A Framework for Compliance of Lgacy Information Systems with Legal Aspect

Welcome to AIS Transactions on Enterprise Systems

We are delighted to welcome this first issue of the AIS Transactions on Enterprise Systems (AIS TES). This is the first journal in the AIS Transactions series which adds to our repertoire of journals. JAIS (The Journal of the Association of Information Systems) and CAIS (the Communications of the Association of Information Systems) are more general research journals in the field whereas the Transactions series aims to publish the best research from our special interest groups (SIGs). We expect to publish the first issue of the second Transactions journal, the AIS Transactions on Human-Computer Interaction in the next few months and others will follow. As for all AIS journals and conference proceedings AIS TES will feature in the AIS eLibrary, together with an impressive array of other content, such as the MIS Quarterly and the Information Systems Journal, as well as much other content. The eLibrary is developing rapidly and is destined to become the 'one-stop shop' for all information systems academic needs. The eLibrary is accessible to search engines, such as Google and Google Scholar, and provides high visibility, ease of access, and powerful searching, to all its content. The eLibrary provides full-text access freely to all AIS members and citation and abstract access to non-members. Shortly the eLibrary will include the facility for non-members to download full-text content on a 'pay-per-view' basis. So as part of the eLibrary AIS TES will be in the best possible position to develop rapidly and become a key resource for the community.

We congratulate Norbert Gronau and colleagues in the SIG Enterprise Systems for publishing an outstanding first issue of a journal which we see quickly becoming established as the leading journal in the field of enterprise systems. On behalf of the AIS we welcome this new venture and would like to thank everyone who has contributed to making it possible.

David Avison, President AIS and
Guy Fitzgerald, VP Publications AIS

I would like to welcome the new AIS-sponsored journal AIS Transaction on Enterprise Systems. This Journal presents a valuable addition to SIG EntSys sponsored academic activities. It would would provide SIG EntSys and other AIS-sponsored SIGs with a publishing arm. Best papers from SIG EntSys- sponsored AMCIS tracks, pre-ICIS workshops, and other workshops will be fast tracked for publishing in the AIS Transaction on Enterprise Systems.

AIS Transaction on Enterprise Systems captures the values and spirit of SIG EntSys of being inclusive, developmental, open, and inspirational. The journal provides an open on-line access to academics and practitioners from all over the world to provide high quality not only double-blind peer reviewed academic contribution but also on-line contributions and comments on papers. The aim is not only to publish high quality peer scrutinised papers but also to provide a comprehensive developmental opportunity. The journal international editorial board representing five continents and 14 countries to ensure the inclusiveness and welcoming of all traditions of research on Enterprise Systems.

Enterprise Systems research plays a vital role in the current research agenda. Enterprise systems continue to represent one of the largest IT investments organisations make and is seen as

one of the prerequisites for doing business in many industries. The introduction, use and maintenance of enterprise systems (ES) is challenging to many organisations. Organizations have not always realized the benefits they anticipate from such a significant investment. I hope this new journal would extend the SIG EntSys strive to stimulate research in this significant area that affect organisations of all different sizes in most organizational sectors.

The success of any journal depends on its editors, editorial board, reviewers, and authors, but in the AIS case, the readers also play invaluable role in providing active contributions and vigorous on-line reviewing. This journal is a significant step towards exploiting web 2 technology in academia which could provide opportunities to enhance rigor and relevance of research. I do encourage members, colleagues, and friends of SIG EntSys to play an active role in the success of this journal as authors, reviewers, and active readers.

Amany Elbanna
Chair of SIG EntSys
Lecturer in Information Systems
Business School
Loughborough University
UK

As editor-in-chief of AIS Transaction on Enterprise Systems I am very delighted to present you the first issue of AIS Transactions on Enterprise Systems as a specially printed journal. The next issues will be available through the journal's website www.enterprise-systems.net.

Some time ago, I felt that a journal covering current research topics and case studies for practitioners in the field of Enterprise Systems was missing. Topics belonging to current research, state of technology, educational issues and development of enterprise systems were never combined in a single magazine. This journal shall cover the mentioned topics and areas. I strongly encourage authors who submitted papers accepted we accepted at conferences where no proceedings are available to submit their work to AIS Transactions on Enterprise Systems.

I would like AIS Transaction on Enterprise Systems be the leading journal in the field of Enterprise Systems, whereas the topics covered could expand from Enterprise Systems towards topics related to Enterprise Systems such as the relationship between general management concepts and the management of IT companies.

I would like to thank all persons and institutions involved in this endeavor. Without the support of the David Avison, president of the Association of Information Systems (AIS) and the special Interest group on Enterprise Systems (SIGEntSys), actually led by Amany Elbanna, this journal would not be like it is right now. Further on, I would like to thank all other members of the AIS and SIGEntSys for their ongoing support. My thanks are also directed to the editors of AIS Transactions on Enterprise Systems for their reviews and the authors for their contributions. My thanks are also to GITO publishing who is responsible for the distribution of this journal and I would also like to thank my employee Carsten Brockmann, who is the head of the editorial office and continually encouraged me to proceed with this journal.

I wish you a joyful reading and would be delighted to receive your contributions for one of the next issues.

Best regards

Norbert Gronau

Imprint

AIS Transactions on Enterprise Systems
www.enterprise-systems.net

Editorial office
Carsten Brockmann
University of Potsdam
August-Bebel-Straße 89, 14482 Potsdam, Germany
Tel.: 0049 331/977-3455
Fax: 0049 331/977-3406
E-Mail: editorial-office@enterprise-systems.net

Publisher
GITO Publishing GmbH
Detmolder Str. 62, 10715 Berlin, Germany
Tel.: 0049 30/41938364, Fax: 0049 30/41938367
© 2008 GITO mbH - Verlag für Industrielle
Informationstechnik und Organisation
1st volume 2009
ISSN 1867-7134

Layout: Wiebke Wegener, Wansdorf
Printing plant: vierC GmbH & Co. KG
Title graphics: Kathleen Goldacker

Contents

Available online at www.enterprise-systems.net

AIS Transactions on Enterprise Systems 1 (2009) 1, 8-14

Probing the Gaps between ERP Education and ERP Implementation Success Factors

S. Mohamed, T. S. McLaren

Abstract

This paper compares two streams of research related to ERP education and ERP implementation success factors. Many of the factors found to be associated with ERP implementation success are covered in the normative research on desirable skills outcomes for ERP educational programs. However, a gap analysis suggests several "soft skills" that are associated with ERP implementation success factors are typically overlooked. These gaps suggest that ERP education must place more emphasis on change management, organizational and employee resistance, and performance incentive schemes. These findings have implications for the design of ERP training programs and university-level curriculum.

1. Introduction

Enterprise Resource Planning Systems (ERPs) are now widely adopted within the business community. A major reason for the popularity of ERPs is their ability to integrate data and business processes throughout an entire organization [4,13]. The dominance of ERPs in industry has created a huge demand for ERP skilled employees. Business schools have responded with an ERP curriculum to deliver market-driven learning outcomes (ERP skills). However, upon comparing the recommended learning outcomes (job skills) from the normative literature on ERP curriculum with the ERP implementation success factors typically mentioned in research literature, there appear to be several gaps where the learning outcomes may not address the success factors.

This paper aims to compare the normative research on suggested skills outcomes of ERP curriculum with research into the factors associated with ERP implementation success. The goal of this paper is to determine if the skills that are typically prescribed for ERP educational programs can be mapped to ERP implementation success factors and if there are other skills required for success that are not receiving enough focus in today's ERP curricula.

2. ERP Job Skills and ERP Implementation Success Factors

In the late 1990s, there was a shortage of workers with ERP related skills [9,10]. As organizations worked to implement these complex systems, university business schools became increasingly interested in incorporating ERP education into school curriculum [5]. With the growing demand for ERP skills and the growing dominance of the technology in industry, it only seemed fitting that business schools adjust to the changing needs. The ERP vendors were also encouraged by academia's interest in ERP curriculum, as they felt that academic theory and ERP practice would further the development of their products. To this end, in 1996 SAP created their Academic Alliance Program to promote the use of ERPs within universities. Today, ERP curriculum has been widely adopted in academia for over a decade [11].

There has been a number of studies documenting how best to incorporate ERP curriculum in academic education. For example, research has been conducted to determine the best format for ERP content delivery: specific ERP package training, business process knowledge in relation to ERPs, Information Systems (IS) concepts using ERP, or teaching general ERP issues and concepts [5]. In general, the direction of curriculum can be placed into four distinct schools of thought: ERP Training, ERP via Business Processes, Information Systems Approach and ERP concepts. The fifth school of thought is a combination of the other four schools of thought known as the Hybrid Approach [9].

Of specific importance to this paper are the numerous studies that look at the specific ERP skills that recent graduates require according to industry needs [5, 6, 9-11]. In analyzing these studies a comprehensive list of skills that have been identified as most important to the success of a graduate entering into implementations or support of ERP systems can be compiled.

There has also been an extensive amount of research on the drivers of success in relation to ERP implementations. Many organizations have reported great success with the implementation of an ERP system while others have suffered difficulties in aligning ERPs with practiced business processes [1]. This paper argues that there have also been difficulties in aligning ERP educational outcomes (skills) with ERP implementation success factors.

2.1. Skills Required by ERP Graduates

Various studies have attempted to define the educational outcomes that are likely to be most important for imparting ERP graduates with the skills required for successfully working with ERPs [5, 6, 9-11]. Boyle & Strong (2006) synthesized many of the prior studies along with additional survey-based research to categorize the key skills required by ERP graduates into five main categories: ERP Technical Knowledge, Technology Management Knowledge, Business Functional Knowledge, Interpersonal Skills and Team Skills [5]. The key skills in each category are summarized in Table 1. These skills encompass a full ERP implementation from back-end to front-end skill sets, from general systems knowledge to specific ERP Knowledge, from Technical to Business Knowledge, and from Personal to Team knowledge.

In the ERP Technical Knowledge category a graduate is expected to be competent in general information technology basics such as systems analysis and data management. Also a wide range of ERP skills are listed as important from back-end technical knowledge of hardware to front-end knowledge in end-user computing support. In the Technology Management category skills listed are ones that should allow graduates to have an understanding of how ERP systems can be used to meet the strategic goals of the organization [5]. Skills from the Business Functional Knowledge category are especially useful to have knowledge in as ERP systems integrate all aspects of the entire organization [4]. ERPs are based on business processes, but business processes rely on an understanding of business functions [5], and thus enhances the importance of skills in this

Table 1: Required ERP Skills for ERP Education (Adapted from Boyle & Strong, 2006)

Skills Category	Required Skills for ERP Education
ERP Technical Knowledge	*ERP Administration *Networks *Operating systems *Systems Analysis *Systems Design/Integration *Systems Life Cycle Management *Relational Databases *ERP related programming language *Data Management *Decision Support Systems
Technology Management Knowledge	*Knowledge of ERP Concepts *Ability to learn new technologies *Ability to focus on technology as a means, not an end *Ability to understand technological trends
Business Functional Knowledge	*Knowledge of Business Functions *Willingness to learn in detail a specific business functional area *Ability to quickly understand the needs of customers *Ability to understand the business environment *Ability to interpret business problems *Ability to develop appropriate technical solutions to business problems.
Interpersonal Skills	*Ability to deal with uncertainty *Ability to accomplish assignments *Ability to write coherently *Ability to learn *Ability to deliver effective presentations *Ability to be proactive *Ability to be sensitive to organizational culture *Ability to teach others
Team Skills	*Ability to work cooperatively in a team environment *Understanding of group dynamics *Ability to plan projects *Ability to lead projects

category.

Category Name	General Description	Associated Criteria
User Knowledge and Involvement	Refers to the knowledge and involvement of employees	*Training *System understanding *User understanding *Top management involvement *Documentation
ERP Project Team and Service	Pertains to end-user relationships with internal ERP project team	*Interaction, association and conduct between ERP Project team and end-users *Communication with ERP project team and end-users *Domain knowledge and expertise exhibited by project team *Willingness and Commitment of ERP project team to support and assist user adoption of the ERP system

Table 2: User Satisfaction Criteria for Successful ERP Implementations (Adapted from Wu & Wang, 2006)

The skills in the interpersonal category are critical for a new graduate as it is important for them to be able to articulate ideas to clients and members inside and out of direct project teams [5]. Recent research has shown the importance of both interpersonal and communication skills. The importance of solid technical skills is not to be disputed but rather research is acknowledging that soft skills also play a very important role in effectively working in a IS environments [10].

Team Skills Category focuses on skills needed to be effective members or managers of an IT team. From an individual's perspective one must be able to work collaboratively in a team environment [5]. From a managerial perspective, team skills are required to reinforce effective leadership and planning. Leadership and Planning can be rolled up and discussed as Project Management skills. In prior research, Project Management skills have been ranked as one of leading skills to have when working on ERP Implementations. In fact Project Management ranked, by importance, as the number one skill among non-technical skills and number two among technical skills [10].

2.2. Success Factors for ERP Implementation

There has been a significant amount of research into the factors that are believed to lead to successful ERP implementations [1-3, 6-8, 12-14]. Although each research paper presents a slightly different view on how successful implementations are evaluated, there is a general convergence in many of the factors which determine the success or failure of an ERP implementation (which we refer to as "success factors").

Table 3: Problems in Failed ERP implementation (Adapted from Furumo & Melcher, 2006)

Technical Problems	Non-technical Problems
Team leader was not enough of an expert in the functioning of ERP systems	Team leader did not lead team effectively
Members wanted a team leader that understood all the business functions and how data elements flowed through the system	Team leader unable to bring team members together
	Team leaders concerned with lack of support from upper management
	No additional compensation for extra work required.
	Older associates resisted ERP implementation
	Older associates unwilling to adapt to change in tasks or responsibility
	IT department resisted ERP because they did not want to administer a system where they did not have full control
	IT department unhelpful in adjustment of task (i.e., refusal to reprogram standard reports leaving SQL work to individual members)

Bingi et al. (1999) identified ten critical implementation concerns that should be addressed before implementations begin [4]. The ten concerns are as follows: Top Management Commitment, Reengineering, Integration, ERP Consultants, Implementation Time, Implementation Costs, ERP Vendors, ERP employee selection, Training Employees and Employee Morale [4]. Their analysis suggests the most significant ERP implementation success factors include people-related issues such as: employee morale, employee training, employee selection, and senior management commitment.

Wu & Wang (2006) identified three critical dimensions that summarize a user's satisfaction in the ERP implementation: ERP System Products, User knowledge and Involvement and ERP Project Team and Service. [14] As this paper is not concerned with how to select the appropriate ERP system, the first category, ERP System Product, will not be evaluated further. The two remaining dimensions along with their general descriptions and associated criteria are listed and explained in Table 2.

Furumo & Melcher (2006) is an example of research into failed ERP implementations. Results from this study indicated that the failure had nothing to do with the ERP product itself, but rather the organizations social structure [8]. Problems voiced from team members of this failed project are summarized in Table 3. The majority of the complaints were related to changes in roles and tasks rather than issues with the technology itself [8] (Table 3).

Similarly, Al-Mashari & Al-Mudimigh (2003) identified five main factors that led to the failure of an ERP implementation. The five areas of concern were: Managing Scope Creep, Lack of Ownership, Lack of Change Management, Lack of Communication, Lack of Performance Measures and Isolation of IT from business affairs [1]. From this list it is shown that once again, the actual ERP system or any technical aspects were involved with the failed implementation. Recommendations from this study identified five core competencies for effective implementations. The top three are as follows: Change strategy development and deployment, Enterprise-wide Project Management and Change Management techniques and tools [1].

The preceding papers illustrate the success factors typically identified in ERP implementation success factors research. Collectively, these papers can provide a clear picture of what it takes to work successfully with ERP systems and have a successful ERP implementation. A consistent theme is that the factors associated with failure or success of a system is related both to the technical and organizational aspects of ERP implementations (see Table 4).

Although it may appear that the findings support the opinion that the importance of technical versus non-technical factors for successful implementations is about equal, in fact the non-technical aspects are mentioned much more frequently and in numerous articles compared with the technical factors. This has to do with the fact that ERPs are now quite robust from a technology standpoint while ERP project success or failure is more often related to the ability of the organization to accommodate the necessary changes to business processes organizational structure [2,12] (Table 4).

3. Gap Analysis between ERP Job Skills and ERP Success Factors

As mentioned earlier, the purpose of this paper is to determine if the skills being taught to ERP graduates are the ones that are most valued in

Technical Factors	
Reengineering Business Processes	(Bingi et al., 1999)
Integration of Data and functional areas	(Bingi et al., 1999)
Engaging ERP Consultants	(Bingi et al., 1999)
Implementation TimeLine	(Bingi et al., 1999)
Implementation Costs	(Bingi et al., 1999; Hawking et al. 2004)
Selecting ERP Vendor	(Bingi et al., 1999)
ERP System Understanding from employees	(Wu & Wang, 2006)
Domain knowledge and expertise exhibited by project team	(Wu & Wang, 2006)
Strong Business Function Knowledge	(Furumo & Melcher, 2006)
Strong ERP specific Knowledge	(Furumo & Melcher, 2006)
Enterprise-wide Project Management	(Al-Mashari & Al-Mudimigh, 2003)
Managing Scope Creep	(Al-Mashari & Al-Mudimigh, 2003)
Organizational Factors	
Top Management Commitment and Involvement	(Bingi et al., 1999; Furumo & Melcher, 2006; Wu & Wang, 2006; Al-Mashari & Al-Mudimigh, 2003)
Identify Performance Measures	(Al-Mashari & Al-Mudimigh, 2003)
Adjusted Performance Rewards	(Furumo & Melcher, 2006)
ERP employee selection	(Bingi et al., 1999)
Training Employees	(Bingi et al., 1999; Wu & Wang, 2006)
Employee Morale	(Bingi et al., 1999)
Strong Change Management	(Al-Mashari & Al-Mudimigh, 2003)
User Understanding	(Wu & Wang, 2006)
System Documentation	(Wu & Wang, 2006)
Lack of Employee And Department Resistance	(Furumo & Melcher, 2006)

Table 4: Factors Related to Successful ERP Implementations

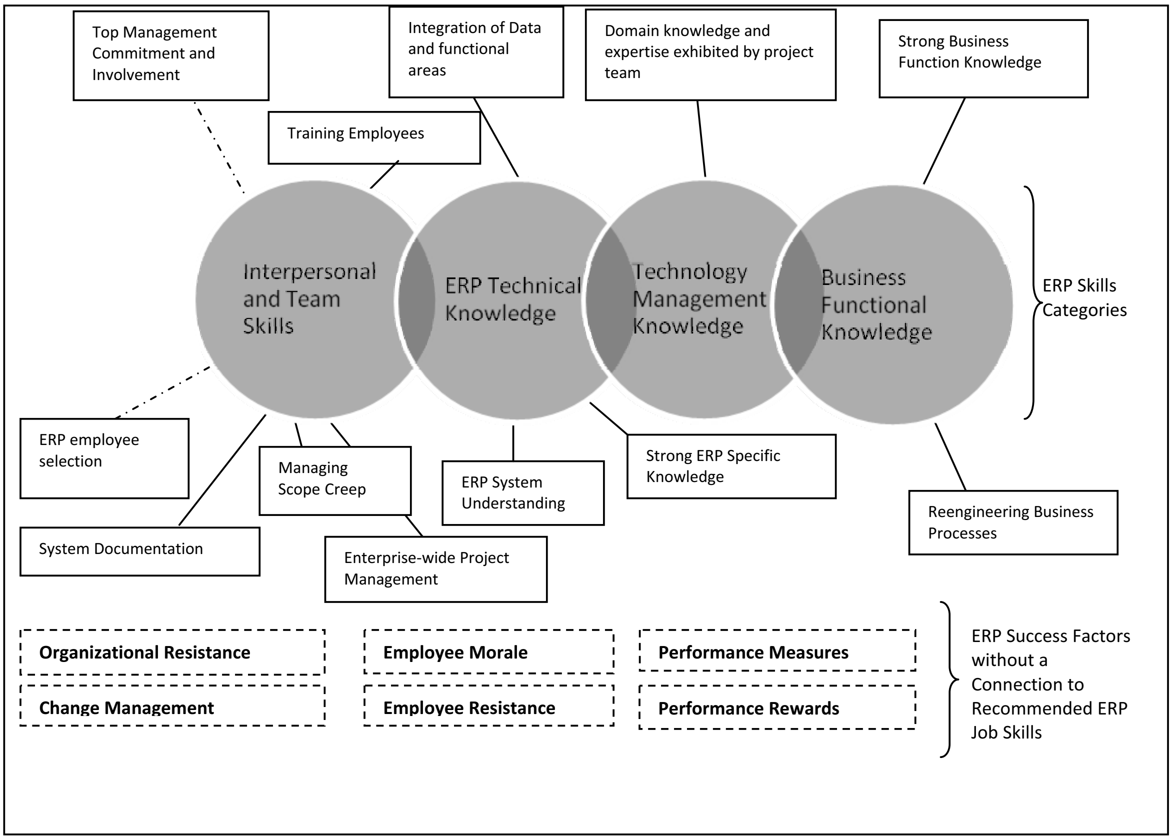

Top Management Commitment and Involvement
Integration of Data and functional areas
Domain knowledge and expertise exhibited by project team
Strong Business Function Knowledge
Training Employees
Interpersonal and Team Skills
ERP Technical Knowledge
Technology Management Knowledge
Business Functional Knowledge
ERP Skills Categories
ERP employee selection
Managing Scope Creep
ERP System Understanding
Strong ERP Specific Knowledge
Reengineering Business Processes
System Documentation
Enterprise-wide Project Management
Organizational Resistance
Employee Morale
Performance Measures
Change Management
Employee Resistance
Performance Rewards
ERP Success Factors without a Connection to Recommended ERP Job Skills

Fig. 1: Authors' Mapping of ERP Implementation Success Factors to ERP Job Skills

successful ERP implementations. Therefore, we will now analyze our finding on the skills being taught in ERP curriculum with the key factors for success we analyzed in the second half of the paper. Figure 1 is a graphical representation of the five main skills categories from Table 1, drawn with lines connecting the skills to the associated success factors from Table 4. A solid line represents a success factor that was exactly listed in one of the skill sets, representing a direct match. A dotted line represents a critical success factor that would logically fit within that skills category, but was not listed as a specific skill in the skills list from Table 1.

The technical factors for ERP success directly map to an identified skill category, as depicted by the solid lines. As for the organizational or non-technical factors for implementation success, only two directly aligned, via solid lines, with specific skills in the Interpersonal and Team skills category. Skills are typically taught that would allow graduates to Train employees and Write clear documentation material. The Interpersonal and Team Skills Category is also connected via dotted lines to two other factors for success: Top Management Commitment and Success as well as ERP employee selection. These have been made dotted lines based on a few assumptions. First, graduates being taught ERP specific skills both technical and organizational will be better equipped to select other team members that display the skill set needed to be successful. Second, as many ERP graduates move directly into implementation and support roles, they will likely have greater influence on top management to show the level of commitment expected from employees. (Figure 1)

What is apparent from this chart is that there are a number of success factors that do not map directly or indirectly back to any of the four skill categories mentioned. The following are all success factors that do not link to any of the skills categories: Department Resistance, Strong Change Management, Performance Measures, Employee Morale, Performance Rewards, and Employee Resistance.

These 'outcast' success factors can in fact be grouped into two separate areas of skill deficiencies. The two groupings would be Change

Management and People Management. Under Change Management the following 'outcast' success factors would be grouped: Department Resistance, Strong Change Management and Employee Resistance. Under People Management the remaining 'outcast' success factors would be grouped: Performance Measures, Employee Morale and Performance Rewards. This would suggest that there is a current skills gap in ERP education that does not adequately address Change Management or People Management skills.

4. Conclusions and Discussion

This paper compares two streams of research related to ERP educational outcomes and ERP implementation success factors. Many of the factors found to be associated with ERP implementation success appear to be covered in the normative research on desired job skills for ERP training programs. However, a gap analysis suggests several "soft skills" that are associated with ERP implementation success factors are typically overlooked. These success factors require ERP education to place more emphasis on Change Management, organizational and employee resistance, and performance incentive schemes.

This analysis would indicate that universities offering ERP education re-evaluate their curriculum to ensure that both Change Management and People Management skills are being instilled into their graduates, as both these areas present a large number of non-technical aspects of successful ERP implementations. Prior research has provided us with a solid list of skills found to be most important for ERP graduates. The breadth and depth of these skills change based on the type of curriculum and intensity of study. For example, business students specializing in the ERP field would be expected to display the majority of skills listed in Table 1. Whereas a student studying ERP as a course in an overall IS degree would be expected to have much fewer ERP skills. Prior research also documents critical success factors for ERP implementations. These factors arise from both successful implementations as well as failed implementations, and they are summarized in this paper to produce a list of areas that should be address to ensure success.

This paper has attempted to reconcile the skills that are currently recommended for graduates of ERP training with the skills that are identified as precursors to ERP success. Our analysis has shown that technical factors for ERP success receive adequate attention in the normative literature on ERP curriculum design. Interpersonal and Team skills receive similar attention; however, the most significant gap appears to be in the areas of Change Management and People Management skills. These last two "soft skills" are frequently emphasized in the literature on ERP success factors, but are not as prominent in the literature on ERP curriculum design. We suspect that, in turn, many ERP educational programs do not place much emphasis on these critically important skills, perhaps because they require students to have first become more proficient in the more technical aspects of ERP. However, as ERP technologies continue to mature and become more standardized, configurable, and commoditized, we argue that ERP education should now divert some of its previous technical focus onto the complex issues of ERP change management and people management. While ERP technical skills can easily be outsourced to low cost service providers, ERP change management and people management skills will continue to be in high demand and command premium wages in the job market. To remain relevant, ERP education must continue to deliver a mix of technical, business, managerial, and interpersonal skills, but the relative emphasis on each needs to be re-evaluated as the demand for different skill sets evolves.

Our preceding analysis has some limitations. First, in most curriculums, ERP courses are not taught in isolation from other business courses. It might be possible that these skills are being delivered to graduates via other courses in the program. Second, it is not feasible to create an exhaustive list of all the skills delivered through every ERP course. Our list of skills is only a preliminary comparison of the most generic ERP curricula. Regardless, there is a clear need for ensuring the right skills are delivered in ERP education. It is our hope that this paper will stimulate further assessment of ERP curricula and lead to new insights into how we can more closely align excellence in ERP implementations with excellence in ERP education.

Acknowledgements

The authors wish to thank Barry Au, the editorial board, and the anonymous reviewers for their assistance and suggestions for improving this manuscript.

References

[1] Al-Mashari, M., & Al-Mudimigh, A. (2003). ERP implementation: Lessons from a case study. Information Technology & People, 16(1), 21-33.

[2] Aladwani, A. M. (2001). Change management strategies for successful ERP implementation. Business Process Management Journal, 7(3), 266-275.

[3] Amoako-Gyampah, K. (2004). ERP implementation factors: A comparison of managerial and end. Business Process Management Journal, 10(2), 171-183.

[4] Bingi, P., Sharma, M. K., & Godla, J. (1999). Critical issues affecting an ERP implementation. Information Systems Management, 16(3), 7-15.

[5] Boyle, T. A., & Strong, S. E. (2006). Skill requirements of ERP graduates. Journal of Information Systems Education, 17(4), 403-412.

[6] Esteves, J. and V. Bohorquez (2007). An Updated ERP Systems Annotated Bibliography: 2001-2005. Communications of the Association for Information Systems (19)18, 386-446.

[7] Ferratt, T. W., Ahire, S., & De, P. (2006). Achieving success in large projects: Implications from a study of ERP implementations. Interfaces, 36(5), 458.

[8] Furumo, K., & Melcher, A. (2006). The importance of social structure in implementing ERP systems: A case study using adaptive structuration theory. Journal of Information Technology Cases and Applications, 8(2), 39-58.

[9] Hawking, P., McCarthy, B., & Stein, A. (2004). Second wave ERP education. Journal of Information Systems Education, 15(3), 327-332.

[10] Kim, Y., Hsu, J., & Stern, M. (2006). An update on the IS/IT skills gap. Journal of Information Systems Education, 17(4), 395-402.

[11] Sager, J., Mensching, J., Corbitt, G., & Connolly, J. (2006). Market power of ERP education-an investigative analysis. Journal of Information Systems Education, 17(2), 151-161.

[12] Skok, W., & Döringer, H. (2001). Potential impact of cultural differences on enterprise resource planning (ERP) projects. EJISDC, 7(5), 1-8.

[13] Willcocks, Leslie P. and Richard Sykes (2000). “The Role of the CIO and IT Function in ERP.” Communications of the ACM 43(4): 32-38.

[14] Wu, J. H., & Wang, Y. M. (2006). Measuring ERP success: The ultimate users’ view. International Journal of Operations & Production Management, 26(8), 882-903.

Available online at www.enterprise-systems.net

AIS Transactions on Enterprise Systems 1 (2009) 1, 15-26

A Framework for Compliance of Legacy Information Systems with Legal Aspect

Abdelaziz Khadraoui, Michel Leonard, Thanh Thoa Pham Thi, Markus Helfert

Abstract

Institutional activities are governed by legal aspects represented by a set of laws that regulate their execution. Therefore, the compliance between laws and Information Systems (IS) is a crucial issue for organisations, not only at the moment of IS development but also in the whole life cycle of IS. At present, compliance issues with legal aspects typically rely on experiences. The reference of the laws is manually done by business stakeholders and IS designers. In other words, the compliance with legal aspects are not explicitly addressed in the IS engineering domain. Our research aims to develop a comprehension framework for compliance of legacy IS with laws. In this framework, the abstract description (or conceptual specifications) of legacy systems are analysed in order to detect incompliant elements with laws. It is based on an ontology that is extracted from laws. The approach helps to improve legacy IS and IS reengineering and thus helps to meet user expectation.

Keywords: legacy information system, modelling, method, compliance, ontology, laws, IASDO model.

1. Introduction

Most activities in organizations are governed by a national or international legal framework that includes a set of laws stipulating lawful activities, procedures or rules for organizations. Information systems (IS) support organizations' activities. Consequently, IS are affected by these laws and legal aspects. All regulations in laws can be considered as a part of user requirements, which are considered in IS development. Ignoring these legal aspects may cause incomplete and/or inaccurate requirement specification. Therefore, the developed IS will not meet expectation from users and other stakeholders. Reviewing related literature shows that one of the main causes of IS failure concerns the requirement specification. McManus and Wood-Harper in [18] for instance identified that one of three main indicators of IS project failure is "no clear project requirement definitions". In [4], Chris Sauer underlined the first cause in five main indicators of information project failure concerns "the definition of the project's scope and original requirements". According to [3], one of three main reasons of information project failure is "ignoring the importance of requirement definition".

Therefore, the compliance of IS with laws is a very important issue in IS engineering. Indeed, meeting the user needs and complying with legal requirements contributes to the IS success.

In order to systematically obtain user requirement definitions, literature propose several methods for requirement engineering. Examples of such approaches include goals-based, scenario-based or goal-scenario-based methods [14],[20],[31],[26],[28]. However all of these methods do not mention the conformity of requirement specification with the legal aspect. Furthermore, actual methodologies for IS development such as Merise [29], RUP method of IBM, Agile methodologies [5] so far do not include a systematic model or framework for referring the legal aspect in the IS development process. All of these methods implicitly consider that referencing laws is manually done by business stakeholders and IS analysts. Certainly, due to the lack of a methodology or systematic approach this is mostly based on experiences.

On the other side, legal aspects are described by law texts, which contain concepts, rules, roles and constraints governing the related institutional domain. The exploitation of these sources of knowledge allows to enhance the IS adequacy and compatibility with institution activities.

Recently an innovative approach for IS engineering based on a law ontology is developed [12]. This approach allows extracting knowledge from laws. This is described by law-based ontology. It includes the initial stage of IS development to discover informational concepts as well as rules in the related IS. This approach helps to develop an IS in co-ordination with systematic compliance with the laws and the laws evolution.

However the approach has also its drawbacks. Indeed the approach has not yet taken into account the compliance of existing or legacy IS with laws. Meanwhile it is important to evaluate that existing IS are in compliance with legal aspects. In order to improve IS reengineering, it is important to discover the root causes of the defects.

Several research and industrial efforts focused on creating new and more efficient development methodologies. Many authors propose processes to increase the quality of the legacy applications in order to adapt the legacy IS to the new changes [7],[17],[25]. However, adapting the legacy system with the laws aspect is not addressed.

The informational aspects of IS are described by conceptual models that are the results of the modelling process in the system development cycle. Therefore, controlling the compliance of an existing IS with laws can be based on the conceptual models of that IS.

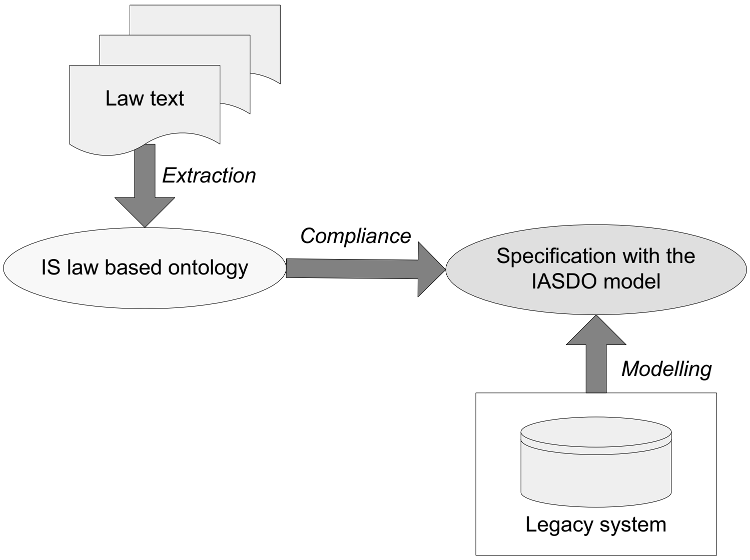
In this paper, we propose a framework for compliance of legacy ISs with the laws. Our frameworks is based on an ontology extracted from the law texts. Firstly the laws of the related domain are studied; a laws-based ontology model is produced. The legacy system is studied and then subsequently reanalysed to produce its conceptual models. Concepts and rules defined in the ontology model allow to detect incompliant elements in the conceptual models of the legacy system.

The paper is organized as follows. Section 2 presents our proposed framework for the compliance of legacy IS with laws. Section 3 explains laws-based ontology concepts in our approach and the method for extracting ontology from laws. Section 4 presents the IASDO model (Integrated Aspects of Static, Dynamic and Organization) used for describing informational aspect of legacy IS. We also summarize the main reasons for using the IASDO model in our approach. Section 5 proposes guidelines for analysing legacy IS to detect incompliant elements with the laws aspect. It is followed by an illustration of our framework with a case study. In section 6 we present our conclusion and further research.

2. Framework for compliance of legacy IS with laws

This section presents our framework for compliance of legacy IS with laws. The laws-based ontology is used as a knowledge source to compare with the specification of the legacy system to detect defects in that system. The framework has three main tasks as follows “Figure 1”.Figure 1. Proposed framework for the compliance with laws of legacy IS

Fig. 1. Proposed framework for the compliance with laws of legacy IS



2.1 Extraction of ontology from laws

In the context of IS engineering, we define an ontology model as a conceptual information model that describes some specific domains in terms of concepts, facts and business rules. An ontology model is a reference model to support information interoperability and to share information: (1) it supports human understanding of the domain under consideration and communication; (2) it facilitates interoperability across different parts of IS.

The meaning of ontology considerably evolved from its origins in philosophy to its current usage in IS. While ontology in the philosophical sense roughly means a categorization of all the entities that exist in the world and the relationships between them, ontology in the IS sense is only considered as a limited universe of discourse [32]. In our research, laws are considered as universe of discourse for IS engineering.

Laws describe concepts, business rules, roles and constraints governing the given institutional domain. The exploitation of these sources of knowledge allows to enhance IS adequacy and compatibility with institution activities and to find stable common information for IS engineering in perspective of sustainable development. The concepts and business rules extracted from the appropriate laws are used to build the ontological aspects of the corresponding domain.

Law based ontology is a new approach for IS engineering that allows establishing and clarifying the links between laws and IS, in particular the alignment between the amendment of laws and the evolution of IS. This link is established by ontology so-called "Laws-based ontology". In other words, we use laws as a source of knowledge to analyse and construct the ontological level of an institutional domain.

IS ontology model extracted from laws provides a reference for IS designers to discuss and understand the ways in which they view and interpret the institutional domain. It allows to conduct the evolution of the IS, which has been adapted to the evolution of the legal texts. Furthermore, the IS ontology level remains independent from technologies and business practices.

2.2 Description of legacy systems with the IASDO model

The second task in the framework concerns the description of informational aspect of legacy IS with a conceptual model. In this task the informational aspects of legacy IS are analysed and captured with the support of reverse-engineering techniques [1],[2],[16],[19].

In order to achieve a thorough understanding of the different aspects of the system, business professionals and domain experts are consulted. We do not aim to propose a method or technique to reverse engineer the legacy system at this phase, which is out of scope of the paper and subject of future research.

We do not use the ontology model to represent the legacy IS, as the ontology model presents knowledge and semantic concepts without a distinction of static and dynamic concepts. In other words, the ontology model is at a very high abstraction level. Meanwhile in the legacy system, concepts of static, dynamic and organization are clearly distinguished and implemented. We will describe this system considering the distinction of these concepts.

The chosen conceptual model should enable us to describe a complete view of the legacy IS including data, process and organizational aspects in a consistent and precise way. Popular languages and models such as UML [6], Entity-Relationship model [24], Data Flow Diagram [8], ORM [9], EPC [11] need several models to capture different aspects of IS. Some models just focus on one aspect of IS without taking into account interrelation of them. As a consequence consistency checks between the diagrams/models representing different aspects of IS are not included. Furthermore a global view on that legacy system is not provided.

In order to represent a legacy IS in such a way, we use a comprehensive approach: the IASDO model. The IASDO (Integrated Aspect of Static, Dynamic and Organization) model [21],[22] allows to describe an IS with static, dynamic and organizational concepts such as data structure, relationships of data, the transaction/processes on data, organizational role as well as interrelations among these concepts. Therefore this model allows to obtain a global view of the legacy IS as well as a precise and consistent specifications.

2.3 Analysing the compliance of legacy IS with laws

This task concerns comparing the specification of legacy IS and the laws-based ontology model according to provided guidelines and then detecting incompliant elements in the legacy IS. Analysing the compliance of legacy IS with a legal framework has two objectives: the study of the information completeness and the study of the information correctness (i.e. information accuracy and consistency) of the legacy IS regarding the ontology model.

- The information completeness means that the legacy IS specifications do not miss statements and elements (concepts, business rules, organizational role, relations) described in laws. These elements are mandatory and relevant for the institutional domain.
- The information correctness means that the contents of the legacy IS specifications are accurate and consistent as they are defined in laws.

The analysis of compliance of legacy IS with laws is based on the correspondent elements of the laws-based ontology model and the specification of the legacy IS (specification with the IASDO model). Determining elements of each specification is based on the correspondent meta-models.

In the following sections, we describe in details each component in our framework.

3. Laws-based ontology

As mentioned in the above section, the laws describe concepts, business rules and roles that govern the institutions activities. These elements are extracted and represented by an ontology model that is considered as the informational kernel applied to the institutions activities. In this paper, we illustrate our approach with the laws Geneva Law K 4 20.06 http://www.geneve.ch/legislation/rsg/f/rsg_k4_20p06.html (legal text in French). This law describes the procedure which medical doctors must follow to file a request of authorization in order to prescribe a narcotic for the treatment of a dependent person (drug addict).

The doctor must obtain an authorization from the cantonal doctor before the prescription of any narcotic. The law also describes how the drug will be distributed and administered. The pharmacist, on the basis of authorization delivered by the cantonal doctor, provides the doctor, or directly the patient with the prescribed drug. Table 1 and Table 2 present the articles 2 and 5 extracted from the Geneva Law K 4 20.06.

Table 1. Article 2 extracted from the Geneva Law K 4 20.06

> **Article.2 Authorization Request**
>
> 1 The department of social action and health (hereafter: the department) is designated to enact instructions addressed to the medical profession that regulate the modalities of general practitioners' responsibility for the administration of narcotics for dependent people.
>
> 2 The medical doctor who prescribes a **narcotic** to a **drug addict** must primarily obtain the **authorization** of the **cantonal doctor**._
>
> 3 The **doctor**, on presentation of an official identity card and a provided photograph, must fill an **authorization request** within following information:
>
> a) last name, first name, complete address of patient;
>
> b) complete date of birth;
>
> c) identity card number and nature;
>
> d) name of the considered substance;
>
> e) modality of administration, dosage and intended duration of cure;
>
> f) name of the pharmacy who provides the product.
>
> 4 The doctor annexes to the authorization request, in each case, the signed documents attesting the respect for instructions mentioned in the subparagraph 1.

> **Article. 5 Recourse procedure**
>
> 1 Within two working days after receiving the authorization request of general practitioners, the cantonal doctor must address his (her) agreement or refusal to the general practitioner and to the indicated pharmacist.
>
> 2 The general practitioner can make recourse within 8 working days after receiving the refusal notification of the cantonal doctor in accordance with the department which comes to a conclusion about the notice of Commission on Healthcare Profession (hereafter: commission).
>
> 3In the event of urgency, the notice of the commission can be given by its sub-commission A.

Table 2. Article 5 extracted from the Geneva Law K 4 20.06

3.1 Laws containing stable and invariant concepts

Laws evolve, however this does not mean that everything in laws modelling is variant. The major conceptions of the legal system are remaining relatively stable over decades of legal practice and jurisprudence. For example, the Geneva Law K 4 20.06 contains concepts {Doctor, Patient (Drug Addict), Authorization Request, Decision, Cantonal Doctor, Drug Prescription, Drug Distribution, Drug Administration, Pharmacist, Cantonal Pharmacist, Accepted Request, Examined Authorizations, Refused Request, Notification Refusal, Appeal} that are stable and invariant. In the case of the domain of the prescription of narcotics (drug) intended for the treatment of the dependent person, these concepts are constantly in the centre of institution activity.

3.2 Laws containing

The business rules are used to assist organizations in better achieving goals, communicate between principals and agents, between the organization and interested third parties, demonstrate fulfilment of legal obligations, operate more efficiently, perform analysis on current practices. Laws contain business rules. Some aspects of these business rules will be expressed with constraints in form of integrity rules in the IS. Their role is to preserve the coherence, correctness and consistency of an IS during its exploitation.

For example, the Geneva Law K 4 20.06 enumerates different cases where patients can benefit of the administration of any narcotic. It describes and specifies the conditions of acceptance or refusal of a request. This law describes also the appeal procedures.

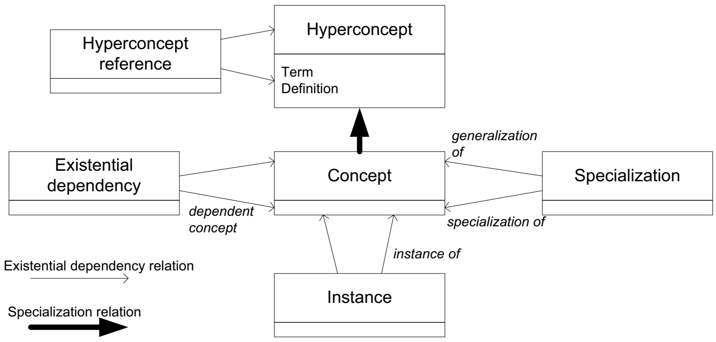
For example, article 5 of the Geneva Law K 4 20.06 describes business rules for specifying the appeal procedure in the case of refusal of the authorization request by the cantonal doctor.

3.3 Laws describing role of persons

An organizational role represents a set of necessary responsibilities, authorities and capabilities to perform the activities of the development process or to survey the activities performed by the other organizational roles [15].

The laws that specify ontological spaces are associated with roles. Each ontological space contains necessary information for performing activities by a certain role. For example, the Geneva Law K 4 20.06 describes different ontological spaces such as the ontological space of a Doctor (general practitioner), the ontological space of a Cantonal Doctor, the ontological space of a Pharmacist, the ontological space of a Cantonal Pharmacist, and the ontological space of an Appeals Commission.

Fig. 2 presents the conceptual basis for hyperconcept representation.



3.4 Representation of laws-based ontology

Figure 2 represents our meta-model for the representation of laws-based ontology. The laws-based ontology model is built from one or several hyperconcepts (Hcp).

As shown in Figure 2, a hyperconcept is constructed on a subset of concepts extracted from laws, forming a unity with precise semantics. The ontology model is represented by an oriented graph where nodes are concepts and edges are links between concepts. A concept is described by a term and a definition. A term is a way to designate the concept in the language of the expert of the corresponding domain. A concept can be the generalization or the specialization of another concept. The generalization-specialization relationship is useful to classify concepts according to their common properties or their specificities. A concept C1 depends existentially on a concept C2 if the existence of C1 is related to the existence of C2: if the concept C2 disappears then the concept C1 disappears too. In our model, a concept can be an instance of another concept. The instance, itself is considered as a concept.

In the ontology model, there is no distinction of static concepts (e.g. concepts correspond to objects in the real world) and dynamic concepts (e.g. concepts correspond to object behaviours, processes in the real world). As discussed above, the model has three types of links: (i) instantiations, (ii) existential dependencies and (iii) generalization-specialization links. The hyperconcept schema must satisfy a set of conformity rules including connectivity and completeness.

The connectivity guarantees that each concept of a hyperconcept is related to at least one other concept in the same hyperconcept. In this case, the hyperconcept represents a homogeneous zone and not a discontinuous unit.

If a concept C1 belongs to a hyperconcept Hcp and is linked to a concept C2, then C2 belongs to Hcp too, this rule concerns the completeness of a hyperconcept.

3.5 Method for extracting ontology from laws

The process model for the construction of ontology from laws is expressed as a map (i.e. strategic guideline) [27]. The

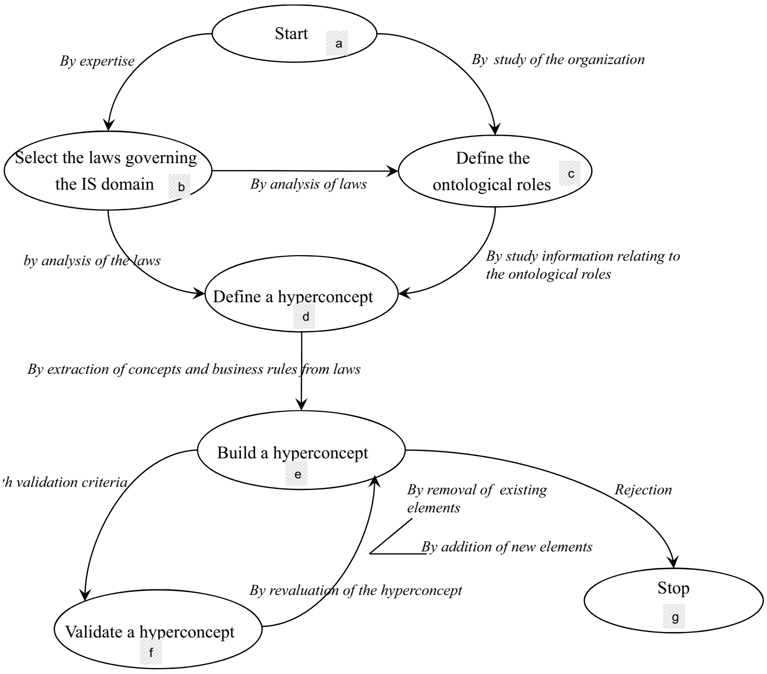

Start
a
By expertise
By study of the organization
Select the laws governing the IS domain
b
By analysis of laws
Define the ontological roles
c
by analysis of the laws
By study information relating to the ontological roles
Define a hyperconcept
d
By extraction of concepts and business rules from laws
Build a hyperconcept
e
h validation criteria
By removal of existing elements
Rejection
By addition of new elements
By revaluation of the hyperconcept
Stop
g
Validate a hyperconcept
f

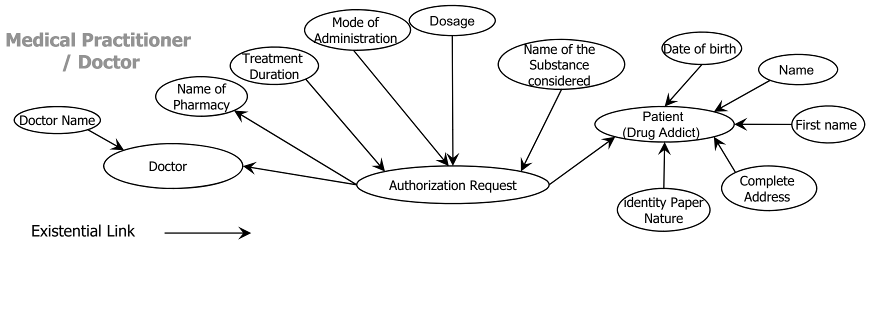
Fig. 3. The process of construction of ontology from laws

nodes represent the intentions and the links between the nodes represent the strategies. An intention indicates the goal to reach and a strategy specifies the manner with which the intention can be carried out.

Figure 3 specifies the processes of the proposed method for extracting ontology from laws. This map comprises five intentions called (i) Select the laws governing the IS domain, (ii) Define the ontological roles, (iii) Define a hyperconcept, (iv) Build a hyperconcept and (v) Validate a hyperconcept.

In [12], we proposed guidelines and method components for the extraction of the laws-based ontology. Each guideline is composed of a set of more detailed sub-guidelines or on the contrary is a part of some more complex guidelines.

Fig. 4. The hyperconcept “Medical Practitioner / Doctor” constructed from the Geneva Law K 4 20.06, Article 2



Using the ontology model and applying this process, the article 2 of the Geneva Law K 4 20.06 enables us to construct the hyperconcept “Medical Practitioner / Doctor” as described in the following (Figure 4):

4. The IASDO model

In the following we present our modelling approach, the IASDO model (Integrated Aspects of Static, Dynamic and Organization) used for describing informational aspect of legacy IS. The IASDO model consists of the following concepts:

- Class: a class has attributes and one or several identifiers formed by attributes of the class.
- Transaction: represents a process, a task or a decision in organizations. A transaction processes information in its input classes and produces information in its output classes. A transaction has pre-condition and/or post-condition describing the rules of transaction execution regarding input information coming from its input class(es) or output information going into its output class(es).
- Role: a role represents an organizational unit or a person in the organization who is assigned responsibilities or functions contributing to common activities and goals of organizations.
- Existential Dependency relation: an existential relation (oriented link) between a predecessor class and a successor class means for each object in the successor class, its existence depends on the existence of one and only one object in the predecessor class.
- Dynamic Specialization relation: a dynamic specialization relation is different from a traditional specialization / generalization relation by dynamic nature of the relation; it allows describing that an object may dynamically gain new classification (become an instance of a class) or loose a classification (is not an instance of a class). This depends

on the state change of the object through its life, after an execution of a transaction or a decision. For instance, an "authorization request" becomes the "accepted request" or "refused request" after a decision is made. Therefore, the "accepted request" class and the "refused request" class are sub-classes of the "authorization request" class, each sub-class is linked to its super-class by a dynamic specialization link.

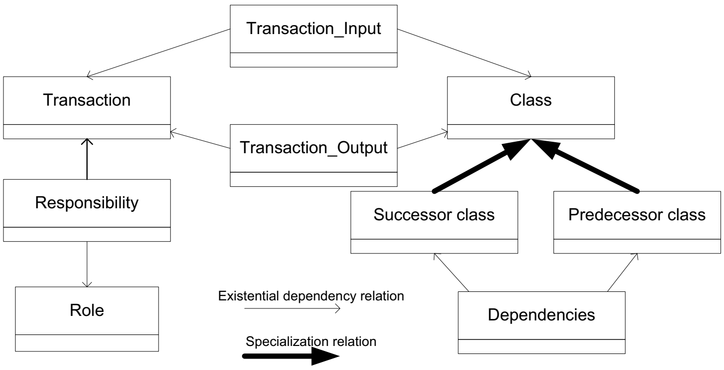


Fig. 5. A simplified Meta-model of the IASDO model

- Dataflow: an oriented link between a class (or transaction) and a transaction (or class), then the class is an input (or output) class of the transaction.
- Responsibility: describes the responsibility of roles on transactions.
- Privilege: describes the privilege of roles on classes. A privilege may be creation, suppression, modification or consultation of information/object of a class.

Figure 5 presents a simplified meta-model of the IASDO model.

A specification with the IASDO model includes the graphical presentation and the textual description, if necessary. The graphical presentation of the IASDO model is a bi-partite graph within two kinds of nodes: class and transaction. The textual description is necessary for the description of the pre/post-condition of a transaction, as well as the description of particular rules.

5. Analysing compliance of legacy information systems with laws

This section presents in detail the guidelines for analysing the incompliant elements of legacy system with laws. Subsequently a case study is presented to illustrate the proposed framework. Finally evaluate the framework and conclude with a discussion of the findings.

5.1 Guidelines for analysing

Suppose the laws-based ontology is Spec1 = {e_{11}, e_{12}, …e_{1n}}, each e_{1i} , i ∈1.. n , may be a concept, a link between concepts, a role, a hyperconcept or a rule; and the specification with the IASDO model is Spec2= {e_{21}, e_{22}, …, e_{2m}}, each e_{2j}, j ∈1..m, may be a class, an attribute, a transaction, a role, a data flow, a link between classes, a responsibility of a role for a transaction (i.e. link between a role and a transaction), a privilege on class (i.e link between a role and a class), or a rule.

We have then the analysing process consisting of four following steps:

- *Step 1: Identifying correspondent elements between Spec1 and Spec2.*
  Table 3 illustrates the correspondent elements of the two models at the meta (construct) level. Based on this table, the correspondent elements of the two specifications are identified.
- *Step 2: Identifying missed concepts in Spec2*
  This step concerns identifying all elements e1i in the ontology specification corresponding the categories defined in Table 3 so that e1i does not correspond to any element e2j in the IASDO specification. An element in Spec2 is missed then relationships come from/to this element are missed too. This step allows detecting incomplete elements in Spec2.
- *Step 3: Identifying missed and incorrect relations in Spec2*

Table 3. Correspondent concepts of the ontology model and the IASDO model

Concepts of the ontology model	**Correspondent concepts of the IASDO model**	**Correspondence definition**
Concept	Class Attribute Transaction	A concept e_{1i} in Spec1 may correspond to a class, an attribute or a transaction e_{2j} in Spec2. e_{1i} corresponds to e_{2j} means e_{1i} and e_{2j} describe the same thing in the real world (i.e. the same semantic).
Instantiation link	Rules	An instantiation link of Spec1corresponds to a rule in Spec2 if they have the same semantic of the domain application.
Roles	Organizational role	A role in Spec1 corresponds to a role in Spec2 if they have the same semantic.
Hyperconcept	-	
Rules	Rules	A rule in Spec1 corresponds to a rule in Spec2 if they have the same semantic.

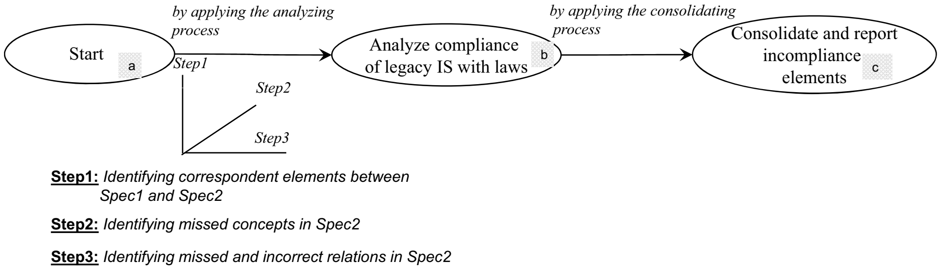

by applying the analyzing process
by applying the consolidating process
Start
a
Step1
Step2
Step3
Analyze compliance of legacy IS with laws
b
Consolidate and report incompliance elements
c
Step1: Identifying correspondent elements between Spec1 and Spec2
Step2: Identifying missed concepts in Spec2
Step3: Identifying missed and incorrect relations in Spec2

Fig. 6. Process of analysing the compliance of legacy IS with laws

This step concerns identifying missed and incorrect relations in Spec2 that is based on existing relations in Spec1. In this context a relation is a link between two concepts such as an existential dependency relationship, a dynamic specialization relationship, an input/ output data flow, an attribute of class. This results in a corresponding link between an attribute concept and a class concept in the ontology model. We also included responsibilities for roles of transactions. Applying this algorithm to identify missed and incorrect relations in Spec2 results in:

For each element e_{2i} in Spec2 corresponding to an element e_{1j} in Spec1:

- Search all relations come from/to e_{1j} in Spec1 and see if each relation is described the same in Spec2 (i.e. the same type, the same direction and the same source and destination concepts).
- A missed relation in Spec2 is a relation that exists in Spec1 but does not exist in Spec2.
- An incorrect relation in Spec2 is a relation that is not correctly described (i.e. the same type, the same direction, the same source and destination concepts) regarding the description in Spec1.

After applying this algorithm, a set of missed relations and incorrect relations are identified.

For example, this is an incorrect responsibility situation: if a concept e1i in Spec1 corresponds to a transaction e_{2k} in Spec2, e_{1i} belongs to an ontological space of a role e_{1j} then e2k must under responsibility of a role e_{2l} corresponding to e_{1j}, otherwise there is an incorrect role situation.

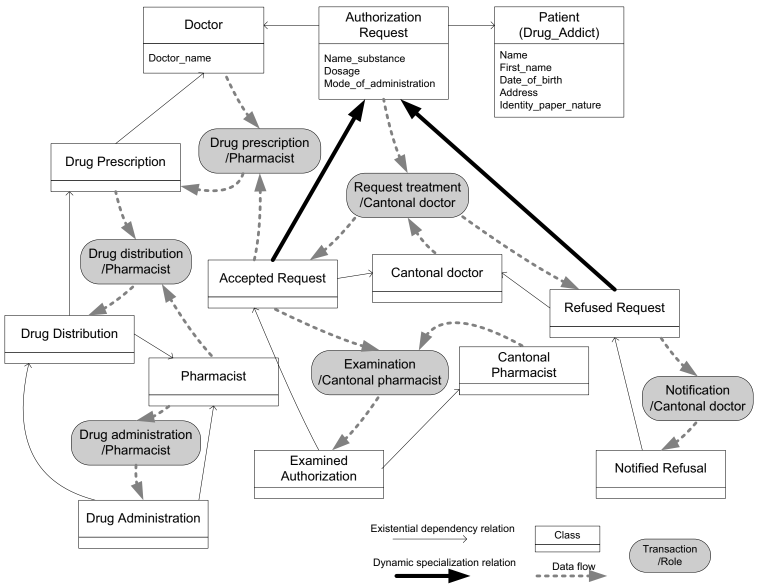
Note that, as discussed previously, the ontology model extracted from laws allows deriving the informational kernel of information systems in related domain activities. However, each organization in the same domain activities may have their own and supplement rules, activities and information. Therefore, incorrect relations in some cases may be supplement parts of the organization application. It needs a validation with the semantic of concrete applications to determine if the incorrect relations are really incorrect.

- *Step 4: Consolidate and Report incompliant elements*

  The incomplete and incorrect elements in the legacy system will be reported and interpreted in the following ways.

- Missing a concept may correspond to missing an attribute, a class, a transaction or a role in a legacy system.
- Missing a relation may correspond to missing a rule, missing a relationship, an attribute of a class, a responsibility assignment, or an input (output) information of a transaction.
- Incorrect relation may produce incorrect semantic or incorrect rules. It may be also a supplement part.

Fig. 7. The specification with the IASDO model describes the legacy information system.



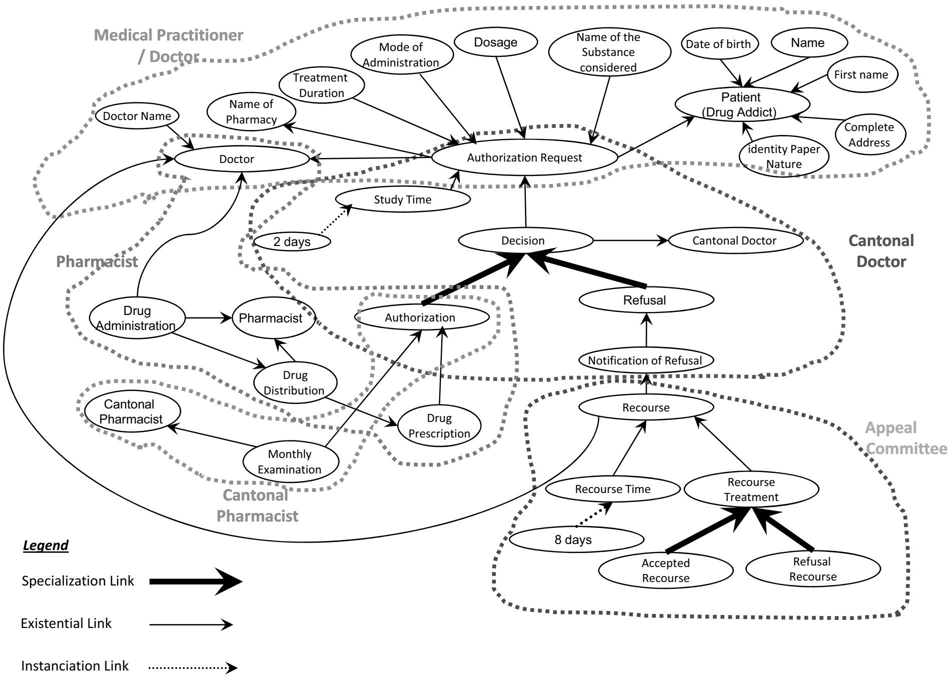


Fig. 8. IS Ontology extracted from the Geneva Law K4 20.06

The process model of analysing laws compliance of existing IS is illustrated in Figure 6.

5.2 Case study

In this section, we illustrate our approach with a case study; analysing the compliance with laws of a legacy system. The legacy system in the case aims to support the prescription of narcotics (drug) intended for the treatment of the dependent people. We describe the information of this system with the IASDO model. An ontology model concerns this domain is extracted from laws text. Then we analyse the specification of the legacy system to detect incompliant elements.

Figure 7 presents the specification of the legacy system with the IASDO model. Note: The creation of "Authorization Request" is under responsibility of Doctor

Figure 8 illustrates an example of ontology model extracted from the Geneva law K4 20.06. This example concerns the domain of the prescription of narcotics (drug) intended for the treatment of the dependent people.

From the law K4 20.06, we define the following hyperconcepts:

- A hyperconcept describing the ontological space of a doctor: this space specifies information concerning the authorization request in order to carry out a treatment. This space is under the responsibility of the doctor.
- A hyperconcept describing the ontological space of a cantonal doctor: this space specifies information concerning the decision-making about the request for authorization. This space is under the responsibility of the cantonal doctor.
- A hyperconcept describing the ontological space of an appeal committee: this space specifies information concerning the decision-making about the appeal formulated by the doctor. This space is under the responsibility of the appeal committee.
- A hyperconcept describing the ontological space of a pharmacist: this space specifies information concerning the dispensation of narcotics. This space is under the responsibility of the pharmacist.
- A hyperconcept describing the ontological space of a cantonal pharmacist: this space specifies information concerning the monthly examination of the authorizations and the register of stocks. This space is under the responsibility of the cantonal pharmacist.

In order to verify the compliance of the legacy IS with the Geneva law K4 20.06, the analysing process is applied.

Step 1: Identifying correspondent elements of the specification with the IASDO model and the ontology model

Table 4 illustrate the correspondent elements of the two specifications.

Step 2 and 3: Identifying missed and incorrect elements

Comparing concepts of the two specifications, the incompliant elements in the legacy system as in the following Table 5:

Laws-based ontology	**Legacy IS**
▪ Doctor ▪ Cantonal Doctor ▪ Pharmacist ▪ Cantonal Pharmacist ▪ Appeal Committee	▪ Not found ▪ Cantonal Doctor ▪ Pharmacist ▪ Cantonal Pharmacist ▪ Not found: The ontological role "Appeal Committee" is not considered in the existing IS.
▪ A hyperconcept describing the ontological space of a doctor.	▪ Informational space for a doctor. Concept: *Patient* → Class: *Patient* Concept: *Authorization Request* → Class: *Authorization Request* Concept: *Doctor* → Class: *Doctor* Concept: {*Name, First name, Date of birth, Complete Address, Identity Paper Nature*} → Attributes of the class "*Patient*" Concept: {*Name of the substance, Dosage, Mode of Administration, Treatment Duration, Name of Pharmacy*} → Attributes of the class "*Patient*"
▪ A hyperconcept describing the ontological space of a Cantonal doctor. ▪ Concept Decision	▪ Informational space for a cantonal doctor: Concept: *Authorization Request* → Class: *Authorization Request* Concept: *Study time* → Attribute of the Class "*Authorization Request*" Concept: *Study time= 2 days* → instance of the Attribute *Study time* Concept : Decision → Class: *Decision* AND Transaction: *Request Treatment* ▪ Not found
▪ A hyperconcept describing the ontological	▪ Informational space for a Pharmacist:

Table 4. The correspondent elements of the two specifications

Step 4: Consolidate and Report incompliant elements

Table 5 presents incompliant elements of the legacy system with the law-based ontology. However, due to the different modelling of the two specifications, some incompliant could be explained as follows:

- The role "Doctor" is not explicitly described in the specification. In fact, the method creation of an "authorization request" is considered as a basic transaction (mostly the creation of object in a class that is not an output class of any transaction is a basic transaction), the role Doctor is associated to this basic transaction although that is not illustrated in the graphic modelling. Therefore, Doctor is not a missing concept. This concept is excluded from the incompliant elements report.
- The concept "Decision" in fact is implicitly included in the two concepts "accepted request" and "refusal request". By dynamic specialization, an "authorization request" becomes an "accepted request" or "refusal request" after it has been treated based on a decision. Therefore, the concept "decision" is not missed as well as the specialization modelled in the legacy system is not incorrect. These one are excluded from the incompliant elements report.

5.3 Evaluation

Our proposed framework includes different steps and guidelines which assist us to detect incompliant elements of legacy information systems with laws. Our framework was illustrated in the case study. Meanwhile other approaches for IS engineering lack a systematic approach to refer to laws. In addition they mostly relay on experiences. Although at the present we only manually applied the proposed framework, we could detect missed and incorrect concepts, relations and rules in the legacy system. We could show a systematic approach that indicates the efficiency of our approach. We aim to develop a tool that allows to semi-automatically perform the steps in the proposed framework.

6. Conclusion

We have presented a framework for the compliance of legacy IS with the legal aspects and illustrated the proposed framework with

Table 5. Incompliant elements

Missed concepts or relations	**Incorrect concepts or relations**
- The role "Appeal committee" by consequence the responsibility of this role is not mentioned - The role "Doctor" - The concept "Decision" - The link between "Authorization request" and "Name of Pharmacy" that represents the rule governed by laws: "The name of the pharmacy which provides the drug must be indicated in the request for authorization". - The link between "2 days" and "Study time" of "Authorization request" represents the rule: "The duration of treatment must be indicated in the request for authorization". - The link between "8 days" and "Recourse time" of "Recourse" represents the rules: "The duration of recourse of a refused authorization requested is limited with 8 days".	- In the ontology model, "Accepted Request" and "Refusal Request" are sub-classes of "Decision", in the specification of legacy system, "Accepted Request" and "Refusal Request" are sub-classes of "Authorization Request" (i.e. relations do not have the same destination concept). - The incorrect responsibility of the transaction "Notification".

a case study. The IS supporting activities of organization is affected by laws as the laws regulate the execution of organization activities. The compliance of IS with laws is a critical issue in organization for obtaining an IS meeting user's expectations, not only at the moment of IS development but also in the whole life cycle of IS. However, this issue is not appropriately addressed in current methodologies and approaches of IS engineering.

In the proposed framework, the ontology extracted from laws is the basis for the controlling this compliance. The framework allows to detect incompliant elements of the legacy IS with laws. It also assist us to facilitate the IS evolution or IS reengineering and thus to meet user's expectations. However, the proposed framework is at a conceptual level. In our future research we aim to develop tools to support tasks proposed in the framework. In addition we aim to apply our approach to more practical scenarios and using the developed tools in practices.

7. Bibliographical references

[1] Arnold, R. (1992). Software engineering, IEEE Press.

[2] Atcherley, P. (1994). Re-engineering legacy information systems into new environments, The Institution of Electrical Engineers, IEEE.

[3] Barry, C. (2000). Who is to blame for ERP failure?, Serverworld Magazine, http://www.serverworldmagazine.com/sunserver/2000/06/erp_fail.shtml.

[4] Beale, I. (1996). Why information systems fail: a case study, Book Review – Readings, Internal Auditor, 53, 12-14.

[5] Boehm, B., & Turner, R. (2004). Balancing agility and discipline, Boston: Addision-Wesley.

[6] Booch, G., & Jacobson, I., & Rumbaugh, J. (2000). Unified modelling language : User Guide, Addison-Wesley.

[7] Brooke, C., & Ramage, M. (2001). Organisational scenarios and legacy systems, International journal of information management, (Int. j. inf. manage.), ISSN 0268-4012 CODEN IJMAED, 21, 365-384.

[8] Demarco, T. (1978). Structured analysis and system specification, Yourdon Press, NewYork (DFD).

[9] Halpin, T. (1998). Object role modeling (ORM/NIAM), Handbook on Architectures of Information Systems, eds P. Bernus, K. Mertins & G. Schmidt, Springer-Verlag, Berlin.

[10] Jiang, J., & Klein, G., & Balloun, J. (1999). Crampton, system analysts' orientations and perceptions of system failure, Information and Software Technology, 41, 101-106.

[11] Keller, G., & Nüttgens, M., & Scheer, A-W. (1992). Semantische prozeßmodellierung auf der Grundlage Ereignisgesteuerter Prozeßketten (EPK), in: Scheer, A.-W. (ed.): Veröffentlichungen des Instituts für Wirtschaftsinformatik, No. 89, Saarbrücken : Universität des Saarlandes (in German).

[12] Khadraoui, A. (2007). Method components for institutional information system engineering, ISBN: 978-2-88903-000-2, Editions SES - University of Geneva.

[13] Khadraoui, A., & Arni-Block, N., & Léonard, M., & Ralyté, J. (2005). Laws-based ontology for e-Government information systems, The Second international conference on innovations in information technology IIT'05, Dubaï.

[14] Kotonya, G. & Sommerville, I. (2002). Requirements engineering: processes and techniques, John Wiley & Sons.

[15] Le Dinh., T. (2004). Information system upon information systems: A conceptual framework, Phd Thesis, University of Geneva.

[16] Liu, K., & Alderson, A., & Qureshi, Z. (1999). Requirement recovery from legacy systems by analyzing and modeling behavior, in the Proceedings of the IEEE International Conference on Software Maintenance, Oxford, UK.

[17] Liu, K., & Alderson, A., & Sharp, B., & Shah, H., & Dix., A. (1998). Using semiotic techniques to derive requirements from legacy systems, SEBPC Legacy Systems Workshop.

[18] McManus, J., & Wood-Harper, T. (2003). Information systems project management: the price of failure, Management Services, 47.

[19] Markel, S. D. (1997). Process definition for capturing legacy system requirements, IEEE.

[20] Nertrand, P., & Darimont, R., & Delor, E., & van Lamsweerde, A. (1997). GRAIL/KAOS: an environment for Goal-Driven Requirements Engineering, the 19th International conference on Software Engineering, Boston.

[21] Pham Thi, T.T., & Dong Thi, B.T., & Bui, M.T.D., & Léonard, M. (2006). Spécification de workflow avec le modèle IASDO (Workflow specification with the IASDO model), IEEE the 4th RIVF, HoChiMinh city.

[22] Pham Thi, T.T., & Helfert, M. (2007). Modelling information manufacturing systems, Int. J.Information Quality, N°1, 5-21.

[23] Pham, T. (2005). Integration of static, dynamic, organizational aspect of information system, Phd Thesis, University of Geneva.

[24] Pin-Shan Chen, P. (1976). The entity-relationship model - Towards an Unified View of Data, ACM Transactions on Database Systems, 1.

[25] Ransom, J., & Sommerville, I. , & Warren, I. (1998). A method for assessing legacy systems for evolution, software maintenance and reengineering, Proceedings of the Second Euromicro Conference. ISBN: 0-8186-8421-6.

[26] Rolland, C. (1999). Experience with goal-scenario coupling in requirements engineering, RE'99: Fourth

IEEE International Symposium on Requirements Engineering.

[27] Rolland, C., & Prakash, N.,& Benjamin, A. (1999). A multi-model view of process modelling, Requirement Engineering , 4 , 169-187.

[28] Sutcliffe, A. (2003). Scenario-based requirements engineering, Proceedings of the 11th IEEE International conference on requirements engineering.

[29] Tardieu, H., & Nancy, D., & Pascot, D. (1979). Conception d'un système d'information; construction de la base de données. Editions d'Organisation.

[30] Turki, S. & Aïdonidis, C., & Khadraoui, A., & Léonard, M. (2004). Ontologies for institutional IS engineering, Open INTEROP Workshop On "Enterprise Modelling and Ontologies for Interoperability"; EMOI-INTEROP 2004; Co-located with CaiSE'04 Conference, Riga (Latvia).

[31] Van Lamsweerde, A. (2001). Goal-oriented requirements engineering: A guided tour, Proceedings RE'01, 5th IEEE International Symposium on Requirements Engineering, Toronto.

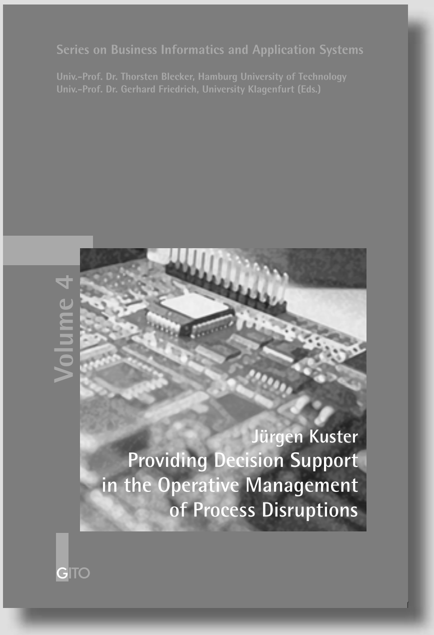
[32] Zúñiga, GL. (2001). Ontology: Its transformation from philosophy to information systems. In Christopher Welty and Barry Smith, editors, Proceedings of the International Conference on Formal Ontology in Information Systems (FOIS'01), ACM Press, Ogunquit, Maine, USA, 187-197.

Abdelaziz Khadraoui
Centre Universitaire d'Informatique
University of Geneva
Battelle - bâtiment A, 7, route de Drize CH-1227
Carouge, Switzerland
Email: khadraoui@cui.unige.ch
Phone: +41 22- 379 - 0231

Michel Leonard
Centre Universitaire d'Informatique
University of Geneva
Battelle - bâtiment A, 7, route de Drize CH-1227
Carouge, Switzerland
E-Mail: leonard@cui.unige.ch
Phone: +41 22- 379 0227

Thanh Thoa Pham Thi
School of Computing
Dublin City University, Glasnevin, Dublin 9,
Ireland
E-Mail: thoa.pham@computing.dcu.ie
Phone: +353 1- 700 - 6911

Markus Helfert
School of Computing
Dublin City University, Glasnevin, Dublin 9,
Ireland
E-Mail: markus.helfert@computing.dcu.ie
Phone: +353 1- 700 - 6911


Series on Business Informatics and Application Systems
Univ.-Prof. Dr. Thorsten Blecker, Hamburg University of Technology
Univ.-Prof. Dr. Gerhard Friedrich, University Klagenfurt (Eds.)
Volume 4
Jürgen Kuster
Providing Decision Support
in the Operative Management
of Process Disruptions
GITO

Available online at www.enterprise-systems.net

AIS Transactions on Enterprise Systems 1 (2009) 1, 27-35

Enhancing the Accessibility of Enterprise System Documentation with Domain Ontologies

Christian Janiesch

Abstract

Accessible process models are vital to the longevity and meaningfulness of model-based enterprise systems documentations. Different parties intend to consult the models for diverse reasons and purposes. Lack alication domain knowledge as well as lack of modeling knowledge of a user is major factors in the perceived usefulness of process models. It is therefore important to provide better accessible models. However, as each user's access to models is different, models need to be adapted to the user's needs. However, customizing models leads to a zoo of variants which cannot be administered properly. One way of preventing this, is to employ configuration mechanisms to increase the reusability of process models already during model design. We propose to enrich these models with domain ontologies to improve the consistency, configurability and, thus, accessibility of the documentation.

Keywords: ???

1. Introduction

The documentation of enterprise systems is of paramount importance for the lasting benefit of the implementation project [3]. However, the documentation needs to comply with the diverse needs of users who use models for different purposes. A system engineer for instance is in need of different information than a manager or a trainee. Approaches to adapt models to suit multiple perspectives exist, but different degrees of preexisting application domain knowledge complicate the designing of one universal model.

Hence, it is not surprising that the concept and importance of application domain knowledge has gained increased attention in the IS community [e.g., 29; 40]. Application domain knowledge or subject matter expertise can be understood as the knowledge of the problem area [24]. Especially in IS development, application domain knowledge about the context of future software, i.e. the target environment and its content, is considered crucial for a project success [12]. Application domain knowledge is opposed to programming or modeling knowledge, also termed IS domain knowledge, which comprise purely technical skills [29].

An intensively discussed means to capture common domain knowledge is provided by ontologies [e.g., 8; 21; 45; 51]. We propose to utilize domain ontologies to enhance the reusability of enterprise systems documentation during the design phase by adding additional application domain knowledge to the models. In the following, we focus on process models as the key model-based documentation of enterprise systems.

This paper follows the critical hermeneutic method to gain knowledge. Hermeneutic research methods focus on an understanding which is based on theoretical and practical arguments. According to hermeneutics, the process of gaining knowledge is influenced by a circle of (previous) understanding, gaining new knowledge, and then achieving a better understanding of the entire [6].

The paper is organized as follows: In the subsequent section we elaborate on the fundamentals of process design and reuse in modeling to provide a common understanding of the problem area. In Section 3 we provide an overview of the concept of ontology and relate it to the domain of process design. Section 4 relates the two concepts and introduces applications which assist reuse in process design. The paper closes with a summary and outlook to further research,

2. Reuse in Process Modeling

2.1. Fundamentals

Processes are generally seen as any activity performed within a company or an organization [33]. Thereby, activities are considered to be the smallest entity used to build a process. An activity can either refer to work that needs to be done manually or to work that can be done automatically [22]. Other definitions [11; 27] emphasize on the timely and local order of activities and also regard different kinds of inputs and value-added outputs. Also, business goals to which processes are tied to, are stated to be significant for the definition of a process [22]. Each of those definitions falls short to integrate all relevant perspectives into one consistent definition. In the context of this work, a process is therefore defined as “a completely closed, timely and logical sequence of activities which are required to work on a process-oriented business object” [5], as this definition integrates all relevant perspectives.

Consequently, a business process is considered a special kind of process that is directed by business objectives of a company and by the business environment [5]. Business processes can be further classified into value creating core business processes and not value adding supplementary processes. Whereas core business processes are considered to contain corporate expertise and produce products or services that are delivered to customers [22; 37], supplementary business processes facilitate the ongoing operation of the core processes. This distinction is not intended to be always selective as one business process might be a core business process for one product and a supplementary business process for another [5].

2.2. Adaptation Mechanisms for Process Models

Process Models contain practical knowledge that may not be documented otherwise. The more effective the process model meets the requirements of a particular need or perspective; the higher is its perceived quality. Ideally, each identified perspective should be provided with a tailor-made version of a process model. This approach is called multi-perspective process modeling [10; 38]. The most significant problem that results from a multiplicity of perspective-specific, tailor-made models is the need to manage possible redundancies inside the model itself. This leads to increased modeling and maintenance cost and the danger of inconsistencies within the model base.

Different reuse mechanisms aim at overcoming this problem. Reuse means to apply the experiences of a former projects to solve an actual problem [51]. This implies that an existing knowledge base is utilized to avoid starting from scratch. Different adaptation approaches for the reuse of knowledge have been developed which can be found in a similar form in software engineering, conceptual modeling, or method engineering. All reuse approaches are based on a common set of reuse mechanisms, which enable their definition and application. The following five mechanisms are taken from the area of reference modeling, which provides a sound basis for the design of processes [2; 47]:

Process models can be designed as configurable artifacts. They are provided with specific attributes which contain configuration rules. Depending on the values assigned to the condition part of the rule it can be decided whether the conclusion part of the rule has to be executed or not. The conclusion part can imply consequences such as the elimination of model elements or the modification of their representation. With this mechanism model variants regarding application-specific characteristics can be created in an automated manner. An important application for this mechanism is for example to tailor a process model to a specific perspective such as a management view or software engineering view.

When the mechanism of instantiation is used, process models are equipped with placeholders. The placeholders are inserted within the construction phase of the process model and annotated with an instantiation domain. When a specific model is created, the placeholders are filled with valid occurrences to the particular circumstances. Depending on the properties of the application domain, numeric or alphanumeric values, distinct model elements, or even composed model clusters can be defined as instantiation domains.

Through specialization a specific model is derived from a more general model by adapting, extending and/or partially modifying the more general

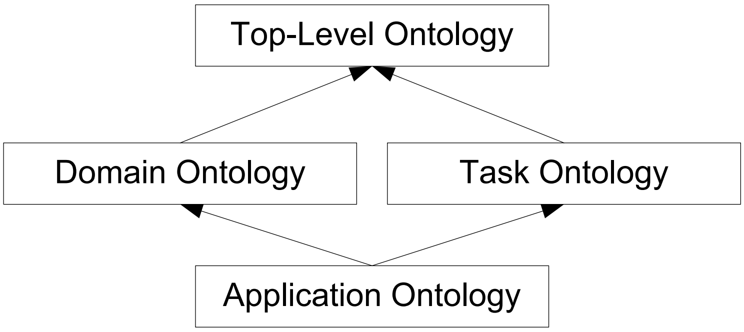


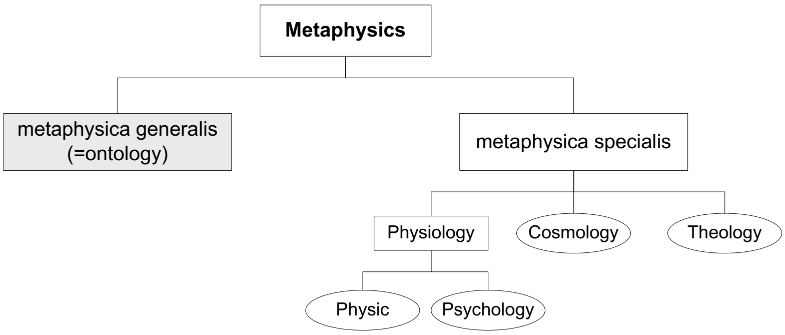
Fig. 1. Levels of ontologies [19].

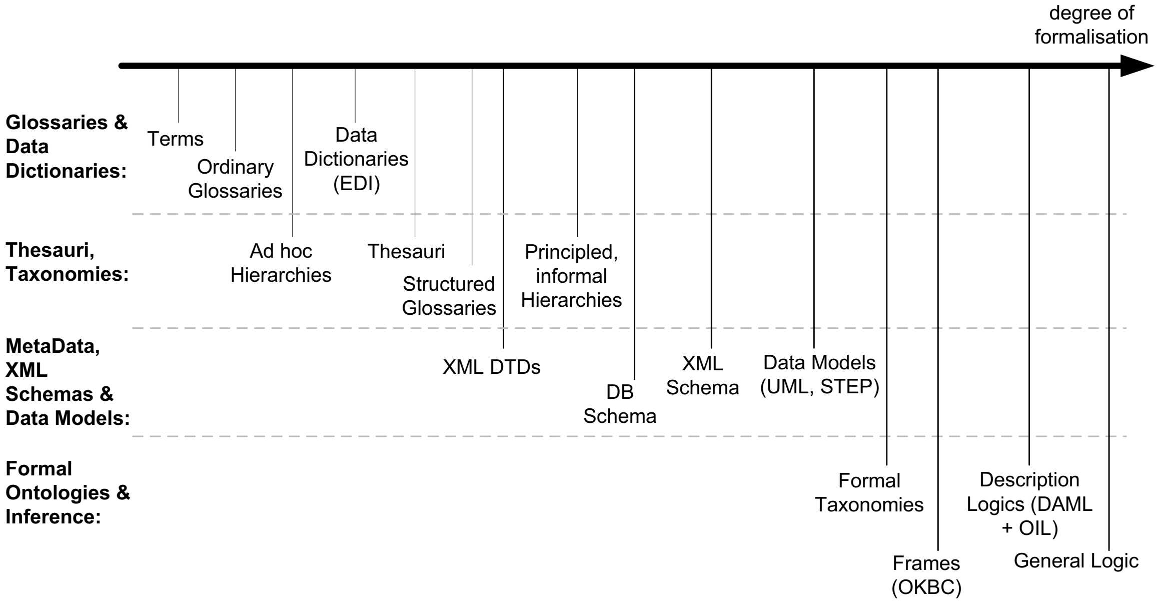
one. For this purpose, the model is annotated with specialization instructions. Process models which support specialization have a higher level of abstraction than the resulting specific models. These more abstract process models offer only a relatively low number of model elements.

Process models, which support aggregation, are not available as monolithic blocks but rather contain independent parts, so called model element components. Within aggregation, a specific model is built by assembling these model components. Interface descriptions of the model components offer information on their general compatibility and how to combine them.

An analogy implies the transfer of information from one subject to another. This mechanism is very flexible as it can be drawn from any aspect of a process model. Patterns employ this mechanism in order to be applicable in domains they were not specifically constructed for. The application of a pattern requires a conclusion by analogy to establish a fit between the problem description in the pattern and the actual situation.

Fig. 2. Different representation forms of ontologies [46]





3. Concept and Design of Ontologies

3.1. Ontology in Philosophy and Information Systems Research

The notion of ontology has arisen long before the term was coined. It traces back to Parmenides and was embellished by Plato [36] and Aristotle [1]. Aristotle sets out from substance as the most profound and first category of every being and elaborates a system of ten categories, which accounts for his idea of ontology, i.e. how the world is constituted. This conceptualization was the most dominant view of ontology until Kant made some central changes [28]. The traditional meaning of ontology is a branch of philosophy dealing with the nature of things in general [50]. Hence it is a total, domain independent concept.

In contrast to philosophical ontology, IS research has inherited and altered the idea. One can speak of informational ontologies, which are partial, domain specific, and committed to an epistemological constructivism [19; 41]. These ontologies are also often denoted as formal ontologies in the literature. The nature of capturing parts of the real world automatically results in a domain specific application of these ontologies [8]. But more importantly it leads to a plurality of coexistent ontologies. Hence, the plural ontologies is brought up which does not exist in philosophical contexts.

This plurality substantiates the introduction of different levels to structure different ontologies according to their specificity. The most prominent classification according to Guarino [19] is distinguishing top-level, domain, task, and application ontologies. Cf. Figure 1 for an overview.

Top-Level ontologies are intimately related to the philosophical notion of ontology and are based on very generic categories [9; 42; 49]. The domain or task ontology level is already developed with regard to the requirements of a certain application [30]. And finally application ontologies are implemented for usage in a particular environment or system.

The plural ontologies is associated with the conception of informational, pluralistic, formal ontologies. The constructivist account renders informational ontologies as contingent human artifacts, which are independent of reality and are necessarily subject to some particular ends or purposes. As a result they are always susceptible to a certain degree of subjectivism caused by the individual ontology engineer. Consequently a popular definition in IS literature reads: "An ontology is a formal, explicit specification of a shared conceptualization for a domain of interest" [16]. Gruber moreover specifies a conceptualization as "an abstract simplified view of the world that we wish to represent for some purpose" [16]. Resulting from this, the foremost motivation for employing ontologies in the context of IS is the hope to alleviate the subjectivism of different models by means of underpinning them with a shared conceptual vocabulary manifested in an ontology. Thus, ontologies aspire to affiliate slightly different worldviews on a common ground.

3.2. Instantiation and Design of Ontologies

More recently IS research has developed a manifold of representational means for illustrating ontologies. A comprehensive list can be taken form Figure 2.

In this schema the different instantiations of ontologies are arranged according to their degree of formalization. The prevailing technique, tying in with the philosophical tradition since Aristotle, is the taxonomy. This is defined as a structure where all entities are grouped to a hierarchy exclusively built with subsumption relations [20]. Whilst the groups of glossaries & data dictionaries and thesauri & taxonomies are characterized by their semi-formal and non-technical approach, the groups meta data, XML schemas & data models as well as formal ontologies are highly specified. The different forms of representation also impose varying degrees of restriction on the logical structure. However, all elements noted in Figure 2 are usually subsumed under the term ontologies.

According to literature, a list of the basic components to substantiate the definition of ontologies is given with the following [32]:

- a set of concepts (basic terms)
- relationships, and
- inference rules.

One of the first adaptations of ontologies to IS goes back to Bunge [7] and has found is manifestation in the Bunge-Wand-Weber Ontology [49; 50]. Since the crucial idea of a philosophical ontology is to capture reality, they argue that ontologies can improve informational models by mapping real world objects into abstract entities. Note that a similar assumption is implicitly underlying other data modeling techniques such as the Entity-Relationship Model. A second popular account of the top-level structure of ontologies is given by Chisholm [9]. He divides all entities in contingent and necessary ones and then further distinguishes states and non-states.

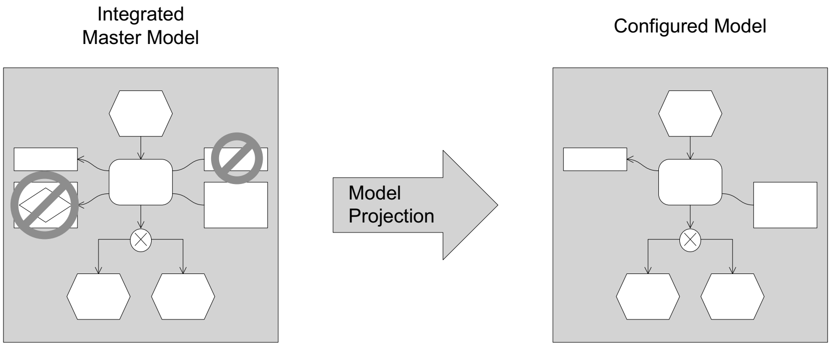


Fig. 3. Configuration procedure for conceptual models [2]

Several concrete engineering processes for ontologies haven been proposed [15; 17; 18, Smith, 2004 #938]. Since developing ontologies is an extensive endeavor, tool support is inevitable.

4. Relating Process Models and Ontologies

Process models as well as modeling methods can be designed as configurable artifacts. They are then provided with explicit configuration points, which specify model variants regarding application-specific characteristics. Based on assigned values to configuration parameters a model is projected into an application-specific model. The actual procedure of model projection is conducted automatically based on the prior selection and annotation of configuration parameters. Cf. Figure 3 for a high-level illustration of the procedure. Here, an integrated master model, e.g. an event-driven process chain [39], is configured according to the users requirements by selecting several configuration parameters. This results in the omission of certain model elements or method elements which are then not part of the configured model after its projection. As it is only a projection they are still part of the integrated master model but not visible in the projection.

Configuration Parameter Ontologies: When offering configurable process models, it is important to specify configuration parameters which allow for the configuration of the model with respect to various practical requirements, for availing the model to as many parties as possible. Therefore, it is necessary to define categories of configuration parameters. Becker et al. [2] propose perspectives (i.e. roles or even individual views) and company characteristics; UN/CEFACT [43] recommends the following eight context categories: business process, product classification, industry classification, geopolitical, official constraints, business process role, supporting role, and system capabilities. Each of the categories has to be instantiated with an unambiguous taxonomy which allows for the detailed specification of a company's context to provide the best suited projection of the master model.

While Becker et al. [2] do not specify a comprehensive amount of parameters, UN/CEFACT [43] proposes to existing nest code lists as a domain ontologies. For example they propose to use ISO 3166.1 for countries and ISO 3166.2 for regions and so on. The elements are ordered in a taxonomy. In their approach it is admissible to combine elements from different categories to denominate a specific overall context, e.g. as laws from a specific country. If however one category is always combined with another and cannot stand alone it is to evaluate whether it is obsolete or should be changed [25]. Currently not all categories can be utilized with code lists. The Unified Context Methodology Project is an ongoing initiative to close this gap [44].

Ensuring Consistency with Ontologies: In order to configure process models, a selection of configuration parameters has to be made. This can either be done implicitly by the use of high level configuration mechanisms [26] or explicitly by the selection of values. However, these configuration parameters can have relations with each other. The selection of one configuration parameter can entail either that another parameter is selected as well (inclusion) or that another configuration parameters may not be selected any more (exclusion). The modeling software utilized must support the user with ontologies that comprise this information. This is especially important for inter-company models which require certain elements, e.g. process functions, to be present at least at one party participating in the process [4].

Consider a process including three parties: a consumer, a broker, and a provider. The consumer uses the broker to find the best possible provider for a given service and the broker relays the consumer order. The provider will only process the consumer order in case a credit check has been performed. The credit check can be either performed by the broker or the provider. The function perform credit check of the process model can be configured as

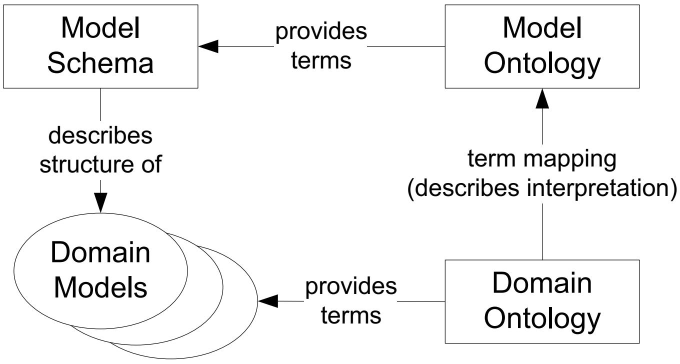


Fig. 4. Modeling of domain knowledge.

optional, mandatory or obsolete for the broker. This configuration has an exclusive effect on the configuration of the provider since this makes his credit check optional, obsolete or mandatory depending on the broker's configuration as the check has to be performed by either one of the parties before the order can be executed. Bearing this in mind, the interdependencies of each configuration parameter has to be carefully considered in order to construct a consistent configurable process model.

Synonym Management: In different divisions of an enterprise different naming conventions may have been established (e.g. procurement employees call a procurement order just order, whereas distribution employees may use the term purchase order). In order to consider these synonymous conventions in line with conceptual modeling, a proper synonym management allows for perspective-specific exchange of model element naming [2].

It stands to reason that an ontology of synonymous terms is utilized as a foundation for the dynamic management of synonyms, homonyms, and polysems. Homonyms are single terms that signify multiple meanings. Vice versa synonyms are unique concepts which can be expressed by different words. A slightly more complex situation is caused by polysems, which are concepts with overlapping meaning [23].

Lexical and logical relations allow for a more comprehensive adaptation of the model in order to avoid new homonyms that might occur when a synonymous term is changed, e.g. if the term standard is replaced by default, all terms utilizing default in the sense of failing should be replaced by mistake or error.

A problem for ontologies arises, when such a broad concept must be classified into a discrete structure, where conflicting categorizations may occur. The second class of problems concerns the consensus which is essential to construct ontologies for a group of people. This becomes manifest in cases, where polysemy is at hand. Then the first problem is to find a consensus on how to classify a concept, before the subsequent problem of having to classify it in a discrete structure follows.

Model Comparison: Although we propose to incorporate configuration mechanisms using domain ontologies already during model design, it may be necessary to relate separately constructed models. It is therefore necessary to compare different models or even to merge their contents. First approaches to model consolidation were of manual nature. With the growing size of process models to document whole process clusters or service landscapes as well as workflows, a demand for conducting at least semi-automatic if not automatic consolidation has arisen. In order to compare or merge models a common domain ontology has to be established.

One of the major challenges here is to overcome the problem of external consistency, because each informational ontology which is the basis for a process model is restricted to its own application domain. And every domain represents a particular part of reality [14; 48]. This nexus is illustrated in Figure 4, where the model ontology provides the different model levels with the necessary terms. The central item here is the domain models. These can be substantiated by various ontologies for various models. Alternatively, different models can be supplied by one unifying domain or global ontology.

However, it might be economically unfeasible to reconcile all ontologies into one global ontology. Shaping one global ontology is hampered by inflexibility, high maintenance costs, and the problem of finding group consensus concerning the underlying concepts [31]. Thus, as argued above, issues about the consistent combination of selected ontologies arise. It can be very useful to compose multiple ontologies to interrelate already existing ontologies within a particular domain.

In general the most prevailing problems of ontologies are inconsistency, incompleteness, and redundancy [13]. They become especially evident, when models based on ontologies from different backgrounds need to be merged [31]. For example a practitioner will use different terms than a theoretically oriented academic lecturer. Even if both try to denote the same entities in one application domain, they will end up with different ontologies.

Pfeiffer [35] proposes to utilize (building-block-based) domain-specific modeling methods to enable comparison already at a language level. Hence, the representation is contingent on the conceptualization of the person reflecting the domain and explicating it into a model. To overcome these issues a standardized engineering process for ontologies is necessary as well as comprehensive modeling conventions [5].

5. Conclusions

Accessible process models are vital to the longevity and meaningfulness of model-based enterprise systems documentations. Different parties intend to consult the models for diverse reasons and purposes. We showed how ontologies in conjunction with configuration mechanisms can increase the reusability of process models already during model design. We also pointed out the problem which arises when models of different application domains, i.e. models based on different application (implicit) domain ontologies, need to be combined.

Our research has theoretical as well as practical implications. First, the results of this paper are relevant for creators of process models such as system analysts, software engineers, or consultants. We give reason to assume that process models should be delivered with a specification of the domain terms used in the model. Such models are better to comprehend and to evaluate for users with varying application domain knowledge. Hence, to link a model to a domain ontology or to supplement it with definitions of the domain vocabulary can increase the model accessibility. To provide process models as a detached artifact and without any additional information may in contrast jeopardize the success of a customer project and easily lead to project documentation with no particular use.

Second, this paper explains the requirements a process model has to meet from the user's perspective. A process model has impact on the results of an IS development or organizational design project. Hence, a user of a process model is interested in the chance to understand the process model thoroughly. A specification of the modeling method and the reference to a domain ontology increase the accessibility and configurability. Thus, a user of a process model should declare at least this information about the model as obligatory. In order to facilitate an organization-wide consensus about a model it is helpful when all stakeholders share an equal understanding. Based on this comprehension the model can then be mapped to their individual domain experiences and be accepted or rejected.

Third, the results of our research also concern the producers of process modeling tools. It shows that the accessibility and consistency of a process model is influenced by the availability of additional information. Therefore producers of process modeling tools should augment their products with the functionality to manage the association between process models and application domain terms. Therefore, a process modeling tool should be able to access and link to domain ontologies, or at least simple glossaries or technical term models. For convenient usage a process modeling tool should (semi-)automatically establish connections between an ontology elements and the model elements. If an ontology is not available for a certain domain, a tool should provide support to develop it. Furthermore, a process modeling tool should allow for exporting a domain ontology [e.g., 41] and a modeling method [e.g., 34] in order to deliver it together with the process model to the model users.

References

[1] Aristotle (2005). Metaphysics. Sioux Falls, SD: Nuvision Publishing.

[2] Becker, J., Delfmann, P., & Knackstedt, R. (2007). Adaptive Reference Modeling: Integrating Configurative and Generic Adaptation Techniques for Information Models. In J. Becker & P. Delfmann (Eds.), Reference Modeling. Efficient Information Systems Design Through Reuse of Information Models (pp. 27-58). Heidelberg: Physica.

[3] Becker, J., Janiesch, C., Delfmann, P., & Fuhr, W. (2006). Perspectives on Process Documentation: A Case Study. In C.-S. Chen, J. Filipe, I. Seruca & J. Cordeiro (Eds.), Enterprise Information Systems VII (pp. 167-177). Dordrecht: Springer.

[4] Becker, J., Janiesch, C., & Dreiling, A. (2006). A Framework for Interdependent Configuration of Enterprise Systems. Inaugural Workshop on Enterprise Systems Research in MIS. (Pre-)ICIS, Milwaukee, WI.

[5] Becker, J., Kugeler, M., & Rosemann, M. (Eds.). (2008). Process Management: A Guide for the Design of Business Processes (2nd ed.). Berlin: Springer.

[6] Becker, J., & Niehaves, B. (2007). Epistemological Perspectives on IS Research: A Framework for Analysing and Systematizing Epistemological Assumptions. Information Systems Journal, 17(2), 197-214.

[7] Bunge, M. (1977). Treatise on Basic Philosophy Vol. 3. Ontology I: The Furniture of the World. Dordrecht: D. Reidel.

[8] Chandrasekaran, B., Joesephson, J., & Benjamins, R. (1999). What are Ontologies and Why Do We Need Them? IEEE Intelligent Systems, 14(1), 20-26.

[9] Chisholm, R. M. (1996). A Realistic Theory of Categories. Cambridge: Cambridge University Press.

[10] Darke, P., & Shanks, G. (1996). Stakeholder Viewpoints in Requirements Definition. Requirements Engineering, 1(1), 88-105.

[11] Davenport, T. (1993). Process Innovation: Reengineering Work Through Information Technology. Boston, MA: Harvard Business School Press.

[12] Davis, G. B., & Olsen, M. H. (1985). Management Information Systems: Conceptual Foundations, Structure, and Development (2nd ed.). New York, NY: McGraw-Hill.

[13] Gómez-Pérez, A. (2004). Ontology Evaluation. In S. Staab & R. Studer (Eds.), Handbook on Ontologies (pp. 250-271). Heidelberg: Springer.

[14] Gómez-Pérez, A., & Benjamin, V. R. (1999). Overview of Knowledge Sharing and Reuse Components: ntologies and Problem-Solving Methods. International Joint Conference on Artificial Intelligence Workshop on Ontologies and Problem-Solving Methods, Stockholm, 1-15.

[15] Gómez-Pérez, A., Fernandez-Lopez, M., & Corcho, O. (2004). Ontology Engineering. Heidelberg: Springer.

[16] Gruber, T. R. (1993). A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition, 5(2), 199-220.

[17] Gruber, T. R. (1995). Towards Principles for the Design of Ontologies Used for Knowledge Sharing. International Journal of Human Computer Studies, 43(5-6), 907-928.

[18] Guarino, N. (1997). Understanding, Building and Using Ontologies. International Journal of Human Computer Studies, 46(2-3), 293-310.

[19] Guarino, N. (1998). Formal Ontology and Information Systems. 1st International Conference on Formal Ontology in Information Systems (FOIS), Trento, 3-15.

[20] Guarino, N., & Welty, C. (2000). A Formal Ontology of Properties. 12th International Conference on Knowledge Engineering and Knowledge Management. Methods, Models and Tools (EKAW), Berlin, 97-112.

[21] Guizzardi, G., Pires, L. F., & van Sinderen, M. J. (2002). On the Role of Domain Ontologies in the Design of Domain-Specific Visual Modeling Languages. 17th ACM Conference on Object-Oriented Programming, Systems, Languages and Applications (OOPSLA), Seattle, WA.

[22] Harmon, P. (2003). Business Process Change: A Manager's Guide to Improving, Redesigning, and Automating Processes. San Francisco, CA: Morgan Kaufmann.

[23] Hirst, G. (2004). Ontology and the Lexicon In S. Staab & R. Studer (Eds.), Handbook on Ontologies (pp. 209-227). Heidelberg: Springer.

[24] Hjørland, B., & Albrechtsen, H. (1995). Toward a New Horizon in Information Science: Domain-Analysis. Journal of the American Society for Information Science, 46(6), 400-425.

[25] Janiesch, C., Dreiling, A., Greiner, U., & Lippe, S. (2006). Configuring Processes and Business Documents: An Integrated Approach to Enterprise Systems Collaboration. 2006 IEEE International Conference on e-Business Engineering (ICEBE), Shanghai, 516-521.

[26] Janiesch, C., Dreiling, A., & Seidel, S. (2006). Document Variant Management: Facilitating Enterprise System Definition, Configuration, and Interoperability. 17th Australasian Conference on Information Systems (ACIS), Adelaide, 1-10.

[27] Johansson, H. J., McHugh, P., Pendlebury, A. H., & Wheeler, W. A. (1993). Business Process Reengineering: Breakpoint Strategies for Market Dominance. Chichester: Hohn Wiley & Sohns.

[28] Kant, I. (1999). Critique of Pure Reason. Cambridge: Cambridge University Press.

[29] Khatri, V., Vessey, I., Ramesh, V., Clay, P., & Park, S.-J. (2006). Understanding Conceptual Schemas: Exploring the Role of Application and IS Domain Knowledge. Information Systems Research, 17(1), 81-99.

[30] Kumar, A., & Smith, B. (2003). The Unified Medical Language System and the Gene Ontology: Some Critical Reflections. In A. Günter, R. Kruse & B. Neumann (Eds.), 26th German Conference on Artificial Intelligence (KI). Lecture Notes in Artificial Intelligence (Vol. 2821, pp. 135-148). Berlin: Springer.

[31] Mitra, P., & Wiederhold, G. (2004). An Ontology Composition Algebra. In S. Staab & R. Studer (Eds.), Handbook on Ontologies (pp. 93-113). Heidelberg: Springer.

[32] Neches, R., Fikes, R., Finin, T., Gruber, T., Patil, R., Senator, T., et al. (1991). Enabling Technology for Knowledge Sharing. AI Magazin, 12(3), 36-56.

[33] Object Management Group Inc. (2006). Business Process Modeling Notation (BPMN) Specification 1.0 Retrieved 2007-06-20, from http://www.omg.org/cgi-bin/apps/doc?dtc/06-02-01.pdf

[34] Object Management Group Inc. (2005). MOF 2.0/XMI Mapping Specification, v2.1 Retrieved 2007-04-28, from http://www.omg.org/cgi-bin/doc?formal/2005-09-01

[35] Pfeiffer, D. (2008). Semantic Business Process Analysis: Building Block-based Construction of Automatically Analyzable Business Process Models. Dissertation, Münster.

[36] Plato (2003). The Republic (2nd ed.). London: Penguin Classics.

[37] Porter, M. E. (1985). Competitive Advantage: Creating and Sustaining Superior Performance. New York, NY: The Free Press.

[38] Rosemann, M. (1998). Managing the Complexity of Multiperspective Information Models using the Guidelines of Modeling. 3rd Australian Conference on Requirements Engineering, Geelong 101-118.

[39] Scheer, A.-W. (2000). ARIS: Business Process Modeling (3rd ed.). Berlin: Springer.

[40] Shaft, T. M., & Vessey, I. (1998). The Relevance of Application Domain Knowledge: Characterizing the Computer Program Comprehension Process. Journal of Management Information Systems, 15(1), 51-78.

[41] Smith, B. (1998). The Basic Tools of Formal Ontology. 1st International Conference on Formal Ontology in Information Systems (FOIS), Trento, 19-28.

[42] Sowa, J. F. (1999). Knowledge Representation: Logical, Philosophical, and Computational Foundations. Pacific Grove, CA: Brooks Cole Publishing.

[43] UN/CEFACT (2008). UN/CEFACT Core Components Technical Specification. Version 3.0 Implementation Verification Retrieved 2008-16-09, from http://www.unstandards.org:8080/download/attachments/3801817/Specification_CCTS+3p0+ODP6+20080207.pdf?version=1

[44] UN/CEFACT Techniques and Methodologies Group (2007). Unified Context Methodology Project Retrieved 2007-06-20, from http://www.untmg.org/

[45] Uschold, M. (1998). Knowledge Level Modelling: Concepts and Terminology. The Knowledge Engineering Review, 13(1), 5-29.

[46] Uschold, M., & Gruninger, M. (2004). Ontologies and Semantics for Seamless Connectivity. SIGMOD Record, 33(4), 58-64.

[47] vom Brocke, J. (2007). Design Principles for Reference Modelling: Reusing Information Models by Means of Aggregation, Specialisation, Instantiation, and Analogy. In P. Fettke & P. Loos (Eds.), Reference Modeling for Business Systems Analysis (pp. 47-75). Hershey, PA: IDEA Group.

[48] Wand, Y. (1996). Ontology as a Foundation for Meta-modelling and Method Engineering. Information and Software Technology, 38(4), 281-287.

[49] Wand, Y., & Weber, R. (1993). On the Deep Structure of Information Systems. Information Systems Journal, 5(3), 203-223.

[50] Weber, R. (1997). Ontological Foundations of Information Systems. Melbourne: Coopers & Lybrand.

[51] Wimmer, K., & Wimmer, N. (1992). Conceptual Modeling Based on Ontological Principles.

Available online at www.enterprise-systems.net

AIS Transactions on Enterprise Systems 1 (2009) 1, 36-43

Enterprise Architecture Design as an Engineering Discipline

Stephan Aier, Stephan Kurpjuweit, Jan Saat, Robert Winter

Abstract

Enterprise architecture can provide systematic support to organizational change, when requirements of respective stakeholders of business and IT are met. This article focuses on the design of enterprise architecture and proposes a "business-to-IT" approach that considers lessons from classical engineering disciplines. A framework for engineering driven enterprise architecture design is presented. Since such an approach creates specific requirements for tool support, an appropriate software implementation is presented.

Keywords: enterprise architecture, business engineering navigator, tool support

1. Introduction

Organizations are subject to constant evolution. Due to the different impact, organizational change can be distinguished into incremental change (optimization) and fundamental change. While most functional methods of business administration, such as marketing, finance and human resources provide support for optimization (e.g. six sigma) [10], the structured design of innovative and fundamental change requires a holistic approach to systematically support organizational transformation [21]. Complex changes require a thorough understanding and therefore a targeted documentation of the artifacts to be designed, their relationships to each other as well as a clear structuring of the transformation procedure. Therefore, architectural as-is documentation, to-be planning, and support of necessary changes are core challenges for enterprise architecture (EA) analysis and design [13]. To meet these challenges, design objects of EA such as strategic aspects, organizational structure, business processes, software components and data structures as well as IT infrastructure components are modeled to enable communication and analysis of the EA [11, 22]. While there is a broad variety of EA literature focusing on evaluation [18] and generalization [12] of EA frameworks or discussing EA modeling [1], only few publications address EA application and its benefits [14, 17]. In particular an engineering approach is missing which deploys EA to systematically support innovative and fundamental change.

In this contribution we analyze mature engineering disciplines to derive characteristics for a framework to systematically support consistent "business-to-IT" transformation. We propose the business engineering navigator (BEN) concept to support construction, navigation and analysis functionalities for artifacts and relationships of all architectural layers – from strategic aspects down to IT infrastructure. BEN therefore provides a framework on how engineering methods can be applied to organizations. BEN delivers insights on how complex design and transformation challenges can be broken down to manageable projects. We therefore discuss how BEN can be used to systematically support to EA design in this article.

The next section identifies core concepts of mature engineering disciplines. Following lessons learnt from classical engineering, section 3 derives requirements for an engineering based approach to EA. Section 4 introduces the BEN concept to support a stakeholder-oriented EA management (EAM) as one of multiple possible applications. Section 5 discusses a "business-to-IT" EAM tool support and proposes ADOben as an appropriate solution. The findings are summarized, and future research is outlined in section 6.

2. Lessons from Mature Engineering Disciplines

Mary Shaw analyzed the development of classical engineering disciplines [20]. She found that engineering disciplines produce cost efficient solutions for relevant problems by using scientific knowledge in the artifact design process in service to society. These aspects are now further characterized:

1. "Cost efficient solutions": Engineering does not only imply the construction of suitable solutions, but also emphasizes reasonable handling of given resources and conditions.
2. "For relevant problems": The constructed solution addresses practically relevant problems.
3. "By using scientific knowledge": The construction process is comprehensible and traceable based on scientific construction languages, methods, and frameworks so that the solutions will most likely fit the requirements.
4. "In service to society": The engineer acts in a responsible way by providing useful innovations to society and environment.

The following subsections give an idea of addressing these aspects by analyzing classical engineering.

2.1. Engineering Knowledge Patterns

Classical engineering disciplines distinguish between innovative construction and construction routine. Innovative constructions have to address new solutions while construction routine involves reusing existing solution patterns for known problems [23].

Construction routine is the usual design form in classical engineering disciplines, while innovation is rather rare. To make the construction process as efficient as possible, the collection, organization, and conditioning of knowledge is necessary to make this knowledge available to less experienced engineers. All disciplines found appropriate media for this knowledge transfer, e.g. engineering handbooks [2, 6] and tool support for collaborative engineering [15].

2.2. Standardized Construction Plan and Construction Language

Mature engineering disciplines use a high level construction plan (architecture) of the design artifact. This plan depicts the main components and their relationship to each other that is needed in order to achieve the desired behavior. (Some engineering disciplines including civil engineering and software engineering use the "architectural blueprint" or "architectural design" (short architecture) as central construction plan. In the following the term "architecture" is used as synonym for the central construction plan of all engineering disciplines.) All mature engineering disciplines have developed standardized construction languages for architectural description. In mechanical engineering, for example, a dozen standards exist on how to design construction plans [9]. These standards are subject to early stages of mechanical engineering education since they are an essential means of communication.

2.3. Division of Labor

Besides structuring the system to be designed, the construction plan is used to structure the design process: the components of a system are constructed in teams and then assembled in order to become a whole according to the architecture. The division of labor during the construction process is a core feature of classical engineering disciplines, since it is the only way to construct complex systems in large teams.

2.4. Architectural Design

Designing the architecture is the supreme discipline in engineering, which involves the transformation of requirements (problem space) into a high level blueprint of the system to be designed (solution space). Designing the architecture involves fundamental design decisions which have impact on the whole design process. An example can be found in the definition of quality characteristics that the system to be constructed must address (e. g. Which changes to the system can be made easily, which not? What is the system's performance? What is the capacity of the system? How scalable is the system?).

Due to the mentioned responsibilities, great attention is paid to architecture and only experienced and highly qualified engineers are involved in the architectural design. By involving internal and external experts as well as complex analysis frameworks, engineers seek to ensure the quality of the architectural blueprint so that the architecture addresses all the required characteristics of the system to be designed.

3. An Engineering Based Approach to Enterprise Architecture

Following the above introduced characteristics of mature engineering disciplines, requirements for an engineering driven approach to EA can be derived. EA can be regarded as the central construction plan for organizational transformation in a “business-to-IT” approach. EA describes the main business and IT components as well as their relationships (c.f. “standardized construction plan” in classical engineering). EA is the result of important design decisions and determines fundamental characteristics of the organization, such as strategic positioning, business process efficiency and effectiveness, business/IT alignment, and information systems capabilities. Indirectly, EA therefore implies e.g. an organization’s capability to rapidly launch new products, to adapt to new regulations, or to exploit business potentials of IT innovations (c.f. “architectural design” in classical engineering).

Following engineering principles, concrete requirements of internal and external stakeholders build the starting point for EA design. Stakeholders may e.g. contribute model information and also consume information of the EA. As far as designing stakeholders are concerned, conventions (c.f. “standardized construction language”) and governance are vital to enable distributed but consistent design (c.f. “division of labor” in classical engineering). Designing EA does not imply to create new models from scratch, but to integrate and aggregate existing knowledge from architectural parts (c.f. “engineering knowledge patterns” in classical engineering). Not all of the stakeholders’ concerns and requirements have effects on the fundamental structure of the organization (or EA), but they partially might still have influence as architectural drivers.

There exist different classes of architectural drivers. One class focuses on the functional development of the organization. Examples can be found in the opening of new markets and sales channels or business process outsourcing. Another class of architectural drivers focuses on optimization of organizational structures, e. g. by consolidation of redundant structures or reuse of existing resources to improve flexibility and prepare the organization for possible future changes.

Architectural drivers tend to have tradeoffs which require compromises in the architectural design. Priorities of the architectural drivers are subject to changes which might cause discontinuities in organizational development. A merger, for example, might change any given situation to set the focus on architectural consolidation.

The sketched complexity of the matter often causes difficulties for enterprise architects to choose the appropriate artifacts and relationships for the EA model. From an engineering perspective and taking experiences from EA projects in companies into account, the following heuristics can be derived.

3.1. Criterion of Width

EA models must address the information demand of their stakeholders. Information demands are implied by management tasks (concerns) of the respective stakeholders. EA can for example deliver crucial data for project portfolio management to support decision making, concerning investment decisions for business applications.

A successful method for stakeholder involvement turned out to be the collection and analysis of precise questions that stakeholders have, e. g. “Can investments in applications by justified by additional revenue, gained from the product or service which is supported by this application?” Situational fragments of the EA model (viewpoints) can help to answer such questions by representing the desired information on an aggregate level and in a form of representation which is appropriate for the respective stakeholder.

Following the criterion of width, all artifacts and relationships needed for the creation of view-points must be reflected in EA. The sum of information demands of all stakeholders therefore determines the maximum EA extent.

3.2. Criterion of Depth

When EA is only designed in respect to the criterion of width, chances are high that a huge number of detailed structures of implementations or detailed inventories of single artifacts types are included.

Architecture strategies which are derived from the architectural drivers, and the desired characteristics of the whole system should also be included in EA. These architecture strategies need to be expressed and documented, so that their realization is measurable. Architecture strategies focus on the entire system or on groups of similar artifacts (This heuristic is based on the locality criterion, initially published by [5] and then adapted by [7] This criterion is adapted for enterprise architecture and informally described.) such as all core business processes, all data flows across domains, or all products which are distributed over a certain channel. Structures which only focus on implementation details of one artifact, and which are only relevant for this object, should not be a part of EA. Exceptions might be useful in certain situations, e. g. to support concerns of a key stakeholder.

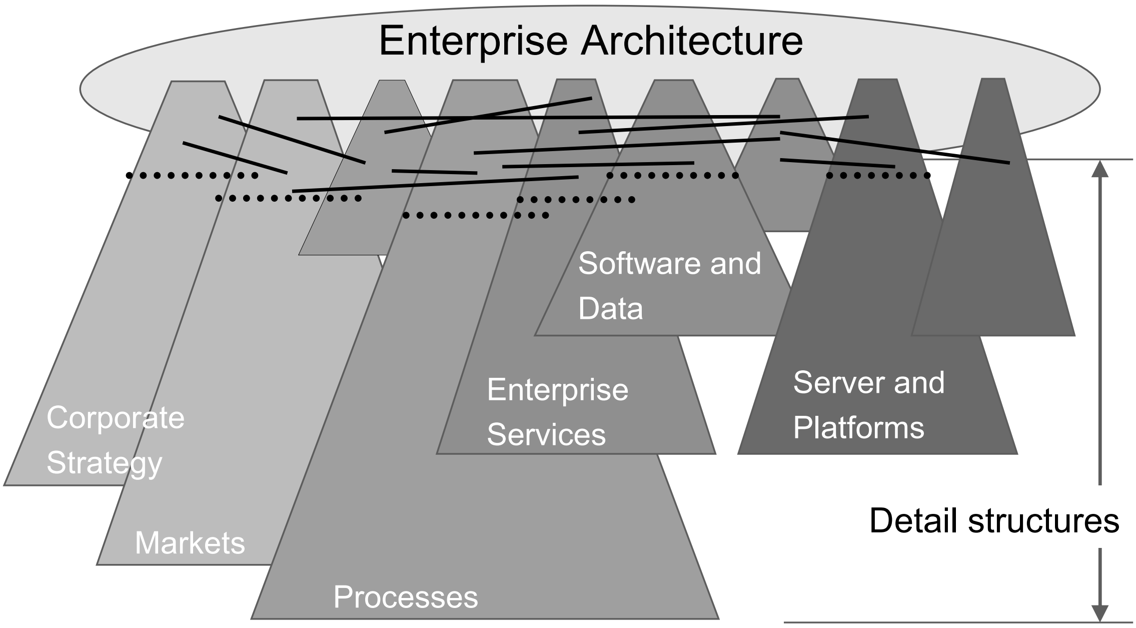
The relevance of an artifact can be indicated by the impact that a change of this artifact has on others (This heuristic is based on encapsulation and information hiding, which originates in object orientation (cf. e.g. [16]).). If a change of an artifact does not influence others at all, it should most likely not be included in EA. Following the idea that EA is the blueprint for change projects, problems can arise from making unnecessary design decisions for the entire architecture which should be better made for individual projects. Therefore, details such as object oriented class structures, detailed data structures, mapping information of network adaptors to servers, structures of teams in individual business units, workflow specifications of business processes, or construction details of products should not be part of EA. Figure 1 illustrates our "broad and aggregate" understanding of EA.

In two cases it can be useful to include detail artifact structures in the EA model. In both cases, changes to the detail structure cause potential changes to other artifacts, which means that the above mentioned heuristic remains valid:

1. Relationships to other detail artifact structures: Examples can be found when deploying single software components on servers or assigning sub-goals to the responsible business units. A relationship on detail level (e. g. application component and server) can always be observed on the respective aggregated level (e. g. respective application and respective server cluster). Detail structures should only be included in EA when they have impact on design decisions with effect on the entire system. This is true for the deployment of application components on servers, since the explicit documentation of this relationship might have considerable impact on the ability of the organization to react in case of a blacked out computing center. An example for a relationship on detailed level without significant impact can be found in the assignment of application functions to detailed activities of a business process. In this case, the aggregate relationship between application and business process delivers sufficient information for EA purposes, while detail documentation can be misleading.
2. Objects on detailed level can be reused in multiple artifacts: Similar to the case above, the detail level should only be taken into account, if reuse has significant impact on the behavior of the entire system. This is the case in examples such as reuse of product components as part of a platform strategy. Contrary, it is not the case when reusing libraries in multiple applications.

Moreover, it cannot be recommended to include many objects of a detail structure which all have similar relationships within the architecture. This

Fig. 1: Enterprise Architecture is Broad and Aggregate



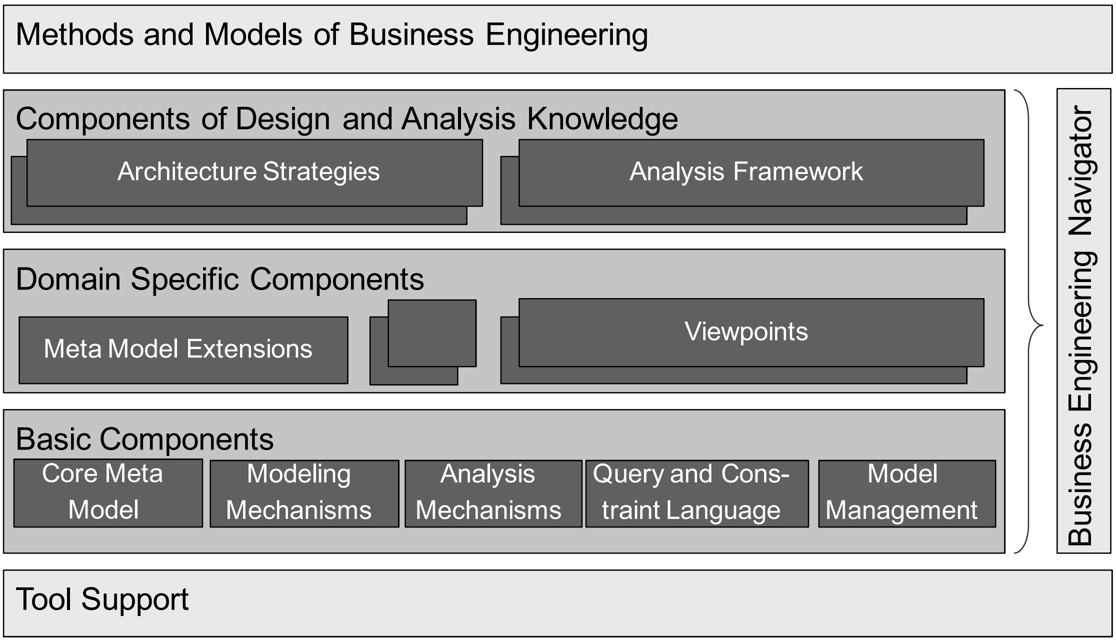

Methods and Models of Business Engineering
Components of Design and Analysis Knowledge
Architecture Strategies
Analysis Framework
Domain Specific Components
Meta Model Extensions
Viewpoints
Basic Components
Core Meta Model
Modeling Mechanisms
Analysis Mechanisms
Query and Constraint Language
Model Management
Tool Support
Business Engineering Navigator

Fig. 2: Architecture of the Business Engineering Navigator Approach

is e.g. the case when considering all client computers (as inventory).

3.3. Pragmatic Criterion

Organizations are subject to constant changes. Therefore EA models need to be updated regularly. Many projects show that continuous maintenance efforts incur high costs. Therefore it needs to be considered if the benefits resulting from covering a stakeholder concern exceed the costs necessary to gather and maintain this information. Not every stakeholder information demand which is claimed by the criterion of width will gain positive revenue. Therefore, the pragmatic criterion proposes to carefully analyze and evaluate the value of artifacts and relationships. No maintenance efforts should be put into artifacts which are not necessary for any concerns [8].

Quantifying costs and benefits of information demand is far from trivial [e.g. 17]: Benefit analysis often results in "reverse" considerations (what if we did not have this information?). Costs arise according to type, origin, necessary conditioning efforts, and frequency of usage. Information demands being served from the same pool of data might realize considerable synergies.

The main feature of the architecture is to provide a high level plan to support long term strategic development of an organization. High frequency in changes of detail information incurs high maintenance costs and can be used as an indicator that the level of aggregation is too low. From our experience, in most cases it is sufficient to use and maintain more aggregate structures (as proposed in the criterion of depth). Usually, high level models can be maintained manually with reasonable efforts, i. e. without having to develop and use automated interfaces to detail repositories (such as configuration management database, process model repository, product configuration system). However, there may be use cases where more detailed model data is needed, automated data imports might be necessary to provide an efficient solution at reasonable maintenance efforts.

4. Business Engineering Navigator

BEN structures the various components of engineering support for EAM. BEN is based on the above mentioned principles of engineering and addresses the main requirements of EAM.

Figure 2 illustrates the components of BEN and their assignment to abstraction layers. This structure can be used as a framework for practical as well as research projects. The components are described in the following subsections.

4.1. Basic Components

Basic components include domain independent functionalities which are used to model, analyze and design EA.

- Core meta model: A common set of vocabulary is a major prerequisite to consistently design the five layers of the business engineering framework. The BEN meta model is based on generic modeling methods and contains artifacts on a strategy layer, organizational layer, integration layer, software layer, and an IT infrastructure layer [22]. This meta model serves as a standardized construction language for organizational transformation.
- Modeling mechanisms: A domain independent description language provides basic mechanisms to create models of the design artifacts. This includes hierarchical refinement of artifacts using "part-of" and "is-in" relationships as well as domain clustering.
- Analysis mechanisms: Generic types of analyses and analysis mechanisms are instantiated for each concrete viewpoint (cf. below). Examples for generic types of analyses include matrix analysis, dependency diagrams, list reports, architecture views, and spider web diagrams [3].
- Query and constraint language: A query language is needed to analyze the models using predefined and ad-hoc queries. Using the constraint language, the architecture strategy and the architectural principles are specified and verified. Both languages are based on formalized modeling mechanisms, e. g. relational algebra.
- Model management: This basic component includes version management functionalities, such variants handling and model history. These aspects are crucial to model life-cycle management

4.2. Domain Specific Components

Domain specific components are instances of generic components for the five different layers listed in section 4.1.

- Meta model extensions: Specific extensions of the core meta model allow the application of the engineering approach in specific contexts (e. g. a certain industry, a certain company size or maturity level) and in specific projects (e. g. business driven changes, IT driven changes, alignment projects).
- Viewpoints: A viewpoint catalogue is comprised of generic analysis mechanisms and types of analyses which are suited to given stakeholder information demands. Queries needed for each viewpoint can be formulated using the above introduced query language [11].

4.3. Components of Design and Analysis Knowledge

Components of design and analysis knowledge help to keep record of the engineers' knowledge.

- Architecture strategies: Generally valid and accepted design patterns and architectural strategies (e.g. handling of redundant master data) and principles can be organized as knowledge repositories [4].
- Analysis framework: An analysis framework implements models of quality and metrics for the design artifacts (e.g. analysis frameworks which help to refine aggregate targets, such as efficiency, into measurable counts, such as scalability, avoidance of redundancies, capability for multi channel usage [19]). Results of the analysis are represented as viewpoints.

The BEN approach proposes to adapt EAM to the respective application scenarios of the respective organization. Therefore, generally valid and accepted components of design and analysis knowledge must be adapted, extended and integrated.

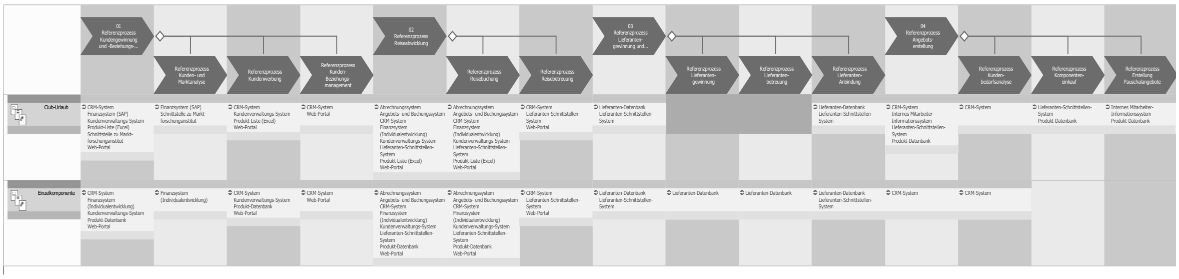
The BEN approach can be understood as interface between methods of business engineering and underlying software tools: On one hand, BEN defines requirements for software systems and gives assistance how to use them in the context of the engineering discipline. On the other hand, BEN is a service layer for different methods, which may give concrete guidance in change and transformation for organizations.

5. Tool Support: A BEN Implementation for Documentation, Analysis and Design of Enterprise Architecture and Learnings from First Applications

Regarding the criterion of width, EA addresses a variety of stakeholders with different information demands and different views on EA. Therefore the implementation of the basic components of BEN (cf. section 4.1) requires a specific tool support where BEN can serve as a foundation for the implementation or configuration of EA software tools. ADOben is such an implementation of BEN requirements based on ADONIS, a commercial modeling tool and meta-modeling platform.

ADOben implements the required model types from a strategy layer down to an IT infrastructure layer as well as the interdependencies between the artifacts and models on these layers. Therefore it is possible to design an architecture plan for the as-is situation. Using means of architecture analysis and a dedicated architecture strategy, a blueprint for the to-be situation can be designed.

To support the application scenarios of potential EA stakeholders, the tool implements



	01 Referenzprozess Kundengewinnung und -Beziehungs-...	Referenzprozess Kunden- und Marktanalyse	Referenzprozess Kundenwerbung	Referenzprozess Kunden-Beziehungs-management	02 Referenzprozess Reiseabwicklung	Referenzprozess Reisebuchung	Referenzprozess Reisebetreuung	03 Referenzprozess Lieferanten-gewinnung und...	Referenzprozess Lieferanten-gewinnung	Referenzprozess Lieferanten-betreuung	Referenzprozess Lieferanten-Anbindung	04 Referenzprozess Angebots-erstellung	Referenzprozess Kunden-bedarfsanalyse	Referenzprozess Komponenten-einkauf	Referenzprozess Erstellung Pauschalangebote
Club-Urlaub	CRM-System Finanzsystem (SAP) Kundenverwaltungs-System Produkt-Liste (Excel) Schnittstelle zu Markt-forschungsinstitut Web-Portal	Finanzsystem (SAP) Schnittstelle zu Markt-forschungsinstitut	CRM-System Kundenverwaltungs-System Produkt-Liste (Excel) Web-Portal	CRM-System Web-Portal	Abrechnungssystem Angebots- und Buchungssystem CRM-System Finanzsystem (Individualentwicklung) Kundenverwaltungs-System Lieferanten-Schnittstellen-System Produkt-Liste (Excel) Web-Portal	Abrechnungssystem Angebots- und Buchungssystem CRM-System Finanzsystem (Individualentwicklung) Kundenverwaltungs-System Lieferanten-Schnittstellen-System Produkt-Liste (Excel) Web-Portal	CRM-System Lieferanten-Schnittstellen-System Web-Portal	Lieferanten-Datenbank Lieferanten-Schnittstellen-System			Lieferanten-Datenbank Lieferanten-Schnittstellen-System	CRM-System Internes Mitarbeiter-Informationssystem Lieferanten-Schnittstellen-System Produkt-Datenbank	CRM-System	Lieferanten-Schnittstellen-System Produkt-Datenbank	Internes Mitarbeiter-Informationssystem Produkt-Datenbank
Einzelkomponente	CRM-System Finanzsystem (Individualentwicklung) Kundenverwaltungs-System Produkt-Datenbank Web-Portal	Finanzsystem (Individualentwicklung)	CRM-System Kundenverwaltungs-System Produkt-Datenbank Web-Portal	CRM-System Web-Portal	Abrechnungssystem Angebots- und Buchungssystem CRM-System Finanzsystem (Individualentwicklung) Kundenverwaltungs-System Lieferanten-Schnittstellen-System Produkt-Datenbank Web-Portal	Abrechnungssystem Angebots- und Buchungssystem CRM-System Finanzsystem (Individualentwicklung) Kundenverwaltungs-System Lieferanten-Schnittstellen-System Produkt-Datenbank Web-Portal	CRM-System Lieferanten-Schnittstellen-System Web-Portal	Lieferanten-Datenbank Lieferanten-Schnittstellen-System	Lieferanten-Datenbank	Lieferanten-Datenbank	Lieferanten-Datenbank Lieferanten-Schnittstellen-System	CRM-System	CRM-System		

Fig. 3: Three Dimensional Matrix Report in ADOben

the respective queries and visualizes their results. The following example illustrates an application scenario in which a business analyst plans the launch of a new product. Information demands of the business analyst could be: "Do we have adequate application support for the new product?", "Where are potential breaks between applications along the process?" Using the query "Which applications are used in which process for which product?" on the architecture model, a matrix report in three dimensions as shown in Figure 3 is created. The matrix shows the products and processes as well as the underlying applications.

Based on a generic core meta model and generic analysis mechanisms as well as specific extensions for a defined application scenario, every other query could be run on the underlying models and visualized in a report.

Since BEN is not particularly developed for EAM, the generic concepts (as presented in section 4) could also be implemented in different tools and for other business engineering methods. As a first means of feasibility evaluation the BEN approach has been implemented in a German financial service provider using ADOben. The application of the approach verified that EA should be positioned as a planning tool, not as a tool focused on operative tasks (like for example a configurations management database system triggering an alarm when a server hard disk fails). To achieve this, the three criteria defining EA scope have proven to be valuable. The criterion of width requires that the EA meta model and the viewpoints are developed in close collaboration with all stakeholders of the EA. To get the buy-in of the stakeholders, the introduction of EAM should be taken as a chance to revise the planning and documentation processes within the organization in order to ensure that the EAM organization concept is integrated seamlessly and does not cause an overhead work load for the stakeholders. The analysis capabilities of ADOben, especially matrix analyses have turned out to be a valuable tool to foster and rationalize the communication between the IT unit and the business units as well as to systematically address alignment questions between business structures and IT structures.

6. Conclusion

Based on analysis of classical engineering disciplines, this paper presents an engineering approach to EAM which has been generalized as BEN. It is shown how EA models can be constructed based on stakeholder requirements in order to create a pragmatic solution representing a "broad and aggregate", business-to-IT architecture – and not a set of enterprise-wide detail models which will never be completed and soon be outdated. BEN delivers a foundation for efficient EA design and EAM. BEN can be implemented in software tools and applied using business engineering methods to enable structured solution design.

Engineering disciplines in general, BEN and ADOben show that the engineering of complex environments involves a complex 'mechanism'. This mechanism can be evaluated according to its applicability and to its connectivity to other approaches, tools, and methods. The development of this mechanism is aimed at a clear structure so that elements can be arranged according to the respective situation as a best-of-breed solution. This means that ADOben is one solution to implement BEN as an EAM tool. At the same time BEN is not limited for the use in the context of EAM. The core idea is to ensure structured engineering. Further research activities in this area will focus on the methods themselves and their situational character. The ultimate goal is to provide engineering support for the situational development and maintenance of "business-to-IT" solutions – in the context of EAM, but also for integration management, for information logistics management, for IT/business alignment and other scenarios in information management.

References

[1] Arbab, F., de Boer, F., Bonsangue, M., Lankhorst, M., Proper, E., & van der Torre, L.: (2007). Integrating Architectural Models. Symbolic, Semantic and Subjective Models in Enterprise Architecture, Enterprise Modelling And Information System Architectures, 2(1), 40-56.

[2] Avallone, E.A., Baumeister, T., & Sadegh, A.: (2007). Marks' Standard Handbook For Mechanical Engineers. Mcgraw-Hill Professional.

[3] Bucher, T., Fischer, R., Kurpjuweit, S., & Winter, R.: (2006). Analysis and Application Scenarios of Enterprise Architecture - An Exploratory Study, EDOC Workshop on Trends in Enterprise Architecture Research (TEAR 2006), Tenth IEEE International EDOC Conference (EDOC 2006). Hong Kong: IEEE Computer Society.

[4] Buckl, S., Ernst, A.M., Lankes, J., & Matthes, F.: (2008). Enterprise Architecture Management Pattern Catalog.

[5] DeRemer, F., & Kron, H.H.: (1976). Programming-in-the-large versus programming-in-the-small, IEEE Transactions on Software Engineering, 2(2), 80-86.

[6] Dubbel, H., Kuttner, K.H., & Beitz, W.: (1994). Dubbel. Handbook of Mechanical Engineering. Berlin: Springer.

[7] Eden, A.H., & Kazman, R.: (2003). Architecture, Design, Implementation, International Conference on Software Engineering (ICSE). Portland, OR:

[8] Fischer, R., Aier, S., & Winter, R.: (2007). A Federated Approach to Enterprise Architecture Model Maintenance, Enterprise Modelling and Information Systems Architectures, 2(2), 14-22.

[9] Giesecke, F.E., Mitchell, A., Spencer, H.C., Hill, I.L., Dygdon, J.T., Novak, J.E., & Lockhart, S.D.: (2008). Technical Drawing. Denver, CO: Pearson Education.

[10] Harry, M.J.: (1988). The Nature of Six Sigma Quality. Rolling Meadows, IL: Motorola University Press.

[11] IEEE: (2000). IEEE Recommended Practice for Architectural Description of Software Intensive Systems (IEEE Std 14712000). New York: IEEE Computer Society.

[12] IFIP-IFAC Task Force on Architectures for Enterprise Integration: (2003). GERAM - The Generalised Enterprise Reference Architecture and Methodology. In Bernus, P., Nemes, L., Schmidt, G. (Eds.), Handbook on Enterprise Architecture. (pp. 22-62). Berlin et al.: Springer.

[13] Johnson, P., & Ekstedt, M.: (2007). Enterprise Architecture - Models and Analyses for Information Systems Decision Making. Pozkal: Studentlitteratur.

[14] Kurpjuweit, S., & Winter, R.: (2007). Viewpoint-based Meta Model Engineering, Enterprise Modelling and Information Systems Architectures - Concepts and Applications, Proceedings of the 2nd Int'l Workshop EMISA 2007. Bonn: Gesellschaft für Informatik, Köllen.

[15] McGuire, J.G., Kuokka, D.R., Weber, J.C., Tenenbaum, J.M., Gruber, T.R., & Olsen, G.R.: (1993). SHADE: Technology for Knowledge-based Collaborative Engineering, Concurrent Engineering, 1(3), 137-146.

[16] Parnas, D.L.: (1972). On the criteria to be used in decomposing systems into modules, Communications of the ACM, 15(12), 1053-1058.

[17] Schekkerman, J.: (2005). The Economic Benefits of Enterprise Architecture: How to Quantify and Manage the Economic Value of Enterprise Architecture. Victoria, British Columbia: Trafford Publishing.

[18] Schekkerman, J.: (2004). How to Survive in the Jungle of Enterprise Architecture Frameworks: Creating or Choosing an Enterprise Architecture Framework. Victoria, British Columbia: Trafford Publishing.

[19] Schelp, J., & Stutz, M.: (2007). A Balanced Scorecard Approach to Measure the Value of Enterprise Architecture, Journal of Enterprise Architecture, 3(4), 8-14.

[20] Shaw, M.: (1990). Prospects for an Engineering Discipline of Software, IEEE Software, 7(6), 15-24.

[21] Winter, R.: (2008). Business Engineering - Betriebswirtschaftliche Konstruktionslehre und ihre Anwendung in der Informationslogistik. In Dinter, B., Winter, R. (Eds.), Integrierte Informationslogistik. (pp. 17-38). Berlin, Heidelberg: Springer.

[22] Winter, R., & Fischer, R.: (2007). Essential Layers, Artifacts, and Dependencies of Enterprise Architecture, Journal of Enterprise Architecture, 3(2), 7-18.

[23] Zwicky, F.: (1948). Morphological Astronomy, The Observatory, 68(845), 121-143.

Stephan Aier
Institute of Information Management,
University of St. Gallen,
St. Gallen, Switzerland,
E-Mail: stephan.aier@unisg.ch,
Phone: +41 71 224 3360

Stephan Kurpjuweit
Institute of Information Management,
University of St. Gallen,
St. Gallen, Switzerland,
E-Mail: stephan.kurpjuweit@unisg.ch,
Phone: +41 71 224 3316

Jan Saat
Institute of Information Management,
University of St. Gallen,
St. Gallen, Switzerland,
E-Mail: jan.saat@unisg.ch,
Phone: +41 71 224 3782

Robert Winter
Institute of Information Management,
University of St. Gallen,
St. Gallen, Switzerland,
E-Mail: robert.winter@unisg.ch,
Phone: +41 71 224 2190

Available online at www.enterprise-systems.net

AIS Transactions on Enterprise Systems 1 (2009) 1, 44-51

Constraints in Conceptual Modeling – A Generic Approach

Martin Juhrisch, Werner Esswein, Gunnar Dietz

Abstract

The increasing use of information models beyond the technical aspects of software and the rising number of model creators as well as the comparability requirement for models resp. model evaluation necessitate a convention-orientated constructive restriction of the degree of freedom in specific conceptual models. In this paper a generic method is represented that by adopting of Description Kits allows a restriction of freedom in modeling regarding the aspects of natural language in specific conceptual models and enables a restrictive use of existing modeling languages. As a use case the configuration of service-orientated architectures is discussed. By using Description Kits we can link SOA services to conceptual models and therewith facilitate the alignment of business processes and the supporting information technology.

Keywords: Conceptual Modeling, Description Kits, Service-oriented Architectures, Web service.

1. Introduction

In the literature the restriction of freedom in conceptual modeling is especially discussed under the aspect of model integration in shared modeling projects [1]. Within this scope companies are systematically analyzed and reconstructed by means of semi-formal modeling languages [2].

However, both the high degree of freedom in modeling and the missing standardization of model elements in semi-formal modeling languages induce a lot of conflicts in model integration [3]. For example, references across different models cannot be disintegrated by different systems of concepts nor can the overlap areas between the sub-models be identified by different levels of abstraction. There are two approaches to avoid conflicts and enable comparability of models: first, the approaches use syntactic transformations and semantic tests for model transformation in order to identify overlap areas [4]. These approaches create only minimal demands on the existing model. Additionally, there are approaches starting from model construction in order to enhance the construction of simply comparable model artifacts. Approaches in this area are based on the assumption that the comparison of the models with a random structure hinders the identification of semantic overlap areas. Becker et al. confirm the fact that in order to get comparable results in a distributed modeling project the degree of freedom of a modeler has to be strongly restricted [1]. This means that if an identical situation comes up, different persons should use the same language constructs for the description [2].

Therefore, one basic hypothesis is the assumption that restrictions in modeling perceptibly simplify the later comparison of the sub-models [3]. As a consequence the conventions restrict the degree of freedom in modeling in order to guarantee a certain extent of compliance [5]. Empiric researches show that sub-models created in a shared modeling process differ from each other in used vocabulary, degree of abstraction and level of detail [HaSo06]. Therefore, the restriction of freedom in modeling with the purpose to establish comparable sub-models applies foremost to the elimination of synonyms and creation of semantically disjoint language constructs [3]. The demand on semantically disjoint language constructs avoids abstraction conflicts – all shared models use the same level of abstraction.

The automatic analysis of models represents the second reason for restricting the freedom in modeling. First, considering the expected diversity of models inside a process landscape, manual evaluation will be strongly complicated. Additionally, conceptual models prescriptively serve as a design for a target-structure of the organization and its information system that needs to be implemented. The challenge is to create a comprehensive model system that is valid at the moment of completion – i. e. that reacts on actual market requirements. Therefore, it is reasonable to be able to accomplish at least semi-automated analyses and evaluations based on the ascertained process landscape as an indicator for future reorganization actions resp. to prepare conceptual models in a way that these could serve as an input for a transformation process in design models. For the automatic analysis the requirement for comparability mentioned above is an essential requisite [3]. Additionally, language constructs must have both a suitable level of abstraction and domain closeness in order to be valuable regarding the content.

In the present paper a new approach is represented that generically enables a development of domain-specific language constructs and a limited use of these in conceptual models. The origin of the method is in the adaptation and optimization of the service-orientated architectures at the University of Muenster [6]. Organizational target-models resp. functional requirements models must fulfill the role as input in a translation in service compositions.

The article is structured as follows: the next chapter shortly introduces the motivation for restricting the degree of freedom in conceptual modeling. In Chapter 3 an own method is presented that establishes a basis for the generic restriction of the degree of freedom in modeling. An application example for configuration of service-orientated architectures is provided at the end of the article and the main ideas and open issues are summarized in the future research.

2. Constraints in Conceptual Modeling

The essence of the restriction of the freedom in modeling in terms of modeling certain aspects in the information model is the limitation of language vocabulary to an amount of domain-specific, semantically disjoint language constructs. With this not only the designed information model but also already the modeling language has a semantic connection to the application domain [3]. Therefore, on the application point of view semantically meaningful operations in conceptual models can be defined already at a language level.

One example for this is the specific problematic situation in case of mapping distributed processes in public institutions. PICTURE [1] presents a method that can be used for restricting the freedom in modeling to an amount of domain-specific language constructs. With the elimination of type, synonym, homonym and abstraction conflicts the semantic model comparison can be reduced to the syntax level [4]. The restriction of the available terminology is flanked by a specific process modeling language with a simple syntax.

Because of the fact that the language constructs are deduced from specific technical languages, the specialists – in their role as a modeler – are aware of the semantic of the language constructs. The choice of the unsuitable modeling constructs is avoided and the interoperability of the models enabled.

However, a natural language is always established over time; its vocabulary and grammar are not static but a result of the language use in a speech community. The determination of domain-specific language constructs at meta-model level seems to be problematic against the background of dynamic change. Furthermore, domain-specific modeling languages do not have a sufficient amount of language constructs to picture all the phenomena of the respective domains. Consequential a vast number of domain-specific modeling languages are necessary. The approach introduced here is focusing on the development of situation-dependent adaptable language constructs designed for a project in a certain operational domain – possibly also only for a short period – and adapted in the course of time. For this reason, either existing language constructs are adapted or entirely new ones are built from existing language fragments. This is enabled by distinguishing between object and meta-modeling language.

If following the Meta-Object-Facility (MOF) Architecture of Object Management Group (OMG) a language is used in the system development in two phases [7]:

- On the one hand it is used in modeling at object model level when analyzing and documenting the discourse area – e.g. the business domain.

- The phase of the language generation on the other hand references the description of the syntax and semantic of a modeling language under the use of meta-language which again can be interpreted as a modeling language. In this case we talk about a meta-model (M2M) [8].

Since a domain-specific modeling language – created as an artificial artifact by a method developer – possesses a modifiable grammar it can moreover be freely adapted to certain conditions. Software supporting the development of a modeling language is summarized in the concept of Meta-CASE Tools. Well-known tools are for instance MetaEdit+ [9] or cubetto® Toolset [10].

3. The Description Kit Approach

In the present paper Description Kits (DK) are introduced that cover restricted describable ancillary information in adequately enriched conceptual models. DK represent the consensus of the speech community in terms of the amount and structure of certain linguistic concepts relevant for the business analysis. The DK approach is generic enough to restrict every kind of modeling information in their description relating to the present modeling purpose. Concrete descriptions of business information in analysis models concretize the imagination of the modeler at purely linguistic level within the scope of given DKs.

3.2. Description Kit Language

The DK approach centers the phase of the language generation (see Figure 1). In the meta-model level at layer 1 the creation of the so-called Description Kit Language (DKL) follows. Here the syntax of every DK is determined. This contains the hierarchization of different DK concepts [11] as well as the determination of their usage. The latter require a linkage of the meta-model of a conceptual modeling language – already existing by the time – to the model of the DKL. It is determined which DK concepts are possible or obligatory to which elements of the model. So the DKL is associated with the meta-model of the goal language.

The DKL can be kept generic in a way that one or more description languages of this kind are created only once in advance and these are then used in different contexts. The ideal case is that there exists a DKL that is generic in a way that each modeling information dependant on the existing modeling purpose can be modeled as a restricted domain-specific language construct.

The DKL represented in this paper distinguishes between composite and atomic DK. Composite DKs are again an aggregation from other DKs from a lower hierarchical level. In contrast, atomic DKs do not consist of other kits. Furthermore, a DK (composite or atomic) in terms of meta-entity can be accumulated by parameters. Parameters are optional components of a DK and have to be completed when modeling at level 0. To all the parameters of an atomic DK optionally either definite values or placeholders can be allocated. For parameters placeholders can be specified for free selectable features and for a not-empty amount of possible parameter value allocations.

However, a DKL would be possible as well that – in terms of modeling the role information – determines the concept of „role“. With this no domain-specific language constructs are modeled

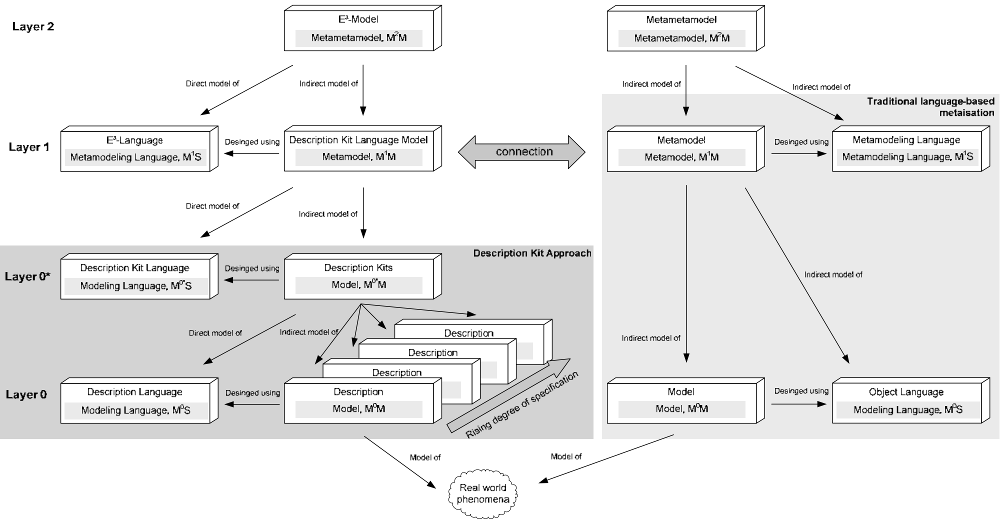


Fig. 1. Classifying the Description Kits in traditional language-based metaisation (see also [12])

at meta-model level but the language for DKs of these constructs. An extension of a meta-modeling language as recommended in the literature [11] is no more necessary.

A forced usage of certain DKs for certain modeling elements can additionally not only mean a restriction of the freedom in modeling in terms of the DKs itself but also in terms of the modeling.

3.2. Modeling of Description Kits

Using a DKL the definition of concrete DKs follows at level 0*. The instantiation from level 1 to level 0* is seen in terms of „this DK corresponds to this and that concept" – and determines so also the usage of the DK in modeling (see Appendix). Depending on the DKs these could be concrete types of roles or documents (e-mail, letter, etc.) or concrete patterns [11]. These are adaptive language constructs that are partly representing natural language concepts making guidelines for the specification of concrete descriptions at layer 0. As already in the pattern approach of [11] the limitation to the concepts established at this layer means to partly resolve the mentioned language conflicts (see Chapter 2).

In the example of role information concrete roles needed to be set. Each role can again be developed differently and completed with different parameters (i. e. field of study by student or institution by a member of staff). Of course this could also be modeled as wildcard parameter. But in the same way organizational structure at level 0* could be applied additionally to the concrete roles so that when modeling only the organizational unit has to be attached to the role "member of staff".

After defining a DK, limited describable model information in the model at level 0 can be modeled. So in a DK the amount of all the description possibilities of a limited domain-specific language constructs is determined which can be modeled later in a concrete description and concretized by setting parameters. Hence, the collectivity of DKs determines the vocabulary and all the possible conditions of domain-specific language constructs. Every instance of a DK – a description – contains a unique name and is a part of the specific conceptual model on layer 0. A DK is instantiated by selecting a set of parameters and sub-DKs of the corresponding DK and by setting certain parameter values. Hence, all the possible descriptions permitted by the DK are concretized when setting the parameters or allocating DKs from lower hierarchical levels.

At each point in time of modeling at layer 0 always exactly one configuration of a DK at layer 0* is active. An existing DK configuration can be modified during the modeling on layer 0. As already mentioned above, the connection possibilities of the DK are determined with the concepts of the conceptual modeling language at meta-model level and these stay constant. Therefore, in case of modifications the DKL should be not affected. In this case, only the DKs of an adaptation would be affected. Modifications in a DK for example when defining additional parameters transfer one DK of one configuration in another one. The adaptation operations should enable an adjustment without endangering the consistency of the existing specific conceptual models. It is problematic insofar as when modeling the DKs no assumptions are made about when certain adaptation operations are implemented.

The advantage of the approach presented here is the generic modeling of domain-specific language constructs. At the same time, the further use of existing modeling languages is possible. With adoption of the additional layer 0* an approach is found that makes an extension of a meta-modeling language like E^3-model unnecessary. A support of the closeness of levels 0* and 0 by the modeling tool makes the adaptation process significantly easier than in the pattern-approach used till now [11]. The possibility to define different kind of DKLs offers a higher flexibility than the original (already generic in principle) pattern approach and enables a direct implementation of approaches such as enrichment of models with role information in terms of [13].

4. Use Case: Configuration of Service-oriented Architectures

With the continuous consolidation of Web service standards a change away from the implementation and deployment of services to the point of service management is taking place in the discussion about the service-oriented architectures. The indicators: a number of standardization requests and the number of large projects in this area are evident for a growing demand on management methods for illustrating business requirements on service compositions. An application of the Description Kit approaches is represented that enables a targeted coordination between the relevant business processes and SOA in specific conceptual models.

Referred to the abstraction levels of the Model Driven Architecture MDA [14] the illustration classifies as a transformation task between computation independent models (CIM) and platform independent models (PIM). Now, from the present modeling purpose it can be deduced that an automatic transfer requires a translation of real-world phenomena in a form that enables an appropriate automation. Reduction of semantic heterogeneity inside the modeling elements in CIM and PIM is the deciding step to an automatable transfer. The main hypothesis is that a connection can be established between an organization and IT domains, provided that an amount of linguistic concepts are used simultaneous in CIM and PIM.

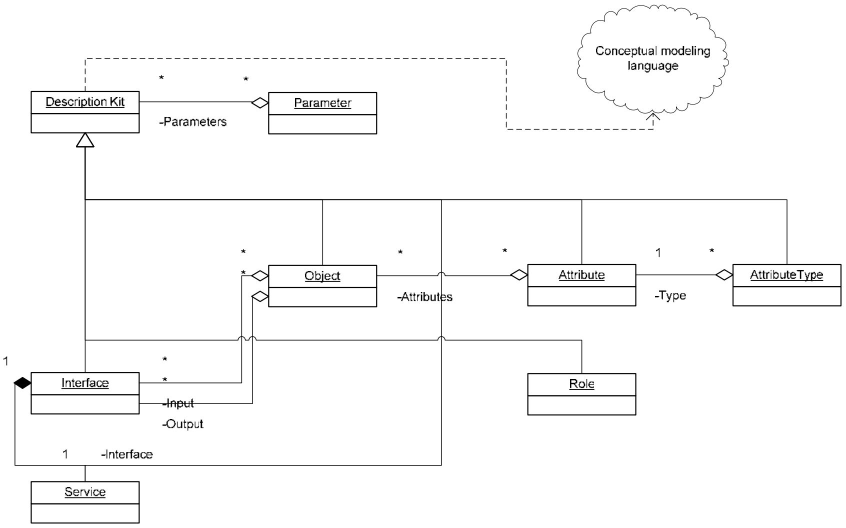


Fig. 2. Description Kit Language (Object, Service, Role)

Therefore, in the next part of this paper an exemplary use of the DK approach to overcome semantic gaps between semi-formal problem description in CIM and formal solution in PIM is used. The goal is to reach the comparability of CIM and PIM.

For the modeling aim here the freedom in matters of modeling of business objects resp. parts of them is restricted. These were chosen as a concept because first, they have an importance both in the analysis and in the design phase in order not to have bad influence to the ordinary modeling task, and second, because they qualify well for an automatic analysis for service candidates in CIM and for adjustments between CIM and PIM. In case of DK method the following steps need to be taken:

- Level 1 (meta-model level): At meta-model level a hierarchic DKL is developed. Figure 2 illustrates a DKL that serves for the modeling of Object, Service and Role Description Kits at level 0* (see Figure 2). The DKL contains general DK concepts and (not shown) the relation to the modeling language used. The concept Description Kit has to be emulated on the existing meta-model of the modeling language – in the present case it is referred to the meta-model of the event-driven process chain (EPC) of ARIS method [Sche00].
- Level 0*: On level 0* concrete DKs are defined using the concepts of layer 1. DKs for business objects, services and role at level 0* are developed (see Appendix). These DKs are adaptive, thus they can be adapted to the modified requirements to the modeling of business objects during the modeling at object model level. In this case, DKs build adaptive model elements corresponding to the consensus between software architects and specialists concerning the amount and structure of certain linguistic concepts relevant for the analysis and the design.
- Level 0 (model level): On the normal modeling level (level 0) the DKs are used to define concrete descriptions (as student) and concrete objects (as the HISSOSService) resp. concrete object states (as an active resp. inactive student). Thus, the modeling of objects and services will be limited to the vocabulary predetermined by the DK.

A description at level 0 is interpreted as a concrete business object or a part of it. The specification of a very certain object can be done when parametrizating DK instances which can combine logical operators with each other if necessary. Therefore, DK have a deciding effect on the activities during the modeling at an object model level because of the fact that the degree of freedom is limited to a premodeled vocabulary of adaptive modeling elements when describing business objects resp. parts of them, services and roles.

4.1. Conceptual Modeling with Descriptions

After defining DKs on layer 0* these can be used on ordinary modeling level (layer 0). To use them they have to be instantiated. The instantiation of a Description Kit creates a concrete description on the model layer. However, such a description doesn't need to describe a concrete real world phenomena, but can also describe only some aspects. E.g. in Figure 3 the description "Student" is still abstract in the sense that it doesn't represent a concrete student but the conditions to a person to be a student. On the analysis phase this description can be used for the definition of concrete business objects. For that reason, state transition models are used that – based on an extension of the EPC – describe functional requirements (see again Figure 3).

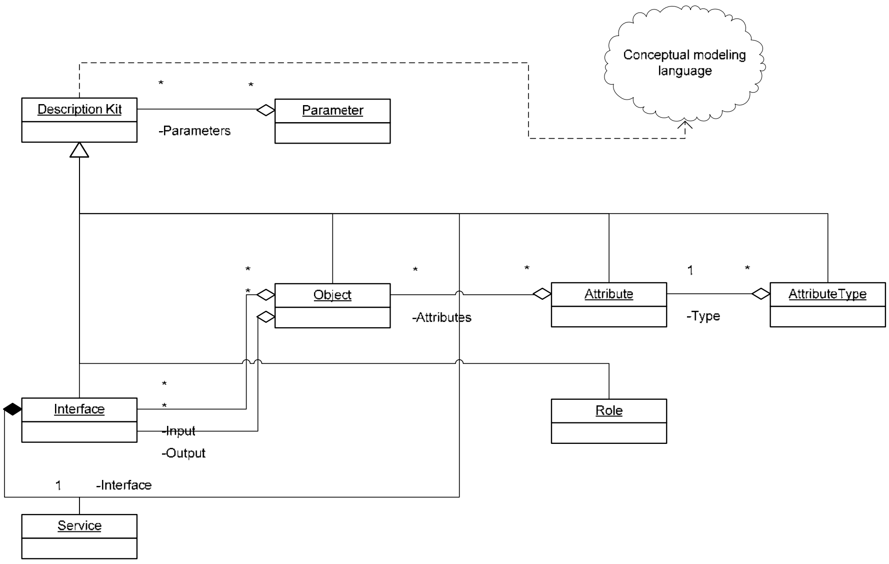


Fig. 3. Extending of EPC as state transition model

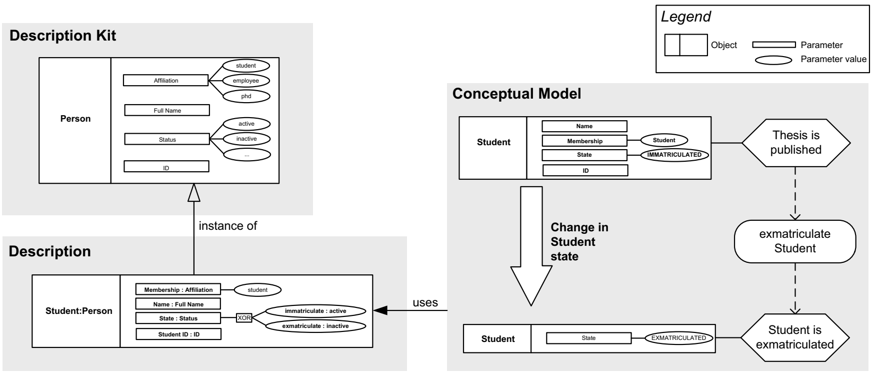
Thereby, a semantic relationship between an event and descriptions is created and interpreted as an operational condition. This becomes clear by gathering the conditions of the allocated descriptions. The business logic to be automated is then documented as an amount of situation transitions in descriptions.

4.2. Service Modeling with Descriptions

With the E^3-method [15] a model for services is created that includes an illustration of a service interface by means of graphic characters and links the method signatures with instantiated DKs "Object" (DKO).

When defining the message parameters the DKOs are again to be used. Due to the fact that when modeling CIM or PIM in both cases DKO instances are used, the bridge between these two domains will be closed. It is assumed that the service function of a Web service is suitable for a certain operational context if the initiating resp. resulting event of a process function in a CIM comprehends DKO instances with a condition according to the one existing in the model of the service function. The configuration automatically chooses suitable service functions and – if necessary – an adaptation of a target model is recommended to the modeler. As a result the EPC model is in the best case adapted to the existing SOA. An adapted target model can be transmitted as a reference model – professional solution with underlying implementation – to other organizations.

Fig. 4. Modeling of Web services using the Web service Description Kit



5. Future research

The approach introduced above achieves the first step towards a model driven configuration of SOA via business process models. Future work lies on the comprehensive evaluation in a pilot study conducted at the University of Munster. Going productive with cubetto® Toolset in the near future and involving decentral IT-support organizations of the university after that, the acceptance of the method will become apparent. The consistent usage of the modeling tool at the university will help to use it as a medium of choice for documenting and managing of SOA and to consider it as a part of the integrated information management.

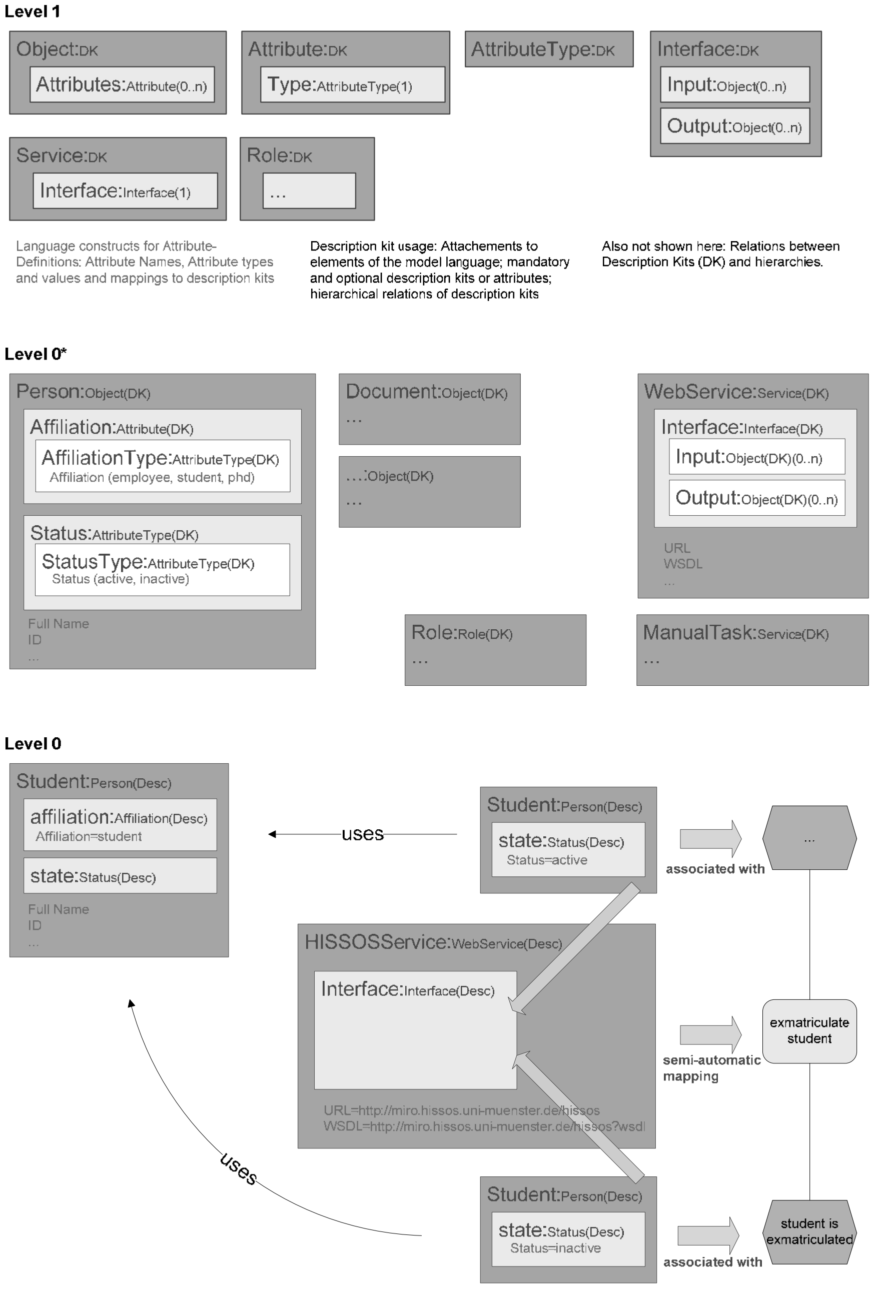

Level 1
Object:DK
Attributes:Attribute(0..n)
Attribute:DK
Type:AttributeType(1)
AttributeType:DK
Interface:DK
Input:Object(0..n)
Output:Object(0..n)
Service:DK
Interface:Interface(1)
Role:DK
...
Language constructs for Attribute-Definitions: Attribute Names, Attribute types and values and mappings to description kits
Description kit usage: Attachements to elements of the model language; mandatory and optional description kits or attributes; hierarchical relations of description kits
Also not shown here: Relations between Description Kits (DK) and hierarchies.
Level 0*
Person:Object(DK)
Affiliation:Attribute(DK)
AffiliationType:AttributeType(DK)
Affiliation (employee, student, phd)
Status:AttributeType(DK)
StatusType:AttributeType(DK)
Status (active, inactive)
Full Name
ID
...
Document:Object(DK)
...
...:Object(DK)
...
WebService:Service(DK)
Interface:Interface(DK)
Input:Object(DK)(0..n)
Output:Object(DK)(0..n)
URL
WSDL
...
Role:Role(DK)
...
ManualTask:Service(DK)
...
Level 0
Student:Person(Desc)
affiliation:Affiliation(Desc)
Affiliation=student
state:Status(Desc)
Full Name
ID
...
uses
Student:Person(Desc)
state:Status(Desc)
Status=active
associated with
...
HISSOSService:WebService(Desc)
Interface:Interface(Desc)
URL=http://miro.hissos.uni-muenster.de/hissos
WSDL=http://miro.hissos.uni-muenster.de/hissos?wsdl
semi-automatic mapping
exmatriculate student
uses
Student:Person(Desc)
state:Status(Desc)
Status=inactive
associated with
student is exmatriculated

Fig. 5. The Description Kit Approach (exemplified)

References

[1] Becker, J., Pfeiffer, D. and Räckers, M. (2007). PICTURE – A new Approach for Domain-Specific Process Modelling. Proceedings of the 19th International Conference on Advanced Information Systems Engineering (CAiSE 2007) Forum. Trondheim, Norway, pp. 45-48.

[2] Becker, J., Rosemann, M. and Schütte, R. (1995). Grundsätze ordnungsgemäßer Modellierung. Wirtschaftsinformatik, Vol. 37 (5), p. 435.

[3] Pfeiffer, D. (2007). Constructing comparable conceptual models with domain specific languages. Proceedings of the 15th European Conference on Information Systems.

[4] Gehlert, A. (2007). Migration fachkonzeptueller Modelle. Logos, Berlin.

[5] Rosemann, M. (2003). Preparation of Process Modeling. Process Management, Berlin, pp. 41-78.

[6] Böhm, B., Held, W. and Tröger, B. (2007). Integrated Information Management at the University of Munster. Changing Infrastructures for Academic Services, Bad Honnef.

[7] OMG (2002). Meta Object Facility (MOF) Specification, Version 2.0. http://www.omg.org/docs/ptc/04-10-15.pdf

[8] Frank, U. (1999). Conceptual Modelling as the Core of the Information System Discipline – Perspectives and Epistemological Challenges. Proceedings of the 5th Americas Conference on Information Systems, Milwaukee, pp. 695-697.

[9] Kelly, S., Rossi, M. and Tolvanen, J. P. (2005). What is Needed in a MetaCASE Environment? Enterprise Modelling and Information Systems Architectures, pp. 22-35.