information technology (IT), the discipline of Enterprise Architecture (EA) management provides a holistic perspective on those elements as well as their relationships. Moreover, EA management fosters communication and provides a consistent information base and therefore is able to contribute to the success of M&A. However, currently there exists only little work investigating on the role, benefit, and usage of EA management in M&A. In this paper, we therefore peruse EA management literature to identify tasks and artifacts beneficial for this type of enterprise transformation. Furthermore, we compare these findings with the results of a survey conducted among experts at three European EA management conferences. Both, literature analysis and expert survey help to build a basis for further research regarding the application of EA management during M&A.

1 MOTIVATION

Over the past century, the appearance of Mergers & Acquisitions (M&A) remained remarkably high. As an ongoing trend, enterprises increasingly establish M&A as a strategic management instrument (Gerds & Schewe 2009; Jansen 2008). Regarding the future, analysts likewise predict a high level of M&A activities (Capital 2011). Consequently, for many enterprises M&A are not considered as individual events, but rather represent common instruments of modern business strategies.

M&A affect the whole enterprise and result in a multitude of complex transformation projects (Gerds & Schewe 2009; Jansen 2008; Penzel 1999). The transformation includes the majority of an enterprise's domains, among others, elements like business strategy, financials, law, products, processes, applications, and infrastructure. Figure 1 depicts a typical M&A process which consists of three phases: merger planning, transaction, and post-merger integration (PMI). Along this way, it includes activities relating to different interdependent management disciplines involving a variety of special experts as well as internal and external stakeholders (Jansen 2008; Picot 2008).

Figure 1: M&A process.

The merger planning phase typically includes strategic planning of M&A activities, analysis of the environment, identification of candidates, and a high-level valuation of possible target scenarios. The transaction phase starts with the initial contact and negotiations with a target enterprise. This phase includes financial planning, due diligence, pre-closing integration planning, and corporate valuation. It ends with the official announcement of the merger, contract signing, antitrust clearance and is completed with the final closing that includes the payment. At this time the formerly independent enterprises close their deal and legally become one single company. During the PMI phase, a post-closing integration plan is worked out allowing to implement the integration of strategy, organization, business processes, systems, administration,

operations, culture, and external relationships of the enterprise. Further activities include monitoring of progress, a formal post-merger audit, and a possible follow-up restructuring (Jansen 2008; Picot 2008).

Enterprise Architecture (EA) management is an approach for analyzing, planning, and controlling as-is and target states of the enterprise in terms of business, information systems, and technology architecture, based on an overarching EA model (Aier, Riege & Winter 2008; Buckl & Schweda 2011). Thereby, the main benefits EA management offers are (cf. (Aier, Riege & Winter 2008; Buckl & Schweda 2011; The Open Group 2009)):

- Creation of a holistic perspective on the enterprise, comprising business & IT elements
- Foster communication by defining a common language for multidisciplinary stakeholders
- Gathering information from differing sources and provisioning of consistent decision base

These aspects are considered as challenges enterprises are confronted with during M&A (Gerds & Schewe 2009; Picot 2008; Penzel 1999). By contrast, different authors (e.g., van den Berg & van Steenbergen 2010; Ross, Weil & Robertson 2006) explicitly propose EA management as a holistic approach for enterprise transformation. However, the application of EA management methods and models in enterprise transformations like M&A has not been subject to in-depth research yet. In this paper we therefore address five central questions detailing the role of EA management in M&A from a literature and industry perspective. Thereby, the questions range from general to specific:

- Is there a general reference to M&A in EA management literature?
- For which phases of an M&A process is EA management relevant?
- Which are typical EA management tasks carried out during M&A?
- Which EA management artifacts are used to perform these tasks?
- Have these artifacts been designed and/or evaluated by empirical means?

To answer these questions we follow a three-fold approach. In the first step, we conducted a literature analysis covering current EA management books. Secondly, we captured the opinion of experts attending three leading European practitioner conferences for EA management with the help of a survey. Lastly, we compared their experience with our findings originating from literature.

The remainder of this article is structured as follows: Section 2 describes the results gained by means of the literature study. Section 3 documents planning and execution of the expert survey and compares its results with those from the literature study. Finally, Section 4 concludes with a summary complemented by a critical reflection and indications on future research topics in this area.

2 LITERATURE ANALYSIS

2.1 Focus and Method

Over the last years, the body of knowledge with regards to EA management matured steadily (Aier, Riege & Winter 2008). However, the application of this knowledge in the light of an enterprise transformation such as M&A just starts to be subject of research (Freitag, Matthes & Schulz, 2010; Ross, Weill & Robertson 2006). We conducted a literature study comprising 13 recent English and German EA management books published between 2005 and 2011, focusing on their contribution towards M&A. We focused our literature study on EA management books, as they summarize findings of continuous and quality-assured research work being gained throughout years of research and knowledge accumulation. Thereby, we build a solid foundation to complement our literature study with journals and conference papers in a next research step.

We studied EA management literature in five stages ranging from the relationship to concrete EA management tasks and artifacts. In Step 1 we examined the general relationship between EA management and M&A, i.e., if the books refer to M&A at all, e.g., as an application domain, a use case, or a driver. In the course of Step 2 we worked out if the author(s) refer(s) to a distinct phase of the M&A process in which EA management can be applied. In Step 3 we investigated if the author(s) address(es) a concrete M&A task being supported by EA management. During Step 4 we identified details on the specific artifact in terms of concrete methods, models, or visualizations. Finally, in Step 5 we examined the sources regarding the usage of empirical means, e.g., interviews, case studies, or surveys.

2.2 Analysis Results

Five EA management books, Bernard (2005), Johnson and Ekstedt (2007), Niemann (2006), Schekkerman (2008), Wagter et al. (2005) do not refer to M&A at all. A second group of books refers to M&A but does not explicitly dwell on this topic. Proper et al. (2008, p. 6) mention M&A as one

driver for change, while Hanschke (2009, p. 328) provides a definition of the term as part of her book's glossary.

Two books speak of the PMI phase as an application domain for EA management and list corresponding tasks (Engels et al. 2008, p. 84-86, 169, 232, 277; Keller 2006, p. 98). Engels et al. (2008) list consolidation of business processes and the application landscape as EA management tasks. Additionally, they mention M&A as a reason for data redundancies. Keller (2006, p. 98) considers application and infrastructure consolidation as EA management tasks during the PMI phase and proposes patterns for application consolidation including influence factors and risks.

Besides integration, Ross et al. (2006, p. 176-181) mention knowledge transfer and provision of standardized best practices as relevant EA management tasks. The authors describe the effects of M&A on the enterprise's foundation for execution and the influence of different architecture maturity levels at the acquirer and seller company side. They propose architectural approaches like unification, replication, coordination that can be applied in M&A situation. Additionally, the authors illustrate these strategies by means of three case studies (UPS, CEMEX, and 7eleven Japan).

In addition to the PMI phase, Schwarzer (2009, p. 85-86) also refers to the merger planning phase, in which EA management provides information about necessary measures that have to be prepared and implemented. The author points out that IT plays an important role during M&A since it provides the basis for the future integration of the business processes. Furthermore, she emphasizes that EA management helps to consolidate of IT organization and IT landscape in the PMI phase by as-is and target architecture planning.

The work of Lankhorst et al. (2005, p. 108-110) focuses on EA modeling. The authors name modeling of processes, organizational structure, business functions, IT, applications, and services, as well as the creation of a common understanding among stakeholders and application consolidation as EA management tasks during M&A. The authors include a PMI example from the insurance industry to demonstrate the applicability of their EA models.

Berg and Steenbergen (2010, p. 4, 25, 27, 37, 50, 134, 137) refer to M&A, including all phases, as one application domain for EA management. Based on a fictitious M&A example from the banking industry, the authors motivate the importance of EA management and the implementation of an EA framework. The researchers propose EA management in order to achieve synergies between both companies being one key goal of M&A. The EA framework can be applied to map the architectural landscape, detect overlaps, and identify what needs to be changed and what needs to continue. As concrete tasks, they mention the homogenization of infrastructure and processes and the support of standardization.

2.3 Summary

Five authors do not bring up M&A at all, while eight publications at least mention the topic. However, these EA management books do not elaborate on M&A in detail, the description of tasks remains vague. If M&A is addressed, authors dedicate a maximum of five pages. Most contributions are limited to the PMI phase. EA tasks mentioned are mostly focused on IT consolidation and integration work as well as support of communication and modeling. The authors do not provide concrete EA artifacts explicitly addressing the challenges of M&A. When it comes to empirical means, the authors stick to fictitious examples or case studies.

3 EA MANAGEMENT SURVEY

After shedding light on the role of M&A in EA management from a literature point of view, this section presents the key results of a survey conducted among EA management practitioners. In line with the literature analysis, the survey centered on how EA management is contributing to M&A today as well as expectations regarding its future role. Subsequently, we summarize the results from the survey and compare them to the literature.

Table 1: EA management survey and participation details.

Conference name	Location / Date	Issued	Returned
EAM Forum 2011	Frankfurt, Germany; Feb 2011	50	14 (28%)
The Open Group Architecture Practitioners Conference	London, UK; May 2011	50	16 (32%)
EAMKON 2011	Stuttgart, Germany; May 2011	35	14 (40%)

3.1 Survey Setup

In terms of the survey setup we followed the questionnaire design process suggested by Frazer and Lawley (2000). Aiming at a high return rate and completeness of the answers, the survey was limited to one page, containing 15 concise questions. The distributed sheet was subdivided into three main parts: participant's background, questions about the company's EA management, and questions referring to M&A. The survey was conducted at three European EA management conferences between February and May 2011 (cf. Table 1). We selected these conferences on purpose as their audience represents a homogenous sample of EA practitioners in Europe. However, the survey sample cannot be considered to be formally representative (cf. Mayer 2008). In total, 44 of 135 participants returned the fully completed survey, resulting in a response rate of 33%.

3.2 Practitioners' Perspective and Comparison to Literature

The first questions addressed the company's industry sector and the individual role of the survey participant. From their industry background, the participants represent a homogenous group. Regarding their roles, the majority of participants follow an EA management profession. Domain architect subsumes different roles that focus on one architecture domain (e.g. business, application, and technology). In both charts (cf. Figure 2, 3), all responses mentioned only once have been assigned to the category 'other'.

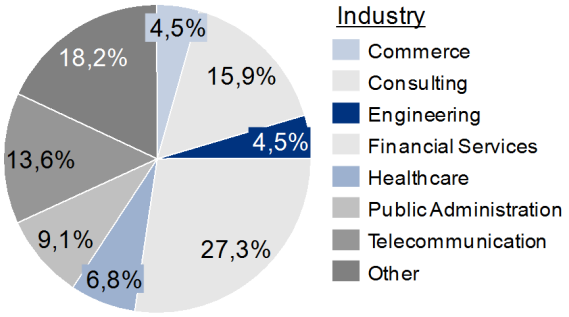


Figure 2: Participants' background (1).

With respect to the overall relevance of M&A, 63.6 % of the participant's companies have been involved in M&A in the past, while 56.8 % expect M&A to be relevant for their company in future.

With a group of four questions, we addressed the current role of EA management during M&A. The

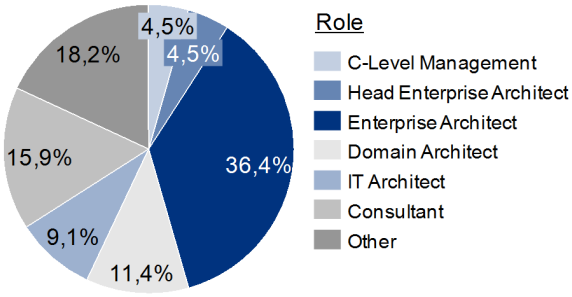


Figure 3: Participants' background (2).

support of M&A projects is seen as a responsibility for EA management in 29.6 % of the participant's companies (47.7 % no, 22.7 % not specified). Looking back, 23.3 % of the participants indicated that in their companies, enterprise architects have been engaged in M&A projects in the past (62.8 % no, 13.9 % not specified). While for current literature, M&A is considered as application domain for EA management, the topic has still not found its way into the minds of the practitioners.

To go into more detail regarding the role of EA management, we asked those participants whose companies did engage enterprise architects in M&A, to indicate the respective process phase. As Figure 3 displays, enterprise architects mainly contribute to the PMI phase with minor involvement in the merger planning and transaction phase. This situation is accounted for in current EA management literature, where the majority of publications refer to the PMI phase while only a small number mentions earlier process phases.

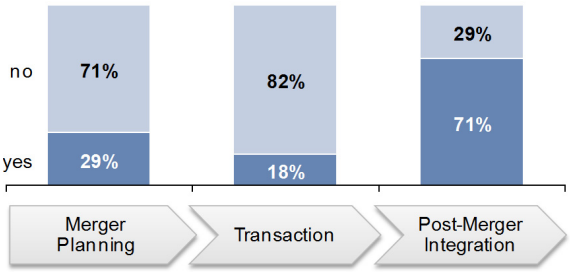


Figure 4: EA management support per M&A phase.

In the survey we offered an additional free text field to collect those tasks which are assigned to enterprise architects during M&A today. The tasks mentioned by the survey participants are mainly part of the PMI phase or general EA management tasks, with one exception. Regarding the PMI phase, subsequent tasks were mentioned: integration planning, consolidation or respectively integration of IT and processes, business and IT integration,

migration of applications and data, and software selection. General EA management tasks included scoping, providing transparency, IT master planning, target architecture design, governance, and project management. One participant brought up due diligence as an activity performed during the transaction phase.

Complementary, we offered a free text field to find out how enterprise architects could support M&A projects in the future. Some of the tasks mentioned here as future EA management responsibilities have also been stated as tasks performed today, namely: due diligence, target architecture design, consolidation of IT and processes, providing transparency, and project management. In addition, the participants mentioned the review of a target enterprise's as-is architecture and support of C-level management (e.g., CIO, CTO) in decision making (e.g. by pointing out costs of integration) which are part of the merger planning phase. Furthermore, they stated the following general EA management tasks: development of integration scenarios, preparation of a business capability roadmap, dependency analysis, providing a consolidated information base, and mapping of business and IT capabilities.

Similar to the asked experts, literature considers

- consolidation of organization and business process, applications, and infrastructure,
- dependency and redundancy analysis,
- identification of focus areas (scoping) and measures required, and
- as-is and target-architecture planning

as tasks which should be performed in the course of the PMI phase. However, instead of providing in-depth details perused sources solely lists those tasks.

Standardization of best-practices as well as knowledge transfer are considered as being general EA tasks from both, literature and participants. Additionally, literature proposes EA modeling and the creation of a common understanding. Both tasks were not mentioned by the participants. In turn, participants considered governance, project management, due diligence, application and data migration, preparation of business capability roadmap, and support of management decision making as essential EA task during M&A. Remarkably, these activities were not addressed by examined EA management literature.

Finally, we approached the participants with our central question straightforward. We asked for their opinion on the potential of EA management to contribute to the success of M&A as a special type of enterprise transformation. In retrospective, 60.0 % of our survey participants stated that EA management would have made an M&A in their company more successful, while only a very small percentage of 2.5 % did not see any value in applying EA management (37.5 % not specified). These figures show that practitioners see a real value of EA management in the context of M&A. Certainly, the outcome for this last question is biased, as the majority of participants (61.4 %) were enterprise architects who work in this management discipline. The literature analyzed does not explicitly confirm that the application of EA management improves the success rate of M&A.

3.3 Summary

The survey revealed that the majority of practitioners think that in retrospective EA management would have made M&A more successful. However, today only one out of three practitioners sees support of M&A as a responsibility of EA management while even a smaller number was actively involved in enterprise transformations. Nonetheless, the survey participants pointed out several EA management tasks they consider beneficial for M&A. These tasks were not limited to the PMI phase. Furthermore, practitioners stated that today EA management mainly contributes to the PMI phase with minor involvement in the merger planning and transaction phase.

4 CONCLUSIONS

As one special type of enterprise transformation, M&A often struggle to capitalize the initially expected benefits. EA management is promoted as an approach of enterprise transformation. Given that only minor research has been conducted regarding the role of EA management in M&A, this paper investigates on the topic. We perused 13 current EA management books focusing on their contribution towards M&A. Afterwards, findings were compared with results from an expert survey conducted at three European EA practitioner conferences.

As we found out in our studies, examined literature discusses the role of EA management in M&A only on a very general level. While current EA practitioners consider M&A as one of their application domains, their actual degree of involvement was low at the time we conducted the survey.

We are aware of the introductory character of our results researching on EA management in the

context M&A. Our primary goal is to provide a starting point by incorporating findings from the literature analysis as well as from a practitioner survey. For this reason and the page limit constraint of six pages, the analyzed literature was limited to recent EA management books. An extension of literature analysis during further research steps could include other publication types and sources from related research domains. Regarding the expert survey, further work should increase the sample size, be extended to groups other than enterprise architects, and could be enhanced with complementary questions detailing on specific aspects in more detail.

In all, the expected applicability of EA management has to be proven in practice. Therefore, enterprise transformation and in particular M&A remain an important research field for EA management. Besides the communication of empirically gained experience and lessons learned, future research should include the design and evaluation of concrete EA management artifacts that can be applied during M&A.

ACKNOWLEDGEMENTS

The research findings presented in this paper were possible thanks to the support of all survey participants. We wish to acknowledge the experts for sharing their experiences and their time spent.

REFERENCES

Aier, S., Riege, C., Winter, R. 2008. Unternehmens-architektur - Literaturüberblick und Stand der Praxis. *Wirtschaftsinformatik*. 50(4):292–304.

van den Berg, M., van Steenbergen, M. 2010. *Building an Enterprise Architecture Practice*. 1st ed. Springer, Dordrecht/Netherlands.

Bernard, S. A. 2005. *An Introduction to Enterprise Architecture*. 2nd ed. AuthorHouse, Breinigsville/ USA.

Buckl, S.; Schweda, C.M. 2011. *On the State-of-the-Art in Enterprise Architecture Management Literature*. Technical Report, Chair for Software Engineering of Business Information Systems. Technische Universität München, Munich/Germany

Capital 2011. Kenn-Ziffern zu Firmenübernahmen.

Engels, G., Hess, A., Humm, B., Juwig, O., Lohmann, M., & Richter, J.-P. 2008. *Quasar Enterprise: Anwendungslandschaften serviceorientiert gestalten*. 1st ed.. Dpunkt.Verlag, Heidelberg/Germany.

Frazer, L., Lawley, M. 2000. *Questionnaire Design and Administration: A Practical Guide*. 1st ed. John Wiley & Sons Ltd, Milton/Australia.

Freitag, A., Matthes, F., & Schulz, C. 2010. A method for consolidating application landscapes during the post-merger integration phase. (ABICT). Wisla/Poland.

Gerds, J., Schewe, G. 2009. *Post Merger Integration: Unternehmenserfolg durch Integration Excellence*. Springer-Verlag, Berlin/Germany.

Hanschke, I. 2009. *Strategic IT Management: A Toolkit for Enterprise Architecture Management*. 1st edition. Springer, Berlin and Heidelberg/Germany.

Jansen, S. A. 2008. *Mergers & Acquisitions: Unternehmensakquisitionen und -kooperationen. Eine strategische, organisatorische und kapitalmarkt-theoretische Einführung*. 5th ed. Gabler, Wiesbaden/Germany.

Johnson, P., Ekstedt, M. 2007. *Enterprise Architecture: Models and Analyses for Information Systems Decision Making*. 1st ed. Professional Pub Serv, Pozkal/Poland.

Keller, W. 2006. *IT-Unternehmensarchitektur. Von der Geschäftsstrategie zur optimalen IT-Unterstützung*. 1st ed. Dpunkt.Verlag; Heidelberg/Germany.

Lankhorst, M. 2005. *Enterprise Architecture at Work. Modeling, Communication and Analysis*. 1st ed. Springer, Berlin and Heidelberg/Germany.

Mayer, H. O. 2008. *Interview und schriftliche Befragung*. 5th ed. Oldenbourg Wissenschaftsverlag, München/ Germany.

Niemann, K. 2006. *From Enterprise Architecture to IT Governance: Elements of Effective IT Management*. 1st ed. Vieweg+Teubner, Wiesbaden/Germany.

Op't Land, M., Proper, E. 2008. *Enterprise Architecture: Creating Value by Informed Governance*. 1st edition. Springer, Berlin and Heidelberg/Germany.

Penzel, H.-G. 1999. Post Merger Management in Banken und die Konsequenzen für das IT-Management. *Wirtschaftsinformatik*, 41(2):105-115.

Picot, G. 2008. *Handbuch Mergers & Acquisitions*. 4th ed. Schäfer-Poeschel, Stuttgart/Germany.

Ross, J. W.; Weill, P.; Robertson, D. 2006. *Enterprise Architecture as Strategy: Creating a Foundation for Business Execution*. 1st ed. Mcgraw-Hill Professional, Boston/USA.

Schekkerman, J. 2008. *Enterprise Architecture Good Practices Guide: How to Manage the Enterprise Architecture Practice*. 1st edition. Trafford Publishing, Victoria/Canada.

Schwarzer, B. 2009. *Einführung in das Enterprise Architecture Management*. 1st edition. Books on Demand GmbH, Norderstedt/Germany.

The Open Group 2009. TOGAFTM Enterprise Edition Version 9.http://pubs.opengroup.org/architecture/ togaf9-doc/arch/. Accessed Aug, 12th 2011.

Wagter, R., van den Berg, M., Luijpers, J., van Steenbergen, M. 2005. *Dynamic Enterprise Architecture: How to Make It Work*. 1st ed. John Wiley & Sons, Hoboken/USA.

A Comparison of Two Business Rules Engineering Approaches

Lex Wedemeijer
School of Computer Science, Open University, Valkenburgerweg 177, Heerlen, Netherlands
lex.wedemeijer@ou.nl

Keywords: Business Rules, Requirements Engineering, Aspect Oriented Modelling, Modelscope, Relation Algebra, Ampersand.

Abstract: We compare two contemporary approaches under development for business rules engineering with the aim to understand their coverage of business rules and their potentials for requirements engineering. One approach, Aspect Oriented Modeling, focuses on events, state transitions and the synchronization of transitions between objects. The other approach, Ampersand, focuses on invariant rules that should be complied with regardless of the business events taking place. Our comparison brings out that either method can adequately capture some types of business rule, but not others. We conclude that a combination of the two approaches may be a significant contribution to the methods and tools for business rules engineering currently available.

1 INTRODUCTION

All businesses operate according to rules. The rules, whether formally acknowledged or tacitly assumed, influence and control the behaviour of the business. Various approaches to capture, model, implement and enforce business rules exist (zur Mühlen and Indulska, 2010). Still, which approach toward requirements engineering for business rules is best suited to what business environment is open for debate. This paper looks at two contemporary methods and tools under development. The first method is called Aspect Oriented Modelling, or AOM, and it comes with a tool ModelScope (McNeile and Roubtsova, 2010). The other method and tool is called Ampersand (Joosten, 2007). We compare the two by taking the leading example put forward by proponents of one method, and redevelop that example using the opposite method and toolset.

Our aim is to learn and understand about the coverage of business rules and potential for requirements engineering, outlining major differences and semantic issues that we encountered. Doing so may provide useful insights to method engineers engaged in the creation of new and improved approaches for business rules engineering. However, our intent is not to present a thorough evaluation of the two approaches, that are as yet immature and lack a track record of business engineering projects.

The outline of the paper is as follows.

Section 2 sets the stage. We introduce the notion of Business Rule as the common ground, and we discuss the selection of our two approaches.

Section 3 introduces the Ampersand leading example, which is about an IT Service Desk, and we discuss its redeveloped version using the AOM approach and ModelScope tool. Section 4 describes the fictitious Banking example of the AOM approach and outline its redeveloped version using the Ampersand approach and tool.

Lessons learned from the two translate-and-redevelop exercises are presented in section 5.

Section 6 presents our conclusions. We advocate as a future direction the integration of the two approaches to combine their powers to capture, model, implement and enforce business rules.

2 BUSINESS RULES IN INFORMATION SYSTEMS

A business rule, as defined by the Business Rules Group is "a statement that defines or constrains some aspect of the business. It is intended to assert business structure or to control or influence the behavior of the business" (Hay, Kolber et al., 2003). Rules can be classified into five broad categories: transformation rules, integrity rules, derivation rules, reaction rules, and production rules (Wagner, 2005). Business rules can and will apply to people, processes, and/or overall corporate behaviour, but in

this paper we will only consider rules explicitly captured in some information system of the business.

Business rules in an information system on the one hand, describe and constrain the business data (definitions and recorded data), i.e. the 'terms' and 'facts' referred to in the Business Rules Manifesto (2003). On the other hand, the rules describe, guide and constrain the business behaviour, i.e. the events affecting the information in the system. It requires that all types of event must be described, their operations on data, and it should be clarified whether a particular event should be prevented, discouraged or encouraged according to the rules.

2.1 Selected Approaches

This paper is restricted to two approaches that we selected for three reasons.

First and foremost, each approach is selected for being based on just a single modelling paradigm, and consequently, each is strong in capturing one particular category of business rule as named above.

Second, the approaches are just emerging from the laboratory phase and the comparison may help to improve their fitness for use.

Third, their tools are open source and can be studied in isolation, not being part of some dedicated ERP system, database, or service bus infrastructure.

2.2 AOM and Modelscope Approach

AOM, Aspect Oriented Modelling in full, focuses on transformation rules and the synchronisation of state transitions of business objects. The AOM approach aims to capture object behaviour and synchronize (the composition of) multiple behaviours in the early modelling stages of engineering projects.

AOM relies on Protocol Modelling (McNeile and Simons 2006), an event-based paradigm in which models are composed from behavioural components called Protocol Machines (McNeile and Roubtsova, 2008). By composing and synchronizing behaviour of distinct Protocol Machines, the approach provides "a basis for defining reusable fine-grained behavioural abstractions" (McNeile and Simons 2006).

The tool to support this approach is called ModelScope.

The Modellers' Guide (ModelScope version 2.0, 2004) describes it as "a state transition diagram interpreter which understands how events change the state of objects, what events an object can accept based on its state, and how events create and destroy relationships between objects" (p.6). The idea is that a transition of a business object is accepted only if both its pre- and post-state comply with the rules expressed in ModelScope. Any violation of a rule is detected by the tool. It then either rejects the transition immediately (behaviourtype 'essential') or prompts the user to explicitly allow it (behaviourtype 'allowed'). Either way, the business behaviour is guided towards rule compliance.

ModelScope offers a default user interface to enter data about events, and a facility to provide persistent data storage. The tool also provides callback features for specifying inferences and calculations in java code. These callbacks are invoked whenever an event is presented to the model.

The ModelScope tool is available for free download at: `www.metamaxim.com`

2.3 The Ampersand Approach

Ampersand focuses on integrity rules, or more exactly on what is called invariant rules. These rules are characterized as "agreements on business conduct" that users in the business should comply with at all times. Basically, Ampersand constitutes "a syntactically sugared version of Relation Algebra" (Michels, Joosten et al., 2011). The idea is that the state of the business should at all times comply with the agreements that are expressed as invariant rules in Relation Algebra. Ampersand will determine all the violations of all the rules and report them to the user(s). Obviously, such listings contain derived data only, but instead of being regarded as superfluous data, the Ampersand approach views these violations as triggers. Each rule violation constitutes a signal to the user that work must be done to remedy the problem, thereby guiding the business behaviour towards rule compliance.

Ampersand is also the name of a software tool to compile scripts written in the Ampersand vernacular. The tool comes with a facility to provide persistent data storage, and a user interface can also be specified for data entry. The tool can present a diagram of the Conceptual Model complete with its populations and rule obeisances. Moreover, it can also produce an extensive functional-specifications document.

The Ampersand tool is available for free download via the SourceForge community at:
`www.sourceforge.com/ampersand`

3 AMPERSAND TO AOM

This section describes the characteristics of the Ampersand leading example, and outlines how this example is redeveloped according to the AOM ap-

proach and using the ModelScope tool. The challenge is to translate the invariant rules from Ampersand into corresponding events with synchronized state transitions in ModelScope.

3.1 IT Service Desk in Ampersand

The leading example for Ampersand is the IT Service Desk. The Conceptual diagram, with 5 concepts and 7 relations, is depicted in figure 1. The example lists several business rules. There are several multiplicity constraints that concern one relation only. The rules marked 1, 2 and 3, involve more than just one relation. In Ampersand, these are known as cyclic rules because the relations involved in them can be seen to form a cycle in the diagram.

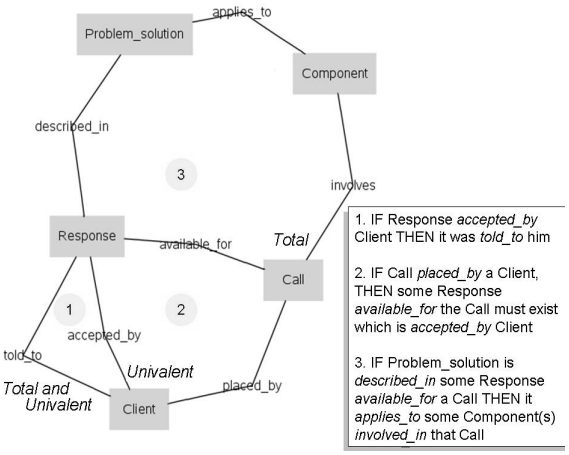


Figure 1: Conceptual diagram for Service Desk example.

The basic Ampersand script for this example contains some 60 lines of code. This example was translated into a ModelScope script of 500 lines of code, plus 20 components with callbacks of some 25 lines of java code each. These counts are slightly unfair though, because the Ampersand script lacks the user interface specifications that had to be added to the ModelScope script.

3.2 Translating to the AOM Approach

Findings from our translation effort are explained in this section. The main point is that we find AOM to be not very well suited to capture the invariant rules which are of prime importance in Ampersand.

In redeveloping the above example as a ModelScope model, we decided to translate each concept, and also each relation as a distinct object. The argument for not capturing the Ampersand relations as ModelScope relations will be explained below.

3.2.1 Integrity Rules

A major difference between Ampersand and AOM was immediately encountered. The Relational paradigm presumes two basic rules: Entity integrity and Referential integrity. Entity integrity dictates that every customer has a name that is unique among all customers. To contrast, the Object-Oriented paradigm does not assume such a rule, and ModelScope does permit different customers to have identical names. There is a demand that "the combination of the OID and machine-type properties of a machine must be unique in a system" (McNeile and Simons 2006) but as the system itself (i.e. ModelScope) creates and controls the OID's, the demand is trivial for the system, but not for the users. We had to make explicit provisions in the ModelScope model to capture the integrity rules:

- Entity Integrity is captured by rules that prohibit the insertion of an object instance whenever its name is either blank, or already present in that concept. Moreover, we also made sure to prevent duplicate tuples, i.e. we prohibit insertion of any tuple already present in the relation.
- Referential Integrity is safeguarded by enforcing certain events, not rules. In the user interface, we ensure that the user can insert a tuple only by selecting one instance of the source object and one instance of the target object. In addition, we provided Cascading Delete events to supersede the default Delete events that the user interface provides.

3.2.2 Lifecycle Support

Another difference between AOM and Ampersand is in their dealing with object life cycles, specifically in dealing with the end of the object life cycle. In Ampersand, a tuple or atom that no longer records relevant data about the real world is deleted from the data store. But in ModelScope, object data is never removed from storage; a tool feature that is deliberately incorporated to keep track of the state and the dynamics of any object instance at all times.

The designer however may introduce an object state 'deleted' and ensure that instances in this state will accept no events. Like integrity, this subtle difference also requires considerable attention when translating from Ampersand to AOM.

3.2.3 Multiplicity Rules

Multiplicity rules are not hard to implement in ModelScope. Two distinct behaviours, named "Univalent" and "Total", are included in the specification

of the designated relations. For ease of use, and for consistency with Ampersand, we assigned both behaviours the 'allowed' behaviourtype, i.e. ModelScope will signal any transition attempting to violate univalence or totality, but the user can override the signal and allow the transition.

Univalence can be handled per tuple of the relation. This behavioural component produces a signal whenever a tuple is inserted but the tuple's source object instance happens to be already represented in another tuple of the same relation.

Total is slightly harder to deal with. A violation of the Total multiplicity cannot be handled per tuple of the relation, but the entire extension of the source object has to be inspected. Even more: the insertion of a new instance of the source object must be allowed even though it means violating the cardinality rule on all relations specified as Total. This is because prohibiting the insertion of source data would effectively halt all data processing.

The Ampersand example happens to have no relations with injective or surjective cardinalities, but these might have been provided for in much the same way. Homogeneous-relation properties such as reflexive, symmetric or transitive, may also be provided for without much ado.

3.2.4 Why not Capture Relations as Relations

We indicated how we translated Ampersand relations to ModelScope objects, not relations. Our reason for doing so is that in ModelScope, the notion of 'relation' is restricted to functional relations only, in keeping with Object-Oriented approaches (Booch, Jacobson et al., 1997). That is, the tool enforces univalence but not referential integrity for its relations. In Ampersand however referential integrity holds rigidly for all relations, but univalence is an option. Turning the Ampersand relations into ModelScope objects allowed us to safeguard the rules that Ampersand had imposed on the various relations.

3.2.5 Rules Involving more than One Relation

Three rules in the Ampersand leading example involve more than one relation. Basically, all three rules are inclusion assertions. The first one is just a basic set inclusion (remember that by definition, a relation is a special kind of set). The two other rules use the relational composition operator, perhaps better known as natural join (Codd 1970).

To translate these rules from Ampersand to ModelScope, they must be written as rules for state transition of business objects, and we must ensure their proper synchronisation. We therefore need to consider all state transitions that might violate the rule; and also those that undo an existing violation.

We decided to create in ModelScope one behavioural component (object type) for each rule. In order to understand the impact of various behaviourtypes for this object, we went so far as to implement separate versions with distinct behaviourtypes for each rule. In doing this, we in fact merged a Business Data Model and an uncoupled Business Rules Model into a single ModelScope script.

For rule number 1, two transitions may produce a violation: the addition of an *accepted_by* tuple, and deletion of a *told_to* tuple. Likewise, a violation can be remedied in two ways: by adding a tuple in the former relation, or deleting one from the latter.

Whenever a composition of relations is involved, the number of state transitions that potentially cause or remedy a violation is multiplied. More complicated rules for relations will require ever more complex callback code to assess the effects of a single event or state transition being presented to the model. Indeed, the java callback code becomes prohibitively complex even for invariant rules of moderate size.

3.2.6 Once a Violation has been Allowed

The focus of Ampersand on invariant rules means that it reports on the rule violations that it detects in the persistent data. However, there is no notion of persistent violations in AOM, and the ModelScope tool does not provide any options to assess the stored data for static violations. Still, a report of rule violations in ModelScope similar to Ampersand can be produced, but it takes some tinkering. This is because rule violations are in fact derived data instances, and the ModelScope tool is not well suited to deal with this kind of derived data.

4 AOM TO AMPERSAND

The previous section described the translation of a leading Ampersand example into a working example for AOM. In this section we attempt the opposite direction: we take a fictitious Banking example that is leading for the AOM approach, and we outline how it may be redeveloped using the Ampersand approach and tool. The challenge is to translate the events and synchronized state transitions from ModelScope into a corresponding set of rules in Ampersand.

4.1 Banking Example

A leading example for AOM is called Banking. A diagram of behavioural components is depicted in figure 2, leaving out some components such as Address, Savings-Account and Transfer. The example covers various events to be synchronized, such as opening or closing a bank account, making deposits or withdrawals, guarding against overdrawing an account, and temporarily freezing an account in order to prevent withdrawals.

The example consists of a ModelScope script of about 100 lines of code, plus a dozen callback components with between 5 and 25 lines of java code. The Ampersand script after translation has 50 lines of code. Again, the count is unfair because Ampersand proved inadequate to capture all the rules, and we omitted details of the user interface specification.

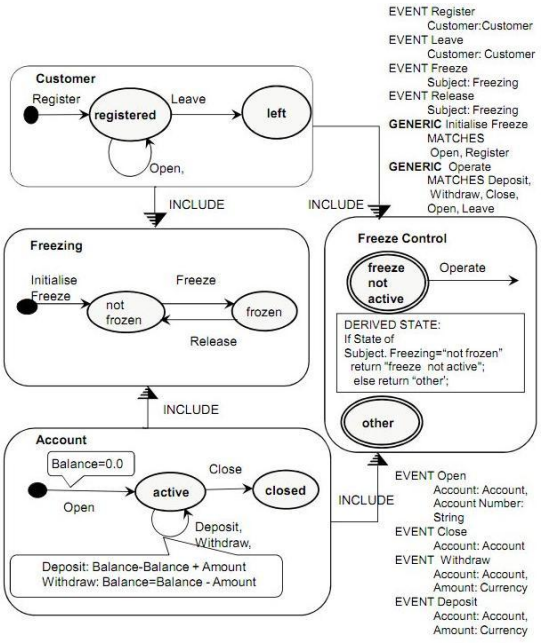


Figure 2: Diagram for Banking example.

4.2 Translating to Ampersand

The main issues encountered in the translation are explained in this section. We find the invariant rules of Ampersand to be excellently suited to capture several of the state transition rules in the example. However, various other rules embedded in the state transitions failed to be captured by Ampersand.

4.2.1 Determining Concepts and Relations

The diagram above is compliant with the AOM approach, but not with the Relational paradigm that Ampersand is based on. To create a working Ampersand script, several adjustments had to be made. For one, we omitted various derived attributes, states and behaviours from the AOM model. Furthermore:

- regular attributes such as the customers' address, and even the attributes that act as user-supplied identifying names (Accountnumber, Full-name), became concepts in their own right,
- as each behaviour-state is recorded by way of an attribute with a pre-assigned (hard-coded) range of possible values, we translated all STATES attributes into distinct concepts, except for those behaviours that have only a single state,
- three events in the AOM model carry an important data item: the amount of the cash deposits and withdrawals. This data must be recorded in the Ampersand model, and we added the appropriate concepts and relations.

Figure 3 is the translated Conceptual diagram.

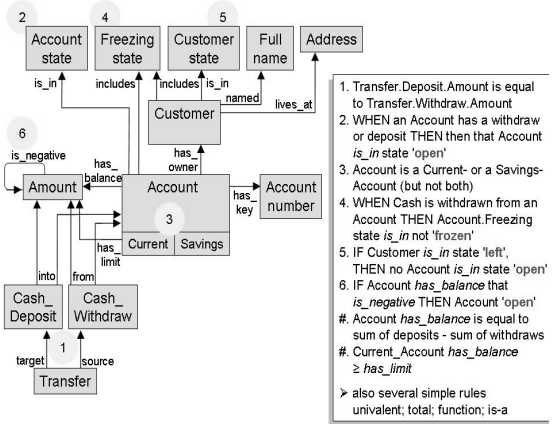


Figure 3: Conceptual diagram for Banking, translated.

4.2.2 Determining the Rules

Rules in the AOM example are expressed as constraints on state transitions, with the aim to guarantee that the data captured in the model cannot enter states that have no counterpart in reality.

We expected that all of the state transition rules in the AOM example would translate into non-cyclic rules on relations in Ampersand. In general, non-cyclic rules are expressed by way of one composite relation and specific (named) states of concepts, such as 'opened' or 'active', their most common type being multiplicity constraints. Our expectation proved to be correct, with just a few exceptions.

4.2.3 Implicit Capture of Cyclic Rule

When transferring money between accounts, the

withdrawn amount is assumed to be exactly equal to the deposited amount. This requirement is implemented implicitly, by having only one "amount" attribute in the "Transfer" event, to be used in both the Cash withdrawal and Cash deposit. We think that equality ought to be made explicit as a business rule.

Likewise, when money is transferred, the two accounts are assumed to be different. However, we found that the ModelScope script did not intercept transfers into the same account. The event was accepted, but resulted in an unexplained error.

4.2.4 State Transitions Capture Invariant Rule

The "Customer" object comes with three possible states: 'registered', 'pending leave', and 'left'. The Modellers Guide explains that before a customer leaves the Bank, all of the accounts must be closed. While the procedure to close the accounts is running, the customer's state is labelled as 'pending leave'.

So the general idea is that a left Customer has no open accounts. This is a splendid example of an invariant rule. However, ModelScope does not capture this rule. Instead, it implements constraints on state transitions simulating the process of a customer leaving. But: once a customer has left, ModelScope still accepts certain events for that customer, such as (un)freezing the status, or even depositing cash into his accounts that suddenly re-open. We attribute this imperfection to complexity: the number of state transitions to be accounted for rapidly rises as the number of objects, behaviours and events increases.

4.2.5 Immutable Property

For some attributes or object states, no transition is specified, e.g. the customer's Full Name, or a customer who has status 'left'. By implication, the attribute value or state will be for ever immutable. A corresponding immutability-rule might be specified, but we did not do so because Ampersand does not support temporal rules nor pre- and post-states.

5 LESSONS TO BE LEARNED

Lessons to be learned from these two translate-and-redevelop exercises are presented in this section. We point out basic differences between the approaches, and some weaknesses in the current tool support for each approach are pointed out.

5.1 Paradigm Mismatches

Both approaches support business rule engineering, but using fundamentally different paradigms.

5.1.1 Object-orientation vs Relation Algebra

A prime difference is the modelling paradigm underlying each approach: OO versus Relation Algebra. As a result, basic issues such as identity, entity and referential integrity are treated widely different as was discussed above.

5.1.2 Instance- vs Set-oriented Rules

A fundamental difference between AOM and the Ampersand approach is that AOM is designed for synchronization of events and state transitions on individual object instances, i.e. it implements transition rules. Such rules are expressed and evaluated on the basis of single instances following a fine-grained behavioural analysis.

Ampersand however is geared towards invariant rules expressed by assertions on entire relations. This is a very compact and powerful way to specify requirements on a large-scale, structural level. The downside is that, if a rule involves many relations, the details are lost of how a single state transition may incur one or multiple rule violations.

5.1.3 Dealing with a Rule Violation

Both AOM and Ampersand acknowledge that in a working business environment, a rule sometimes needs to be violated. But here again, the approaches are fundamental different.

The AOM approach focuses on state transitions, and rules are maintained by permitting or forbidding object transitions depending on pre- and post-states of the objects involved in the event. If ModelScope determines that some state transition rule is violated, the event is either rejected offhand, or it is presented to the user for acceptance (behaviourtype 'allowed'). But once accepted, the data is stored. Thus, violation is regarded a dynamic, temporary phenomenon.

The Ampersand tool does not deal with transitions or pre- and post-states. It inspects the stored data and calculates the (static) violations of the rules. These are reported to the user, leaving it to the user to assess the errors in the data and make amends. The tool is not concerned with the particular events, or chain of events, that may have caused the violations.

5.2 Unresolved Issues

Apart from the fundamental mismatches mentioned above, we also noticed several issues that need attention in both approaches.

5.2.1 Workflow Support

It is a matter of opinion (Hofstede, van der Aalst, et al., 2003) whether workflow specifications constitute business rules, or whether they are just a way to implement the underlying, more fundamental business rules.

ModelScope takes a somewhat ambivalent position by providing a behaviourtype 'desired'. It is expressed only in the user interface, where it indicates to the user which next step (event or transition) is desired for an object instance, when that instance happens to be on display. The engineer may apply this behaviourtype, but the tool provides no proper guidance or control. There is no link to some encompassing workflow; nor is it clear how to go about if different workflows desire conflicting state transitions for a particular behaviour.

Ampersand takes the position that a workflow is merely an implementation, one possible chain of actions that a user may undertake to remedy violations and comply with all the invariant rules. Therefore, Ampersand supports the user in remedying violations, but it offers no features to support detailed workflow design.

5.2.2 Derived Data

In practice, there is lots of derived data about, either stored as persistent data or used on the fly in the software code. Rules to control such data are sometimes called 'projector' rules (Dietz, 2008). It is a rule of thumb that derived data ought to be eliminated from data models, but we encountered several types of derived data in the examples.

One subtype of derived data is derived attribute value, such as the balance of a bank account. ModelScope can handle this kind of data very well by way of java callback routines. Ampersand however, relying on Relation Algebra only, does not deal very well with derived values.

However, another subtype is derived existence (or demise) of some instance of a concept or relation. In particular, we may consider every rule violation to be just such a derived instance. Here, the situation is reversed: Ampersand is well suited to handle such instances, but ModelScope can hardly deal with derived object instances or life cycles.

5.3 Shortcomings of the Tools

Both the Ampersand and ModelScope approaches, and their tools, are under development. Our translation efforts brought out various points of lacking user functionality, frustrating our goal of comparising the two approaches. Such shortcomings can well be mended in upcoming tool releases.

5.3.1 ModelScope Tool

The ModelScope interface offers a NAME attribute for the user to identify objects. However, the tool ignores these identifiers and uses internal object-identifiers instead. This is a cause of confusion whenever the user makes a mistake with identifiers, such as accidentally entering the same identifying name twice, which is accepted by the tool without signalling the duplicate.

Another issue that needs attention is derived data and code redundancy. It was mentioned how a compound business rule will involve more than one relation. Hence, every transition on any of those relations has to assess the same rule. The implication is that either the program code for that rule has to be duplicated for each transition (with a certain risk of becoming inconsistent if the code is edited), or some derived data object must be designed in order to encapsulate the code. ModelScope provides little support for either solution.

Regrettably, the current tool version does not generate diagrams or specifications for end users or engineers to ease their understanding, or to help in reviewing of the set of objects and events after completion. Such documentation would be helpful particularly if models grow in size and the numbers of events that must be synchronized increase.

5.3.2 Ampersand Tool

The current implementation of Ampersand cannot deal with rules that involve numeric calculations or comparisons, nor timestamps and date intervals. As a result, important business rules defy to be implemented, such as the basic rule that the balance of a bank account equals the sum of deposits minus the sum of withdrawals, or the rule that the negative balance of a Current account shall never be lower than the limit set for it.

A further drawback of the current Ampersand version is that it currently lacks a proper user interface for entering and editing data.

6 CONCLUSIONS

We conducted a detailed comparison of two emerging approaches and tools for modelling business rules. The first one is called Aspect Oriented Modelling (AOM) and comes with a tool named ModelScope. The other approach and tool is called Ampersand. By comparing these two, we aimed to learn and understand about the impact and consequences brought about by the choice of modelling approach underlying each method.

6.1 Conflicting Approaches

The two approaches were selected for focusing on one particular category of rules. The AOM approach is strong in transformation rules, and Ampersand is strong in integrity (invariant) rules.

Our comparison efforts show the negative consequences of this: AOM is weak in dealing with invariant rules, especially compound ones. And Ampersand provides poor support for state transition rules. Moreover, our translate-and-redevelop exercises disclose semantic issues about each method that were not previously noticed. The two approaches are conflicting in important aspects:

- in the modelling paradigm, relational or object-oriented, causing engineers to produce fundamentally different models,
- in the instance- or set-oriented perspective to be taken for rules, and
- in how a violation is dealt with.

The implication is that prior to conducting an actual analysis of business rules, a business engineer must decide which approach is most suited to the case at hand. If the case at hand is primarily concerned with the proper processing of events, and synchronization of dynamic state transitions for multiple objects, then the AOM-approach is at an advantage. A point of concern then is the size of the model, and the number of state transitions to be accounted for. If the focus is more on the static states of large data collections, and compliance to invariant rules, then the Ampersand approach seems to be more appropriate.

In our view, the two approaches provide as yet inadequate support for requirements engineering. This may be no surprise as both approaches, and their support tools, are still under development. The current state of affairs is that both approaches have serious drawbacks, and one approach is not superior to the other. At present, neither supports all needs of developers engaged in business rules engineering.

6.2 Limitations of the Comparison

In our comparison, we looked at only two small examples of business context. A thorough evaluation of the approaches would require a comparison of quality, flexibility, maintainability and other features of the delivered systems designs. But as the approaches and their ways of working are just emerging from the laboratory phase, no realistic operational models and implementations were available to be scrutinized and compared in our comparison.

Moreover, we selected the approaches for being based on just a single modelling paradigm. We found little support in either approach for the other categories of business rules: workflow (reaction and production rules), and derivation rules. From this, it may be speculated that other approaches that target a single modelling paradigm and a single category of business rules, will also fall short in providing adequate support for rule engineers in operational business environments.

6.3 Direction for Development

We stipulate that approaches may well be combined to augment one another. This is because system engineering efforts generally involve aspects of large-scale structural requirement analysis as well as fine-grained behavioural analysis.

Ampersand enables to express invariant rules that are attractively simple, yet have a wide impact, as a single rule can encompass multiple relations each with extensive populations. These features are needed in early stages of requirements engineering, when overall structure and consistency is the issue, not detail.

The AOM approach is strong in the modelling of events and synchronization of multiple behaviours, important features in later stages of engineering, when precise details are at stake.

An engineering approach and companion tool combining the expressive power of invariant rules of Ampersand with the detailed capabilities of controlling state transitions of AOM, would significantly contribute to the field of business rules engineering. It would permit to capture the invariant rules having a large scope in the early stages of engineering, whereas fine-grained rules for state transitions and workflow rules could be added later on. We expect that it is possible to reduce the size of the overall models and to keep the number of state transitions moderate. Combining the capabilities would reduce the drawbacks of either approach, and provide a more complete coverage of the engineering needs,

which in turn would result in a higher quality of deliverables. Work to bring event synchronization capabilities to the Ampersand metamodel has begun, but it has not resolved all the fundamental differences that must be overcome in the conflicting approaches (Roubtsova, Joosten et al, 2010).

The final word has yet to be said for the best approach and tool, depending on the kind of business environment, to supporting business rules and requirements engineering.

ACKNOWLEDGEMENTS

We wish to thank the organizers and reviewers of the symposium on Business Modeling and Software Design for their helpful comments on an earlier version of this work. We also thank dr. Roubtsova and prof. Joosten for their assistance with the Modelscope and Ampersand toolsets.

REFERENCES

Booch G, Jacobson I, Rumbaugh J. Unified Modeling Language, Rational Software Corporation, version 1.0, (1997)

Business Rules Manifesto (2003). At: www.business rulesgroup.org/brmanifesto.htm. Version 2.0. Edited R.G. Ross. Last accessed 24 march 2012

Codd EF. (1970). A relational model of data for large shared data banks. In: *Communications of the ACM* 13(6): 377-387.

Dietz JLG. (2008). On the Nature of Business Rules. In: *Advances in Enterprise Engineering* eds. Dietz, Albani and Barjis, Springer Berlin Heidelberg. 10: 1-15.

Hay JD, Kolber A, Anderson Healy K. (2003) *Defining Business Rules - what are they really*. Final report. BusinessRulesGroup. At: www.businessrulesgroup .org/first_paper/BRG-whatisBR_3ed.pdf. Last accessed 24 march 2012

Hofstede A ter, Aalst W van der, et al. (2003). Business Process Management: A Survey. In: *Business Process Management*. M. Weske, Springer Berlin / Heidelberg. 2678: 1019-1019.

Joosten S. (2007) Deriving Functional Specification from Business Requirements with Ampersand. At: icommas .ou.nl/wikiowi/images/e/e0/ampersand_ draft_2007nov.pdf. Last accessed 24 march 2012

McNeile A, Simons N. (2006) Protocol modelling: A modelling approach that supports reusable behavioural abstractions. In: *Software and Systems Modeling* 5(1): 91-107.

McNeile A, Roubtsova E. (2008). CSP parallel composition of aspect models. In: *Proceedings of the 2008 AOSD workshop on Aspect-oriented modeling*. Brussels, Belgium, ACM: 13-18.

McNeile A, Roubtsova E. (2010). Aspect-Oriented Development Using Protocol Modeling. In: *Transactions on Aspect-Oriented Software Development* VII. S. Katz, M. Mezini and J. Kienzle, Springer Berlin / Heidelberg. 6210: 115-150.

Michels G, Joosten Se, van der Woude J, Joosten S. (2011). Ampersand, Applying Relation Algebra in Practice. In: *RAMICS 2011, Relational and Algebraic Methods in Computer Science*. Rotterdam, The Netherlands, May 30-June 3. H. de Swart, Springer Berlin / Heidelberg. LNCS 6663: 280-293.

ModelScope Version 2.0 (2004), *Modellers' Guide Version 15*. At: www.metamaxim.com. Last accessed 24 march 2012

Roubtsova E, Joosten S, Wedemeijer L. (2010) Behavioural model for a business rules based approach to model services. June 2010. In: *BM-FA '10: Proceedings of the Second International Workshop on Behaviour Modelling: Foundation and Applications*.

Wagner G. (2005). Rule Modeling and Markup. In: *Reasoning Web*. eds. N. Eisinger and J. Maluszynski, Springer Berlin / Heidelberg 3564: 96-96.

zur Mühlen M, Indulska M. (2010). Modeling languages for business processes and business rules: A representational analysis. In: *Information Systems* 35(4): 379-390.

Semantic Business Analysis Model
Considering Association Rules Mining

Anna Rozeva[1], Boryana Deliyska[1] and Roumiana Tsankova[2]
[1]*Department of Business Management, University of Forestry, 10 Kliment Ohridski blv., Sofia, Bulgaria*
[2]*Department of Management, Technical University, 8 Kliment Ohridski blv.Sofia, Bulgaria*
arozeva@hotmail.com, delijska@gmail.com, rts@tu-sofia.bg

Keywords: Business Analysis, Knowledge Discovery in Databases, Association Rules, Domain Ontology, Semantic Model, Ontology Reasoning.

Abstract: Deriving models for intelligent business analysis by generation of knowledge through data mining techniques has proved to be highly theoretically researched and practically implemented topic in the field of decision support and business intelligence systems in the last decade. A general data mining task concerns discovery and description of relationships among items recorded in business transactions. The model of association rules is the one most implemented for revealing such relationships. In order to increase the decision support value of the output associative models the necessity for capturing and involving semantics from the domain of discourse has emerged. Ontologies represent the tool for structuring the concepts and their relationships as knowledge for a subject area that was established with the growth of the Semantic Web. The paper is intended to design a framework for implementing ontologies in the association rule analysis model that provides for involving semantics in the extracted rules by means of initial verification and optimization of the mining task by database scheme ontology and exploration of rules' interestingness by ontology reasoning process.

1 INTRODUCTION

Modern business analysis is inevitably information based. Therefore it faces the problem of dealing with continuously growing amounts of structured or unstructured data from a variety of sources. The main challenge consists in getting the "big picture" out of it for the sake of best serving the decision making. The general technology for processing data resulting in deriving summarized models is data mining (Larose, 2006). Models contain knowledge about a related domain. Mining tasks perform mainly classification, clustering or extraction of associations. The model describing associations between items on the basis of their mutual occurrence in transactions has proven to be of particular value for business analysis. It's represented by rules relating certain items X and Y with assigned support and confidence (Maragatham and Lakshmi, 2012). These rules represent new knowledge and are derived by processing raw data from transactions by data mining techniques which is also often referred to as knowledge discovery in databases (KDD) (Frawley et al, 1992). Models extracted by summarizing significant amounts of data are related to instances of objects from the domain of discourse and hence they lack the abstractness that is inherent to models in general.

At the same time knowledge about domains exist which is accumulated, stored for being used and shared through the resources of the Semantic Web. The knowledge represents conceptualization that articulates abstractions of certain state in reality (Guizzardi, 2007). The tool for its engineering is referred to as ontology. Obviously ontologies capture the domain semantics. The task is to map ontologies to extracted analytical models for verifying their correctness and for inferring new knowledge. "The role to be played by ontologies in KDD (and even their mere usability) depends on the given mining task and method, on the stage of the KDD process, and also on some characteristics of the domain and dataset" (Svatek and Rauch, 2006, p. 163). We propose a framework for implementation of domain ontologies in association rule mining with the goal for mining task verification on the database scheme, extracted rules conceptualization and further refinement through domain ontologies.

The remainder of the paper is organized as follows: The second section is a review on approaches for application of ontologies in mining databases for association rules extraction and results obtained. The third section presents a framework for extracting semantic association rules analysis model for knowledge generation from a database by ontology reasonong. The fourth section presents application results on sample database and ontology instantiations. The last section concludes with discussion on the effect for the semantic enrichment of the business analysis model.

2 ASSOCIATION RULES AND ONTOLOGIES

Association rule as defined by (Agrawal et al., 1993) is a triple (*I, S, C*), where I is an implication of the form $X \rightarrow Y$ denoting *if X then also Y*, *S* and *C* are interestingness measures for support and confidence. *X* and *Y* are items in database transactions and the rule correlates the presence of both sets of items in transactions, i.e. transactions that contain items of *X* tend to contain items of *Y* as well. S indicates the statistical significance indicating the percent of transactions that contain items of both *X* and *Y*. *C* measures rule's strength as the probability of their mutual occurrence. The extracted association rules have support and confidence greater than the predefined ones.

Ontology is represented in (Maedche, 2002) as 5-tuple of the form (C, R, H^C, F, A), where C is a set of concepts, R is a set of nonhierarchical relations among concepts, H^C is taxonomy of the concept hierarchy that defines relations among concepts c_1 and c_2 of type 'is-a' and 'has-a' mainly. F is a function that instantiates the relationships from R and A is a set of axioms that describes constraints. The definition is a formal description of the concepts and their hierarchical relationships in a specific domain as a piece of reality. Further on it's instantiated for an element of the domain by application ontology. Task ontology is designed and implemented with the purpose of modelling the knowledge for solving specific task within the application as shown in (Deliyska and Manoilov, 2012).

The ontology support of the KDD process has been within the scope of a lot of studies recently. In (Gottgtroy, Kasabov and MacDonell, 2004) a general framework for the mutual interdependence of ontology building and maintenance and the knowledge discovery is suggested. It's argued that ontologies facilitate each stage of the knowledge discovery process from improving the quality of the source data, feature selection by navigation through hierarchy and finally production of improved results by reasoning within ontology's links and relationships.

Framework for implementation of domain knowledge into association rules generation is proposed in (Antunes, 2008). It provides for formulation of constraints that control the mining process by using domain ontology. Two constraint types are defined, i.e. interestingness and content. While the interestingness measure refers to quantitative conditions on the frequency of mutual appearance of the items in transactions, the content constraint considers characteristics of the items that are present in the domain ontology. They are qualified as taxonomical when based on restriction among concepts, defined in the ontology taxonomy. Non-taxonomical constraints are referred to as relational and they are based on the ontology relations among concepts. The constraints guide the knowledge discovery process, providing the desired level of abstraction.

The framework presented in (Bellandi et al., 2007) provides for the extraction of constraint-based multilevel association rules with ontology support. The constraints for the mining process are defined from domain ontology as domain specific. The ontology is used for filtering the transaction instances sourcing the mining process. The system architecture involves interpretation module translating user constraints and passing them to an ontology query engine for excluding non-interesting rules and for presenting the interesting ones at the relevant abstraction level. It's stated that this approach improves association rules support and provides for decreasing the amount of useless rules discovered when source data are sparse.

While the frameworks discussed so far address the input of the mining process, the one proposed by (Marinica and Guillet, 2010) considers the association rules post mining phase with the aim to decrease the number of delivered rules so that they are useful and understandable for the user. An approach for pruning and filtering the discovered rules is designed. Ontologies are used for the integration of user knowledge at the post processing stage. Besides this the quality of discovered rules can be validated at different points in an interactive process by the domain expert.

The notion of multidimensional association rules has been introduced in (Wu et. al., 2007). The

definition refers to the star scheme of the data warehouse as source of the transactions for the mining process. The approach is focused on the stored data and aims to overcome the lack in their structural and semantic exploration. It proposes functions for the effective maintenance of the discovered knowledge. The stated problems are solved by designing two types of ontologies. The scheme ontology contains the warehouse metadata. The domain ontology constructs the domain knowledge for the mining subject as conceptual layer and relationships among the related concepts. They are implemented at loading data in the warehouse. It's pointed out that by this approach minimization of data mining searching boundaries and prevention of repeated mining are achieved. On the other hand by extending the association rule mining to items from the domain ontology generalization of items to concepts with richer semantics is achieved.

In our previous work (Rozeva et. al., 2011) we've designed a framework for generation of knowledge models from text documents which consisted of structure and knowledge models. Current work extends the categorization knowledge model presented there by exploring the association rules model. It is designed on mining set containing transaction items and acquires business semantics by reference to ontologies. The results obtained will support its involvement in mining a text document corpus. The related work review presented is the background for designing a semantic analysis model.

3 SEMANTIC ANALYSIS MODEL DESIGN

The proposed framework implements ontological reference both in the step of setting up the input to the association rules mining task and extending the value of extracted patterns. The functionality implemented in the task definition step provides for the optimization analysis of mining task parameters. It examines the input and predictable item sets and performs reasoning on designed database scheme ontology for ensuring non-redundant rule generation. On the basis of exploring key-based and hierarchical dependencies included in the scheme ontology, both the input and predictable item sets will be optimized. At the evaluation stage obtained rules are explored by reasoning on provided domain ontologies. The goal is to put the focus on the interesting rules. Such rules are considered the ones with items belonging to different domains. The proposed architecture is shown in Figure 1.

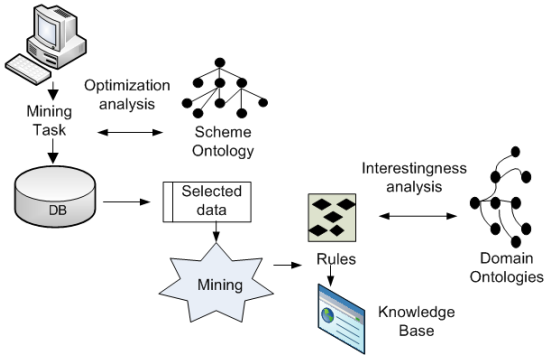


Figure 1: Framework of semantic analysis model.

The ontological reference at the front and back ends of the association rule generation process provides for the reduction of the number of rules obtained and for enhancing their value for business analysis purposes.

3.1 Scheme Ontology Design

The scheme ontology contains metadata of the database scheme. An excerpt of the designed scheme ontology is shown in Figure 2.

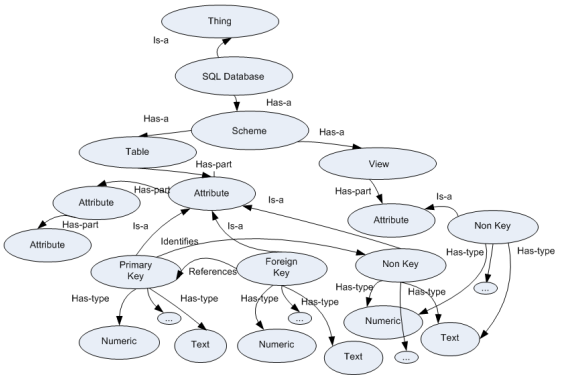


Figure 2: Database scheme ontology.

The top ontology concept *SQL Database* has subconcept *Scheme* and its subconcepts are the database objects *Table* and *View*. Their concept relationships to the ontology root are 'Has-a'. The concepts *Table* and *View* have subconcept *Attribute*. It is related to the superconcepts by 'Has-part' relationship. The *Attribute* concept which is subsumed by the *Table* concept has three subconcepts, i.e. *Primary key*, *Foreign key* and *Non-key*, related to it with 'Is-a' relationships. The *Attribute* subconcept of the *View* concept has just a

Non-key subconcept. The primary, foreign and non-key concepts have properties which are name and value type. Value types are: *numeric, text, date*, etc., which have 'Is-of-type/Has-type' relationships to the *Attribute* concept. The *Attribute* concept which is subsumed by the *Table* concept may represent hierarchy with levels defined by *Attribute* concept. The relationships between the hierarchy levels are of type 'Has-part'. For shortness concept instances are not shown in the scheme ontology.

A mining task MT for association rules specifies database tables, views or table and view related with many-to-one relationship; input and predictable attribute(s). The MT query-like representation adapted from (Wu et al., 2007) is:

```
Mine Association Rules
InputSet {IAttr1, IAttr2,…}
PredictSet {PAttr1, PAttr2, …}
From CaseTable Inner Join NestedTable
With MinSup%, MinConf%
```

The aim of the MT query optimization analysis is to identify input attributes which are functionally dependent. The dependency type is specified in the scheme ontology as being either on the primary key or between levels in a hierarchy.

3.2 Ontological Optimization of Mining Query Definition

The optimization of MT definition is proposed to be performed by reasoning on the database scheme ontology. A description logic reasoning tool Pellet is described in (Sirin, E., Parsia, B. et al., 2007). Reasoning concerns finding implicit facts in the ontology on the basis of explicitly stated facts. Basic reasoning tasks refer to proving satisfiability of a concept, subsumption of concepts, check an individual as instance of a concept, retrieving all the individuals that are instances of a concept and finding all the concepts an individual belongs to. Answering queries over ontology classes and instances for finding more general/ specific classes and retrieving the individuals matching it is a basic service performed in ontology reasoning. The reasoning tasks for checking MT definition retrieve the individuals of the *Attribute* concept and their descriptions and axioms. Mining query parameters are checked to match some of the retrieved individuals. Further on parameters are checked for consistency against the descriptions and constraints of the examined concept. Consistencies on primary key and hierarchical inclusion are considered. If the parameter set is consistent on the examined descriptions then the dependent parameter has to be removed from the mining query item set. For shortness parameter set with 2 items is considered. Further on is the specification of the ontology reasoning process:

```
InputParameterSet: {IAttr1, IAttr2}, SchemeOntology;
Concept→'Attribute';
Retrieve individuals of 'Attribute' → ABox;
Retrieve descriptions for ABox → TBox;
Check IAttr1∈ABox,
Check IAttr2∈ABox;
Check ∃ 'Identifies'.IAttr1,IAttr2 ⇒ Outputset {IAttr1};
Check ∃ 'Has-part'.IAttr1⊃IAttr2 ⇒Outputset {IAttr1}.
```

The optimization of mining query definition performed by ontological reasoning process decreases model's size and training time by preventing the generation of redundant association rules.

3.3 Analysis of Rules Interestingness

The interestingness analysis of mined association rules has been adapted from (Marinica and Guillet, 2010). We propose to perform it by reasoning on domain ontologies. The analysis is targeted at:

- Conceptualization / individualization along the domain ontology taxonomy;
- Filtering obvious rules, i.e. with the same subsuming concept.

The first task provides for the generalization / specialization of the left side (condition) and the right side (consequent) items of the extracted rules by implementing the subsumption or individual retrieval reasoning operations on the domain ontology. This analysis can be applied when the condition and consequent items refer to the same or to different domain ontologies. The second task aims at focusing on non-obvious rules. Obvious are considered rules with condition and consequent items having the same subsuming concept. The rule involving such items represents association between items from a common domain. The semantic value that is added by the analysis consists in revealing associations between items from different domains. The associations are considered more interesting when the condition and the consequent domains differ to the greatest extent possible. The measure

for the differentiation is the number of subsumptions that are to be performed for reaching their common subsuming concept as in the following reasoning process:

```
InputRuleSet: R, DomainOntology: DO;
R:{Condition, Consequent};
DO:Statement;
[a, b rdf:Statement;
rdf:subject:s;
rdf:predicate:p;
rdf:object:o]
Condition:a.o ∩ Consequent:a.o
⇒ {Condition, Consequent}⊂ a.s;
R → Non interesting;
Condition:a.o;
Consequent:b:o;
⇒ {Condition, Consequent} ⊄ a.s;
R → Interesting.
```

4 SEMANTIC MODEL IMPLEMENTATION

The proposed approach for semantic association rule model design has been implemented on purchase transaction tables from the sample database (Microsoft SQL Server Database Product Samples, 2012). The database scheme ontology from Figure 2 has been filled in with the corresponding instances. Administrative location and product domain ontologies have been designed. The MT queries that have been defined and the result from ontological reasoning analysis are presented in Table 1.

Association rules model has been trained with the Apriori algorithm with input and predictable parameters CustId and ProdId. The minimum probability was set to 10% and minimum support to 1%. The number of generated rules without and with optimization as shown in the last column is approximately 50%. By varying the support and probability values the rule number will be different.

Enhancement of model interestingness is performed by filtering the non-obvious rules. They are identified as belonging to different subsuming concepts. The case of rules that involve single condition item is considered.

Table 1: Mining task optimization.

MT Input	MT Predict	Optimization	Rules
CustId, CustName	ProdId	CustId, ProdId	158→49
CustId, Country	ProdId	CustId, ProdId	111→51

The task is performed by extraction of mined rules from the model as RDF triples {id, condition, consequent} from the tree-like structure shown in Figure 3.

ATTRIBUTE_NAME	ATTRIBUTE_VALUE	SUPPORT	PROBABILITY
Customer Key	11223	8	0,00053767054237516
v Assoc Seq Line Items(Sport-100)	Existing	8	0,00053767054237516

Figure 3: Model's node content.

The tree nodes are from the following types: with condition item only, with consequent item only and with both condition and consequent item values. Each node has the respective probability and support values attached. The nodes with both condition and consequent items are filtered and the RDF triples are obtained. The extracted subsuming concepts of the rule items are compared and the rule is either kept or discarded. An initial model with ProdId as input and predictable parameter and resulting predicted associations is shown in Figure 4.

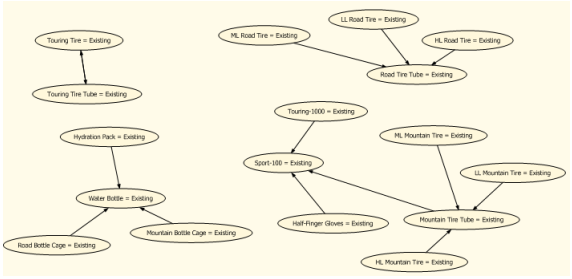


Figure 4: Initial model rules.

By applying interestingness reasoning on the Product domain ontology where the Product concept is subsumed by the Category concept the model shown in Figure 5 has been obtained. The rules which remained after performing the analysis display associations between ontologically remote items only.

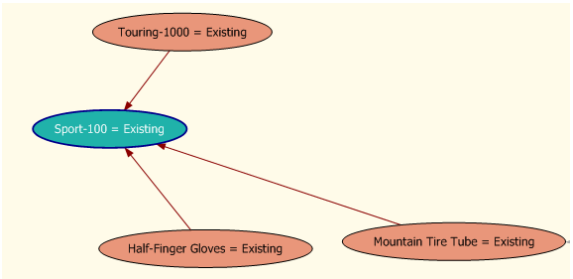


Figure 5: Interesting rules.

By subsumption operation on the RDF triples the rules can be further ontologically generalized.

5 CONCLUSIONS

The analysis model for extraction of associations between items from business transactions stored in a database presented in the paper introduces an innovative approach for capturing and using domain knowledge. It is intended to filling the gap between definition of the analysis task and the interpretation of obtained results and the examined business domain. Ontologies have been recognized and widely adopted as model for capturing this background knowledge. The proposed framework for designing semantic analysis association rules model implements two types of ontologies that provide background knowledge for the data source structure and the domain of discourse. The ontology content is made use of by means of reasoning process based on description logic. The reasoning on the data source ontology provides for the support and optimization of mining task definition. Key dependent and hierarchy related query parameters are identified by the reasoning process and discarded for the sake of generating non-redundant set of rules. Domain ontology reasoning is implemented for tuning rule interestingness. Interesting rules are considered those involving items from different domains. Reasoning process procedures have been presented. The proposed methodology has been evaluated on sample transaction database with reasoning on instantiated structure and domain ontologies.

Future work is intended in refining the reasoning process in order to be applied further on to mining associations between terms extracted from text document corpus with available ontology referring to e-Governance services. Application of the designed framework in automatic generation of ontologies will be researched as well.

ACKNOWLEDGEMENTS

The paper presents results of the project “Research and Education Centre for e-Governance” funded by the Ministry of Education in Bulgaria.

REFERENCES

Agrawal, R., Imielinsky, T. and Swami, A., 1993, Mining Association Rules between Sets of Items in Large Databases., In *Proc. ACM SIGMOD Conf. on Management of Data*, p.207-216.

Antunes, C., 2008, An Ontology-Based Framework For Mining Patterns In The Presence Of Background Knowledge, In *Proceedings of International conference on advanced intelligence (ICAI 2008)*, Post and Telecom Press, Baijing, China, p.163-168.

Bellandi, A., Furletti, B., Grossi, V. and Romei, A., 2007, Ontology-driven association rule extraction: A case study, In *Proceedings Workshop Contexts and Ontologies: Representation and reasoning in 16th European conference on artificial intelligence*, p.1-10.

Deliyska, B., Manoilov, P., 2012, Ontologies in intelligent learning systems, In Jin, Q. (ed.), *Intelligent learning systems and advancements in computer-aided interaction*, Information science reference. An imprint of IGI Global, p.31-48.

Gottgtroy, P., Kasabov, N. and MacDonell, S., 2004, An ontology driven approach for knowledge discovery in biomedicine, In *Proc. 8th Pacific Rim Int’l Conf. on Artificial Intelligence*.

Guizardi, G., 2007. On ontology, ontologies, conceptualizations, modelling languages and (meta) models. In Vasilecas, O. (eds.), In *Frontiers in artificial intelligence and applications, databases and information systems IV*, Amsterdam, The Nederlands: IOS Press, p.18-39.

Frawley, W., Piatetsky-Shapiro, G., and Matheus, C., 1992, Knowledge Discovery in Databases: An Overview. *AI Magazine*, Vol. 13, p.57-70.

Larose, D., 2006. *Data mining methods and models,* Wiley-IEEE Press.

Maedche, A., 2002, *Ontology Learning for the Semantic Web*, Kluwer Academic Publishers.

Maragatham, G., Lakshmi, M., 2012, A recent review on association rule mining, *Indian journal of computer science and engineering*, Vol. 2(6), p.831-836

Marinica, C. and Guillet, F., 2010, Knowledge-Based Interactive Postmining of Association Rules Using Ontologies, *IEEE Transactions On Knowledge And Data Engineering*, Vol. 22, No. 6, p. 784-797.

Microsoft SQL Server Database Product Samples, *AdventureWorks2008_SR4*, http://msftdbprodsamples.codeplex.com/releases/view/37109, Retrieved March 24th 2012.

Rozeva, A., Ivanov, M. and Tsankova, R., 2011, Business modelling for generation of knowledge from explicit data, In *Proceedings of First International Symposium on Business Modeling and Software Design (B.Shishkov, ed.),* Sofia, Bulgaria, p.114-121

Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A. and Katz, Y., 2007, Pellet: A Practical OWL-DL Reasoner, *Web Semantics*, 5 (2), p.51-53

Svatek V., Rauch J., 2005, Ontology-Enhanced Association Mining. In: *Semantics, Web and Mining, Joint International Workshops, EWMF 2005and KDO 2005*, p.163–179.

Wu, C.-A., Lin, W.-Y., Tseng, M.-C. and Wu, C.-C., 2007, Ontology-Incorporated Mining of Association Rules in Data Warehouse, *Journal of Internet Technology*, Vol.8, No4, p.1-9.

On the Applicability of the Notion of Entropy for Business Process Analysis

Peter De Bruyn, Philip Huysmans, Gilles Oorts and Herwig Mannaert
Normalized Systems Institute
Department of Management Information Systems, University of Antwerp, Prinsstraat 13, 2000 Antwerp, Belgium
{*peter.debruyn, philip.huysmans, gilles.oorts, herwig.mannaert*}*@ua.ac.be*

Keywords: Entropy, Business Process Analysis, Normalized Systems.

Abstract: Contemporary organizations need to be able to dynamically adapt, improve and analyze their business processes. While many approaches are available in literature, few of them tend to use typical engineering concepts for this purpose. In this paper, we employ the concept of entropy as defined in statistical thermodynamics to advance the field of business process analysis. After reviewing some existing literature on entropy and its application in business topics, we show how earlier insights from entropy reasoning in Normalized Systems theory may offer opportunities to be applied in business process engineering as well. The necessary entropy concepts are defined in a business process context, entropy occurence in business processes is illustrated and some initial principles for controlling the resulting entropy are discussed. Finally, some implications for both theory and practice (e.g., Service-Oriented Architectures) are reviewed.

1 INTRODUCTION

In current business environments, organizations are increasingly confronted with more demanding customers and fiercer competitors forcing them to continuously change and adapt their business plans, products, services and business processes. Consequently, a lot of research on the *improvement, optimization and change of business processes* has been performed recently. Typical variations of terms referring to this issue include Business Process Reengineering (BPR), Core process redesign, process innovation, business process transformation, organizational reengineering, Total Quality Management (TQM), etcetera (O'Neill & Sohal, 1999). Many of those approaches mainly provide overall project management related best practices and life cycles such as "secure management commitment", "discover process opportunities" and "inform stakeholders" (Kettinger, Guha & J.T.C., 1995) or general optimization techniques such as "have those who use the output of the process perform the process" and "link parallel activities instead of integrating their results" (Hammer, 1990). Such general guidelines have clearly proven their value in the past. However, while often claiming terms as "*design*" and "*engineering*", it is remarkable to note how few approaches actually apply traditional engineering concepts as the core of their method to optimize or change the considered business processes.

Hence, in this paper we will try to advance the field of business process analysis by applying the *entropy* concept from thermodynamics for this purpose. We will do so by applying the theoretical framework of *Normalized Systems (NS)*. NS is an approach to design evolvable modular structures, based on theoretically proven theorems. While it was originally applied at the level of software architectures (Mannaert, Verelst & Ven, 2011, 2012), its relevance at the organizational level has already been demonstrated previously (Van Nuffel, 2011; Huysmans, 2011). However, the NS theorems were initially proven and derived from the concept of stability as defined in systems theory. Recently, the existing theorems have been confirmed by reasoning based on the concept of entropy while simultaneously suggesting new insights (and two new theorems) at the software architecture level (Mannaert, De Bruyn & Verelst, 2012). Consequently, we will try to make an initial attempt in this paper to verify whether it is valuable to analyze business processes from the NS entropy viewpoint as well and whether new insights seem to emerge correspondingly.

The remainder of this paper will be structured as follows. In Section 2 some related work on the concept of entropy will be discussed, including some previous attempts to use the concept in management and

organizational research. Next, we will briefly summarize the essence of Normalized Systems theory in Section 3 and its extension based on entropy. Section 4 will discuss the usefulness of analyzing business processes from the NS entropy viewpoint, whereas Section 5 will deal with some of the resulting implications for theory and practice. Finally, some conclusions will be presented in Section 6.

2 RELATED WORK ON ENTROPY

In this section we will first provide some definitions and context regarding the concept of entropy. Next, we will discuss some earlier attempts of applying entropy reasoning to business and management topics.

2.1 Basic Concepts

Entropy, referring to the second law of thermodynamics, is generally considered to be a fundamental property in engineering sciences. The concept has been described and studied from many different perspectives, but all have basically the intention of describing the irreversibility of nature. Typically, more specific interpretations associated with entropy include (1) complexity, perceived chaos or disorder (Anderson, 2005), (2) uncertainty or lack of information (Shannon, 1948) and (3) the tendency of constituent particles in a system to dissipate or spread out (Leff, 1996). At one point, most interpretations can be brought back to the phenonomen that the modules (or in their most elementary form: particles) in a system have the natural tendency to interact (being coupled) in an uncontrolled way unless additional structure (i.e, energy or effort) is introduced in the system.

In this paper — as done previously in NS (Mannaert et al., 2012) — we will start from the statistical thermodynamics perspective towards entropy. Here, it was defined by Boltzmann in 1872 as the number of possible microstates consistent to the same macrostate of that system (Boltzmann, 1995). The *macrostate* refers to the whole of externally observable and measurable (macroscopic) properties of a system (typically temperature or pressure of a gas), whereas the *microstate* depicts the whole of microscopic properties of the constituent parts of the system (i.e., modules and particles). Generally, a particular macrostate (e.g., a certain temperature in a container) can be obtained by a myriad of different combinations of microstates (i.e., many different configurations of the molecules embedded into the container resulting in the same temperature). The higher the number of microstates consistent with that macrostate, the larger the degree of entropy becomes according to statistical thermodynamics. This relation can also be expressed in the following formula:

$$S = k_B log(W) \quad (1)$$

where S stands for the amount of entropy regarding a particular macrostate of a system, k_B equals the Boltzmann constant and W refers to the possible number of microstates consistent with the considered macrostate, given the assumption that each microstate is equally probable. According to this definition, entropy can then be seen as a measure of the information (or lack thereof) we have of the system, complying with the above mentioned interpretations of entropy as uncertainty or perceived disorder. In terms of the natural tendency of particles to interact, it can be seen as if they all "contribute" to the final resulting (observable) macrostate of the system through their mutual interactions, while it is unclear for the observer which exact configuration of particles (out of many possible configurations) brought it into being.

2.2 Entropy Applied in Business Topics

Several attempts have been made in the past to relate entropy concepts to business challenges and situations. Not claiming to be exhaustive or complete, we will illustrate some of them in this section.

For example, Trienekens, Kusters, Kriek & Siemons (2009) have elaborated the concept of entropy in the context of software development processes. First, they operationalized entropy as the amount of disorder, lack of structure and 'instability' [1] apparent in a system, being measured by the complexity (i.e., the number of interacting components) and change (i.e., the amount of changes over time) in the system. Also, they made the distinction between internal and external entropy: the former referring to the degree of ad-hoc organization inside the organization itself, the latter denoting the dynamism of its surrounding environment. As such, they conclude that both types of entropy are required to be in balance.

Janow (2004) studied the productivity and organizational decision making within firms based on Shannon's entropy approach. In doing so, he found theoretical arguments to support the finding that organizations tend to become slower in their decision making

[1] The notion of stability according to Trienekens et al. (2009) was not formally defined in their article, although the meaning of invariability (absence of change) seems to be clearly suggested. This interpretation should not be confused with the definition as proposed by Mannaert et al. (2011, 2012) and employed in the remainder of this paper, as both interpretations clearly differ.

process as well as lose productivity when they grow. By analogy with the information theory developed for communication systems, an organization is considered as a network consisting of nodes (here: human beings) taking decisions and communicating them with each other. Each organization is then proven to reach "saturation" at a certain organizational size, resulting into organizational trashing and productivity implosion.

Next, entropy was also considered as being a measure for the degree of industry concentration (Horowitz, 1970) or corporate diversification (Jacquemin & Berry, 1979; Palepu, 1985). Again, the relation to the uncertainty interpretation of entropy is made is the sense that highly concentrated industries are considered to have a higher degree of entropy as it is more difficult is such situations to predict which of the several available companies will obtain the ultimate preference of a particular consumer.

Finally, the intent of Jung (2008) and Jung, Chin & Cardoso (2011) of measuring the degree of entropy present in business process models resulting in some uncertainty measures of process models, seems to be most closely related to our approach. Starting from Shannon's entropy as defined in information theory (Shannon, 1948), their aim is to measure the uncertainty or variability of workflow process models or the information gained by the process design. For example, it is concluded that the entropy of workflows consisting of purely serialized tasks or AND-splits have zero entropy as there is no uncertainty regarding which tasks are going be executed in process instantiations. On the other hand, the inclusion of XOR-plits, OR-splits and loops in a process increase entropy as one is not aware upfront which tasks are going to be executed (or even their frequency). Each of these uncertainties can then be derived, given an assumed probability of each branch or loop iteration. However, this approach differs from the approach we will take in Section 4 and onwards. First, Jung (2008) employs the entropy definition from information theory, whereas our approach will focus on the statistical thermodynamics perspective. Next, their measures are aimed at studying the design-time structure of business processes, whereas we will use entropy to focus on the run- or execution-time analysis of business processes.

3 NORMALIZED SYSTEMS

Normalized Systems theory (NS) is about the deterministic creation of evolvable modular structures based on a limited set of proven and unambiguous design theorems, primarily aimed at the design of evolvable software architectures. First, we will discuss the essence of NS in its initial form, i.e., starting from the stability point of view from systems theory. Next, the recent association and indications towards conformance with entropy concepts from thermodynamics will be highlighted.

3.1 NS and Stability

Normalized Systems theory initially started from the well-known maintenance problems in software applications, as was for instance already articulated earlier by Manny Lehman in his "*Law of Increasing Complexity*". This law states that software programs become ever more complex and badly structured as they are changed and evolve over time and hence become more and more difficult to adapt (Lehman, 1980). Based on the systems theoretic stability, this phenomenon was related to the concept of *combinatorial effects*: a change in modular structure of which the impact or effort to implement it, is related to the size of the system (on which the change is applied to) (Mannaert et al., 2011, 2012). Indeed, given the assumption that software applications keep on growing, this means that the same type of change requires more effort as time goes by. In contrast, systems which are free of such combinatorial effects (for a defined set of anticipated changes) are called *Normalized Systems*. These systems comply with stability as defined in systems theory as a bounded impact always results in a bounded output function (effort), even if time $t \rightarrow \infty$.

For this stability to be reached, the following design theorems were proposed and formally proven to be necessary conditions regarding the avoidance of combinatorial effects (Mannaert et al., 2011, 2012):

- *Separation of Concerns:* each concern (in terms of change drivers) should be separated in its own distinct action entity;
- *Separation of States:* the calling of action entities by other action entities should be performed in a stateful way;
- *Action Version Transparency:* the updating of action entities should not not have any impact on its calling action entities;
- *Data Version Transparency:* the updating of data entities should not have any impact on the action entities receiving the data entity as input or producing it as output.

As the construction of such stable software—strictly adhering to the above described principles—is not straightforward and current software constructs

do not offer by themselves any mechanisms for software developers to obey them, a set of five elements was proposed: data elements, action elements, workflow elements, connector elements and trigger elements. As these elements offer recurring structures of constructs to facilitate the application of the previous principles, NS applications are traditionally build as an aggregation of instances of these elements.

3.2 NS and Entropy

Recently, efforts were made to explain the abovementioned in terms of entropy as defined in thermodynamics (Mannaert et al., 2012). First, the Boltzman definition in statistical thermodynamics was adopted considering entropy as the number of microstates consistent with a certain macrostate. As such, *microstates* were defined as binary values representing the correct or erroneous execution of a construct of a programming language. The *macrostate* is then to be seen in terms of loggings or database entries representing the correct or erroneous processing of the considered software system. In order to control the defined entropy, the earlier proposed theorems seem to be useful as well. Regarding the *Separation of States* principle, for instance, synchronous stateless pipeliness typically do not keep state when calling other action entities. As such, in case an error occurs, it not clear which particular action entity 'caused' the failure. In terms of the *Separation of Concerns* principle, each concern should again be isolated in its specific construct to avoid the creation of multiple microstates for one macrostates. This time however, concerns should be identified based on so-called uncertainty drivers instead of change drivers. The other two remaining principles, Data Version Transparency and Action Version Transparency, seem less applicable as they are related to compile-time and not for run-time analysis. However, two new theorems were suggested from this viewpoint:

- *Data Instance Traceability*, requiring each version and values of an instance of a data structure to be tracked;
- *Action Instance Traceability*, requiring each version and thread of an instance of a processing structure to be tracked.

Indeed, not exporting this information to an observable macrostate would lead to multiple possible microstates consistent with the same macrostate.

4 USING ENTROPY FOR BUSINESS PROCESS ANALYSIS

In order to extend the concept of entropy to business process analysis, we will first propose a definition of entropy, microstates and macrostates in a business process context. Next, we will illustrate the existence of such entropy by means of an example and discuss how NS principles can be helpful in reducing the amount of entropy in business process systems.

4.1 Defining Entropy Concepts in Business Process Systems

Our purpose is to apply the concept of entropy as defined in statistical thermodynamics (i.e., the number of microstates consistent with the same macrostate) to business processes. Consequently, a first effort should be directed towards interpreting macro- and microstates in such context. While the stability viewpoint (cf. Section 3) analyzes modular structures at design time, an entropy based analysis tends to investigate the modular structures during or after their execution (i.e., run time). Hence, regarding the *macrostate* (i.e., the whole of macroscopic properties of a system), typical externally observable properties of a business process might include:

- *throughput or cycle time* (how long did the process take to be executed?);
- *quality* and other output related measures (e.g., succesful or non-succesful completion of the process as a whole or the number of defects detected after the execution of the process);
- *costs* involved in the process;
- other *resources* consumed by the process (such as raw materials, electricity, human resources, etcetera).

Typical *microstates* (i.e., the whole of microscopic properties of a constituent modules or particles of the system) related to the above sketched macrostate might then comprise the throughput time of a single task in the process, the correct or erroneous outcome of a single task, the costs related to one activity or the resources consumed by one particular task of the considered business process. Analyzing instantiated business processes in terms of these defined macro- and microstates would then come down to management questions such as:

- which task or tasks in the business process was (were) reponsible for the extremely slow (fast) completion of this particular instance of the business process? ;

- which task or tasks in the business process was (were) responsible for the failure of the considered instantiated business process? ;
- which activities contributed substantially or only marginally to the overall cost or resource consumption of the considered business process (cf. cost-accounting and management approaches like Activity Based Costing)?

In case the answer to these questions is unambiguous and clear, the entropy in the system (here: business process) is low (or ideally zero) as a particular macrostate (e.g., the extremely long throughput time) can be related to only one or a few microstates (e.g., activity X took three times the normal duration to be carried out, whereas all other activities finished in their regular time span). On the other hand, when no direct answer to these questions can be found, entropy increases: multiple microstates (e.g., prolonged execution of activities X and/or Y and/or Z) could have resulted in the observed and possibly problematic macrostate (e.g., the lengthy execution of the overall process). This phenomenon seems to correlate well with the three basic entropy interpretations we listed in Section 2.1. First, business process analysis is more *complex* as the analyst has to consider the whole system at once, not being able to refine his problem analysis to certain clearly isolated parts of the system. Second, *uncertainty* is present when trying to perform remedial measures or optimizations. Indeed, it is simply not known where possible problems are situated, and the outcome or success of specific adaptations in the business process repository is uncertain as well. Finally, the *tendency of particles to dissipate* is reflected in the fact that the "traces" of one problematic activity are dispersed over the considered system as a whole. In essence, unless a consciously introduced separation is introduced, the information and outcome of all three possible problem causing activities (X, Y and Z) *interacts* before being exposed to the observer (in this case the measurements are aggregated). Hence, from an (enterprise) engineering viewpoint, it would seem appealing to control and reduce the entropy confronted with.

4.2 Illustrating Entropy in Business Process Models

In order to illustrate our conceptualization of the manifestation of entropy in business processes, consider the following example as depicted in the BPMN notation of Figures 1(a) and 1(b). Let us assume that both processes represent a part of a typical assembly line in which automobiles are finalized during their manufacturing. More specifically, we will claim that both process parts represent the manual attachment of the weels (X) and doors to the car (Y), as well as the spraying of a final liquid for the preservation of the color of the car's skeleton or "body" (Z). In Figure 1(a) all these activities are considered as one "finalization" activity P and only the throughput time for this entity of work is registered. A possible frequency graph of the throughput time of this way of modeling is represented in Figure 2(a) by means of a typical normal curve. Further, the sample mean $\overline{T_p}$ is given by the solid curve in this figure, while the "target mean"[2] set by the production manager is drawn by a dashed line. In case statistical hypothesis testing would point out that the observed mean actually significantly differs from the target mean, the production manager has no direct clue to determine which action is accountable for the prolonged production throughput time: both the attachment of wheels, doors or the preservation spray as well as any combination among them might be causing the delay. In entropy terms, the macrostate in this situation is the throughput time of process part P as a whole. The microstate is the combination of the throughput times of each of the constituting activities X, Y and Z. Consequently, the modeling as in Figure 1(a) exhibits a certain amount of entropy as it is not clear which activity causes the prolonged overall throughput time. We can call this confusion about the origin of a macrostate of a business process in terms of its microstate a *business process uncertainty effect*. Due to the inherent uncertainty and doubt in this situation, an imperative and profound study of the whole "finalization" activity should be performed in order to retrieve the cause and solve the problem identified.

On the other hand, Figure 1(b) makes the obvious break up of P into the constituting tasks X (placement of the wheels), Y (placement of the doors) and Z (preservation spray) and their throughput measurements and means $\overline{T_X}$, $\overline{T_Y}$ and $\overline{T_Z}$. The corresponding normal curves of the frequency graphs are depicted in Figure 2(b). In case of the same problem of an enlarged overall average throughput time $\overline{T_p}$, the production manager is now able to refine his analysis towards the individual throughput times of X, Y and Z. Again, the real observed sample means are depicted in solid curves, the target means by the dashed line.

[2] In realistic production environments, typical upper control limits (UCL) and lower control limits (LCL) might be employed to determine significant deviations from the predefined goals. However, as this extra complexity does not add any further insight to our conceptual example (nor does it take the edge off our argument), we will leave this feature out of scope in this paper as its extention is straightforward.

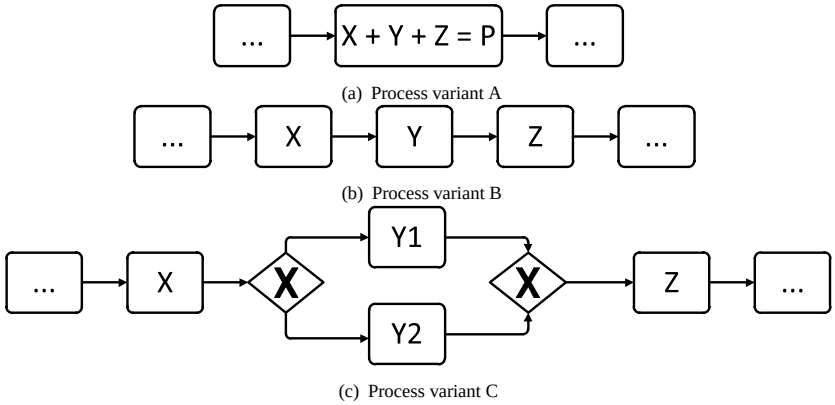


Figure 1: Three business process variants in BPMN, illustrating different degrees of entropy.

For tasks X and Z, the dashed lines are not visible as they coincide with the solide lines. Consequently, no issues regarding a prolonged execution time of these steps seem to be at hand. Regarding task Y however, a larger standard deviation and significant difference between the target mean and observed mean can be found. In order to improve the observed overall average throughput time $\overline{T_p}$, improvements regarding this specific business process task should be aimed for. Hence, entropy (and uncertainty) can be said to be reduced as the production manager can clearly see that the extended throughput time (i.e., the same macrostate as in our previous example) is caused by only one particular task, the other activities having regular throughput times (i.e., the macrostate is consistent with only one microstate configuration).

4.3 Illustrating the Need for Principles to Control Business Process Entropy

In the previous subsection, we illustrated how entropy generation and uncertainty effects can occur in business process models. Now we will illustrate how NS principles facilitate the control and reduction of entropy in the design of business processes.

Starting with the *Separation of States* principle, this theorem would call for the stateful executions of business processes. This would entail to include and keep a unique state after the execution of each business process activity. First, in terms of our previously discussed examples as visually represented in Figures 1(a) and 1(b), this implies that after each step a state should be pertained, registering the successful or unsuccessful completion of the step as well as the relevant observed system properties as costs, throughput time, resource consumption, etcetera. Not including these measure points or 'mashing them up' into one activity as in Figure 1(a) would after all lead to the uncertainty effect as described above. Second, these states need to be uniquely defined after each activity for maintaining transactional integrity and keeping record of exactly which activities have been executed. For example, consider the case of a business proces performing an application procedure for the entrance of potential future university students. The process contains multiple checks each resulting in a 'positive' state (after which the process can continue) or 'negative' state (in which the applicant is refused and the process terminated). Here, a unique state should be defined for each of the refusal situations in order to keep record of the precise reason why one person has been rejected. In case states are not uniquely defined (e.g., each negative outcome receives the same state 'refused') one will not be able to trace the obtained macrostate (i.e., a person has been refused) to the correct microstate configuration (i.e., which reason—which check—the person has been rejected for).

While the previous principle forces the usage of states in order to isolate activities in the system of which the macrostate is studied, the *Separation of Concerns* principle discusses the nature of the 'particles' that have to be separated by the states or measurement points. Stated otherwise, the division of the system into its constituent parts (and hence microstates) should not be done in an arbitrary way. Rather, concerns should be identified based on so-called *uncertainty drivers*: each separate part of the system of which the information should remain traceable for analysis purposes has to be isolated in its own construct (e.g., task or process). This implies amongst others that each task in a business process can contain only one single non-redundant concern. Suppose that in Figure 1(b) both tasks X and Y are combined with an electronic circuit general test inspection activity Q (or another registration, measurement or quality assessment activity) as both the wheel and door at-

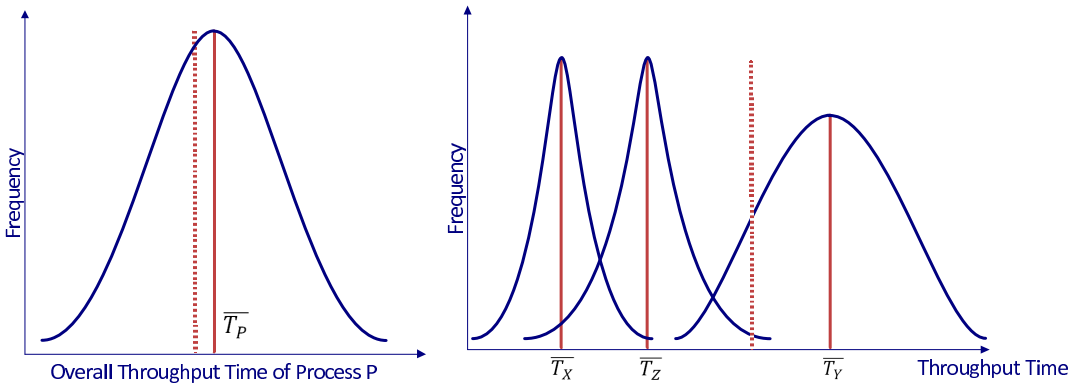


Figure 2: Corresponding frequency graphs of the throughput time, based on the various ways of modeling.

tachment are supposed to have a possible influence on the proper functioning of the electronic circuit of the car. In case Q is not properly isolated in a distinct task and not resulting in its own unique states, problem analysis of the overall troughput time pointing to X and/or Y is no longer unambiguous. First, the block $X+Q$ or $Y+Q$ should be further scrutinized to determine whether X, Y or Q was causing the throughput extension. Second, if investigation reveals a problem in the electronic circuit test Q, the problematic observation is escalated into several intermediate states and remediating actions should be taken at multiple process steps as well. As again multiple microstates can be associated with the same macrostate, entropy and uncertainty increase in such situations. Moreover, recurring task sequences with a clear business meaning of which information (e.g., its progress) is required to be recorded, should be isolated in their separate construct (here: business process) as well according to the same principle. Imagine again the application procedure in which the applicants have to pay an administration fee in order to be able to be assessed by the procedure and a registration fee later on if they have succesfully passed the whole procedure and decide to enroll. In case this redundant payment procedure (most likely repeated in many other business processes as well) is not properly separated, a bottleneck in this fixed task sequence might be much more difficult to be noticed. Consistent problems in the payment procedure would show up in many states (among many different processes) but hide its relation to the payment concern. Again, problems in such application procedure (macrostate) might be related to many interwoven subsequences (and hence microstates), resulting in a higher degree of entropy.

The principle of *Action Instance Traceability* would force us to keep track of the specific version of a task which is executed and relate every state (and measurement) to this specific task version. Also in a business process context, it is not unlikely to imagine situations in which a certain version of a task replaces a predecing task or multiple versions (variants) of one task exist concurrently. Consider for instance the situation in which our assembly line assembles both cars with two (Y_1) and four (Y_2) doors respectively. Elaborating on our investigation of througput time, it might seem reasonable to assume that (given the same technical equipment and resource availability) the attachment of doors for a two-door or four-door car variant might differ significantly. Consequently, the specific version should be recognized and traced individually as depicted in Figure 1(c) in order to allow further business process analysis and unambiguously relate particular macrostates to the correct microstate configuration (i.e., the specific versions) and avoid entropy generation.

Finally, *Data Instance Traceability* in a business process setting would prescribe us to keep track of the specificities of the information object processed by the business process in question. Applying the concept to our car assembly example, this means that characteristics of the car being assembled on a specific time slot should be tracked. Indeed, one can imagine that specific difficulties reflected in the states (e.g., extra costs, resource consumption, throughput time) can arise depending on the type (model) of car assembled on the same assembly line. Not tracking these specificities results in multiple possible ‘causes’ (microstates) consistent with the same ‘problem’ or ’fact’ (macrostate). Hence, registration of the relevant particularities of each processed information object can be considered as another ‘rule’ to control entropy.

5 REFLECTIONS

The concepts and principles discussed above might have several consequences for the design and analysis of business processes. We will first discuss some theoretical implications, followed by a few practical implications.

5.1 Theoretical Implications

Regarding the theoretical implications of employing an entropy viewpoint for analyzing business processes, several issues can be noticed. First, as was already mentioned in our analysis at the software level (Mannaert et al., 2012), the proposed analysis method includes the assumption that the introduced states are independent and decoupled. This means that they should only reflect the outcome of the activities performed in the module (here: task in a business process) they are attached to. Stated otherwise, the resulting state should not be dependent on the activities which have taken place earlier (e.g., in the business process) and with which the studied activity is coupled in a hidden (i.e., not explicitated) way. In a business interpretation, this could for instance be the case when the throughput time of task *B* is dependent on decisions taken earlier in task *A*. For example, it might be realistic to imagine a situation in which the employees responsible for the execution of task *A* choose to quickly (but poorly) finalize their task (in order to minimize their own throughput time), but having a pernicious consequence on the attainable throughput time of the execution of task *B* (for which the employees might then be forced to invest some extra time caused by the low-quality output of their predecessors). This obviously only leads to local optimizations, while preventing global optimization. Entropy reduction by imposing such a "structure" is limited and even misleading as the states here do not reflect the outcome of an isolated subsystem of the regarded overall system. Indeed, from an analysis viewpoint, it is no longer clear how the different subsystems (hence, microstate configuration) brought about the resulting macrostate. Hence, a clear interface between both activities should be defined, including (for example) unambiguous quality conditions as output criteria for activity *A* and preconditions for the execution of activity *B*. With the aim of preventing such phenomena, this assumption adds to our earlier call for the definition of completely and unambiguously defined interfaces for (organizational) modules in the first place (De Bruyn & Mannaert, 2012). Next, efforts should be directed towards avoiding such coupled modules.

Second, the initial application of the NS (entropy related) principles at the business process level demonstrated above, proved to be rather similar and parallel to the software level, and not to contradict with the guidelines by Van Nuffel based on the stability point of view (Van Nuffel, 2011). However, the work of Van Nuffel primarily concentrated on identifying business process instantiations of concerns to be separated in terms of *change drivers* based on the stability rationale of NS . Hence, it would be interesting to perform a similar study with the aim of identifying typical business process instantiations of concerns in terms of *uncertainty drivers* based on the entropy rationale of NS. This could lead to a parallel set of practical recommendations on how to design the modular structure of business processes and their constituting tasks. These guidelines would specifically allow for maximum entropy control and facilitate unambiguous analyses and tracking of outcomes obtained during execution time.

Finally, in the NS rationale at software level, the formulation of the principles (theorems) resulted in a limited set of elements or patterns (recurring structures of constructs) which are instantiated and aggregated consistently to build a software application. At the organizational level, such patterns (fixed structures of business processes) would be appealing as well when exhibiting both stability (i.e., proven absence of combinatorial effects towards a defined set of anticipated changes) and controlled entropy (i.e., ex-ante known measurements or metrics to allow for ex-post process analysis and optimization). The construction of such organizational patterns would facilitate pattern instantiation which is both stable and isentropic (i.e., having observable macrostates of the overall system which can be unambiguously traced to a microstate configuration). The formulation of such patterns is obviously a very challenging effort, and subject to future research. However, at the end of the following subsection, we will give an illustration of the usage of a recurring fixed pattern in reality for the execution of certain business functionalities, leading to entropy reduction.

5.2 Practical Implications

The reasoning in this paper holds irrespective of the implementation method. Therefore, concrete insights for supporting platforms for business processes such as Service-Oriented Architectures (SOA) can be made. In a SOA context, entropy is introduced in a business process when information regarding the micro-states of the service execution cannot be captured. However, the service concept specifically aims

to hide the implementation of its functionality behind its interface. Consequently, micro-states which provide necessary knowledge during a service invocation cannot be captured. Consider the following definition of a service from an often-cited author: "*Services are self-describing, platform-agnostic computational elements that support rapid, low-cost composition of distributed applications. Services perform functions, which can be anything from simple requests to complicated business processes*" (Papazoglou, 2003, p. 3). This definition explicitly mentions the use of different distributed applications. In the context of the throughput example, the usage of such a service can result in delays caused by any application (e.g., because of the program logic, its hardware or its network connection). Consequently, the distributed nature of a service by itself increases the entropy when analyzing the business process.

Moreover, the design of services can introduce entropy as well. The definition mentions the service granularity. It seems that both fine-grained (i.e., "simple requests") and coarse-grained services (i.e., "complicated business processes") are considered valid by the definition. Nevertheless, our analysis indicates that information concerning fine-grained modular building blocks is required to lower the business process entropy. However, various authors discuss how currently a trend towards more coarse-grained services can be observed (e.g., Feuerlicht, 2006). These so-called "enterprise services" attempt to minimize the number of interactions needed to implement a given business function in order to reduce the complexity of the message interchange dialog (Feuerlicht, 2006). This example illustrates how considerations from a technical point of view (e.g., lowering the complexity of the message interchange dialog) may conflict with considerations from a business point of view (e.g., increasing the entropy during business process analysis). Dealing with such conflicts is an important issue for a paradigm which positions itself as the integration for business and IT perspectives. Nevertheless, the service definition by itself does not seem to provide guidance for entropy reduction. This task remains the responsibility of the service designer.

While no silver bullet is currently available to determine the "right" service granularity, certain interesting domain-specific solutions are emerging. These solutions apply the approach to controlling entropy as presented in Section 2 (i.e., enforcing a certain recurring *structure*) in order to ensure the availability of required knowledge. Consider the example of manufacturing organizations who are operating in a global supply chain. Governments of the national and international (e.g., the European Union) level define norms and regulations on these supply chains. The monitoring of these norms and regulations is executed by customs or quality inspections. In order to be able to perform these controls, certain information is required. Because of the heteroginity of processes used in these supply chains, any instance of these processes (e.g., a shipment) needed to be checked individually in order to ensure compliance. Consequently, these controls performed by the customs were very labor- and time-intensive, and organizations considered them to be a "necessary evil". The common goal of governments and organizations should be to ensure compliance to regulations, while having a minimal impact on the planning and execution of supply chains. Based on the introduction of structure in the organizational processes, results have already been achieved towards this goal. For example, member states of the European Union can grant the certificate of Authorized Economic Operator (AEO)[3] to organizations which follow, amongst others, customs compliance and appropriate record-keeping criteria. AEOs can benefit from simplified control procedures (e.g., fewer physical and document-based controls) during customs or quality controls. An AEO commits himself to structure his processes according to certain landmarks, which resemble the process states as discussed in the Separation of States theorem in Section 4.3. Governments which publish such landmarks can ensure that the information required for their controls can easily be gathered based on the registration of these process states. As a result, AOEs cannot use any SOA service with a granularity which spans multiple landmarks. Consequently, these landmarks effectively guide the selection of service granularity by imposing an appropriate structure on the processes, which is shown to be required to control the entropy in them.

6 CONCLUSIONS

In this paper, we explored the usage of the entropy concept for business process analysis. As such, the paper has multiple contributions, while suggesting several avenues for future research. First, we proposed a specific way for analyzing business processes from the typical engineering concept of entropy as defined in statistical thermodynamics. This approach seems to be contrasting with many ad-hoc or qualitative best practice approaches suggested in extant literature. By means of some pedagogical and conceptual examples, we showed that entropy has the tendency to show up in business processes which are arbitrarily

[3]See: http://ec.europa.eu/taxation_customs/customs/policy_issues/customs_security/aeo/

conceived, making it difficult to analyze them ex-post (e.g., in terms of quality, costs, resource consumption or throughput time). Consequently, in order to optimize and control such business processes, we argue that they should be purposefully *engineered* with the aim of controlling the entropy. Second, we proposed some initial principles for entropy control in business process systems (in analogy with previously defined principles at the software level). Third, our reflections demonstrated some of the implications of this entropy based reasoning for both theoretical and practical purposes, such as the design and usage of Service-Oriented Architectures. Obviously, the proposed principles might be further confined later on and business process instantiations of concerns from the entropy viewpoint (i.e., uncertainty drivers) could constitute an interesting path for future research, as would be the construction of organizational elements or patterns incorporating these issues.

ACKNOWLEDGEMENTS

P.D.B. is supported by a Research Grant of the Agency for Innovation by Science and Technology in Flanders (IWT).

REFERENCES

Anderson, G. (2005). *Thermodynamics of Natural Systems*. Cambridge University Press.

Boltzmann, L. (1995). *Lectures on Gas Theory*. Dover Publications.

De Bruyn, P. & Mannaert, H. (2012). Towards applying normalized systems concepts to modularity and the systems engineering process. In *Proceedings of the Seventh International Conference on Systems (ICONS)*.

Feuerlicht, G. (2006). Service granularity considerations based on data properties of interface parameters. *International Journal of Computer Systems science and Engineering*, *21*(4), 315–327.

Hammer, M. (1990). Reengineering work: Don't automate, obliterate. *Harvard Business Review*, *68*(4), 104 – 112.

Horowitz, I. (1970). Employment concentration in the common market: An entropy approach. *Journal of the Royal Statistical Society. Series A (General)*, *133*(3), 463–479.

Huysmans, P. (2011). *On the Feasibility of Normalized Enterprises: Applying Normalized Systems Theory to the High-Level Design of Enterprises*. PhD thesis, University of Antwerp.

Jacquemin, A. P. & Berry, C. H. (1979). Entropy measure of diversification and corporate growth. *The Journal of Industrial Economics*, *27*(4), 359–369.

Janow, R. (2004). Shannon entropy and productivity: Why big organizations can seem stupid. *Journal of the Washington Academy of Sciences*, *90*.

Jung, J.-Y. (2008). Measuring entropy in business process models. In *3rd International Conference on Innovative Computing Information and Control, 2008 (ICICIC '08)*.

Jung, J.-Y., Chin, C.-H., & Cardoso, J. (2011). An entropy-based uncertainty measure of process models. *Information Processing Letters*, *111*(3), 135 – 141.

Kettinger, W., Guha, S., & J.T.C., T. (1995). *Business process change: reengineering concepts, methods, and technologies*, chapter The Process Reengineering Life Cycle Methodology: A Case Study, (pp. 211–244). Idea Group Inc (IGI).

Leff, H. (1996). Thermodynamic entropy: The spreading and sharing of energy. *American Journal of Physics*, *64*(10), 1261–1271.

Lehman, M. (1980). Programs, life cycles, and laws of software evolution. *Proceedings of the IEEE*, *68*(9), 1060 – 1076.

Mannaert, H., De Bruyn, P., & Verelst, J. (2012). Exploring entropy in software systems: Towards a precise definition and design rules. In *Proceedings of the Seventh International Conference on Systems (ICONS)*.

Mannaert, H., Verelst, J., & Ven, K. (2011). The transformation of requirements into software primitives: Studying evolvability based on systems theoretic stability. *Science of Computer Programming*, *76*(12), 1210–1222. Special Issue on Software Evolution, Adaptability and Variability.

Mannaert, H., Verelst, J., & Ven, K. (2012). Towards evolvable software architectures based on systems theoretic stability. *Software: Practice and Experience*, *42*(1), 89–116.

O'Neill, P. & Sohal, A. S. (1999). Business process reengineering a review of recent literature. *Technovation*, *19*(9), 571–581.

Palepu, K. (1985). Diversification strategy, profit performance and the entropy measure. *Strategic Management Journal*, *6*(3), 239–255.

Papazoglou, M. P. (2003). Service-oriented computing: Concepts, characteristics and directions. In *Proceedings of the Fourth International Conference on Web Information Systems Engineering (WISE 2003)*, (pp. 3–12)., Washington, DC, USA. IEEE Computer Society.

Shannon, C. (1948). A mathemathical theory of communication. *Bell System Technical Journal*, *28*, 379–423.

Trienekens, J., Kusters, R., Kriek, D., & Siemons, P. (2009). Entropy based software processes improvement. *Software Quality Journal*, *17*, 231–243. 10.1007/s11219-008-9063-6.

Van Nuffel, D. (2011). *Towards Designing Modular and Evolvable Business Processes*. PhD thesis, University of Antwerp.

On Advancing the Field of Organizational Diagnosis based on Insights from Entropy

Motivating the Need for Constructional Models

Gilles Oorts, Philip Huysmans and Peter De Bruyn
University of Antwerp, Prinsstraat 13, 2000 Antwerp, Belgium
{*gilles.oorts, philip.huysmans, peter.debruyn*}*@ua.ac.be*

Keywords: Entropy, Organizational Diagnosis, Enterprise Engineering.

Abstract: In this paper, we explore how the field of organizational diagnosis can benefit from lessons learned from entropy reduction in other fields. In an organizational context, entropy is related to the lack of knowledge concerning the way of how management-level KPIs (observable system macrostate) are brought about by operational elements (which are considered to be the causing microstate). Because of this lack of knowledge, the goal and scope of projects to remedy problematic KPIs cannot be determined unambiguously. Organizational diagnosis aims to further the insight in these decisions by providing conceptual models to find causal explanations between observations and their causes. In related fields, reduction of entropy is achieved by introducing and analyzing structure in a system, which is described in a constructional perspective. However, we will show in this paper that many diagnostic approaches do not support this constructional perspective adequately.

1 INTRODUCTION

Contemporary markets are characterized by volatility, both on the demand side as a result of changing customer preferences, as on the supply side as a result of mergers and takeovers. Therefore, the competitive environment of organizations is changing at a rapid pace. Consequently, organizations need to be able to react quickly to observed business performance issues in order to satisfy customer expectations. In order to be able to detect these issues, management often defines key performance indicators (KPIs) which capture relevant scores on various criteria. When a problematic KPI is observed, projects can be initiated to remedy the issue, and adapt the organization to its changing environment. However, the influencing factors of KPIs are often diverse and complex. As a result, it is not straightforward to define the concrete scope of such projects to achieve effective improvement of problematic KPI values. Various authors argue that it is indeed naive to expect that simple measures can provide insight in organizations which are complex and variable (Sitkin et al., 1994).

The field of organizational diagnosis attempts to provide conceptual models to find causal explanations of unwanted observations (Harrison, 1994). Such unwanted observations can be indicated by problematic KPIs. Without adequate understanding of the root causes of a problem, decision makers cannot efficiently remedy that problem (Senge, 1990). Consequently, the field of organizational diagnosis is very relevant in this context. However, it is still faced with significant challenges. First, the inherent complexity of organizations makes the diagnosing activity extremely challenging. Various authors suggest that the search for cause and effect relations in an operational organization is very difficult (Harrison, 1994; Harry, 1988). Second, organizational diagnosis depends largely on heuristics. One can expect a different diagnosis from a novice or an experienced diagnostician. In order to better teach or develop methods for performing organizational diagnosis, a more systematic approach is required.

The lack of a systematic approach and the difficulty of handling complexity indicate the need for a clear theoretical basis to approach these issues. A theoretical basis clarifies the concepts which are needed to explain how complexity can be dealt with, and allows to introduce prescriptive elements in a diagnosis approach. While the selection of a certain theoretical basis invariantly results in a focus on certain dimensions, and neglects others, we believe that a relevant theoretical basis can make significant contributions to an immature field. Therefore, we explore the use of the theoretical concept of entropy as defined in the field of thermodynamics in order to gain more insight

in the field of organizational diagnosis. Entropy has already been applied in a wide variety of fields. Insight in dealing with entropy has already matured in those fields. Therefore, the field of organizational diagnosis can progress based on lessons learned from these fields. According to some research methodologies, such as design science, the application of proven solutions in new research fields is the way towards scientific progress (Hevner and Chatterjee, 2010).

This paper is structured as follows. First, we introduce the field of organizational diagnosis in Section 2. We then explore the entropy concept and the reduction of entropy in Section 3. In Section 4, we apply the concept of entropy on the field of organizational diagnosis. Finally, we summarize our conclusions and the contribution of this paper in Section 5.

2 ORGANIZATIONAL DIAGNOSIS

In organizational diagnosis consultants, managers or researchers use conceptual models to find causal explanations of observed and unwanted effects (Alderfer, 2010). A diagnostician works beyond an observational role since he attempts to explain why certain issues occur. He formulates questions (e.g., why are five percent of the produced products defective?) and aims to formulate adequate answers. When an enterprise diagnostician understands a problematic situation, a hypothesis can be formulated to explain how an observed issue can originate. Then, evidence needs to be gathered to confirm or falsify this hypothesis. Based on evidence, the hypothesis can be rejected or refined through an iterative process (Alderfer, 2010).

A popular approach used for diagnosing is Lean Six Sigma (LSS). LSS applies a specific analytic thinking pattern to support the problem solving performed by the diagnostician (de Mast and Bisgaard, 2007). According to this pattern, the analytic mind oscillates between on the one hand the theories, hypotheses, conjectures, ideas one has in mind (i.e., the interpretative world) and on the other hand the observations, measurements, experimental results empirically retrieved from the real world (i.e., the factual world) (Box and Liu, 1999). The pattern is graphically represented in Figure 1. The oscillation in this pattern can start from any of both worlds. It could for example start with an hypothesis one has in mind and the gathering of facts to justify it. These discovered facts might influence the hypothesis, which then again needs to be justified with new facts. However, the process can also start by observing facts from which a hypothesis is built, which is then justified by new facts

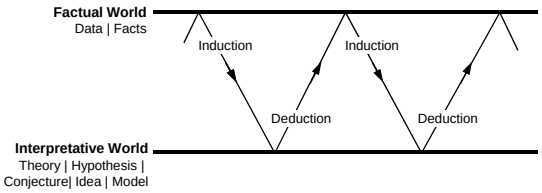


Figure 1: Learning by iteration between data and models.

until a satisfactory hypothesis has been formulated. This is called sawtooth thinking, i.e., the repeated alternation of discovery and justification in which we develop causal explanations (de Mast and Bisgaard, 2007).

Based on this sawtooth-thinking pattern, a wide variety of causal explanations, formulated as hypotheses, can be gathered. In LSS, a so-called *logic filter* is used to select the most important causal explanations when faced with a business performance problem (Harry, 1988). The application of the logic filter is organized in iterative optimization cycles. Each cycle uses a specific collection of tools and techniques to guide an applicant to the vital key correlations between influence variables and business performance outcome variables. However, no theoretical basis is provided to select or evaluate these tools and techniques. We do not claim that this indicates that these tools and techniques are lacking or insufficient. Instead, we believe that a theoretical evaluation can indicate more improvements in a structured way. Therefore, the goal of this paper is to assess whether these tools and techniques can be improved based on insights from the theoretical concept of entropy.

3 THEORETICAL FRAMEWORK: ENTROPY

In this section, we introduce entropy as a theoretical basis to interpret the current diagnosis approaches such as LSS and to analyze how they can be improved. In Section 3.1, we introduce the entropy concept and its definition. In Section 3.2, we explore how entropy is controlled in different fields. Based on this insight, we will be able to formulate improvements for organizational diagnosis approaches.

3.1 Defining Entropy

Entropy as expressed in the second law of thermodynamics is considered to be a fundamental principle. There are many versions of this law, but they all have the same intent. Mathematical derivations

of the entropy principle start in general from a formula describing the number of possible combinations. In statistical thermodynamics, entropy was defined by Boltzmann in 1872 as the number of possible microstates corresponding to the same macrostate (Boltzmann, 1995). The aim is to understand and to interpret the externally observable and measurable macroscopic properties of materials — the macrostate — in terms of the properties of the constituent parts — the microstate — and the interactions between these parts. In Boltzmann's definition, entropy is a measure of the number of possible microstates of a system, consistent with its macrostate. Mathematically, the entropy of a particular macrostate (S) is equal to the Boltzmann constant (k_B) times the natural logarithm of the number of microstates corresponding to that macrostate ($ln\Omega$).

$$S = k_B ln\Omega \quad (1)$$

In thermodynamics, examples of properties related to such a macrostate are the temperature, pressure, or volume of gas in a containment. The studied gas containment consists of a collection of molecules. The observed values of this macrostate are brought about by a certain arrangement of these molecules. However, many different arrangements could result in a certain macrostate: therefore, one cannot be sure of the exact arrangement of molecules represented by a single macrostate. The number of arrangements which can correspond to a single macrostate is the number of microstates referred to in the formula above. This notion of entropy can be seen as a measure of our lack of knowledge about a system.

This definition of entropy can be further clarified by the example of a set of 100 coins, each of which is either heads up or tails up. The macrostate is specified by the total number of heads and tails, whereas the microstate is specified by the possible configuration of the facings of each individual coin. For the macrostate of 100 heads or 100 tails, there is exactly one possible configuration, so our knowledge about the system is complete. At the opposite extreme, the macrostate which gives us the least knowledge about the system consists of 50 heads and 50 tails in any order, for which there are 10^{92} possible microstates (Wikipedia, 2011a). It is clear that the entropy is extremely large in the latter case because we have no knowledge of the internals of the system.

3.2 On Controlling Entropy

A common way of dealing with entropy, is to increase the *structure* or the knowledge of the internals of the system. Consider the coin example. The entropy in this example can be reduced when we add structure to the studied system. Suppose we would have 10 groups of 10 coins, each with 5 heads and 5 tails, the number of possible microstates would only be 2520 (Wikipedia, 2011a). Consequently, the entropy for this system would be much lower. Structure can be used to control entropy, in the sense that by allowing less interaction between the constituing components, a lower number of valid combinations are possible. This leads to less uncertainty concerning the actual microstate configuration.

In complex systems, one has to consider that structure needs to be applied to the constituent parts of the system, not on the macrostate measurement. In the example of the gas container, it is clear that it would not make sense to make more detailed temperature measurements. This would be an example of a more precise macrostate measurement. Instead, the lacking knowledge refers to the characteristics of the individual molecules. Consequently, it is important to be able to distinguish between the nature of the macrostate and microstate. In systems theory, such a distinction is made between the functional and constructional perspective of a system (Weinberg, 1975; Gero and Kannengiesser, 2004). The constructional perspective describes the composition of a system. In a constructional perspective, the different subsystems of which a system consists and their relations (i.e., how do the different subsystems cooperate) are described. In contrast, the functional perspective describes what a system does or what its function is, i.e., how it is perceived by its environment. In a functional perspective, the input variables (i.e., what does the system need in order to perform its functionality?), transfer functions (i.e., what does the system do with its input?) and output variables (i.e., what does the system deliver after performing its functionality?) are described. In order to reduce entropy, the structure of a system needs to be studied from a *constructional perspective*.

Models from a functional or constructional perspective are different in nature (Dietz, 2006). Models created from a functional perspective are called black-box models. These models only depict the input and output parameters by means of an interface, describing the way how the system interacts with its environment. Consequently, the user of the system does not need to know any details about the inner workings of the system. Put differently, the complexity of the system internals is hidden in these models. Models created from a constructional perspective are called white-box models. These models depict the different components of which the system consists, and the way these components work together. Each of

these components can be considered to be a subsystem. Consequently, each component can be regarded as a system on its own and can therefore be described using a functional (i.e., black-box) or constructional (i.e., white-box) model. However, this alternation between black-box and white-box models should be clearly distinguished from merely adding detail to existing models. As argued by Dietz, additional detail within a single perspective can be added by performing functional or constructional decomposition (Dietz, 2006). However, functional decomposition, which elaborates on a certain model from a functional perspective, cannot be used to obtain a constructional model. As discussed, it is in the constructional perspective that the structure of a system is described. Consequently, reducing entropy requires a constructional perspective. As a result, a functional decomposition cannot be used to reduce entropy, since the required structure cannot be applied in this perspective.

4 APPLYING INSIGHTS FROM ENTROPY ON ORGANIZATIONAL DIAGNOSIS

The concept of entropy already received attention in management literature. Various authors applied it to the organizational level:

- First, entropy is considered to be a measure for waste in organizational processes. Originally, entropy was a term to describe the loss of useful energy of mechanic devices such as heat engines when converting energy to work. Several authors argue that waste in organizational processes can be described similarly (Katz and Kahn, 1978).
- Second, entropy is used as a measure of uncertainty with regard to a random variable in information theory (Shannon, 1948). The so-called Shannon entropy quantifies the expected value of a specific instance of the random variable. For example, a coin toss of a fair coin (i.e., a coin toss which has an exact 50% chance of resulting in head) has an entropy of 1 bit (Wikipedia, 2011b). If the coin is not fair, the entropy will be lower since one can expect a certain value to occur more. In other words, the uncertainty of the outcome has been reduced. The entropy of a coin toss with a double-sided coin is zero.
- Third, entropy has been proposed as a measure of industry concentration (Horowitz, 1970) and corporate diversification (Jacquemin and Berry, 1979; Palepu, 1985). In a concentrated industry, entropy is considered to be low. The higher the entropy, the greater the uncertainteny will be with which one can predict which firm will gain the preference of a random buyer.
- Fourth, Janow has studied organizations and productivity based on entropy (Janow, 2004). Janow concluded that entropy offered an interesting means to explain why organizations tend to become gradually more slow in their decisionmaking processes, as well as lose productivity over time.

While the interpretation of entropy in the first type is related to waste, the interpretation of entropy in the second, third and fourth types are related to uncertainty. We will follow the latter interpretation of entropy. For our purpose, entropy can be interpreted as a measure of the number of microstates consistent with a given macrostate. In an organization, a KPI can be considered to be such a macrostate. However, when the influencing factors of this KPI are not known, many different microstates can be relevant for the values of this macrostate. Consequently, the entropy is considered to be high.

By itself, the interpretation of KPIs as a macrostate with high entropy does not contribute much to the field of organizational diagnosis. However, we can now analyze how other fields achieve a reduction of entropy, and compare their approach to the current practice of organizational diagnosis. In Section 3.2, we argued that a constructional perspective is required to reduce entropy. Organizational measurements such as KPIs are defined in relation to the behaviour of the organization in its environment, and are therefore mostly described from a functional perspective.

When insight is needed in problematic KPIs, most approaches only propose to use functional decomposition. Consider an analysis of the return on equity (RoE) according to the strategic profit model. The RoE is defined as the net income divided by the average stockholder assets. The strategic profit model proposes the DuPont formula, which breaks the RoE down into operation efficiency (Net Income divided by Sales), asset use efficiency (Sales divided by Total Assets), and financial leverage (Total Assets divided by Average Stockholder Assets).

$$RoE = \frac{NetInc.}{Sales} * \frac{Sales}{TotalAss.} * \frac{TotalAssets}{Avg.Stckh.Ass.}$$

However, such a decomposition does not coincide with the constructional model of an organization.

Such a model will likely exist of, amongst others, the different products. Consider a lacking product feature as a negative impact on the sales of the organization. In the DuPont formula, such aspects will impact both the operation efficiency and asset use efficiency. Consequently, it will be very hard to arrive at a correct and precise analysis of the cause of the declining ROE by using functional decomposition. Moreover, other terms can easily be added to the DuPont formula, or a completely different decomposition can be made. As a result, different analysts will arrive at different conclusions. Based on constructional models, a more objective analysis can be made (Dietz, 2006). Therefore, we expect the integration of constructional models in organizational diagnosis approaches. However, different shortcomings with regard to this expectation can be observed.

First, many causal diagrams only focus on creating finer grained black-box models. Put differently, they decompose a big black box into smaller black boxes. However, they do not consider the relevance of including constructional mechanisms. As a result, these organizational diagnosis approaches limit themselves to functional decomposition, and exhibit similar shortcomings as described in the example above. For example, Russo describes how Causal Loops Diagramming (CLD) can be used to specify correlation relations between variables (Russo, 2008). However, such approaches are only considered to be able to predict the behavior of organizations, not to explain the observed phenomenon (Craver, 2006). Moreover, Woodward argues that such approaches may even fall short when used for predicting behavior (Woodward, 2005): without constructional knowledge, it is not possible to foresee the "conditions under which those relations might change or fail to hold altogether".

Second, certain approaches seem to propose to include constructional elements in the functional decomposition in order to claim causality. By including constructional elements, a direct relationship between observed functional elements and constructional elements can be made. Craver calls such models "mechanism sketches" (Craver, 2006). Mechanism sketches are incomplete models of a mechanism, which "characterize some parts, activities, and features of the mechanism's organization, but [which have] gaps" (Craver, 2006). With regard to this approach, several reservations can be made.

- It has been argued that functional and constructional models are different in nature (Dietz, 2006). Consequently, different modeling constructs need to be used, which makes a model harder to interpret.
- Modeling the functional variables of an organization would already result in an enormous amount of variables (Ettema, 2011). Adding additional variables will result in increasing complexity, which makes the models harder to manage and interpret.
- Adding constructional elements in an ad-hoc manner fails to identify dependencies between various constructional elements.
- Craver argues that the missing gaps in such mechanism sketches can function as "veil for a failure of understanding" (Craver, 2006).

Third, approaches which explicitly incorporate a constructive perspective, and separate it from behavorial observations, do not offer any support on how to model or select such a constructive perspective. For example, we discussed the sawtooth thinking approach in LSS (see Figure 1). In this approach, the behavioral measurements belong to the factual world, while a constructional model would belong to the interpretative world. However, no guidance to identify relevant constructional elements is available: cause and effect thinking in LSS is supposed to be performed through "brainstorming" (Ettema, 2011).

This analysis shows that, in order to deal with the presented complexity, current diagnosis approaches (1) only consider functional decomposition, or (2) include constructional elements partially, or (3) include explicitly constructional models, but do not provide guidelines on how to construct them. Based on this classification, it can be concluded that it is useful to develop a method, based on a current organizational diagnosis approach, which explicitly includes a concrete approach for constructing constructional models (such as, for example, Enterprise Ontology (Dietz, 2006)).

5 CONCLUSIONS

In the introduction, we started by positioning two issues in the field of organizational diagnosis. We can summarize the contributions of this paper with regard to these issues. First, the inherent complexity of organizations makes diagnosing challenging. In regard to this issue, this paper makes a contribution by using the concept of entropy to interpret the origin of this complexity. Moreover, the paper shows how entropy can be controlled based on insights from related fields. We identified the presence of structure in the constructional perspective to be primordial in controlling entropy. Consequently, a diagnosing approach which attempts to address this complexity should explicitly incorporate a constructional perspective. Sec-

ond, no systematic approach is currently available to perform organizational diagnosis. In order to demonstrate this point we argued that current diagnosis approaches do not adequately incorporate the explicit usage of constructional models. We believe that the thorough application of engineering concepts such as entropy in organizational research is important to further the scientific field described in the Enterprise Engineering Manifesto (Dietz, 2010). Therefore, such a method would indeed further the field of organizational diagnosis.

ACKNOWLEDGEMENTS

P.D.B. is supported by a Research Grant of the Agency for Innovation by Science and Technology in Flanders (IWT).

REFERENCES

Alderfer, C. P. (2010). *The Practice of Organizational Diagnosis: Theory and Methods*. Oxford University Press, USA; 1 edition.

Boltzmann, L. (1995). *Lectures on gas theory*. Dover Publications.

Box, G. E. P. and Liu, P. Y. T. (1999). Statistics as a catalyst to learning by scientific method. *Journal of Quality Technology*, 31(1):1–15.

Craver, C. F. (2006). When mechanistic models explain. *Synthese*, 153(3):355–376.

de Mast, J. and Bisgaard, S. (2007). The science in six sigma. *The American Heart Hospital Journal*, 40(1):25–29.

Dietz, J. L. (2010). Enterprise engineering manifesto. Online available at: `http://www.ciaonetwork.org/publications/EEManifesto.pdf`.

Dietz, J. L. G. (2006). *Enterprise Ontology: Theory and Methodology*. Springer.

Ettema, R. (2011). Diagnosing the enterprise, towards a true causal claim. In *Proceedings of the 2011 CIAO DC*.

Gero, J. S. and Kannengiesser, U. (2004). The situated function-behaviour-structure framework. *Design Studies*, 25(4):373–391.

Harrison, M. I. (1994). *Diagnosing Organizations: Methods, Models, and Processes*. Sage.

Harry, M. (1988). *The nature of six sigma quality*. Motorola University Press.

Hevner, A. and Chatterjee, S. (2010). *Design Research in Information Systems: Theory and Practice*, volume 22 of *Integrated Series in Information Systems*. Springer.

Horowitz, I. (1970). Employment concentration in the common market: An entropy approach. *Journal of the Royal Statistical Society*, 133(3):463–479.

Jacquemin, A. P. and Berry, C. H. (1979). Entropy measure of diversification and corporate growth. *The Journal of Industrial Economics*, 27(4):359–369.

Janow, R. (2004). Shannon entropy and productivity: Why big organizations can seem stupid. *Journal of the Washington Academy of Sciences*, 90(1):39–50.

Katz, D. and Kahn, R. (1978). *The social psychology of organizations*. Wiley.

Palepu, K. (1985). Diversification strategy, profit performance and the entropy measure. *Strategic Management Journal*, 6(3):pp. 239–255.

Russo, F. (2008). *Causality and Causal Modelling in the Social Sciences: Measuring Variations (Methodos Series)*. Springer.

Senge, P. M. (1990). *The Fifth Discipline: The Art and Practice of the Learning Organization*. Doubleday.

Shannon, C. (1948). A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423.

Sitkin, S. B., Sutcliffe, K. M., and Schroeder, R. G. (1994). Distinguishing control from learning in total quality management: A contingency perspective. *The Academy of Management Review*, 19(3):537–564.

Weinberg, G. M. (1975). *An Introduction to General Systems Thinking*. Wiley-Interscience.

Wikipedia (2011a). Entropy.

Wikipedia (2011b). Entropy (information theory).

Woodward, J. (2005). *Making Things Happen: A Theory of Causal Explanation*. Oxford University Press, USA.

Multi-tenant Database Clusters for SaaS

Evgeny Boytsov and Valery Sokolov
Department of Computer Science,Yaroslavl State University, Yaroslavl, Russia
{boytsovea, valery-sokolov}@yandex.ru

Keywords: Databases, SaaS, Multi-tenancy, Scalability.

Abstract: SaaS paradigm brings both many benefits to end users and many problems to software developers. One of such problems is an implementation of a data storage, which is able to satisfy needs of clients of a service provider, at the same time providing easy application interface for software developers and great opportunities for administration and scaling. This paper provides a brief review of existing problems in the field of organizing cloud data storages that are based on the relational data model and proposes the concept of architecture of RDBMS cluster dedicated to serve multi-tenant cloud applications.

1 INTRODUCTION

One of most notable tendencies in the modern software development industry is the shift to Software as a Service (SaaS) paradigm. The main ideas of this approach are the following.

- An application is developed as a system of distributed services interacting with each other.
- All computing power and infrastructure needed for operating an application is supplied by a service provider.
- A fee for an application is taken on the basis of actual usage.

The main advantage of this development approach for customers is that all expenditures for deploying infrastructure, required for correct and stable operation of software suit, are taken by a service provider. This fact should eliminate the need for a customer to have his own IT staff and purchase new computer equipment with every new release of an application. Besides, this approach allows to completely solve the problem of software updating, because now it is done in a centralized manner by the software company itself, that means, that all customers always use the most recent (i.e. the most safe and featured) version of an application.

However the «jump into clouds» brings not only benefits, but also new problems mostly for developers and administrators of such systems.

It is known that most of enterprise-level applications are based on interaction with relational databases. The de-facto standard for such data storages are RDBMS. In recent years there was a tendency to move the most of application logic to the database tier expressed in appearing procedural extensions of the SQL language. Modern RDBMS are able to process very large arrays of data, fulfill very complex data selection and data manipulation queries. Most of software development specialists are familiar with the SQL language and principles of data organization in RDBMS.

In traditional on-premice applications data and a database server are hosted by a customer, thus their safety and availability are under responsibility of customer's IT-staff. In the case of a cloud application, data are hosted by (and thus are under responsibility of) a service-provider which undertakes to provide instant and fast access to them for tens or hundreds of thousands of its clients concurrently. Non-fulfillment of any of these two requirements (speed and availability) would cause penalties to the service provider, and, that is much more important in the cloud industry, would worsen the image of the provider. A typical service level agreement for a cloud service guarantees its availability of 99% (Candan et al., 2009). Thus, maintenance of a cloud application implies large expenditures to organization of the data storage, caused by a need to store data of hundreds of thousands of clients and their backup copies in order to restore in case of failure. These expenditures can drastically limit a barrier of entry to cloud business and decrease provider’s profits. That is why a common desire of SaaS vendors is to minimize costs of data storing and to find architectural solutions that would lead as much as possible to such a minimization without compromising performance and functionality of the application.

One of such solutions is a multi-tenant application (thus, also database) architecture. The main idea of this approach is to share one instance of an application among many tenants (companies subscribed to the service), thus drastically reducing expenditures to application servers, web-servers and associated infrastructural elements. An application design according to such architectural principles imposes some restrictions to functionality, but it brings unprecedented opportunities to scale the solution and allows, having sufficient physical (or virtual) computing power, to set up an unlimited amount of application instances to serve clients.

However, these considerations do not apply to database servers which are the first candidates to become a bottleneck as the system grows. The reason for this lies in the fact that, in a contrast to application servers, database servers scale poorly. To be more precise, application servers are able to scale well just because they descend most of load to the level of database servers, often just generating SQL queries and performing simple post-processing of the result. The database server should provide reliable data storage, fast access, transactional integrity and much more. A trend in recent years, when the most of the application logic moved to the database level, increased the load on this component of the system even more. The total amount of data of all provider's customers in cloud solutions and the number of different queries that they have to perform, make traditional database scaling techniques (like vertical scaling or database partitioning) even less ineffective than they were earlier.

2 BACKGROUND: MODERN WAYS OF ORGANIZING A MULTI-TENANT ARCHITECTURE

There has already been some experience in the field of organizing cloud data storages (Chong et al., 2006b). At the moment, there are two main approaches to designing multi-tenant relational database.

- Usage of shared tables for all clients with attachment of tenant identifier to every record — this is the shared table approach.
- Creation of the own set of tables for every tenant (usually, these tables are united into one database schema) — this is the shared process approach.

Both approaches have its pros and cons.

2.1 Shared Table Approach

This approach is the most radical in answering a question of sharing server resources. Its usage requires adding a special column to every table in a database which stores a tenant identifier to distinguish data of different clients. Every SQL query to the database of such architecture should be supplemented with an additional WHERE/HAVING predicate, that leaves in the query result only records that belong to a specific client. There are also some projects of SQL extensions (Schiller et al., 2011), that allow to add such predicates automatically. The advantages of the shared table approach are the following:

- better usage of a disk space;
- small size of a data dictionary;
- better usage of a query planner's cache (i.e. shorter time of query analyzing and generation of its execution plan).

This approach also has some drawbacks. First of all, it is the enlarging of the size of database tables and their indexes (Jacobs and Aulbach, 2007). This drawback results in a requirement of very high qualification of developer of database queries, since any mistake or inefficient solution can lead to significant degradation of an application performance. In second, usage of this approach implies a need to always add predicate of selection of data of a current tenant. This drawback leads to access errors, when users of one tenant can see data of another tenant in a case of a programmer error. The above mentioned (Schiller et al., 2011) concepts of extensions of the SQL language are possibly able to solve this issue. The third issue of this approach is a complexity of replication and backup copying of data of a separate tenant.

In general, this approach shows good results, when application's data schema does not contain many tables, and a typical query is relatively simple. If the above conditions are met, this approach allows the most effective usage of hardware resources.

2.2 Shared Process Approach

This approach occupies an intermediate position in solving a problem of sharing server resources, between complete isolation of tenant's data in a separate database and a shared storage of them in the shared table approach. The separation of tenant's data is achieved by creation of the own set of database objects (tables, views, e.t.c.) for each tenant. This approach has some advantages.

- Unification of the code of database queries and ease of writing new ones, because, in contrast to the shared table approach, queries known to operate only the current tenant's data, which usually have relatively small size, and, therefore, do not require a lot of memory and other database server resources for their execution (Jacobs and Aulbach, 2007).
- Relative ease of backup copying and data replication of data of a single tenant.
- Decrease of data security risks since tenant's data are grouped together in the own schema;
- Simplification of system administration.

But there are also some drawbacks of this approach. Its usage makes data dictionary of a database very large and heavyweight, decreasing overall database performance. Because of that a query planner is unable to use its cache effectively, that makes him generate a new plan of execution for almost every incoming query (Schiller et al., 2011). Compared to the shared table approach, a disk space is used less effectively (Schiller et al., 2011). Large amount of database objects results in a very long and hard procedure of a data structure change if it is required.

In general, this approach shows good results, if an application data structure is complex, and a typical query selects data from a large set of tables, makes nested subqueries and other complex data manipulations.

3 MOTIVATION: LIMITATIONS OF EXISTING APPROACHES AND GOALS OF THE RESEARCH

Despite the fact that they are not directly supported by most of database engines, both approaches are successfully used by the software development industry. However, generated databases are very large and complex, and therefore they are hard to manage. But every cloud application that aims to have a large user base has to operate on dozens of databases of such a complex structure. It is physically impossible to place all clients into one database. The highest level of database resource consolidation known today is about 20 000 tenants in one database with a relatively simple data structure (Candan et al., 2009). A simple calculation shows that even with such a high degree of resource consolidation, a company would require 50 database servers to serve 500 000 tenants, storing one backup copy of data for each of them for load balancing and data protection against failures and errors. In reality such system would require much more database servers.

But the quantity of database servers is not the only problem in organization of a cloud cluster. Even a more significant point is a load balancing for the optimal usage of computing power and disk space at the entire cluster level. The nature of a cloud application is that the load on it is unpredictable, it may rise and fall like an avalanche, and "burst" of activity can occur from a variety of tenants. To provide the required level of service, an application should be able to dynamically adapt to changing conditions by automatically redistributing available resources. At the moment, there are no software systems that are able to solve this problem, as there are no clear requirements and approved algorithms for them. This general problem assumes the study of the following directions of research:

- development of algorithms of load balancing for multi-tenant cloud database clusters;
- research of developed algorithms for efficiency and safety, including imitation modelling and stress testing;
- development of complex solution for organizing multi-tenant cloud database clusters using ordinary servers.

In the work we will focus our attention on the third goal. The first and the second ones are supposed to be studied in future.

4 SOLUTION REQUIREMENTS

The developed system would be intended for using by small and medium-sized software companies, and, therefore, should be designed to meet their needs and capabilities. The following points are important.

- Reliability and maximum guarantee of data safety. The reputation is extremely important for a cloud service provider, because his clients have to trust him their data.
- Efficient usage of available resources, providing maximum performance of an application.
- Similarity to traditional DBMS with minimal possible corrections for the cloud application specifics.
- Maximum horizontal scalability. The horizontal scalability is preferred to vertical, as it is cheaper and potentially allows infinitely increase the performance of the system.
- Ease and automation of administration as the manual administration of a very large and complex infrastructure could lead to

management chaos and system unmaintainability.

Let us list the main characteristics of multi-tenant databases for cloud applications, which should be taken into account when designing a cluster management system.

First of all, it is the huge aggregate size of stored data. As the provider must serve dozens and hundreds of thousands of clients, the total amount of data, which it is responsible for, is huge and constantly growing with an unpredictable speed. But the size of data of an average client is small. Since we are talking about multi-tenant solutions, this solution is likely to aim at small and medium-sized companies (so called «Long Tail» (Chong and Carraro, 2006a)), and therefore the number of users and the size of data of an average tenant are not large.

Another common point is the presence of shared data. Usually any cloud application has some set of data, which is shared among all tenants of the provider. The size of that data is usually relatively large, and modifications are very rare.

Since cloud solutions have centralized architecture and a service provider is responsible for the large amount of client's data, it needs good facilities for data backup and replication. Like in traditional DBMS, data replication is used to balance the load of database servers. The distinction is that often the replication in cloud solutions is partial, i.e. only data of one or some tenants are replicated. The above mentioned shared data are also replicated.

Based on these requirements and features, we present the proposed project of the cloud cluster management system.

5 THE ARCHITECTURE OF MULTI-TENANT DATABASE CLUSTER MANAGEMENT SYSTEM

The main idea of the proposed solution is to add a new layer of abstraction between application and database servers, functions of which are listed below.

- Routing of queries from an application server to an appropriate database server by the tenant identifier.
- Management of tenant data distribution among database servers, dynamic data redistribution according to an average load of servers and characteristics of different tenants activity in time.
- Management of data replication between database servers in the cluster.
- Management of data backuping.
- Providing a fault tolerance in the case of failure of one or some of the cluster databases.
- Analysis of resource usage and system diagnostics.

The system should be implemented as a set of interconnected services, using its own database to support a map of the cluster and collect statistics on the system usage by tenants and characteristics of the load. Shared process approach is going to be used for tenants data separation at the level of a single database. The choice of this approach is explained by the fact that the system must be sufficiently general and will not have in advance the knowledge about a data structure required for an application. Because one of the requirements of the shared table approach is to add a service column to every table in the database, it assumes much closer familiarity with application data structure, and thus, its usage is difficult for the generic system. Moreover, the usage of a shared table approach requires very good query optimization skills, and thus does not hide the underlying structure of the cluster from the developer. The general architecture of the proposed system is shown in Figure 1.

We proceed to a more detailed consideration of the above-mentioned functions of the system. The proposed solution assumes an appearance of a new element in a chain of interaction between application and database servers. This new element is a dedicated server which, transparently for application servers, routes their queries to an appropriate database server, basing on a tenant identifier provided with a query, characteristics of the query and statistics of the current system load. This is the component application developers will deal with. In fact, this component of the system is just a kind of a proxy server which hides the details of the cluster structure, and whose main purpose is to find as fast as possible an executor for a query and route the query to him. It makes a decision basing on a map of a cluster.

It is important to note that a query routing server has a small choice of executors for each query. If the query implies data modification, there are no alternative than to route it to the master database for the tenant, because only there data modification is permitted. If the query is read-only, it also could be routed to a slave server, but in a general case there would be just one or two slaves for a given master, so even in this case the choice is very limited.

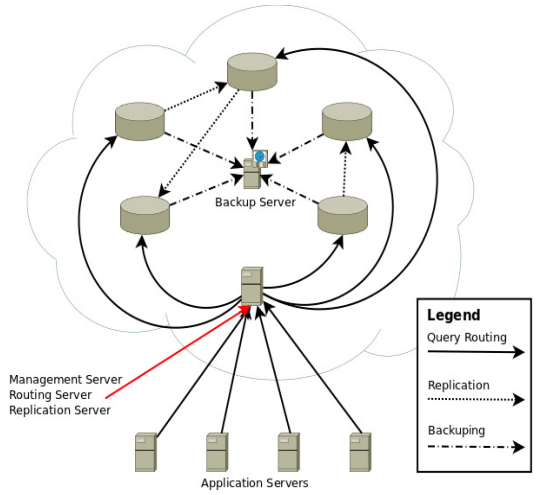


Figure 1: Multi-tenant database cluster architecture.

Besides, it is important to mention that the discussed component of the system can not use expensive load balancing algorithms, because it must operate in real-time. All it can use is its own statistics on the number of queries sent to a specific database server of a cluster, the execution of which has not yet been completed. Basing on this runtime data, it must make a decision on where to send the next query.

The implementation of this component should give a significant benefit in performance and ease of cluster administration.

The second component of the system is the replication and backup management server. Its functions are clear from the title, but it is important to note, that, unlike the traditional databases, in multi-tenant solutions replication is almost always partial, i.e. only a part of data is replicated. For example, data from the first tenant schema could be replicated to one database, from the second tenant schema — to another, and the third tenant schema itself could be a replica of a part of another database from a cluster. Once again we recall that the data change request can only be executed by the master database of the tenant, and this consideration should be taken into account during the distribution of tenant data among servers to avoid hot spots.

The third component of the system will be a set of agent-services placed at the same machines as database servers. These small programs-daemons should collect statistics about the load of the server and monitor server state in a case of failure. All the information collected would be sent to a central server for processing and analysing and would be used as an input data for the load balancing algorithm.

The last and most important and complicated component of the system is the data distribution and load balancing server. Its main functions are:

- initial distribution of tenants data among servers of a cluster during the system deployment or addition of new servers or tenants;
- collecting the statistics about the system usage by different tenants and their users;
- analyzing the load on the cluster, generation of management reports;
- management of tenant data distribution, based on the collected statistics, including the creation of additional data copies and moving data to other server;
- diagnosis of the system for the need of adding new computing nodes and storage devices;
- managing the replication server.

This component of the system has the highest value, since the performance of an application depends on the success of its work. There are several key indicators that can be used to evaluate its effectiveness. First of all, it is the average response time of a service i.e. an average time between the arrival of a request and receiving a response to it. In second, it is an availability of a service, i.e. what percent of requests from the total number has been executed successfully, which failed to meet a time limit or other parameters, and which is not executed at all. Both previous criterions are affected by the average load of database servers. The cluster management system must provide conditions, when servers are relatively equally loaded, and there are no idling servers when others fail to serve all requests.

The core of load balancing system should become an algorithm of cluster load analysis and need for data redistribution. There are several considerations that this algorithm should take into account when making its decision about data redistribution. First of all, it is a performance of cluster servers. If the system is not homogenous, proportions of its parts should be taken into account. In second, it is free resources available. If the system has free resources in its disposal, it makes sense to use them by creating additional copies of tenant data to increase the performance and the reliability of an application. However, if the number of tenants begins to grow, created redundant copies should be removed. Some data distribution strategies can also take into account the history of the individual tenant activity. If users of tenant A actively use an application, and users of tenant B don't, it makes sense to move the data of tenant B to a busier server and create fewer copies of them, since they unlikely will cause problems for the service. The algorithm

should also prevent the creation of hot spots on writing the data in a context of organization of replication. The system should distribute master servers for all tenants in an appropriate way, taking into account the history of tenant activity.

6 CONCLUSIONS AND FUTURE WORK

Thus, in this paper we presented some principles of the architectural design of a multi-tenant database cluster. The implementation of the proposed architectural solutions should provide the framework the usage of which will simplify the development of applications according to the SaaS paradigm. Also, the proposed approach should facilitate the administration and maintenance of the cluster.

The above discussed algorithms for query routing and tenant data distribution can be based on a variety of strategies, and currently it is not clear which of them should be preferred. From this it follows that the most reasonable solution would be to implement several variants of the algorithms and choose the best in a general case according to the results of imitation modelling and stress testing. It is very likely that such versions will not be found and a final implementation of the system will contain several modifications of them, which showed themselves as the best ones under some external conditions. In this case a choice of an appropriate version of algorithms would become a task of a cluster management system administrator.

There are also some questions for a future work:

- the study of fault-tolerance of clusters in case of failure of one or more servers;
- the study of customization complexity issues on the side of application developers to satisfy specific needs of clients;
- the study of effective strategies of data replication.

REFERENCES

Chong, F., Carraro, G. (2006a). *Architecture Strategies for Catching the Long Tail*. Microsoft Corp. Website.

Chong, F., Carraro, G., Wolter, R. (2006b). *Multi-Tenant Data Architecture*. Microsoft Corp. Website.

Candan, K.S., Li, W., Phan, T., Zhou, M. (2009). *Frontiers in Information and Software as Services*. in Proc. ICDE, pages 1761-1768.

Jacobs, D., Aulbach, S. (2007). *Ruminations on Multi-Tenant Databases*. In Proc. of BTW Conf., pages 514–521.

Schiller, O., Schiller, B., Brodt, A., Mitschang, B. (2011). *Native Support of Multi-tenancy in RDBMS for Software as a Service*. Proceedings of the 14th International Conference on Extending Database Technology EDBT '11.

A Goal-based Method for Automatic Generation of Analytic Needs

Ben Abdallah Mounira, Zaaboub Haddar Nahla and Ben-Abdallah Hanene
MIR@CL Laboratory, University of Sfax, Tunisia
{Mounira.Benabdallah, Nahla.Haddar, Hanene.Benabdallah}@fsegs.rnu.tn

Keywords: Analytical Requirements Generation, Goal Requirements Language (Grl), Uml Modeling.

Abstract: In this paper, we are interested in the requirements engineering of decision support systems. In particular, we propose a method, called *Analytic Requirements Generation Method* (ARGeM), for automatic generation of analytic requirements. Our method meets the strategic goals of the enterprise and produces loadable DW schemas. It begins with modeling the goals of the enterprise and uses the UML IS modeling artifacts to generate automatically a complete set of candidate analytic needs. These needs are, subsequently validated by the decision makers who are thus directly involved in the specification process. Once validated, needs contribute to the design of the DW.

1 INTRODUCTION

For decision making, analytic needs that must be satisfied by the system's data warehouse (DW) are specified during the analysis phase. As one of the early stages of system development, this phase implies major problems, if it is inaccurate or incomplete and does not meet the entire user's needs. Thus, it should attract special attention and must be fully supported by effective methods.

On the other hand, several surveys indicate that a significant percentage of DWs fail to achieve the business goals or are spectacular failures. One reason for this is that the requirements analysis is typically overlooked in real projects (Giorgini et al., 2008). Thus, this stage should be based on a goal oriented framework for requirements engineering as the DW aims at providing adequate information to support decision making and to achieve the goals of the organization.

Moreover, existing approaches to decision system development, such as (Golfarelli et al., 1998) (Moody et al., 2000) (Giorgini 2008) works, always consider that the information system (IS) of the enterprise is already computerized and operational for a large period. Therefore, these approaches often encounter some problems such as the lack of source schemas. Moreover, the lack of a decision support system aligned with the IS since its implementation, can threaten its survival. To remedy these problems, it is important to have a decision support built at the same time as the IS and constantly aligned with it.

This work proposes a method, called *Analytic Requirements Generation Method* (ARGeM for short), for automatic generation of analytic requirements that meet the strategic goals of the enterprise and produce loadable DW schemas. Our method begins with modeling the goals of the enterprise and uses the UML IS modeling artifacts to generate automatically a complete set of candidate analytic needs. (The use of UML is due to the fact that this language is a defacto standard for IS modeling). These needs are, subsequently validated by the decision makers who are thus directly involved in the specification process. Once validated, these needs contribute to the DW design.

In the following, Section 2 gives a state of the art of works on the analytic requirements engineering. Section 3 presents our approach to generate analytic needs. The last Section concludes this work and discusses its prospects.

2 RELATED WORKS ON ANALYTIC REQUIREMENTS ENGINEERING

Although most DW design methods claim that there must be a phase devoted to analyze the requirements of an organization (Golfarelli et al., 1998) (Kimball 2002) (Lujan-Mora et al., 2006), this phase does not generate the same interest in both types of DW design approaches: bottom-up and top-down. Indeed,

bottom-up approaches start from a detailed analysis of the data sources (Golfarelli et al., 1998) (Moody et al., 2000). Analytical needs are expressed directly by the designer who must select relevant blocks of data to decision making and determine their structuring according to the multidimensional model (Golfarelli et al., 1998) (Moody et al., 2000) (Cabibbo et al., 1998) (Prat et al, 2006).

Therefore, these approaches assume that decision makers have a good knowledge about the models of operational data, and a perfect understanding of the structures of the data source. Thus, they marginalize the analysis phase of the OLAP requirements in a decision system design. Therefore, the DW may not satisfy all its future users, and may, therefore, probably fail (Giorgini et al., 2008). In addition, all these approaches produce multidimensional schemas regardless of the needs of the decision makers. Thus, the produced schemas are far from covering the goals of the organization.

Unlike bottom-up approaches, top-down ones start by determining information needs of the DW users. These approaches collect and specify the user requirements using different formalisms: goal based models, UML use cases, query languages or decision oriented models. The problem of matching user requirements with the available data sources is treated only a posteriori.

Most goal based approaches are essentially founded on the conceptual framework i* (Giorgini et al., 2008) (Zepeda et al., 2008) (Franch et al., 2011). Requirements' specification is carried out manually from diagrams modeling the enterprise and its goals. Thus, these approaches may overlook some requirements as they may specify needs not covered by the sources. Moreover, they do not directly involve the decision maker. Besides, i* is not a standard and does not provide all the concepts necessary for modeling purposes. So, it requires specific training and tools that support it in order to be used.

The use case (UC) based approaches adopt the UCs of UML to represent the analytic needs (Luján-Mora 2006) (Shiefer et al., 2002). Thus, because of the absence of a precise oriented decision syntax for enouncing UC actions, it becomes very difficult, to identify potential decision elements from the specification. Moreover, the fact that the UML does not model the organization goals, using UC cannot guarantee the coverage of all enterprise goals.

The query based approaches are the most used in literature (Romero et al., 2006) (Bargui et al., 2008), because queries expressed in natural language or pseudo language are easy to understand by decision makers. However, the non-exploitation of the source information impedes obtaining, from the beginning, an optimal set of analytic needs.

In the decision based approaches, needs are specified using decision concepts expressed in a given formalism (Kimball 2002) (Golfarelli et al., 1998). Although the models used by these approaches are characterized by their decision orientation, they remain difficult to understand by decision makers who lack design expertise. Moreover, the fact that there is no well-defined framework for defining goals, the specified needs do not guarantee the achievement of these goals.

The overview presented above on top-down approaches reveals two important criticisms. First, none of the presented methods propose joint modeling of the DW and the IS. This may impede the alignment of the DW to the IS which in turn may produce unloadable schemas. In addition, it does not guarentee the completeness of the analytic needs. Second, specifying needs without linking them to their goals may not lead to the achievement of the expected goals.

To remedy these problems, we propose an analysis method called *ARGeM* (*Analytic Requirements Generation Method*) for automatic generation of analytic requirements that meet the strategic goals of the enterprise and produce loadable DW schemas. Our method begins with modeling the goals of the enterprise and uses the UML IS modeling artifacts to generate automatically a complete set of candidate analytic needs. The aligned modeling of the DW and the IS facilitates the co-evolution of both systems. Another advantage of our method is that it involves directly the decision makers in the specification of their needs by validating the generated requirements.

3 GOAL DRIVEN ANALYTIC REQUIREMENTS GENERATION METHOD

ARGeM consists of three steps (cf. Figure1): i) GRL model construction, ii) analytic element identification and iii) analytic requirements generation. In the following sub-sections, we detail these steps.

3.1 Construction of the GRL Model

Since achieving the qualitative goals of an enterprise is the main purpose behind modeling a DW, it is obvious to begin with determining these goals and taking them as a start point for deriving any analytic

needs. Thus, the first step in our approach is to construct automatically a model representing the qualitative goals of the enterprise. As a pre-condition for this step, we suppose the existence of a business strategy definition for the enterprise, represented with one of the current models such as the ISO1 model. This document, which defines all the strategic goals of the enterprise, usually exists since the establishment of an enterprise requires its presence. We also consider the use case model of the UML IS modeling documentation which gives the functionalities of the IS that helps to reach the strategic goals.

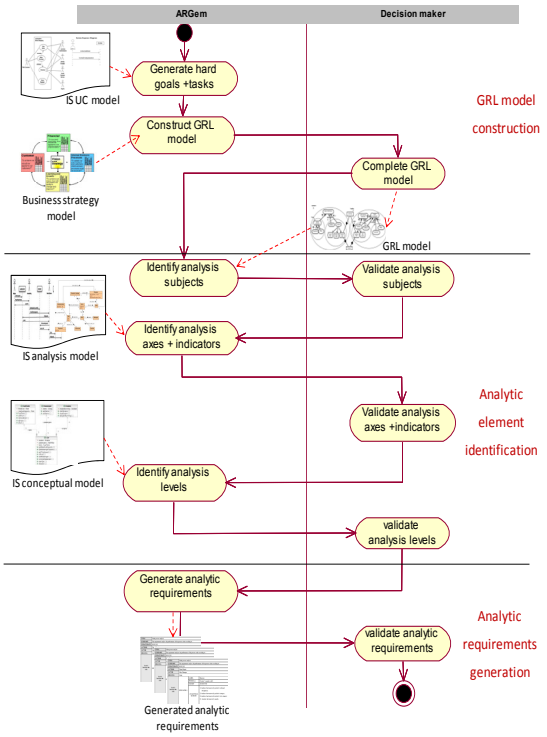


Figure1: The steps of ARGeM.

The product of the first step is a goal model represented with the standard goal requirements language GRL (ITU-T, 2008). With this language, it is possible to represent both functional (low level) goals and their performing tasks, and qualitative (high level) goals that the former tend to meet.

Furthermore, the functional goals of the enterprise are realized by its IS. Their specification is part of the SI design. Indeed, the UC model represents these goals as UCs and scenarios. Thus, it is possible to transform these latter into functional goals and tasks in the GRL model. Subsequently, this model will be completed by the high level goals and their dependencies with all other model elements basing on the business strategy model (BSM) and decision makers directives.

To ensure the generation of the goals from the UC model, we are inspired from the works of (Vicente 2009) (Cysneiros et al., 2003). Basing on these works, we define three rules for transforming the UC concepts into GRL.

Rule1: Transformation of UML Actors

Each UML actor can be transformed into a GRL actor that has the same name. A generalization/specialization relationship between two UML actors becomes an inclusion relationship between the two corresponding GRL actors.

In UML, UCs can be classified into three types: business, support and decision ones (Morley et al., 2008). A business UC describes a business activity while a support UC manages a system resource that is necessary for a business activity. A decision UC provides useful information for decision making. This classification is not standard in UML but can be easily carried out by stereotyping all UCs with one of the three stereotypes: “business”, “support” or “decision”.

UCs are described through nominal and alternative scenarios. In GRL, the goals of an enterprise are of two types: hard and soft. A hard goal is a low level goal that represents a state or a condition that the stakeholders would like to achieve in the enterprise. While a soft goal is a high level one and describes qualitative aspects rather than functional ones. The GRL goals are achieved by executing a set of activities called tasks (ITU-T 2008).

Rule2: Transformation of UCs

Business UCs become hard goals. The scenarios of a UC are potential tasks composing the corresponding hard goal. By contrast, a support UC becomes a GRL resource representing a physical or an information entity.

Rule 3: Transformation of Relationships between Actors and UCs

The communication relationship between an actor and a UC results in the placement of the corresponding goal in the GRL actor generated from the UML one.

Applying these three rules produces a GRL model containing all the elements modeling the enterprise goals except the soft ones. To complete

[1]International Organization for Standardization. http://www.iso.ch

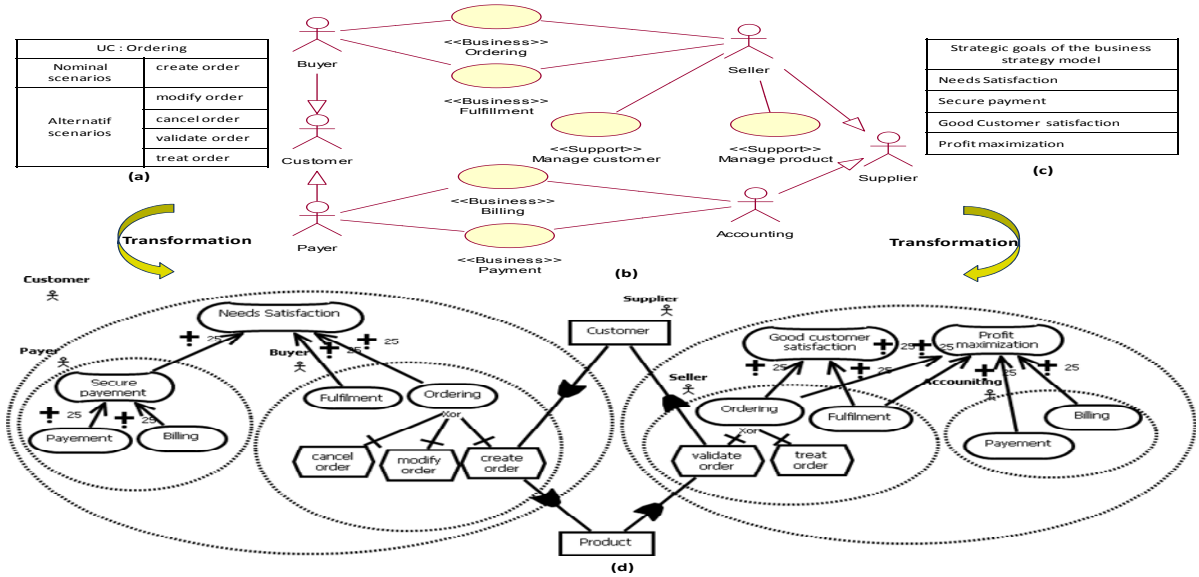


Figure 2: Extract of the GRL model (d) constructed from a UC model ((a) and (b)) and a BSM (c) from the "online sales".

this model, the soft goals specified in the BSMare automatically copied in the GRL one. Then, the missing relationships between soft and hard goals are added by the decision maker.

Figure 2 shows an extract of the GRL model (d) constructed from a UC model ((a) and (b)) and a BSM(c) from the "online sales" domain.

Since the analytic needs aim to analyze information from the IS in order to achieve the stated goals, we must identify the analytic elements from the IS (analysis subjects, analysis axes, indicators) that contribute to formulate these needs. To do this, in the second step of our method, we start from the UML IS modeling artifacts and the GRL model built in the first step. The relationship between the goals of the enterprise and the system functionalities made by the above defined three rules is used to identify these elements.

3.2 Identification of Analytic Elements

This step aims to identify the elements contributing to formulate analytic needs in terms of subjects, indicators, axes and the analysis levels.

3.2.1 Identification of Analysis Subjects

An analysis subject is an activity of the target functional system in order to achieve the company goals. Thus :

RS: each hard goal contributing to satisfy a soft goal is a potential analysis subject.

This rule is justified by the fact that a subject to be analyzed represents business missions. These latter are supported by business UCs. On the other hand, generating subjects (indirectly) from UCs guarantees the loading of these subjects from the source. From the GRL model of Fig 2, the rule RS identifies the subjects "Ordering", "Fulfillment", "Billing" and "Payment".

3.2.2 Identification of Analysis Indicators and Axes

Recall that a subject is formed of indicators and is analyzed from different perspectives called analysis axes representing the observing and the recording context of the indicators. Therefore, the identification of the indicators and the axes of an analysis subject which amounts to analyzing the corresponding business UC. This analysis takes into consideration all the artifacts related to the UC. In particular, we focus on the interaction diagrams (ID) describing the scenarios of the UC and on the class diagram. Since an ID describes the communication between objects that participate in the execution of the UC, this communication serves to identify the potential axes and indicators of the subject. Then, using the class diagram, we consolidate the identified elements and we determine the analysis levels of each axis.

To identify the analysis axes, we define the following rule:

RA: In the ID describing the scenario of creation of a business object corresponding to a subject *S*, let *A* the set of business objects created during this

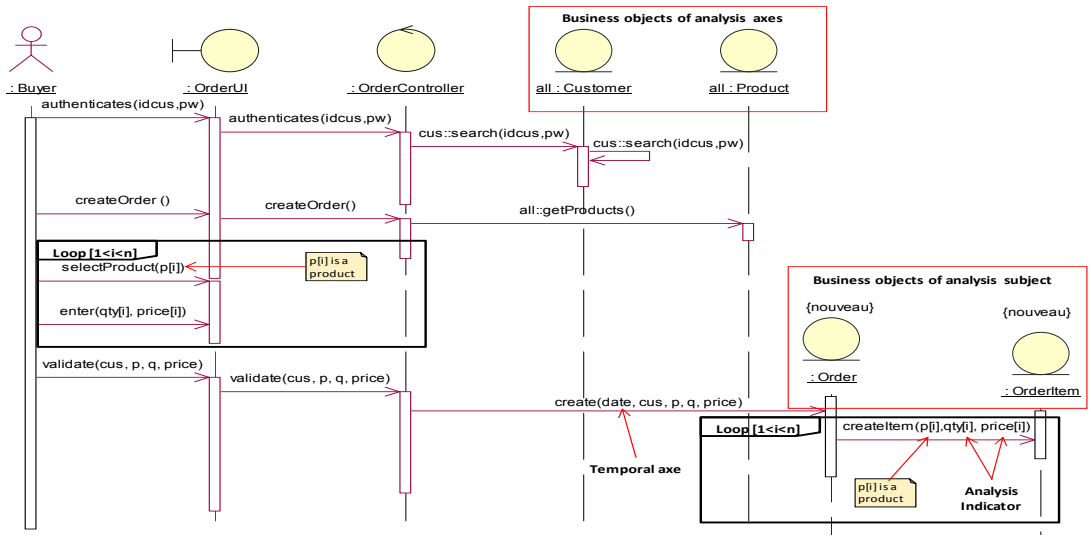


Figure 3: Identification of subject classes, analysis axes and indicators of the « Ordering » subject.

scenario. A business object that is involved in the scenario but does not belong to *A* is a potential axis for *S*. In fact, such an object provides required information for creating the objects of *A*. Moreover, a date parameter of a message having as destination an object belonging to A is a potential temporal axis for S.

It is rather interesting to mention here that a scenario of creation of a business object is easy to identify in UML because this language provides a different notation for the creation message.

Fig 3 illustrates the application of rule *RA* on the sequence diagram formalizing the scenario "Create order" of the "Ordering" UC. The business objects of type "Order" and "OrderItem" created in this scenario correspond to the "Ordering" subject. The business objects of type "Customer" and "Product" correspond to the analysis axes of this subject. The parameter "date" of the message "create" sent to the object "Order" represents a temporal axis for the analysis subject.

We identify indicators by defining the following rule:

RI: in the ID of the creation scenario of a business object corresponding to a subject *S*, a numerical parameter of a message having as destination an object corresponding to *S* and which does not refer to an axis represents a potential indicator for *S*. Indeed, such a parameter will be used in the creation of the destination object.

In the sequence diagram of Fig 3, *RI* produces the indicators "qty" and "price" for the subject "Ordering".

Identification of subjects within business UCs and IDs is more accurate than within class diagrams because pure structural information such as attribute types and multiplicity of relationships is not sufficient to decide of the relevance of a class as being a subject.

In contrast, functional information, provided by UCs, and dynamic information of IDs are more efficient for the subject classes' detection. Indeed, a subject class constitutes a central class around which all interactions take place. Thus, it is easy to identify such a class in a business UC and in an ID.

To consolidate the results provided by rules *RS*, *RA* and *RI*, and to identify the analysis levels of each axis, we use the UML class diagram. To do this, we partition this diagram into *clusters*. Each cluster contains all classes representing a single analysis subject and all classes that are related to it directly or indirectly. Thus, we obtain as many clusters as analysis subjects. The goal of clustering is to facilitate the consolidation phase and, subsequently, to identify the analysis levels.

Recall, first, a primary rule of consistency between the class diagram and IDs: all communication between objects in a system must be supported by static relationships between their classes. Thus, regarding this rule, all classes in a cluster that are directly connected to those corresponding to the analysis subject correspond to axes.

In addition, to consolidate indicators, we define the following rule:

RI': an indicator *m* identified for a subject *S* is an attribute of a class corresponding to *S*.

Figure 4 illustrates the consolidation of the

indicators "qty" and “price” as attributes of the class "OrderItem" by applying the rule *RI*' in the class cluster of the "Ordering" analysis subject.

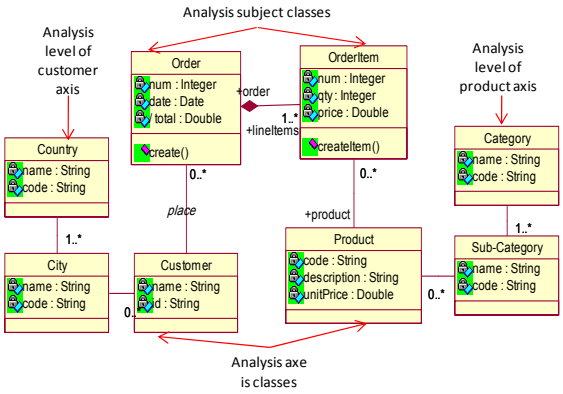


Figure 4: Identification of analysis levels of the product and the customer axes in the “Order” cluster.

3.2.3 Identification of Analysis Levels

Since a subject is analyzed according to different axes, each of which has one o many levels, then, these levels correspond to the attributes of the class axis and those of all classes that are related to it. Moreover, since the levels of an axis are generally non- numerical, we must examine the types of identified attributes.

In addition, during the OLAP process, data are usually analyzed starting from a low level detail to the most detailed one. To formulate analytic needs according to the OLAP process, ie, requirements that analyze a subject according to different levels of axes starting from the lowest to the most detailed one, it is crucial to determine the order of each level in an axis. So, to identify analysis levels, we define two rules *RN1* and *RN2*.

RN1: in a cluster, each class directly or indirectly connected to a class axis is a potential level for this axis. This level has the same range as the number of relationships that separates the class level to class axis. The name of this level is the same as the attribute playing the identifier role in the class level.

RN2: each non-numerical attribute of a class axis (class level) is a potential level for this axis (level). In particular, the levels of a temporal axis are the attributes composing a date such as year, month, and day. The decision maker can also add his own temporal levels.

Since our method does not automatically distinguish between a descriptive attribute and a level attribute, we expect to the decision maker to make this distinction.

In Fig 4, *RN1* identifies, for example, the levels "sub-category" and "category" for the axis "Product". *RN2* generates, for example, the analysis levels "id" and "name" of the axis "Customer".

3.3 Analytic Requirement Generation

For the specification of analytic requirements, we propose to use the template and the syntax proposed in (Bargui et al., 2008) as a means used by decision maker to express his needs. This template is instantiated with the analytic elements identified in the previous phase.

Figure 5 shows the analytic requirement of analyzing the performance of the "ordering" process for an online selling enterprise in order to maximize the profit.

TITLE	Order process analysis		
SUMMARY	This requirement analysis the performance of the process order according to....		
UPDATE DATE	11/04/2012		
ACTOR	Seller		
PROCESS	Order		
Soft Goal 1: Maximize profit	INDICATOR	LABEL	**Total amount**
		FORMULA	**qty** * **price**
		ANALYTIC QUERIES	1) Analyze the **total amount** by *sub-category* and *category* of a **product** according to *month* of a **date**. 2) Analyze the **total amount** by *city* and *country* of a **customer** according to *year* of a **date**. 3) Analyze the **total amount** by *code* and *description* of a **product** by *id* and *name* of a **customer** according to *month* and *year* of a **date**.

Figure 5: Extract of the generated analytic requirements for the process “Ordering”.

4 CONCLUSIONS

In this paper, we proposed an analysis method for automatic generation of analytic requirements which meet the strategic goals of the enterprise and produce loadable DW schemas. Our method begins with modeling the goals of the enterprise and uses the UML IS modeling artifacts to generate automatically a complete set of candidate analytic needs.

The novelty of our method is the aligned modeling of the DW and the IS which facilitates the co-evolution of both systems. Another advantage of our method is that it involves directly the decision makers in the specification of their needs by validating the generated requirements.

As future work, we are examining how we can extract multidimensional concepts from generated requirements.

REFERENCES

Bargui, F., Feki, J., Ben-Abdallah, H., 2008. A natural language approach for data mart schema design. *9th International Arabic Conference on Information Technology (ACIT)*, Hammamet-Tunisia.

Cabibbo, L., Torlone, R., 1998. A logical Approach to Multidimensional Databases. *6th International conference on EDBT*, Valencia, Spain, LNCS 1377, pp. 183-197.

Cysneiros, G. A.A., Zisman, A., Spanoudakis, G., 2003. A Traceability Approach for i* and UML Models. *2nd International Workshop on Software Engineering for Large-Scale Multi-Agent Systems - ICSE 2003*, Portland, May 2003.

Franch, X., Maté, A., Trujillo, J., Cares, C., 2011. On the joint use of i* with other modelling frameworks: A vision paper. *RE*, 133-142.

Giorgini, P., Rizzi, S., Garzetti, M. (2008). GRAnD: A Goal-Oriented Approach to Requirement Analysis in Data Warehouses. *Decision Support Systems (DSS) jounal*, Elsevier, Vol 45, Issue 1, pp. 4-21.

Golfarelli, M., Rizzi, S., 1998. A methodological framework for data warehouse design. *DOLAP*, pages 3–9.

ITU-T: International Telecommunications Union: Recommendation Z.151, 2008. *User Requirements Notation (URN) – Language definition*. Geneva, Switzerland.

Kimball R., 2002. *The Data Warehouse Toolkit*, Wiley, New York, 2nd edition.

Luján-Mora, S., Trujillo, J., Song, I. 2006. A UML profile for multidimensional modeling in data warehouses. *Data & Knowledge Engineering*, 59(3), pp. 725-769.

Moody, D., Kortink, M., 2000. From enterprise models to dimensional models: A methodology for data warehouse and data mart design. *2nd DMDW*, Stockholm, Sweden.

Morley, C., Hugues, J., Leblanc, B. 2008. *UML 2 pour l'analyse d'un système d'information - Le cahier des charges du maître d'ouvrage*. Dunod, 4ème édition, Paris. ISBN 978-2100520978.

Prat, N., Akoka, J., Comyn-Wattiau I., 2006. "A UML-based data warehouse design method". *Decision Support Systems*, vol. 42, pp. 1449-1473.

Romero, O., Abelló, A., 2006. Multidimensional design by examples. *DaWaK*, LNCS 4081, pp. 85–94.

Shiefer, J., List, B., Bruckner, R. 2002. A holistic approach for managing requirements of data warehouse systems. *Americas Conference on Information Systems*.

Vicente, A.A., Santander, V.F.A., Castro, J.F.B., Freitas, I., Matus, F.G.R., 2009. JGOOSE: A Requirements Engineering Tool to Integrate I* Organizational Modeling with Use Cases In UML. *Ingeniare. Revista Chilena de Ingenier,* 17(1) 6-20.

Zepeda, L., Celma, M. , Zatarain, R. 2008. A Mixed Approach for Data Warehouse Conceptual Design with MDA, O. Gervasi et al. (Eds.), *ICCSA 2008*, Part II, LNCS 5073, pp. 1204–1217.

SPECIAL SESSION ON BUSINESS MODELS AND SERVICE APPLICATIONS

Structuring of Growth Funds with the Purpose of SME's Evolution under the JEREMIE Initiative

George L. Shahpazov and Lyubka A. Doukovska
Institute of Information and Communication Technologies, Bulgarian Academy of Sciences
Acad. G. Bonchev str., bl. 2, 1113 Sofia, Bulgaria
atlhemus@abv.bg, doukovska@iit.bas.bg

Keywords: Growth Funds, SME, JEREMIE Initiative, Energy Efficiency

Abstract: Recent surveys in regards to Country's economy development and especially SME's progress in the past few years, shows the decline in development. The recovery and restructuring of the economy would go through rehabilitation and modernization of long-term value creating industries. Continuous development of already established SME's in some of the analysed sectors of the economy is backed by the identified potential for above average growth. Initially targeted favourable sectors, subject of investments would be under the main focus for investments in the upcoming future by any structured fund under the JEREMIE initiative, but will not limit the exploration of other opportunities depending on market developments.

1 INTRODUCTION

Between 2000 and 2008 Bulgaria faced its first real boom period for the last 25 years. The EU accession plan, the currency board and western oriented governments combined with booming banking industry and cheap credit resources created an investor friendly business environment that attracted in total more than EUR 25 billion of FDI and assured steady growth of the economy with rates double than the EU average. Unfortunately more than 70% of these investments went into non-productive, highly speculative and cyclical businesses or were triggered by arbitrage opportunities in privatisation deals. Manufacturing and service industries (excl. financial services) did not benefit proportionally from the growth. Looking back in a period of 25 years Bulgaria has lost more than 50% of its light and heavy industry production and more than 60% of agriculture production, turning from a net exporter into net importer for many goods.

Statistics show that the next growth wave in the country will be driven by the rehabilitation and modernization of long-term value creating industries led by the manufacturing sector, which will profit from a boost in the local agriculture sector and foreign demand (e.g. exports are surpassing pre-crisis levels). Modernization of production assets is closely related to the implementation of Government's initiative for energy affiances improvements. Service industries, excluding financial services and telecom, are currently underdeveloped, and will grab higher share of the economy and outperform.

Manufacturing was heavily hit in the last few years due to strongly decreased internal and external (export) demand, out of date business processes and weak financial management. The liquidity reserves of the sector decreased significantly, fresh liquidity is scarce on the local market and hinders the fast recovery of the sector from the crisis. This situation creates good entry opportunity at low-to-reasonable valuations enabling investors to extract maximum return on the provided capital.

Bulgaria slowly recovers from the crisis; signs of recovery in selected industries are already visible. The Bulgarian government expected GDP to rise by 3.7% in 2011; foreign institutions and banks were more moderate and forecasted an average annual growth of 1.5%. Even though prognoses for a new

Recession in Europe is close to becoming a reality, our expectancy is that after 2013 a partial recovery and additional growth of the export goods demand from Western Europe will be anticipated. Combined with recovery of local consumption and resumed capital inflows this should result in an average GDP growth of 5.0% yoy over the next 10 years.

Table 1: Development and future Economic Forecasting of most Sectors of Economy of Bulgaria.

Forecast by Sectors							
	2004-2008	2009	2010f	2011f	2012f	2013f	2014-2019f
Agriculture	**-4.3%**	**-4.2%**	**2.6%**	**2.6%**	**2.6%**	**2.6%**	**2.6%**
Production	**5.6%**	**-8.1%**	**2.0%**	**8.8%**	**9.6%**	**8.0%**	**5.8%**
Extraction	1.5%	-18.6%	4.8%	6.6%	7.6%	6.5%	4.7%
Manufacturing	6.7%	-8.2%	3.5%	9.6%	10.3%	8.2%	5.8%
Utilities	2.4%	-4.8%	-3.5%	6.3%	7.0%	7.5%	5.8%
Construction	**12.4%**	**-6.4%**	**-8.9%**	**5.5%**	**9.1%**	**8.7%**	**6.3%**
Services	**6.9%**	**-1.7%**	**-1.5%**	**1.6%**	**4.6%**	**6.6%**	**5.9%**

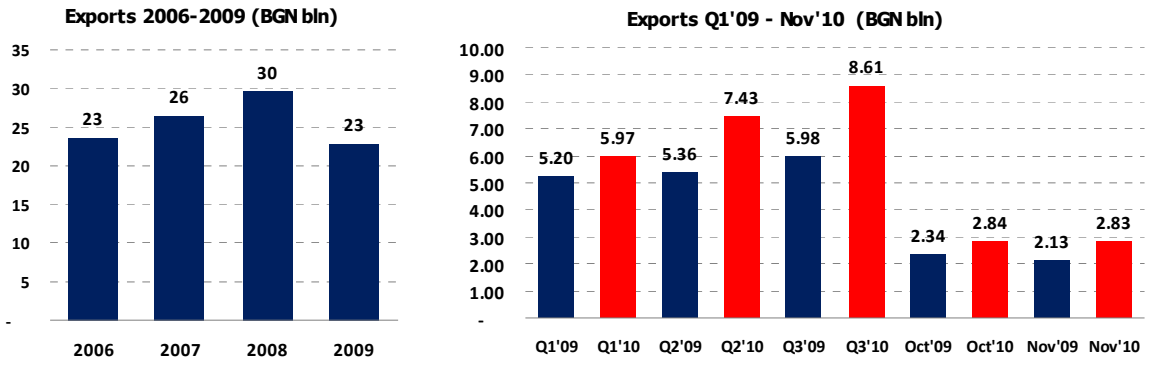


Figure 1: Data of BG export.

Manufacturing output growth is forecasted to be faster than GDP growth over the next decade. Manufacturing output is expected to rise by 9.6% in 2012 and on average by 7% yoy over the next 10 years.

As a result, the share of manufacturing output in GDP is projected to rise from 18.0% in 2009 to 20.9% by 2014 and rise to 21.1% by 2019. Over the same period, the share of service sector output in GDP is expected to fall from 63.4% in 2009 to 61.0% in 2014 and rise to 61.5% in 2019. (Oxford Economic Forecasting)

Figures show that the timing for investment in Bulgarian SME's growth from the manufacturing sector is perfect for the following reasons:

- Rising external demand for the manufacturing sector is already visible in the increase of exports which surpassed pre-crisis peak levels. The decrease in local demand is slowing down to zero, and the reverse trend is already visible in the past year, along with pick up process expected to continue in 2012. This means that the economic cycle will support the investments.
- Most of the companies in the targeted industries have been privatized or established in the period 2001-2007, which makes them attractive for different kinds of growth investments categorised mainly into 2 types: a) expansion – e.g. in production or product range, b) developing the company to the next level – e.g. vertical or horizontal integration or new markets strategy.

The service sector growth prior to the credit crunch was dominated by financial services, telecommunications, and real estate related activities. This led to disproportionate allocation of capital and investments leaving other promising segments of the sector underinvested. Each fund's management should see the potential for above average growth coupled with demand for capital in two major industries, namely: energy efficiency and healthcare.

2 TARGETED MARKET SEGMENT AND ENERGY EFFICIENCY

The analysis seeks to locate and target the most attractive for investments, important and

underdeveloped segment of all sectors of the economy, including several major industries within these sectors. The development of these industries will support sustainable growth in the country's economy in general. The segment will be in the main focus for investments in the upcoming future by any structured fund under the Jeremie initiative, but will not limit the exploration of other opportunities depending on the market developments.

The Energy Efficiency space is an attractive investing segment due to the enormous lag of Bulgaria to achieving EU-wide standard and the active supporting policies implemented over the last years. The government, in line EU targets and initiatives, has provided financial and legislative incentives for improving energy affiance, lowering overall energy consumptions and increasing renewable energy in the total consumptions mix. The country occupies one of the places in terms of energy intensity in Europe with energy intensity coefficients of the GDP standing approximately 90% above EU averages.

The latest Energy Strategy drafted by the government in line with European 20/20/20 goals envisages reduction in green-house emissions, raising the share of RES contribution to 16% of the final consumption, and reducing energy intensity of GDP by 50% by 2020. The interim target for reducing energy intensity of GDP is 25% reduction by the year 2013. The state plans to reduce the energy intensity of GDP from 913 toe/M€05 in 2005 to 456 toe/M€05 by 2020. According to different estimates, the country needs to invest approximate BGN 4.2-4.5 billion to reach the outlined targets and to lower the overall energy intensity of the economy. The achievement of the targets requires implementations efficiency and savings solutions and investments in industry (38% share in total consumption), households (21.8%), transportation (28%), and services (9.4%) and it has opened a market niche for business with above average growth opportunities.

Energy efficiency in Bulgaria is a segment, which is below the average in the EU, not only because it has not received the much needed improvement, but also because priority development was given to targeted industries that are generally energy intensive.

The market of energy efficiency solution providers and services companies is relatively fragmented and consists primarily of SMEs in earlier stages of development, thus offering ample opportunities for investment in innovative technology applications, engineering companies, and complex service providers specialized in the household and industry projects.

Prioritized SMEs in terms of energy efficiency improvement will be businesses, focusing on investments into new machinery, equipment, technologies of higher-energy class, and reduced emissions, along with companies looking for energy efficiency achievement by switching fuel consumptions (gas, etc.). (Bulgarian Small and Medium Enterprise Promotion Agency)

3 INVESTMENT STRATEGY

The individual investments in each fund's portfolio should be selected based on the combination between the mandatory and at least on of the optional criteria:

Mandatory Criteria:

- Management team and human resources' potential;
- Profound market and industry knowledge;
- Business model scalability;
- Distinctive competitive advantages;
- Double digit growth potential of the companies revenues;
- Clear Exit Route.

Optional Criteria:

- Value-adding opportunities through process optimization, strategy fine-tuning;
- Market scalability of the products (export);
- Potential for horizontal or vertical integration.

The majority of SME companies in Bulgaria experience difficulties in maintaining a normal life cycle and tend to suffer from early maturity and decline without being able to materialize its full potential. There are many reasons for this, with the most common being – poor management and lack of financing. The Fund will aim in this cases at eliminating these factors with different optimization strategies, so the company converges to its natural development path and then seek expansion opportunities. Companies that have already accomplished this stage of their life cycle will be prepared for the next level.

Business cycle stage of the investment targets:

By providing equity financing, business expansion and optimization can be achieved primarily through the implementation of various strategies: production capacity expansion; new product or a new line of products launch;

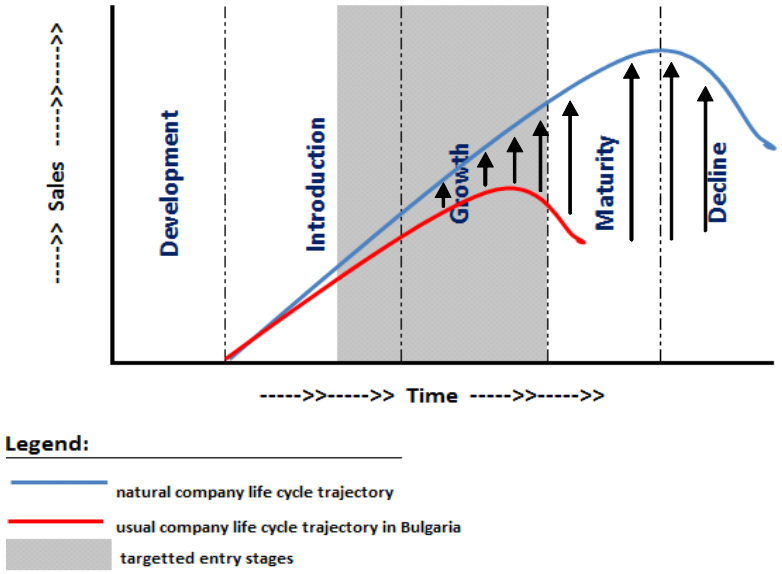


Figure 2: Expansion and optimization.

commercial network development, process improvement and efficiency.

More than 80% of the companies are managed with outdated structures, based on personal skills and single person's authority. We believe that implementation of modern business processes and process management would increase significantly profitability.

Optimization of the marketing strategy and establishment of adequate financial management will be in most of the investment cases the other substantial driver for successful expansion.

- Upgrading to the next level

The step to the next lifecycle stage of the company will be achieved by providing equity capital and financial structuring of the implementation of one or several of the following strategies:

- Organic growth for companies with interesting and multipliable business models;
- Non-organic growth, horizontal integration;
- Non-organic growth, vertical integration across the value chain;
- Creating regional leaders and consolidation plays.

4 EXPECTED NUMBER OF INVESTEE COMPANIES PLANNED INVESTMENT RATE INCLUDING FOLLOW-ON POLICY AND ENVISAGED STRATEGY FOR RISK DIVERSIFICATION OF FUND'S CAPITAL

The size of established Funds under the JEREMIE initiative should be between EUR 45-60 million, thus utilizing the whole amount available from the OP Competitiveness. The amount of the funds should be planned to be at the maximum level in order to fulfill the main targets of each Fund manager, with a main focus on:

- Fund diversification to be aimed at mitigating the various risks.
- Private investor commitment – based on the already confirmed participation by private investors (Banks, Insurance Companies, Mutual Fund and local companies) – the indication should be that the overall commitment of Private Investors will exceed EUR 30 million for each fund.
- Built – in Pipeline- the fund managers will dispose with an immediate pipeline of 15 potential deals with total investment of near EUR 70 million, which should be the base

for the first few executed deals in the first year. In our case the pipeline is partially represented by the described investment cases.

- The demand for financial growth instrument in the SME segment is at its peak. Traditional bank financing remains currently hardly accessible for SMEs, due to the ongoing cautious approach by the banks to lend investment loans with longer tenors following the continuing process of deterioration of the banks' loan portfolios. Banks are currently predominantly focusing their efforts in consumer and mortgage lending.

Therefore, it is considered that managers of each fund should be in a position to grow private equity portfolio of companies within 3 years surpassing the set target of the growth fund of EUR 60 million.

Investing in growth capital in the SME sector involves substantial risk in general and particularly in emerging markets like Bulgaria.

A significant portion of this risks results from the lack of business ethics in the market and a legislation, which doesn't support in particular this kind of investments. Several cases from the experience of international PE players in Bulgaria have shown that even a complete loss of the investments is possible due to fraud and weak legal execution. We believe that the combination between the accumulated experience in each fund's team, combined with previous successful financial deals in the local business environment and the necessary understanding of the peculiarities of the execution of financial deals in Bulgaria will be crucial for mitigating the legislative and fraud risk.

In order to mitigate the business and industry risks, it is necessary to achieve a relative diversification in stages/ types of investment, industries, size and number of portfolio companies. We believe that each fund needs to be able to invest in no less than eight companies in its total lifetime and not more than twelve at any moment of it.

The main purpose of these funds per definition is to support SME growth and not to takeover companies. Therefore the general intention under the initiative of the fund is to hold not more than 50% of the company's equity. Although, as the mentioned negative experience of other PE investors in the country shows, even as minority shareholder it is appropriate to implement irrevocable control mechanisms over the decisions process of the company's management as guarantee that the invested capital is used for its original goals. Attendance in the management board meetings of each company will be just one of these mechanisms.

Generally the management processes of the companies will be reviewed and if needed adjusted. Preferable to invest in companies that have already existing or are willing to implement modern business and management processes, which are detached and independent from individual talent skills and single persons authority. The latter is unfortunately still the business standard for the majority of SMEs in Bulgaria, and bares a high potential business risk in the cases of disloyalty of this key people. (Bulgarian Small and Medium Enterprise Promotion Agency, Bulgarian National Bank).

As funds will be investing in growth, the equity investments as a general rule should be done as a capital increase and not as a partial or full shareholders exit. Exceptions to this rule can be evaluated if one or some of the shareholders hinder the development of the company.

Considering the required experience of each developed structure and the targeted industries, the ideal investment sizes should be between EUR 1.5 million (smaller investments) and EUR 8 million (large). This numbers show the initial investment size. For follow-up capital increases funds are advised to keep special reserves of 10% to 15% of the total fund capital. Ideally, capital injections should be scheduled in tranches tied to performance and/or investment cornerstones.

The general holding period of an investment is projected to be around 5 years, depending on the industry, life cycle of the company and the general economic cycle. Overall targets should be an IRR of 18%. Some of the companies might need to be prepared for acquisition by international buyers due to the natural limitation of the local market. Such companies need to have grown to a size and stage that will make such acquisitions possible.

Additional investment rules have to be made applicable, in order to cover the principles described above:

- A single investment should not exceed EUR 10 million, and if it does, then a decision of the supervisory board will be needed. Single investments below EUR 1.5 million will be not evaluated.
- To assure diversification of companies, Top 4 investments should not exceed EUR 30 million.
- To assure diversification in the targeted industries, the limit per single industry will be 30% of one funds capital.

- A balance (50/50) between the two types of investment will be targeted.

Each fund is to aim and complete at least 3 deals from different industries and different investment types within the first year of structuring. The investment cases show a generalized summary of some of the existing projects/ deals under the specific pipeline. In the following years, performance speed should be kept at 3-4 deals per year (set as target). (Bulgarian Small and Medium Enterprise Promotion Agency, Investor.bg)

5 CONCLUSIONS

It's been proven that given the development stage and nature of the SMEs in Bulgaria the most suitable instruments created by Funds management have to be as plain and simple as possible. Sophisticated financial products generally create mistrust on the local market. Thus each Fund must intend to use for its investment needs primarily direct participation in the companies via investing in common stock and in certain cases trough a combination with investments in preferred stock of the company.

Structured Funds under JEREMIE most likely will aim at purchasing a significant portion of a particular company in order to be able to have a larger influence in its governing and to speed up its growth via the experience and know-how of its investment team. Typically Funds will seek to participate via a capital increase aiming at further strengthening the shareholder's equity, and support the continued growth through acquisitions as well as organic growth.

In order to protect its investment each Fund might seek also participation trough preferred stock as it has many advantages including a greater claim of the assets than common stock thus limiting the downside of the investment. Buying preferred stock could include the option of converting them into common stock at any point of time, in which case the owners will lose the right of a dividend, but will gain the ability to participate in the decision taking process. Preferred stocks could be flexible in terms of the dividend rates that they hold, which could be adjusted along the way so that it does not interfere with the company's sustainable growth.

In limited number of cases each Fund have to aim at lending different types of hybrid loan products, suited to best fit the business needs of each company. A common type of debt product that Funds will be looking at will be the convertible debt, where the loan is secured via the right to convert it to common stocks at certain predetermined conditions. This will reduce both the risk to each Fund and the requirement to the company to provide collateral, which as we have mentioned before proves to be a major obstacle for the SMEs on their way to receiving a proper financing.

Each structured Fund must target an investment with a clear potential to generate above 30% internal rate of return (IRR). As some of them could be expected to not realize their full potential and reach all financial targets at the predefined time horizon, managers should expect that the overall performance that one Fund will be able to achieve will be equivalent to IRR of 18%.

REFERENCES

Oxford Economic Forecasting www.oxfordeconomics.com

BNB (Bulgarian National Bank) - www.bnb.bg

Bulgarian Small and Medium Enterprise Promotion Agency – www.sme.government.bg

www.Investor.bg

Business Models and Service Applications for Traffic Management

Brahmananda Sapkota[1], Marten van Sinderen[1] and Boris Shishkov[2]
[1]*EWI – University of Twente, Drienerlolaan 5, Enschede, The Netherlands*
[2]*IICREST, 53 Iv. Susanin Str., Sofia, Bulgaria*
{b.sapkota, m.j.vansinderen}@utwente.nl, b.b.shishkov@iicrest.eu

Keywords: Business Modeling, Service-oriented Architecture, Web Services, Traffic Management.

Abstract: Managing traffic becomes more and more dependent on ICT – Information and Communication Technology, in general and particularly on ICT services support. Safety, pollution, congestion, and travel time are all important concerns that point to needs for improving traffic management and realizing this would concern the supportive services (including ICT services). Technology-independent functionality models are not only needed for better understanding the (used and/or desired) (IT) services and discussing them of full value with both developers and users but also for establishing appropriate traceability that would allow updating the underlying technology accordingly based on desired updates in the service support. That is why business models and service applications need to be considered together. Hence, in addressing service applications for traffic management, we would emphasize on the crucial role of business process analysis and technology-independent modeling. Further, service applications relate to corresponding ICT-based service platforms that provide relevant support – in the case of traffic management, it would be for example: localized monitoring and management of traffic and environmental information collected from various information sources such as sensors, surveillance cameras, and weather stations. Such information should be made available through the services in order to increase reusability, loose coupling and management of different information and their analysis. With regard to this, two significant challenges relate to service discovery and interoperability. This paper, reporting research in progress, emphasizes not only on service applications (particularly for traffic) and relations to business modeling, but also on the challenges mentioned above. In this way, we present some visions on how to better benefit from business models and IT services, for usefully improving traffic management, partially exemplifying this.

1 INTRODUCTION

The use of Information and Communication Technology (ICT) is increasingly proliferating in transport vehicles. It is applied to support the main functions like car management and supporting the driver to navigate the vehicle from A to B. We begin to see intelligent systems using sensors and actuators to prevent accidents, for example when the car is approaching other cars too closely. Other applications include information systems like navigators to guide the driver to their destination taking traffic information into account. Other ICT systems are there to entertain the travelers with music and/or video movies. The above mentioned functionalities have been developed independently of each other: car sensors, engine management, traffic information systems, entertainment, and telecommunications. A number of these developments are currently specific to the car manufacturer and therefore the brand of the car. In near future, cars (of different brands) will be able to exchange information (FIATS-M, 2012). Nevertheless, with more and more cars appearing in urban streets due to commuting and increasing transportation needs, we experience severe traffic jams, especially during peak hours resulting to: (i) increasing CO2 emissions; (ii) people spending more time in traffic jams; (iii) wasting of fuel; (iv) stress levels increasing with drivers. Managing traffic in a better way is thus crucially important currently.

Managing traffic itself becomes in turn more and more dependent on ICT services support. Safety, pollution, congestion, and travel time (as mentioned before in this section) are all important concerns that point to needs for improving traffic management and realizing this would concern the supportive services (including ICT services). Technology-independent functionality models are not only needed for better

understanding the (used and/or desired) (IT) services and discussing them of full value with both developers and users but also for establishing appropriate traceability that would allow updating the underlying technology accordingly based on desired updates in the service support. That is why business models and service applications need to be considered together (Shishkov, 2011). Hence, in addressing service applications for traffic management, we would emphasize on the crucial role of business process analysis and technology-independent modeling. Further, service applications relate to corresponding ICT-based service platforms that provide relevant support – in the case of traffic management, it would be for example: localized monitoring and management of traffic and environmental information collected from various information sources such as sensors, surveillance cameras, and weather stations. Such information should be made available through the services in order to increase reusability, loose coupling and management of different information and their analysis. With regard to this, two significant challenges relate to service discovery and interoperability.

This paper, reporting research in progress, emphasizes not only on service applications (particularly for traffic) and relations to business modeling, but also on the challenges mentioned above. In this way, we present some visions on how to better benefit from business models and IT services, for usefully improving traffic management, partially exemplifying this.

The remaining of this paper is as follows: Section 2 discusses IT services and also their relation to business modeling. Section 3 discusses the challenges as mentioned already. Section 4 outlines some envisioned solution directions. Section 5 provides partial exemplification. Finally, Section 6 contains the Conclusions.

2 IT SERVICES

In this section, we consider IT services in general (broader) and web services, in particular (these are those IT services which are delivered particularly through Internet). Let's nevertheless start from the service concept: from an abstract point of view, a service represents a piece of well-defined functionality that is available at some network endpoint and is accessible via various transport protocols and specialization formats. The functionalities provided by services cover a vast spectrum reaching from low level features like offering storage capabilities, over simple application functions like changing a customer address, to complex business processes like hiring a new employee (Alonso, 2004).

The ability to create new ICT applications from existing services, independently on who provides these services, where they are provided, and how they are implemented, would mean usefully utilizing the service perspective in application development (Van Sinderen, 2012). Such kind of application development is innovative not only because the application is not constructed from the scratch (actually, this is true also for component-based application development) but also because the development itself is fully centered around the desired end functionality to be consumed by users (this leads to service compositions and hence developers would no longer possess full control over all software components that play roles in delivering the application functionality). Hence, the application development task (as considered in general) might split into: (i) development of small software modules delivering generic adjustable services to whoever might be interested in using them, and (ii) composition of complex functionalities, by using available generic services. This all inspires new middleware developments also (Shishkov, 2011).

Furthermore, in order to be of actual use, such services would demand enabling technology standards and some recent views of Papazoglou (2008) appear to be actual in this respect. Transportation protocols are to be mentioned firstly because logically, web services' relying on a transportation protocol is crucial. Although not tied to any specific transportation protocol, web services build on ubiquitous Internet connectivity and infrastructure to ensure nearly universal reach and support. Hence, their mostly relying on HTTP (the connection protocol that is used by web services and browsers) and XML (a widely accepted format for all exchanging data and its corresponding semantics) looks logical. Having this as foundation, we have to briefly discuss three core web service standards, namely SOAP, WSDL, and UDDI: (i) SOAP (Simple Object Access Protocol) is a simple XML-based messaging protocol on which web services rely in exchanging among themselves information. SOAP implements a request/response model for communication between interacting web services. (ii) WSDL (Web Service Description Language) is a language that specifies the inter-face of a web service, providing to the requestors a description of the service in this way. (iii) UDDI (Universal

Description, Discovery, and Integration) represents a public directory that not only provides the publication of online services but also facilitates their eventual discovery. And finally, as part of the web services composition, we need to introduce some orchestration defining their control flows (Alonso, 2004), such as sequential, parallel, conditional, and so on, and to also determine complex processes that would usually span many parties. BPEL4WS (Business Process Execution Language for Web Services) can usefully support such composition activities. Finally, as the collaboration among many parties (through their web services) is concerned, a common observable behavior (choreography) would often need to be defined. CDL4WS (Choreography Description Language for Web Services) can usefully support such collaboration descriptions.

In SUMMARY: all those details related to IT (Web) services essentially point to business analysis and business modeling activities to be done as 'background'.

3 CHALLENGES

As mentioned, two essential challenges with regard to what has been discussed so far, are: (i) service discovery; (ii) interoperability.

SERVICE DISCOVERY is of crucial importance in identifying, composing, and delivering a service but still here some issues have not been convincingly resolved yet (Sapkota & Van Sinderen, 2011). In an open environment, it is difficult to support on-demand collaboration if the published service descriptions are outdated consequently providing incorrect information. This difficulty escalates when service descriptions contain limited information, i.e., some information may be relevant to the discovery of services but since this information depends on the runtime state, it cannot be included in the service descriptions (Van Sinderen, 2012). So, besides the "correctness" (or "outdated information") problem, we also have the "state-dependency" (or "dynamic information") problem that leads to poor discovery results. For more information on analyzing the service discovery challenge, interested readers are referred to (Sapkota & Van Sinderen, 2011).

SERVICE INTEROPERABILITY concerns the ability of a service to collaborate with other services (Van Sinderen, 2012). This requires that different service 'owners' have to devise their processes and agree on a shared universe of discourse, such that their respective collaboration goals can be fulfilled. Furthermore, it requires the ability of exchanging information and of using the information that has been exchanged in accordance to the collaboration goals. If the collaboration is to be supported by Information and Internet Technology, underlying automated systems send and receive messages containing user data to represent the information. Communication protocols and data formats are to be standardized to achieve syntactic interoperability (exchange of data), and ontology definitions and ontology languages have to be developed to facilitate semantic interoperability (interpretation of data). Hence, in order to achieve interoperability, business requirements and technology solutions have to be aligned.

4 SOLUTION DIRECTIONS

Taking all above in consideration, we can formulate a strategic GOAL as follows: to design and develop powerful and deployable service-oriented (road) traffic management system able to support individual mobility and network wide operations. Fulfilling this goal is claimed to be non-trivial and we have thus outlined several solution directions whose justification is left beyond the scope of this paper and whose further elaboration is considered as future work:

- considering technology-independent functionality models as 'bridges' between user demands and technological solutions;
- aligning technical solutions to the existing standards for service discovery and interoperability;
- balancing individual and group interests through sophisticated rules and regulations;
- aiming at solutions that are adequate with regard to observing privacy and security.

5 EXAMPLE

In the current exemplification section, we would consider a scenario presented at the FIATS-M'11 International Workshop (Sapkota, 2011):

Bob lives in the outskirt of Enschede with his wife and two children. He is scheduled to have a project meeting in Sofia at 11:00 PM on Friday. He

is occupied the entire day because of the kick-off meeting of his recently acquired project on Thursday. Because Bob is mostly busy with his work (delivering lectures, attending meetings, and doing research) during the weekdays, he spends his weekend with his family as much as possible. When his children know about his forthcoming trip to Sofia on Friday, they were sad that they will not see him during the weekend. So he promises his children that he will return to take them to the world-famous zoological garden in Emmen at the weekend.

He decides to travel Friday morning to Schiphol where he will take an early flight to Sofia. Since taking a train would not leave him enough time to check in, he takes his car, which is equipped. with Intelligent Route Planning (IRP) agent, radio and Global Positioning System (GPS) devices.

He books the flight accordingly and downloads his e-ticket to his smart phone. When the e-ticket is downloaded, his smart phone recognizes it and wirelessly communicates with an IRP agent installed on his car. This agent communicates with the GPS device installed on the car and determines the required travel time to reach to Schiphol airport. The IRP agent knows that Bob normally wants to arrive at the airport 30 minutes before the normal time as suggested by the airlines and thus calculates the time Bob needs to start his journey. The IRP agent communicates this information to Bob's smart phone. Bob's mart phone then uses this information and sets its alarm accordingly.

When he follows the route shown on his GPS system, he suddenly encounters that the road is blocked because of construction works. He then ignores the advice that comes from the GPS system and drives on a different road that is thus NOT suggested by the GPS system. The GPS system apparently does not know about this situation and what Bob is doing because Bob is driving on a newly constructed road; the GPS system keeps advising Bob to take a U-turn if possible and Bob keeps ignoring the advice and keeps driving using his own instinct and sense of direction. After a while, the GPS system recognizes the stretch of road that Bob is driving and recalculates the route for Bob. The route Bob has taken based on his own sense of direction turns out to be a fast solution, especially in early morning travel time. The IRP agent on his car records this newly discovered route and updates the map and broadcasts the plan to the passerby cars.

While on his way, the IRP agent installed on a car coming from the opposite direction communicates information of long jam of cars 10 KM ahead because of a recent accident to the IRP agent installed on Bob's car. The IRP agent then communicates this information to the GPS system to re-calculate the route.

When he is driving on the re-calculated route, the IRP agent communicates with the Road-Side Infrastructure (RSI) and finds out that the traffic near the next junction where Bob has to turn right is congested (the RSI can determine such a situation by using information from loop detectors). The IRP agent informs Bob to change the lane well in advance. The IRP agent also predicts, based on the current weather conditions, total number of current road users and their average speed, that the joining road ahead of the next junction could have black ice. The IRP agent then informs Bob to drive at safe speed to avoid a possible slippery road condition.

When Bob drives some 100 KM, the IRP agent receives information from the RSI that there is a poor visibility 20 KM ahead of the road and schedules the light control system to brighten their light calculating the time required to reach that spot. When Bob passes the poor visibility area, the IRP agent identifies that the visibility is OK and resets the high to their original intensity through the light control system.

When at parking lot at the airport, Bob's car recognises that his friend Dave is also at the airport, and sends him an invitation for a coffee if he has time. Dave replies with a call and they meet at a nearby coffee shop. After having a chat with his friend, Bob goes to check-in his flight and leaves for Sofia.

After the meetings in Sofia, Bob returns to The Netherlands. When he lands at the Schiphol airport, he turns his smart phone on. His smart phone then wirelessly communicates with the IRP Agent at his car. The agent then communicates with the GPS system and calculates the time required to reach his home and informs his wife Alice about his arrival time. Bob then continues his journey towards his home following the route displayed on his GPS system.

After driving 45KM, the road RSI communicates to the radio device installed on his car that the road further ahead is busy (which is expected because it is a Friday night). The RPI agent receives this information through the radio device installed on Bob's car and communicates with the GPS system to recalculate the new route and new time required to reach Bob's home. It appears that Bob will arrive home 30 minutes later than previously expected, the IRP agent then informs Alice that Bob will be late by 30 minutes because of busy traffic.

The new road that Bob is driving now is relatively empty ahead of him with just few cars behind him. When he approaches Enschede, the IRP agent communicates with the RSI and finds that an ambulance is coming on the joining road at the junction ahead and Bob will not be able to cross it safely. The IRP agent then informs Bob to slow down because the traffic light at the junction is going to turn red because of the high priority vehicle on the other road. When he starts decelerating, the IRP agent communicates with the IRP agent on the car behind Bob (which was out of the range of RSI communication) and informs that Bob is decelerating. The IRP agent on the car behind Bob then informs his driver Tim to start decelerating to avoid possible environmental pollution (noise, air) and a possible collision because the car in front is decelerating for some reason.

When the ambulance crosses the junction, RSI broadcasts the message that it is going to turn the traffic light to green because there are no other vehicles on the joining road. The IRP agent informs Bob to smoothly accelerate and move forward. Finally, when Bob arrives at home, Alice is waiting for him with a hot cup of coffee, he starts talking with Alice while drinking his coffee.

Hence, a service platform is necessary, for supporting communication between vehicles as well as between vehicles and road infrastructure through the concept of services orientation. The concept of service orientation is used to integrate various types of systems and services. It is also used for supporting interoperation between these services and systems. Furthermore, the service orientation allows us to deploy services in the cloud to achieve performance requirements such as scalability and efficiency.

Such a service platform needs to provide support for different communication protocols as well as service descriptions. To support this requirement, the service platform would provide a standard communication interface which bridges the protocol heterogeneity through the use of adapter. The heterogeneity between service descriptions can be handled by defining an intermediate description language which can allows us to define mappings without knowing the target description language. The back-end information system infrastructure is used to process the collected information and to derive useful information or the composition of services for the user.

In realizing all this, it is essential keeping full consistency between:

(i) what is desired (from user perspective) and
(ii) what is delivered (from system perspective).

We find this example and the follow up discussion as further justification of the claim that technology-independent functionality models are to adequately BRIDGE the gap between the two.

6 CONCLUSIONS

The contribution of this paper is two-fold:

(i) We establish and justify the role of technology-independent functionality models, especially with regard to the technical and technological facilitation of (road) traffic.
(ii) Analyzing strengths and challenges concerning IT services, we propose service-oriented solution directions and provide partial exemplification concerning our proposed visions.

We plan to develop further these views, elaborating them from a privacy/security perspective. The reason is that, as according to our view, without convincingly touching upon this, it would not be possible to implement and deploy such systems in at a larger scale.

REFERENCES

FIATS-M Home: http://www.fiats-m.org.

Shishkov, B.: Technology-Independent Functionality Models for Road Traffic Systems. Proc.: FIATS-M'11 (2011).

Alonso, G., Casati, F., Kuno, H., Machiraju, V.: Web Services, Concepts, Architectures and Applications. Springer-Verlag, Berlin-Heidelberg (2004).

Van Sinderen, M.: Modeling Internet-Based Goal-Oriented Enterprise Interoperability (Keynote Lecture). Proc.: BMSD'12 (2012).

Papazoglou, M. P.: Web Services: Principles and Technology. Pearson Pr. Hall (2008).

Sapkota, B. & M. Van Sinderen: Leveraging the Publish-Find-Use Paradigm of SOA, Supporting Enterprise Collaboration Across Organizational Boundaries. Proc.: BMSD'11 (2011).

Sapkota, B.: An Adaptive Service Platform for Traffic Management and Surveillance. Proc.: FIATS-M'11 (2011).

AUTHOR INDEX

Proceedings of BMSD 2012
Second International Symposium on Business Modeling and Software Design
ISBN: 978-989-8565-26-6
http://www.is-bmsd.org