Marite Kirikova

Janis Stirna (eds.)

CAiSE Forum 2012

Proceedings of the CAiSE'12 Forum at the 24th International Conference on Advanced Information Systems Engineering (CAiSE), Gdańsk, Poland, June 25-29, 2012

CEUR-WS.org/Vol-855

Preface

CAiSE 2012 was the 24th in the series of International Conferences on Advanced Information System Engineering. The theme of CAiSE 2012 is Information Services. The notion of service plays an increasingly extensive role in the enterprise development. Indeed, most organizations are based on the exchange of services: services to the customers and/or citizens, services to support the inter-organizational collaboration as well as services to accomplish intra-organizational activities. Many organizations are sharing services with others, interfacing services from others, or outsourcing their ICT resources to various locations worldwide aided by the internet. For all of them, the concept of service becomes a cornerstone of their collaboration, innovation and value creation. In this context, the information systems (IS) engineering is moving towards the adoption of service-driven architectures where intra- and inter-organisational business activities are carried out with the help of information services.

The CAiSE Forum 2012 is a place within the CAiSE conference for presenting and discussing new ideas and tools related to information systems engineering. Intended to serve as an interactive platform, the Forum aims at the presentation of fresh ideas, emerging new topics, controversial positions, as well as demonstration of innovative systems, tools and applications. Two types of submissions have been invited to the Forum:

(1) Visionary short papers that present innovative research projects, which are still at a relatively early stage and do not necessarily include a full-scale validation.

(2) Demo papers describing innovative tools and prototypes that implement the results of research efforts.

The CAiSE Forum 2012 proceedings represent a collection of 22 excellent short research papers and demos. The selection of Forum papers was very competitive, due to the high standard of the submitted papers. Several innovative papers initially submitted to the CAiSE conference were selected for the CAiSE Forum to stimulate open discussions of high-quality on-going research.

We devote a special thanks to the members of the international programme committee for promoting the Forum and for providing excellent reviews of the submitted papers. Their dedicated work was vital for putting together a high-quality working conference. We also thank the external reviewers. Special thanks go to the University of Gdańsk, Riga Technical University, and Stockholm University for supporting the organization of the CAiSE Forum.

June 2012

Marite Kirikova
Janis Stirna

CAiSE Forum 2012 Organization

General Chair

Sjaak Brinkkemper
Utrecht University, The Netherlands

CAiSE Programme Committee Chairs

Jolita Ralyté
University of Geneva, Switzerland

Xavier Franch
Universitat Politècnica de Catalunya, Spain

Forum Chairs

Marite Kirikova
Riga Technical University, Latvia

Janis Stirna
Stockholm University, Sweden

Organizing Chair

Stanisław Wrycza
University of Gdańsk, Poland

Publication Chair

Lota Endzele
Riga Technical University, Latvia

Forum Programme Committee

Joao Paulo Almeida, Federal University of Espirito Santo, Brazil
Per Backlund, University of Skövde, Sweden
Judith Barrios Albornoz, University of Los Andes, Venezuela
Peter Bellström, Karlstad University, Sweden
Wojciech Cellary, Poznan University of Economics, Poland
Remi Coletta, LIRMM, University Montpellier 2, France
Maya Daneva, University of Twente, The Netherlands

External Reviewers

Table of Contents

A Model Transformation from Misuse Cases to Secure Tropos

Naved Ahmed[1], Raimundas Matulevičius[1], and Haralambos Mouratidis[2]

[1] Institute of Computer Science, University of Tartu, Estonia
{naved,rma}@ut.ee
[2] School of Computing and Technology, University of East London, UK
h.mouratidis@uel.ac.uk

Abstract. In current practices security concerns are typically addressed at the design or implementation stages, leaving aside the rationale for security analysis. The reason is that a systematic approach to address security from late development stages to early analysis stages does not exist. This paper presents transformation rules to perform model translation from misuse case diagram to Secure Tropos model. The translation justifies the system security concerns, and keep the traceability of the security decisions. Our proposal is based on the systematic domain model for information systems security risk management (ISSRM); thus, it preserves the semantics of both security languages' constructs and synchronise the mechanisms across language boundaries to elicit, correct and complete security requirements. An example from banking sector demonstrates the applicability of our proposal.

Keywords: Information System (IS), Requirements engineering, Secure Tropos, Misuse cases, Model transformation.

1 Introduction

It is recognised that blemishes in requirements, on one hand, cost 10 to 200 times more once handled [3], and glitches in early requirements analysis stages outcomes a high percentage of system failures [12]. On another hand current practice starts develop security only after the system design or implementation is done [6]. However, this might lead to a gap between requirement analysis and the actual implementation. Although security modelling languages are used at different stages of the system development, they still lack dedicated constructs to identify the security concerns [7, 8], such as vulnerabilities, risks and their countermeasures. There exists little effort to integrate different security modelling languages into the coherent modelling approach so that developers could benefit from various modelling viewpoints along different system development stages. Such integration could also contribute to security traceability across the development cycle, thus, also keeping the rationale for the security decisions.

In this paper we introduces a set of transformation rules to translate misuse case diagrams [11] to Secure Tropos models [10]. This is a continuation of our previous effort [1], where we reported on the opposite transformation from Secure Tropos to misuse case diagrams. Both these model translations are based on the language semantic alignment [7, 8] to the domain model [9] of the Information Systems Security Risk Management (ISSRM). Since the major question of the goal modelling languages, like Secure Tropos, are to understand why certain system is build, in this paper we focus on capturing the security decision rationale from the misuse case models and representing it using Secure Tropos.

The structure of the paper is organised as follows: in Section 2 we give the background knowledge of security languages and introduce their alignment to the ISSRM domain model. In Section 3 we introduce the transformation rules to translate misuse cases to Secure Tropos. We illustrate our proposal through an online banking example [6]. In Section 4, we discuss benefits, completeness and limitations. Finally, we conclude our study in Section 5.

2 Background

2.1 ISSRM Domain Model

The ISSRM domain model [9] is used to align the security languages. It provides a systematic guidance for security risk analysis and supports modelling, assessing and treating risks on the basis of the likelihood and severity of failures as Tropos Goal-Risk framework [2]. The ISSRM domain model [9] (see Fig. 1) is inspired by, and compliant with the existing security standards (see details in [9]). Additionally as compared to Tropos Goal-Risk framework, ISSRM supports the definition of security for the key IS constituents and addresses the IS security risk management process at three different conceptual levels, i.e., asset-related, risk-related, and risk treatment-related concepts (described later). This gives details about the IS which is abstractly defined in a 3-layer architecture of Tropos Goal-Risk framework and helps to quantitatively measure the risk its likelihood, impact and cost of implementing security controls with respect to asset's value.

***i*) Assets-related concepts** describe the organisation's assets classified as *business* and *IS assets* along with the *security criteria* for business assets expressed in terms of confidentiality, integrity and availability.

***ii*) Risk-related concepts** define *risk*, composed of a threat with one or more vulnerabilities. An *impact* is the consequences of an event that negates the security criterion. An *event* is an aggregation of threat and one or more vulnerabilities. A *vulnerability* is the characteristics of IS assets that expose weakness or flaw. A *threat* is an incident initiated by a threat agent to target one or more IS assets. A *threat agent* is an agent who has means to harm IS assets intentionally. An *attack method* is a standard means to execute threat.

***iii*) Risk-treatment related concepts** describe a *decision* (e.g., avoidance, reduction, retention, or transfer) to treat the risk and *security requirement* is its refinement. A *control* is the implementation of requirements.

The ISSRM application follows the general risk management process based on the security standards (see details in [9]). Firstly, define *organisational context and identify assets.* Then, determine *security objectives for assets.*

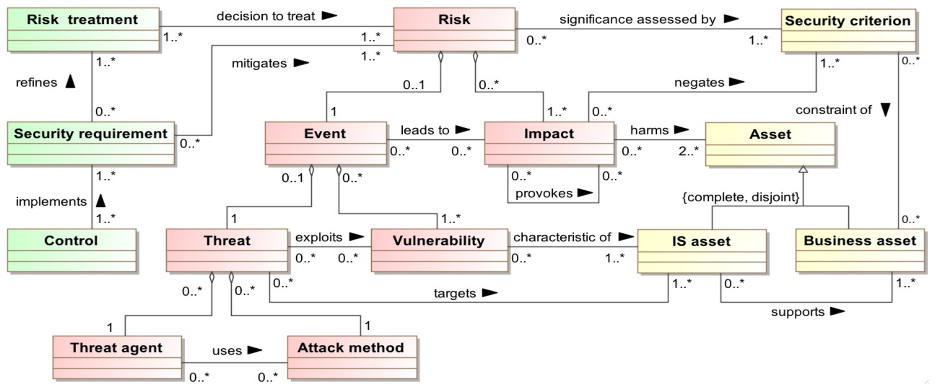


Fig. 1. ISSRM Domain Model, adapted from [9]

Next, *risk analysis and assessment* to identify potential risks and their impacts. Then, *risk treatment decisions* are taken resulting in *security requirements*. Finally, security requirements are implemented into *security controls*. This process is iterative, because new security controls might originate new security risks.

2.2 Misuse Cases

Misuse cases [11] are a security-oriented extension of the Use cases. Misuse case diagrams are extended with misuser, misuse case, and security use cases constructs including threatens and mitigates relationships (see Fig. 2). A *misuser* intends to harm the software system. A *misuse case* is a goal of misuser, the association is represented by a communication association. Misuser executes misuse case either by combine efforts of several misuse cases, or independently. *Threatens* relationship means a misuse case is potentially a threat to the use case. *Mitigates* relationship indicates that a use case is countermeasure against misuse case. *Security use case* performs countermeasure against the identified threat.

As illustrated in Fig. 2 misuse cases are integrated in use case diagrams to express the system unwanted behaviour (e.g., misuse cases `Money stolen`, `Enter pin code result repeatedly`, and `Transfer money to own account`) initiated by a misuser (e.g., `Attacker`). This depiction results in security use cases e.g., `Perform cryptographic procedures`.

2.3 Secure Tropos

Secure Tropos [10] is an extension of Tropos [4]. It enriches Tropos by introducing security related constructs (see Fig. 3). In Tropos, an *actor* (e.g., `Customer`, `Bank officer` and `Banking IS`) is an entity that has strategic goals and interests within the system. A *goal* (e.g., `Transaction be performed`, `Account privacy guaranteed`) is an actor's strategic interest. A *plan* (e.g., `Perform transaction`, `Keep data up to date`) represents means to satisfy actors' goals.

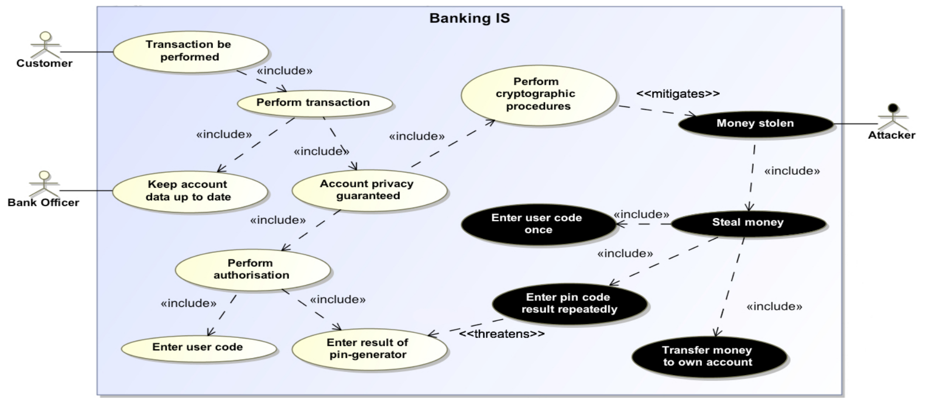


Fig. 2. Misuse Case Diagram

A *resource* (e.g., `Account`) is an entity required by actors. In Secure Tropos, *security constraint* (e.g., `Only by bank customer` and `Only by bank officer`) is a constraint that the system must possess. A *threat* (e.g., `Money stolen`) represents an event that endangers the security features of system. Additionally, vulnerability point is represented by a black circle in Fig.3 (adapted from [5]).

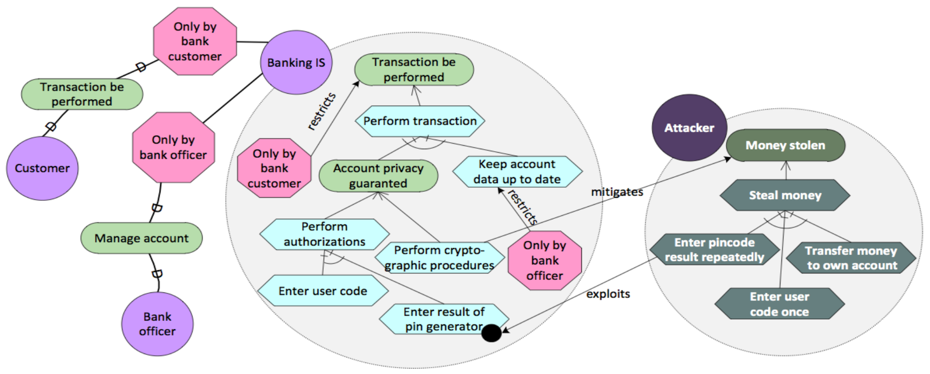


Fig. 3. Secure Tropos Diagram

Secure Tropos uses relationships to connect constructs. *Dependency link* shows that one actor (depender) depends on another actor (dependee) to attain some dependum (e.g., goal, plan or resource). A *secure dependency* is restricted by the security constraint that must be respected by both actors (e.g., relationship between `Customer` and `Banking IS`). A *means-end link* indicates how the goal (end) is satisfied. A *decomposition relationship* represents a breakdown of plan into several plans or goals. *Restricts* and *attacks relationships* are introduced in Secure Tropos where former shows a security constraint restriction on a goal achievement and prior indicates the target of attacker's plan.

Tropos methodology covers the overall IS development, however we limit our scope to the goal and security attack scenario modelling (which correspond to the Tropos late requirements stage [4]).

2.4 Alignment of ISSRM and Security Modelling Languages

As discussed in [9] the ISSRM domain model guides the application of the security modelling languages with respect to the security risks analysis. The detailed alignment of ISSRM domain model with Secure Tropos and Misuse Cases is provided in [8] and [7], and summarised in Table 1 (column **1** & **2**).

Table 1. Alignment of ISSRM Concepts with Modelling Languages Constructs

ISSRM Model Concepts			**Secure Tropos Constructs**	**Misuse Case Constructs**
0			**1**	**2**
Asset related concepts	**a**	Asset	Actor, goal, plan, resource	Actor and use case
	b	Business asset		
	c	IS asset		System
	d	Security Criteria	Security constraint	–
Risk related concepts	**e**	Risk	–	–
	f	Impact	Contribution between threat and other construct	–
	g	Event	Threat	–
	h	Threat	Goal, plan	Misuser & Misuse case
	i	Vulnerability	Vulnerability point	–
	j	Threat agent	Actor	Misuser
	k	Attack method	Plan, attacks relationship	Misuse case
Risk-treatment related concepts	**l**	Risk treatment	–	–
	m	Security requirement	Actor, goal, plan, resource, security constraint	Security use case
	n	Control	–	–

3 Transformation Rules

3.1 Transformation from Misuse Cases to Secure Tropos

This section introduces a set of rules for translating Misuse cases to Secure Tropos model. They are based on ISSRM model and its application process.

Asset-related concepts are translated using following transformation rules:

Tms1. *A system boundary that presents software system in the misuse case diagram is translated to the Secure Tropos actor.*

This rule is based on alignment between the Secure Tropos *actor* and misuse case *system boundary* to the ISSRM *IS asset* as introduced in Table 1 (line ***c***). In Fig.3 we present a Secure Tropos actor `Banking IS` with its boundary.

Tms2. *A use case is translated either to Secure Tropos goal or plan belonging to the boundary of the system actor. Correspondingly, an includes link is translated either to means-ends relationship (where ends is the goal and means is the*

plan) or to decomposition relationship (where some plan is decomposed).
Note: we assume *OR⇒means-ends*, and *AND⇒decomposition* in Secure Tropos model.

It is defined according to the lines ***a*** and ***b***. Here the developer decides whether a use case is translated to Secure Tropos *goal* or *plan*. In Fig. 3, we translate the use case `Transaction be performed` to goal meaning that the use case `Perform transaction` should be *plan*, because only a plan could be means to achieve the goal (ends) in Secure Tropos. On the other hand, the use case `Account privacy guaranteed` is translated to a goal. Here we also define two plans `Perform authorisation` and `Perform cryptographic procedures` that are the means to achieve this goal. We illustrate the *OR relationship* to specify two alternates to achieve the goal `Account privacy guaranteed`.

In Fig. 2, two actors (e.g., `Customer` and `Bank officer`) communicate to the `Banking IS`. Based on the Table 1 lines ***a*** and ***b*** we translate these actors to the Secure Tropos actors in Fig. 3 by introducing the following rule:

TMS3. *An actor from the misuse case is translated to a Secure Tropos actor.*

An interaction of actor with system presents how actors collaborate to achieve their goals. In misuse cases it is defined by *communication links* while Secure Tropos uses *dependency links*. A communication link would be translated using either of the three following cases:

TMS4. *(i) If the system is dependee, then the communication link is translated as depender and the use case to which the misuse case actor communicates is defined as dependum (according to TMS3) in the Secure Tropos dependency;*
(ii) If the system is depender, then the communication link is translated as dependee and the developer specify the dependum manually, since it is not possible to capture it from the misuse case diagram;
(iii) A security constraint could be defined to restrict the goal/plan (as well as the dependum). The restricted goal/plan is translated from the use case, to which the actor communicates in the misuse case diagram.

Following TMS4, the *communication links* (see Fig. 2) between actors `Bank officer` and `Customer` with `Banking IS` are translated to the dependency links (see Fig. 3). However, it is not possible to capture security constraints (Table 1, col ***2***, line ***d***). Although, we defined them manually (e.g., `Only by bank customer` and `Only by authorised bank officer`) by identifying the elements that needs to be restricted e.g. dependum goal `Manage account` in Fig. 3.

Translating ***risk-related concepts***, generate the Secure Tropos attack scenario (see Fig. 3) using the following transformation rules:

TMS5. *A misuser is translated to Secure Tropos actor. In the discussion below we recall this actor as a threat agent.*

It is based on line ***j*** in Table 1, which identifies that the *misuser* and the Secure Tropos *actor* are aligned to the ISSRM *threat agent*. Thus in Fig. 3 we identify a threat agent as *Attacker*.

TMS6. *A misuse case is translated to the plan of threat agent. Using TMS2, an includes link is translated to the Secure Tropos decomposition relationship.*

In Table 1, this rule refers to lines ***h*** and ***k***, according to which ISSRM *threat* and *attack method* are presented as *misuse case* and *plan* (and *goal*). Therefore, `Money stolen`, `Steal money`, `Enter pin code repeatedly`, `Enter user code once` and `Transfer money to own account` are translated to plan constructs in Secure Tropos model (see Fig. 3). To simplify the translation misuse cases are transformed to only Secure Tropos goals.

TMS7. *A threatens relationship is translated to the Secure Tropos exploits link. The exploits link is pointed to the vulnerability point, which needs to be added to the appropriate Secure Tropos construct.*

In the example, *threatens relationship* is translated to Secure Tropos *exploits link* from the threat agent's plan `Enter pin code repeatedly` to the vulnerability point identified in the `Enter result of pin generator` (see Fig. 3). Secure Tropos *threat agent* and his plans; correspond to the combination of the ISSRM *threat agent*, *attack method* and *threat*. Following Table 1 define a Secure Tropos *threat* (aligned to the ISSRM *event*) as a generalisation of the Secure Tropos threat agent and its boundary. For example, `Money stolen`.

Translating ***risk treatment-related concepts***, a security use case `Perform cryptographic procedures` is already translated to the Secure Tropos plan (see Fig. 3) as discussed in rule TMS2. Now we introduce that:

TMS8. *A mitigates relationship from the misuse case diagram is translated to the mitigates link in Secure Tropos.*

In the ISSRM domain model (Fig. 1) the *mitigates relationship* indicates the mitigation of potential risk event by introducing appropriate security requirements. The security use case `Perform cryptographic procedures` mitigates the threat `Money Stolen`, thus it is translated to the Secure Tropos mitigates to reduce the risk event `Money stolen`.

4 Discussion

Semi-automated Transformation: The transformation rules could support a semi-automatic model translation. When translating the models, the developer needs to indicate if the *(mis)use cases* need to be translated to the *goal* or *plan* (see rule TMS2). It influences the translation of *includes relationship* either to *means-ends* or *decomposition*. Also in TMS4, the developer indicates whether the Secure Tropos actor (translated from the misuse case *software boundary*) plays the role of *dependee* or *depender* in the translated *dependency link(s)*. Additionally, the developer defines the labels for *dependum* and *security constraint(s)* (as illustrated in Fig. 3). The remaining rules could be applied automatically.

Transformation Completeness: The transformation does not contribute with complete model in the target language but helps developers to concentrate on the details, which give the added value for the target model. The transformation highlights the major overlapping semantic areas of two security languages. The translated Secure Tropos model give reason for the system security.

Transformations and Misuse Case Textual Template: Matulevičius *et al.* [7] have aligned misuse case textual template and ISSRM domain model. Although we do not have enough space to discuss the template translation. We acknowledge that the template would complement and strengthen the transformation.

5 Conclusion

In this paper we tackled to eradicate the gap between the functional (software) system requirements and their relation to early security requirement analysis. We define a set of transformation rules from misuse case diagrams to Secure Tropos models. The transformation highlights and preserves the security-related semantics. The resulted model helps understanding the environment and gives reasoning on the benefits and trade-offs of the security decisions taken. Therefore, it benefits the overall model maintainability management between the two different presentations of security problem. In the example we have illustrated the applicability of our proposal, we acknowledge the importance of the industrial case study to validate the rules. The translation can be applied to existing or legacy systems to find the missing rationale for implemented security primitives and can provide alternate security solutions to solve the problem.

We agree to the importance of validating the current work and as a future work we encourage to empirically validate the translation through perception, performance and correctness tests. Furthermore, we plan to expand the scope by introducing a semi-automated transformation rules for other security languages. Such approach would result in a systematic model-driven security engineering, which would facilitate systematic security definition from the early requirements to system design and implementation.

References

1. Ahmed, N., Matulevičius, R.: Towards Transformation Guidelines from Secure Tropos to Misuse Cases. pp. 36–42. SESS'11, ACM (2011)
2. Asnar, Y., Giorgini, P., Massacci, F., Zannone, N.: From Trust to Dependability through Risk Analysis. In: Proceedings of ARES. pp. 19–26. IEEE (2007)
3. Boehm, B.W., Papaccio, P.N.: Understanding and Controlling Software Costs. IEEE Trans. Software Eng. 14(10), 1462–1477 (1988)
4. Castro, J., Kolp, M., Mylopoulos, J.: Towards Requirements-driven Information Systems Engineering: The Tropos Project. Inf. Syst. 27(6), 365–389 (2002)
5. Elahi, G., Yu, E.S.K.: A Goal Oriented Approach for Modeling and Analyzing Security Trade-Offs. In: Conceptual Modeling - ER. pp. 375–390. Springer (2007)
6. van Lamsweerde, A.: Elaborating Security Requirements by Construction of Intentional Anti-Models. In: ICSE 2004, UK. pp. 148–157. IEEE (2004)
7. Matulevičius, R., Mayer, N., Heymans, P.: Alignment of Misuse Cases with Security Risk Management. In: Proceedings of ARES. pp. 1397–1404. IEEE (2008)
8. Matulevičius, R., Mayer, N., Mouratidis, H., Dubois, E., Heymans, P., Genon, N.: Adapting Secure Tropos for Security Risk Management in the Early Phases of Information Systems Development. In: CAiSE, Proc. pp. 541–555. Springer (2008)
9. Mayer, N.: Model-based Management of Information System Security Risk. Ph.D. thesis (2009)
10. Mouratidis, H., Giorgini, P.: Secure Tropos: A Security-oriented Extension of the Tropos Methodology. International Journal of SEKE 17(2), 285–309 (2007)
11. Sindre, G., Opdahl, A.L.: Eliciting Security Requirements with Misuse Cases. Requir. Eng. 10(1), 34–44 (2005)
12. Sommerville, I.: Software Engineering. Addison Wesley, 6th edn. (August 2000)

An Aspect-Oriented Approach to Relating Security Requirements and Access Control[⋆]

Azadeh Alebrahim[1], Thein Than Tun[2], Yijun Yu[2],
Maritta Heisel[1], and Bashar Nuseibeh[2,3]

[1] paluno – The Ruhr Institute for Software Technology,
University of Duisburg-Essen, Germany
{azadeh.alebrahim,maritta.heisel}@paluno.uni-due.de
[2] Department of Computing, The Open University, Milton Keynes, UK
{t.t.tun,y.yu,b.nuseibeh}@open.ac.uk
[3] Lero – The Irish Software Engineering Research Centre,
University of Limerick, Ireland

Abstract. Affecting multiple parts in software systems, security requirements often tangle with functional requirements. In order to separate crosscutting concerns and increase modularity, we propose to represent security requirements as aspects that can be woven into functional requirements. Using problem frames to model the functional requirements, weaving is achieved by composing the modules representing security aspects with the requirement models. Moreover, we provide guidance on how such security aspects are structured to implement a particular access control solution. As a result, such security aspects become reusable solution patterns to refine the structure of security-related problem.

Key words: security requirement, aspect-oriented requirements engineering, security pattern, access control, problem frames

1 Introduction

Aspect-Oriented Programming (AOP) [4] is a programming paradigm that deals with crosscutting concerns at the implementation level in order to achieve a *separation of crosscutting concerns*. The crosscutting concerns are encapsulated into separate modules, known as *aspects*, that can be woven into a base system without altering its structure. This provides support for modularity and maintainability. The aspect-oriented concept has been adapted for earlier stages of software development, known as Aspect-Oriented Requirements Engineering (AORE) [11] to deal with crosscutting concerns at the requirements level. Quality concerns [1] such as security affect several parts of software systems, and are considered as crosscutting concerns. The first focus of this paper is on providing support for the separation of security requirements from functional requirements by modularising them into aspects.
The second focus of this paper is on providing guidance for refining the security aspects at the requirements level by reusing knowledge located in the solution space to bridge the gap between security problems and their solutions. The

[⋆] Part of this work is supported by the German Research Foundation (DFG) grant HE3322/4-1 and the Science Foundation Ireland grant 10/CE/I1855.

elaborated security aspects can be transformed into a particular solution at the design level. We believe that requirement descriptions cannot be considered in isolation and should be developed with architectural descriptions concurrently, as described by the Twin Peaks model [9]. Patterns describe solutions for recurring problems in software development, thus providing a means to reuse knowledge. Security patterns [12] provide solutions for software problems in the context of security. We aim to leverage security patterns as solution artefacts to refine the security aspects.
The remainder of this paper is structured as follows. Section 2 describes our approach by taking into account access control as a crosscutting security concern and problem frames as a requirements engineering method. Section 3 discusses related work. Conclusions and discussions of future work are given in Section 4.

2 Our Approach using Problem Frames and Access Control

The proposed approach has four steps. As suggested earlier, requirements and architectural descriptions should be considered as intertwining artefacts influencing each other. Therefore our method is illustrated as an instantiation of the Twin Peaks model [9] (see Figure 1). Considering security requirements as crosscutting concerns that can be modularised into separate aspects, we refine them by using security patterns as solution artefacts. The weaving of aspects into the functionality of the software system is achieved by composing the refined aspects with the functional artefacts. To illustrate these concepts, we use access control as the example and problem frames [3] as the requirements engineering method. We use the problem frames approach, because it allows us to navigate between the problem space and the solution space, while exploring problem structures using problem diagrams and solutions structures using patterns. This ability to express and relate the structures of problems and solutions is crucial for the proposed approach.
Problem frames are means to describe and classify software development problems. A problem frame represents a class of software problems. It is described by a frame diagram, which consists of domains, interfaces between them, and a requirement (see Figure 3). The objective of problem solving is to construct a machine (i.e., software) that controls the behaviour of the environment (in which it is integrated) in accordance with the requirements. Requirements analysis with problem frames decomposes the overall problem into subproblems, which can also be represented by problem diagrams. A problem diagram is an instance of a problem frame. Machines in problem diagrams represent solutions for functional requirements. We call them functional machines to distinguish them from machines representing solutions for security requirements that we introduce later in this section.

2.1 Identifying Access Control (AC) as an Aspect

Different modules representing different functional requirements in a system usually share common security concerns. Treating security requirements in isolation

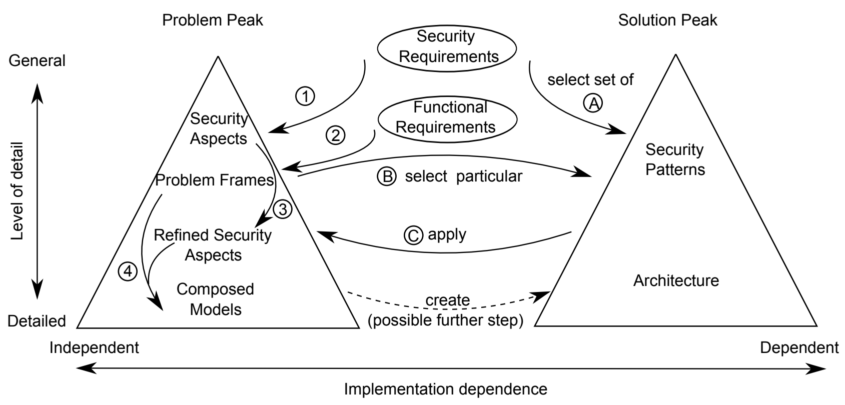


Fig. 1. Overview of our method embedded in the Twin Peaks Model [9]

from the functional requirements enables increased modularity and maintainability of the software. This idea is supported by AORE, which seems to be a promising approach to deal with quality requirements as crosscutting concerns to be separated from the functional requirements [11]. The first step in our method is therefore to identify the crosscutting security concerns that can be captured as a single security requirement, represented as an aspect (step 1 in Figure 1). Access control verifies whether a subject has the permission to access an object within the system. Therefore each user (*subject*) requesting access to the sensitive parts of a system (*object*) should be checked for a permission. Thus, we could express the security requirement addressing access control over a functional requirement as follows:

– SR: Only *subject* with permission to access the *object* before carrying out a function.

We introduce the *advice frame* to express such an access control requirement. The advice frame is illustrated in Figure 2. There is a *Subject* assumed to be a *biddable* domain, as shown by B in the lower right of the rectangle. The subject issues *commands* requesting access to an *Object*, which can be modelled as either a causal domain or a lexical domain. The *Controller* machine shall authorize the subject, validate the command and change the state of the object according to the command. If a user is not authorized or a command is not valid, the Controller machine shall do nothing. We make the behaviour of the identified security machine more concrete by describing its specification as follows:

```
SUBJECT INPUT: userId,command,object
IF SUBJECT INPUT is valid
THEN (Controller (C) does changeState;
Controller (C) performs do(command,object);)
ENDIF
```

Note that the identified security concern is considered in isolation in this step. Therefore it cannot be fully specified. We refine the specification as we proceed.

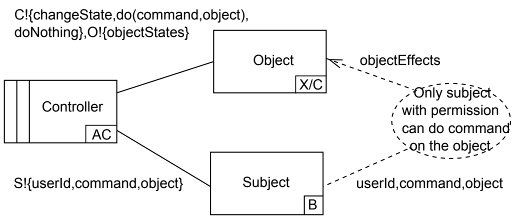

C!{changeState,do(command,object),
doNothing},O!{objectStates}
Object
X/C
objectEffects
Controller
AC
Only subject with permission can do command on the object
S!{userId,command,object}
Subject
B
userId,command,object

Fig. 2. *Advice frame* representing access control

2.2 Capturing Functional Requirements into Problem Diagrams

At the step two, we model functional requirements by using problem frames (see step 2 in Figure 1). Most security relevant systems contain sensitive information that should not be accessible to all users of the system. Sensitive information to be protected in a system is represented as either a causal or a lexical domain. Users are represented as biddable domains. Therefore such problems are generally modeled in problem frames either by the *commanded behaviour frame* containing a causal domain as sensitive information or by the *simple workpieces frame* containing a lexical domain as sensitive information, illustrated in Figure 3.
The commanded behaviour problem frame represents the problem of controlling some parts of the (*Controlled domain*) in the physical world by the machine (*Control machine*) according to commands issued by the *Operator*. The simple workpieces problem frame describes the problem of creating or editing a text or graphic (*Workpieces*) by the machine (*Editing tool*) according to the *User* commands (for details, see [3]).

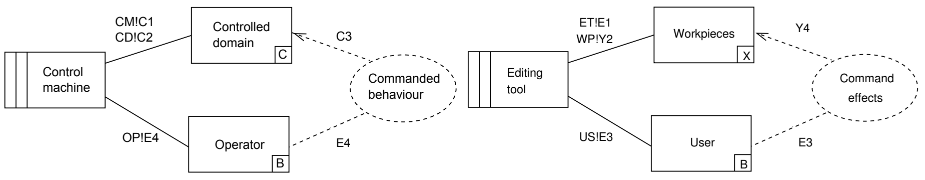


Fig. 3. Commanded Behaviour and Simple Workpieces Frames

2.3 Refining the Access Control Aspect using the RBAC Pattern

Security patterns [12], located in the solution space, provide a widely accepted means to build secure software. Usage of security patterns as solution artefacts aids to address security aspects in the problem space, which is the aim of step 3. Substeps A–C illustrated in Figure 1 deal with selecting and applying the most appropriate security pattern in order to refine the identified security aspect. Exactly how a pattern is selected in this approach is a topic for further research. In this work, we describe a way to structure the security machines, which are considered as black boxes with an unknown structure so far, using using security patterns.

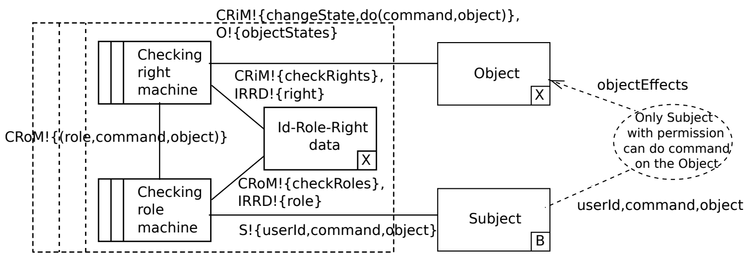


Fig. 4. Refined advice frame by using RBAC pattern

Since verifying permission is a frequently recurring problem in security relevant systems, it has been treated by several access control patterns [12]. Access control patterns define security constraints regarding access to resources. *Role-Based Access Control (RBAC)* provides access to resources based on functions of people in an environment (*roles*) and the kind of permission they have (*rights*). The *User* represents a registered user with certain *id* assigned to a predefined *Role*. Roles are assigned *Rights* in accordance with their functions. *Rights* define and check what resource the user is authorized to access.

Looking at the RBAC pattern in the solution space gives aid to decompose the advice machine. We identify two subproblems, *Checking role* and *Checking right* (see Figure 4). The *Checking role* subproblem represents the problem of checking the role assigned to the *User*, who is represented by the biddable domain *Subject* in Figure 4. The *Checking role machine* verifies whether the subject id is contained in the *Id-Role-Right data*. If there is no id-role relation the machine does nothing. If such a relation exists the machine passes a pair of role and command on to the *Checking right machine*. We specify the *Checking role machine* as stated below:

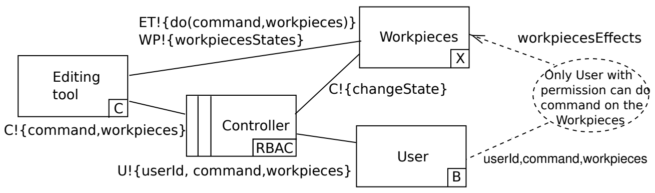
```
SUBJECT INPUT: userId,command,object
Checking role machine (CRoM) identifies the role of userId
IF there is a role for userId THEN Checking role machine (CRoM) passes
(role,command,object) to Checking right machine (CRiM);
ENDIF
```

The *Checking right machine* checks if a particular role is authorized to perform an operation on the object. If the subject with the particular role holds the right to perform the command, the machine changes the state of the object and performs the operation. Otherwise the machine does nothing:

```
INPUT: role,command,object
Checking right machine (CRiM) checks whether the role is allowed to perform command on the object
IF the role is allowed THEN (Checking right machine (CRiM) does changeState;
Checking right machine (CRiM) performs do(command,object);)
ENDIF
```

2.4 Weaving Aspects into Problem Diagrams

We now introduce a *weaving frame* to compose the refined security aspect with the problem diagrams (step 4 in Figure 1). The weaving frame includes all the



ET!{do(command,workpieces)}
WP!{workpiecesStates}
Workpieces
X
workpiecesEffects
Editing tool
C
Only User with permission can do command on the Workpieces
C!{changeState}
Controller
RBAC
C!{command,workpieces}
User
B
userId,command,workpieces
U!{userId, command,workpieces}

Fig. 5. Abstract *weaving frame*

domains from the basic problem frame (commanded behaviour or simple workpieces), including both the functional machine and the advice machine. We complete the specification of the external behaviour of the security aspect outlined in step 1. The internal behaviour remains unchanged as specified in step 3.
The weaving is achieved by mapping domains in the basic problem frame to domains in the advice frame. The domains *Controlled domain/Workpieces* in Figure 3 are mapped to the *Object* domain in the advice frame, and the domain *User* to the domain *Subject*. We consider the case for the simple workpieces frame as functional frame in the following. The weaving of the functional frame commanded behaviour is carried out analogously. We define *join points*, which represent transformation rules to transform the functional frame into the weaving frame by means of weaving of the advice frame. Changes in addition to mappings are italicised. In order to affect the behaviour of the functional machine by verifying user inputs, we place the advice machine on the interface between the *User* domain and the *Editing tool* (see Figure 5).

Join points	User = Subject, Workpieces = Object, Editing tool = Editing tool, E3 = {userId,command,object}, Y4 = objectEffects, Y2 = {objectState}, E1 = {do(command,workpieces)}, *ADD domain Controller, ADD interface with phenomena C!{command,workpieces}, ADD interface with phenomena C!{changeState}*

The advice machine passes the valid command to the functional machine, which performs directly the operation on the Workpieces domain according to the User command. We specify the advice machine as follows:

```
USER INPUT: userId,command,workpieces
IF USER INPUT is valid
THEN (Controller (C) passes (command,workpieces) on Editing tool;
Controller (C) does changeState;)
ENDIF
```

Figure 6 illustrates how the refined RBAC aspect woven into the simple workpieces frame diagram.
Applying our method to a software development problem, we achieve modularity as the access control requirement is now captured as an RBAC aspect. As a result potential changes to this module do not affect the functional models, increasing the maintainability of the system. We made the internal structure and behaviour of the access control machine more concrete by applying the RBAC pattern as solution and describing its specification in detail. Here the access control machine

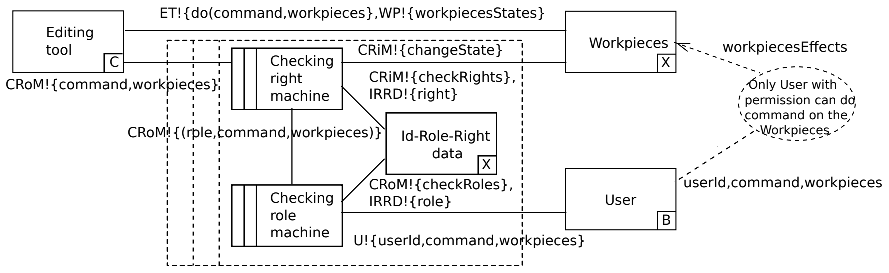

ET!{do(command,workpieces},WP!{workpiecesStates}
Editing tool
C
CRoM!{command,workpieces}
Checking right machine
CRiM!{changeState}
CRiM!{checkRights}, IRRD!{right}
Workpieces
X
workpiecesEffects
Only User with permission can do command on the Workpieces
CRoM!{(role,command,workpieces)}
Id-Role-Right data
X
Checking role machine
CRoM!{checkRoles}, IRRD!{role}
User
B
userId,command,workpieces
U!{userId,command,workpieces}

Fig. 6. Refined weaving frame

is not considered as a black box anymore: we have taken one step further towards implementing RBAC as a particular solution for the access control requirement, thus bridging the gap between the problem and the solution space.

3 Related Work

An AORE model to support the separation of functional and quality concerns is proposed by Rashid et al. [11]. Quality concerns are identified and refined as aspects, which are prioritised in order to resolve conflicts among them. In contrast to our work, Moreira et al. [7] augment the AORE model by a uniform treatment of functional and quality concerns. The method proposed in [8] integrates crosscutting quality attributes into the functional description after identifying and specifying them. However it does not consider solution approaches to refine the crosscutting quality attributes as our approach does. Unlike the goal aspect approach [13], where quality softgoals are refined as aspectual tasks, problem frames-based approach allows navigating between them through physically connected interfaces.

There exists some work that relates aspect concepts to problem frames. Laney et al. [5] propose resolving inconsistencies when composing multiple problem frames. Here we specialise the composition frames to weave security aspects into functional structures. The approach proposed by Lencastre et al. [6] incorporates aspect concepts into problem frames by extending an existing meta-model to express crosscutting relationships between different element types of problem frames. Their work does not focus on treating quality requirements as aspects.

The aforementioned approaches in contrast to our work only discuss methods to incorporate crosscutting concerns into the requirement models. We take one step further towards bridging the gap between the problem and the solution space.

The security Twin Peaks model [2] (an elaboration of the original Twin Peaks model [9]) is a framework for developing security in the requirements and the architectural artefacts in parallel. Taking architectural security patterns into account, the model supports the elaboration of the problem and the solution artefacts. Similar to our work, a method to bridge the gap between security requirements and the design is proposed by Okubo et al. [10]. This method introduces new security patterns at the requirements and the design level, in contrast to our approach that reuses the existing security design patterns at the requirements level.

4 Conclusions and Future Work

We have proposed a method using problem frames to refine the security aspects in the problem space by using the artefacts located in the solution space. We have selected access control as one important security concern to illustrate the refining process. The benefits of our approach are twofold. The first is that we separate security requirements from functional requirements and encapsulate them into separate modules as aspects. Thus we achieve a separation of concerns that increases the modularity of the software. The second benefit is that we give guidance how the security aspects need to be structured to fit a particular solution. To this end we have used security patterns as solutions artefacts to refine the problem structure.

In future work, we will investigate how to find the most suitable security pattern in the set of available security patterns. Finding the most suitable security pattern depends on the context and also on the functional requirements. We will extend the scope of this work by considering different security requirement aspects that need different security patterns to be satisfied.

Acknowledgments. We would like to thank Takao Okubo, Nobukazu Yoshioka, and Haruhiko Kaiya for useful discussions.

References

1. L. Chung, B. Nixon, E. Yu, and J. Mylopoulos. *Non-functional Requirements in Software Engineering.* Kluwer Academic Publishers, 2000.
2. T. Heyman, K. Yskout, R. Scandariato, H. Schmidt, and Y. Yu. The security twin peaks. In *ESSoS'11*, LNCS 6542, pages 167–180. Springer, 2011.
3. M. Jackson. *Problem Frames. Analyzing and structuring software development problems.* Addison-Wesley, 2001.
4. G. Kiczales and E. Hilsdale. Aspect-oriented programming. In *ESEC'01/FSE-9*, pages 313–, USA, 2001. ACM.
5. R. Laney, L. Barroca, M. Jackson, and B. Nuseibeh. Composing requirements using problem frames. In *RE'04*, pages 122–131. IEEE Computer Society, 2004.
6. M. Lencastre, J. Araujo, A. Moreira, and J. Castro. Analyzing crosscutting in the problem frames approach. In *IWAAPF'06*, pages 59–64, USA, 2006. ACM.
7. A. Moreira, J. ao Araújo, and A. Rashid. A concern-oriented requirements engineering model. In *CAiSE'05*, pages 293–308. Springer, 2005.
8. A. Moreira, J. a. Araújo, and I. Brito. Crosscutting quality attributes for requirements engineering. In *SEKE'02*, pages 167–174, USA, 2002. ACM.
9. B. Nuseibeh. Weaving together requirements and architectures. *IEEE Computer*, 34(3):115–117, 2001.
10. T. Okubo, H. Kaiya, and N. Yoshioka. Effective Security Impact Analysis with Patterns for Software Enhancement. In *ARES'11*, pages 527–534, 2011.
11. A. Rashid, A. Moreira, and J. Araújo. Modularisation and composition of aspectual requirements. In *AOSD'03*, pages 11–20, USA, 2003. ACM.
12. M. Schumacher, E. Fernandez-Buglioni, D. Hybertson, F. Buschmann, and P. Sommerlad. *Security patterns: integrating security and systems engineering.* John Wiley & Sons, 2005.
13. Y. Yu, J. C. S. do Prado Leite, and J. Mylopoulos. From goals to aspects: Discovering aspects from requirements goal models. In *RE*, pages 38–47. IEEE Computer Society, 2004.

Support for Eviation Detections in the Context of Multi-Viewpoint-Based Development Processes

Reda Bendraou[1], Marcos Aurélio Almeida da Silva[1,2], Marie-Pierre Gervais[1,2] and Xavier Blanc[3]

[1] LIP6, UPMC Paris Universitas, France
[2] UPO, Université Paris Ouest, France
{reda.bendraou, marcos.almeida, marie-pierre.gervais}@lip6.fr
[3] Labri, Université de Bordeaux 1{xavier.blanc@labri.fr}

Abstract. One recurrent issue in software development processes are developer's deviations from the process model. This problem is amplified in the context of multi-viewpoint-based development of complex systems where the system's specification comes in form of different and intertwined viewpoints. Without a methodological support, these deviations become inevitable. They can be of different kinds: 1) behavioral deviations related to inappropriate actions performed by the developer when realizing process's activities or 2) structural deviations due to inconsistencies in deliverables, which can be in conflict with other viewpoint's outcomes. This paper proposes an approach to overcome these issues. To demonstrate the approach, a prototype was developed and the RM-ODP standard and a viewpoint-based development process were used.

1 Introduction

Recently, multi-viewpoint modeling appeared to be a promising approach for dealing with system's complexity. The system is described through the composition of different viewpoints, each one focusing on a precise concern such as security, persistency, GUI, and so on. The main idea is to focus the developer's attention on a specific aspect of the system thus, abstracting away all the irrelevant details. Different modeling languages can be used for the specification of the system's viewpoints and they can be at different levels of abstraction. This inevitably raises problems related to the heterogeneity of the viewpoints, the overlapping of the concepts between them, and the fact that consistency should be constantly maintained between them. Large-scale systems span multiple and intertwined viewpoints; involve long and multidisciplinary design activities which are not supposed to be known by all project's developers.

In such context, it becomes essential to provide a methodological support during the development process. Indeed, when a developer is performing modeling actions inside a process's activity on a given viewpoint, he usually does not have a global view of the whole system design and is far away from assessing instantly the effects of his actions on the other system's viewpoints

Many approaches provide methodological support with the help of process execution and monitoring engines, also called PSEE (Process-centered Software Engineering Environment) [2][3]. They mainly ensure that the process is correctly applied by the developers by verifying that the activities are executed in the appropriate order and timing, and that the roles are correctly assigned to the right developers. However, if the developer is performing inappropriate actions inside a given activity, this will not be reported by the PSEE. Moreover, if these actions are in contradiction with the process guidelines, they will straightforwardly impact the other viewpoints. As a consequence, the effects are discovered very late, and hence will require costly maintenance actions. Detecting such deviations as soon as they occur can improve the process organization and prevent from risks of project failure and unnecessary delays.

In this work we propose an approach for providing a process support in the context of multi-viewpoint development processes. This support is manifold. First it ensures that developers perform the process in the specified order. Second, it makes sure that in each viewpoint, the developers are following the process guidelines and that they are not violating the project's methodological constraints. It also enforces that developer's actions in a viewpoint do not impact negatively the other system's viewpoints. Finally, it is able to detect developer's deviations from the process model as soon as they occur and to warn the developer of the deviation's cause.

To illustrate our approach, we use RM-ODP (Reference Model of Open Distributed Processing) [5,7], a multi-view-based standard for the specification of distributed systems. The next section categorizes the kinds of process deviations that may occur in such a context and their effects on the process execution. Section 3 presents in details our approach. It is based on Praxis Rules, our language for expressing process and methodological constraints in the context of multi-viewpoint design. The approach is then evaluated in Section 4 through a case study using RM-ODP. Section 5 concludes this paper and sketches some perspectives of this contribution.

2 Categorization of process deviations

Developer's deviations that may occur during a development process can be categorized into four kinds. The first one is what we call **organizational deviations**. They occur when an activity's deadline is not respected, when a role is not fulfilled or assigned to inappropriate developer. The second kind is called **micro behavioral deviations.** These deviations occur when a developer is performing inappropriate actions inside a modeling activity and thus, violating methodological guidelines or business constraints (e.g. a developer is applying a design pattern in a wrong way). This kind of deviation may be the consequences of developer's misunderstanding of the work to accomplish or his willingness to perform the activity by following his intuitions and experience. In the context of multi-viewpoint modeling, if developers are performing modeling actions that are in complete contradiction with what was specified in other related viewpoints, they won't be notified with the eventual conflicts until the end of the activity i.e., until they submit their deliverables to the PSEE for a structural check. We believe that the early detection of micro behavioral deviations can avoid rework actions which represents a considerable gain in terms of time and efforts. Our proposition aims at detecting them as soon as they occur in order to prevent project managers from process failures.

Structural deviations are triggered when a model delivered by an activity has some inconsistencies. In the context of multi-viewpoint modeling the fact that different modeling languages can be used in each viewpoint adds more complexity in maintaining the structural consistency between the different deliverables. The challenge is then to provide an independent-modeling language approach to ensure such consistency. In section 3 we present our proposition to this problem.

Finally, **macro behavioral deviations** occur when a developer decides to execute process's activities in a different order than the one prescribed by the process model. This can be due to an expected project's constraints. In all cases, it is primordial that deviations, whatever their kind, have to be captured by the PSEE and reported instantly to the project manager in order to assist him in taking the appropriate decisions.

3 Praxis and Praxis Rules

Praxis and Praxis Rules are the building blocks of our approach. Due to the lack of space, in the following we focus on demonstrating their use for detecting only structural and micro behavioral deviations. The same principle applies for the other kinds of deviations.

3.1 Praxis

Praxis is used to represent each elementary modeling action performed by a developer within a modeling tool [1]. This representation has already been used in the context of artifacts described in different languages like EMF, UML, XML and Java[1]. It consists of six classes of atomic actions inspired from the MOF reflexive API [4]. The *create(me, mc, t)* and *delete(me; t)* actions respectively create and delete a model element *me*, that is an instance of the meta-class *mc* at the timestamp *t*. The *addProperty(me, p, value, t)* and *remProperty(me, p, value, t)* add or remove the value *value* to or from the property *p* of the model element *me* at timestamp *t*. Similarly, the actions *addReference(me, r, target, t)* and *remReference(me, r, target, t)* add or remove the model element *target* to or from the reference *r* of model element *me* at timestamp *t*.

In the present work, we extend every Praxis action with an extra parameter *v* to represent the *viewpoint* in which it has been performed. For example, the action *create(me, mc, t, v)* represents the creation of the model element *me*, an instance of the metaclass *mc*, in the timestamp *t* in the viewpoint *v*. The same applies for the other kinds of actions. With this extension, we can then represent the different actions performed by developers over different kinds of models. Figures 2 and 3 exemplify two different viewpoints in a given system. The former represents its structural viewpoint with the package *Azureus* and the *Client* and *Server* classes. The second figure represents the behavioral viewpoint by means of a sequence diagram.

Now suppose that a modeler renames server role in the behavioral viewpoint from *Role2* to *MainServer* and that, later, another modeler deletes an operation from the structural viewpoint. These low level actions can be represented by the sequence of Praxis actions below. Notice that Praxis is able to represent changes in different viewpoints in a way that is independent of the meta-models used to represent them.

[1] http://code.google.com/p/harmony.

```
remProperty(role2, name, 'Role2', 12, 'behavioral').
addProperty(role2, name, 'MainServer', 13, 'behavioral').
delete(op1, operation, 14, 'structural').
```

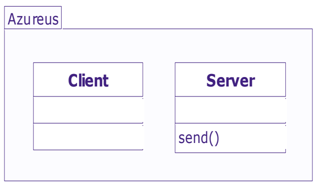


Fig. 2. Structural viewpoint: a UML package with its content

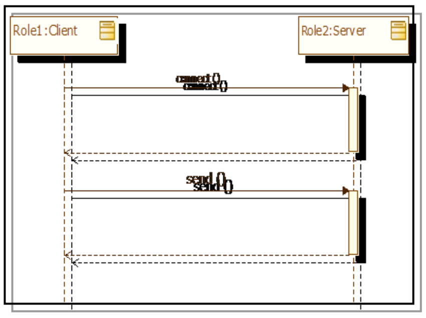


Fig. 3. Behavioral viewpoint : Sequence Diagram

3.2 Praxis Rules

PraxisRules is the rule based language that is used by the PSEE to detect both structural and behavioral deviations during process execution. This language has already been used to detect structural deviations in industrial multi-view models [6] and in structural and behavioral deviations in single viewpoint PSEEs [9]. The PSEE detects deviations by comparing each rule with a Praxis trace captured from the process execution. There are two kinds of rule in PraxisRules, 1) activity post-condition rules that define structural constraints over a sequence of praxis actions and, 2) activity invariant rules that define behavioral constraints over a sequence of praxis actions.

Activity post-condition rules have the form "*ruleName(Variables) <=>expression.*" where *ruleName* is the name of the rule, *Variables* is a list of variables in the rule and *expression* is a logical expression. The meaning of such rule is that *ruleName(Variables)* is *true* in the sequence if and only if *expression* holds. In the expression, a combination of logic predicates such as *and{…}* for conjunctions, *or{…}* for disjunctions and *not{…}* for negations and references to Praxis actions (such as *create(ME,MC,T,VP))* is allowed.

Activity invariant rules have the form "*ruleName(Variables) @ action <=> expression.*", where *ruleName* represents the name of the rule and *Variables* list of variables in the rule. *@action* is an action variable that refers to a particular action in the sequence. The expression *expression* is then used to validate the presence of *@action* in the sequence or to define the allowed order of other actions taking it as reference. The order of actions is defined by temporal operators like *@before{…}* and *@after{…}*. For example, the expression *@before{ action1 @ addReference(A, B, C), action2 @ remReference(A, B, C) }* means that the action matching *addReference(A, B, C)*, represented by the action variable *@action1*, should appear before the action matching *remReference(A, B, C)*,represented by the action variable *@action2*, in the Praxis sequence. Notice that variables are represented by words starting in uppercase letters and in lower case letters for action variables; that timestamps may be omitted from Praxis actions; and that the syntax *[@action] call ruleName(Parameters)* is used to call the rule *ruleName* with the parameters *Parameters* and with the action *@action* for activity invariant rules.

In order to check a constraint that may span multiple models i.e. inter-model constraints, in this paper we allow the viewpoint information in the Praxis representation of actions to be used in PraxisRules. Indeed, contrarily to languages like OCL [8], PraxisRules does **not** impose unique context constraints and can be used to express constraints over a sequence of editing actions among a set of viewpoints. Thus, every time a developer performs a modeling action in a specific viewpoint, the latter is captured by Praxis and annotated with the timestamp and the viewpoint information. Inter-model constraints represented in the form of a PraxisRule are then checked over the combination of the entire viewpoints' sequences of actions. If the rule does not hold, a deviation is triggered.

For instance, let us consider the structural viewpoint given in Figure 2 and the behavioral viewpoint given in Figure 3. In this example, one inter-model behavioral constraint could be that during an activity called *renameMessages*, the developer would be asked to rename the messages in the behavioral viewpoint, but that the names he provides need to correspond to names of operations in the structural viewpoint. Using Praxis Rule this is how such a constraint is expressed:

```
renameMessagesInv(M) @ action <=> or {
   action @  remProperty(E, name, Name, 'behavioral'),
   and {
      action @  addProperty(E, name, Name, 'behavioral'),
       call existsElementInViewpointByName(Name, operation, 'structural')
   }}
existsElementInViewpointByName(Name, MC, V) {
    create(E, MC, V),
    addProperty(E,name, Name)
}
```

This rule called *renameMessagesInv* states that an action *@action* should be either a *remProperty(E, name, Name, 'behavioral')*, meaning a removal of a name of some element in the behavioral viewpoint, or an *addProperty(E, name, Name, 'behavioral')*, meaning the addition of a name for some element in the behavioral viewpoint. The *addProperty* action is further constrained by the existence of an operation *OP* in the structural viewpoint having the same *Name* as the element *E* renamed by the developer. This extra constraint is enforced by the rule called *existsElementInViewpointByName*. This rule is verified by the PSEE by replacing the action variable *@action* with every action executed during the activity. A micro behavioral deviation is then raised if the developer executes any action that does not conform to the constraint. That is the case when he renames a message with the name of inexistent operation or when other actions like creating new elements are performed.

4 Evaluation

In order to evaluate the feasibility of our approach we developed a prototype and tested it in the context of a development process. We used RM-ODP, a multi-viewpoint-based standard for the specification of distributed systems [7]. As a process example, we borrowed the one presented in the UML4ODP profile specification [5]. It describes the design process of the "Templeman Library system" using RM-ODP.

In the case study presented in Section 4.2, Praxis Rules are used 1) to specify RM-ODP consistency rules and to illustrate the occurrence of a structural deviation (inter-viewpoints) and its detection in case of one of these rules is violated; 2) to define the set of allowed actions during the modeling of a given viewpoint of the Library system. The process part consisting in defining the Enterprise viewpoint was taken as an example to illustrate the occurrence of micro behavioral deviations and how our prototype detects them. The RM-ODP defines 5 viewpoints, namely the Enterprise, the Information, the Computational, the Engineering and the Technology viewpoints. For the interested reader, more details can be found on RM-ODP in [7]. In the following, we present our prototype.

4.1 Prototype

In our prototype, we adopted MagicDraw[2] as a modeling tool. Our choice was influenced by the fact that MagicDraw provides a UML Profile for RM-ODP called UML4ODP. Each viewpoint is specified as a stereotyped package (e.g., <<Enterprise_spec>>). Of course, any modeling tool can be used in place of MagicDraw.

The following picture displays a screenshot of our prototype (c.f., Figure 4). It shows a scenario of a process enactment. The parts (1), (2) and (3) represent an extension to MagicDraw in order to display the RM-ODP viewpoint that the developer is working on. Part (4) is also an extension that shows the activity being enacted by the developer. In this sample scenario, the developer executed an action that was not allowed by the process model. That is why a dialog box (5) is prompted to indicate that a deviation occurred due to the execution of an action that violates the process modeling rules. The same kind of dialog box is used to display guidelines during process enactment.

4.2 Case Study

The case study consisted in running a multi-viewpoint-based process on top of our prototype. During the process enactment, we deliberately caused different kinds of deviations i.e. structural and micro behavioral and we controlled if the tool succeeded in detecting all of them instantly. In the following, we present the process modeling rules represented using PraxisRules.

The process model and modeling rules.

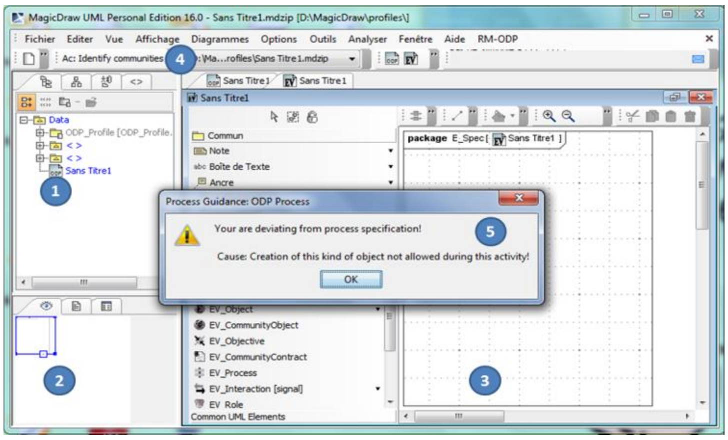
As an example of a development process using RM-ODP, we used the one initially described in natural language in the UML4ODP specification, page 68 of [5]. For the sake of clarity, we focus on the part of the process activities required for the specification of the Enterprise viewpoint of the library application. A first step was to map each activity of the process to Eclipse cheat sheet steps. The second step consisted in representing the consistency rules between the different viewpoints in form of Praxis Rules (i.e., Activity post-condition or invariant rules).

Let us take the first activity of the process as an example i.e., “Identify the communities, with which the system is involved, and their objectives”. For this activity, a process modeling rule was defined using Praxis Rule (see Figure 5). This rule comes

[2] Site, http://www.magic-draw.com

in the form of an activity invariant rule that states that in the Enterprise viewpoint, one can only create elements that relate to that viewpoint. Additionally, it states that only three kinds of elements can be created: *communities*, *objectives* and *"objective of" associations*, which link communities to objectives. These elements are represented in UML respectively by components tagged with the *EV_Community* stereotype; classes tagged with the *EV_Objective* stereotype and associations tagged with the *EV_ObjectiveOf* stereotype. During the process execution, if the developer performs an action not allowed by this rule, a micro behavioral deviation is triggered instantly and its cause is displayed to the developer.

For brevity reasons, since we need one activity post-condition and an activity invariant rule per activity, which would account for 16 rules for the selected process, we are not able to present all the praxis rules in this paper. However, the complete source code of our prototype, along with the process model that was used to detect structural and micro/macro behavioral deviations is available at our WebSite[3]



Process Guidance: ODP Process

Your are deviating from process specification!

Cause: Creation of this kind of object not allowed during this activity!

OK

Fig. 4. Screenshot of a process sample executed in our prototype

```
public IdentifyCommunities() @ a <=> and {
   a @ call inViewpoint("enterprise"),
   not {  t @ create(C,MC),
   not {  or {   and { addProperty(C, stereotype, S),
                           call goodCombination(MC,S)   },
                MC = property  }}
  }
}.
goodCombination(MC, S) <=> or { and { MC = "component", S = "ev_community" }, … }
inViewpoint(V) @ action <=> or { action @ create(E,MC,V), … }
```

Fig. 5. Sample of process modeling rule

[3] (*http://lip6.fr/Marcos.Almeida/publications.html*).

4.3 Discussion

The realization of the prototype and the case study was an important step for us and revealed the feasibility of the approach. Most of all, we were able to ensure the detection of both structural and behavioral deviations. These deviations are detected instantly and the developer is informed with the cause of its deviations. Regarding structural deviations, we were able to detect both intra- and inter-viewpoint inconsistencies, which is of prime importance in the context of multi-viewpoint modeling. In our case study, the language used for defining the different viewpoint was the same but thanks to Praxis Rule, having different modeling languages would not change anything to our solution

A step further in the validation process would be to conduct an empirical study to assess the benefits, in terms of time and quality, of offering such support to the developers. In a previous work, we realized such a study but this was done with a process example which did not include the modeling of a system with several viewpoints [10].

We assessed the "effort of adoption" by considering the coding efforts required for extending the MagicDraw case tool to implement our approach. Thanks to MagicDraw Open API, implementing the MAL component has been implemented as a 360 lines of Java (PropertyChangeListener). This component took one day of work for a Java experimented developer. When it comes to the PEE, our approach is mostly independent from it. During this experiment it consisted in a simplistic extension of MagicDraw interface which amounted to less than 200 lines of Java code. The only dependency of the other components to this one is that the DDE needs to know which is the current activity being executed by the developer so that it can verify the chosen behavioral and structural rules. Our approach would then be able to be reused by any existing PEE that is able to provide these pieces of information to the other components of our approach.

5 Conclusion

In the context of multi-viewpoint-based projects, the risks that developers deviate during the development process are amplified. The heterogeneity of the viewpoints, there overlapping and the need to ensure consistency between them inevitably introduce many chances for developer deviations. If not handled on time, these deviations may cause the failure of the project in terms of reliability of the project's outcome, delays and costs. In this paper we proposed an approach that allows capturing developer's deviations during process realization. Whatever their kind i.e., behavioral or structural, these deviations are detected instantly as they occur and their causes are reported to the developer or project manager. They can then take the appropriate decisions and anticipate the risks that may penalize the course of the project.

As a perspective of this work we are currently studying the resolution of the optimal path to reconcile the developer with the process description in case of late deviation detections i.e. the early deviation detection is turned off by the developer. We also plan to put in place a more important empirical study for validating our approach.

References

1. X. Blanc, et al. Detecting model inconsistency through operation-based model construction. In Robby, editor, Proc. Int. Conf. Software engineering (ICSE'08), volume 1, pages 511–520. ACM, 2008.
2. G. Cugola. Tolerating deviations in process support systems via flexible enactment of process models. IEEE Trans. Software Eng., 24(11):982–1001, 1998.
3. M. Kabbaj, R. Lbath, and B. Coulette. A deviation management system for handling software process enactment evolution. In Q. Wang, D. Pfahl, and D. M. Raffo, editors, ICSP, volume 5007 of Lecture Notes in Computer Science, pages 186–197. Springer, 2008.
4. OMG: Meta Object Facility (MOF) 2.0 Core Specification (January 2006)
5. Information technology — Open distributed processing — Use of UML for ODP system specifications ITU-T Recommendation X.906, ISO/IEC 19793. At: http://www.lcc.uma.es/~av/download/UML4ODP_IS_V2.pdf
6. J. Le Noir, O. Delande, D. Exertier, M. A. A. da Silva, and X. Blanc. 2011. Operation based model representation: experiences on inconsistency detection. In Proceedings of ECMFA'11, Springer-Verlag, Berlin, Heidelberg, 85-96.
7. ITU-T Recommendation X.902 (1995) | ISO/IEC 10746-2:1996, Information technology – Open Distributed Processing – Reference Model: Foundations.
8. OMG: UML 2.0 OCL Specification, November 2003
9. M. A. A. da Silva, R. Bendraou, X. Blanc, and M.-P. Gervais. Early deviation detection in modeling activities of mde processes. In Petriu et al. [11], pages 303–317.
10. M. A. A. da Silva, A. Mougenot, R. Bendraou, J. Robin, and X. Blanc. Artifact or process guidance, an empirical study. In Petriu et al. [11], pages 318–330.
11. D. C. Petriu, N. Rouquette, and Ø. Haugen, editors. Model Driven Engineering Languages and Systems - 13th International Conference, MODELS 2010, Oslo, Norway, October 3-8, 2010, Proceedings, Part II, volume 6395 of LNCS. Springer, 2010.

SDWM: An Enhanced Spatial Data Warehouse Metamodel

Alfredo Cuzzocrea[1], Robson do Nascimento Fidalgo[2]

[1] ICAR-CNR & University of Calabria, 87036 Rende (CS), ITALY
cuzzocrea@si.deis.unical.it.
[2.] CIN, Federal University of Pernambuco, 50.732970 Recife (PE), BRAZIL
rdnf@cin.ufpe.br.

Abstract. Some research has been done in order to define metamodels for Spatial Data Warehouses (SDW) modeling. However, we observe that most of these works propose metamodels that mix concepts of DW modeling (i.e. dimensions and their descriptive attributes) with concepts of OLAP modeling (i.e. hierarchies and their levels). We understand that this mix is a possible limitation, because a DW (conventional or spatial) is essentially a large data repository, which can be analyzed/queried by any data analysis technology (e.g. GIS, Data Mining and OLAP). With aim of overcoming such limitation, in this paper we propose a SDW metamodel, named *Spatial Data Warehouse Metamodel* (SDWM), which describes constructors and restrictions needed to model SDW schemas. We have implemented a CASE tool according to our metamodel and, by exploiting this tool, we have designed two demonstrative SDW schemas.

Keywords: Data Models, Spatial Databases, Data Warehouse.

1 Introduction

The process of decision-making may involve the use of tools, such as Data Warehouse (DW) [6], On-Line Analytical Processing (OLAP) [11], Geographical Information Systems (GIS) [13] and Data Mining [12]. In this context, there are two different types of technologies: one for storing data (DW) and other for querying data (OLAP, GIS and Data Mining). That is, from a database, any query tool may be used to analyze its data. Hence the DW concepts must be independent of those associated to query tools, in particular, with regard to OLAP tools, which has concepts (i.e. hierarchy and its level) that are frequently mixed with the concepts of DW modeling (i.e. dimensions and its descriptive attributes). That is, in DW there are neither hierarchies nor levels, because if these concepts were intrinsic to a DW, any query tool for DW would be able to process multilevel queries, but only OLAP query tools can do it. In the next paragraphs of this Section we present the main concepts and techniques that may be used for DW/SDW modeling, our research problem and how our paper is organized.

DW is a typically-large data repository that is usually designed using the star model [6], which has two types of tables: fact and dimension. A fact table stores some metrics of a business, while a dimension table to hold its descriptive information. There are many techniques/concepts for modeling dimensions and fact tables. However, only some techniques/concepts bring additional information, required to generate correct code, namely: degenerate dimensions, role-playing dimensions and bridge tables (or many-to-many relationship). That is, a degenerate dimension ensures that it can be used only in a fact table and that it will be part of the identifier of the fact table, but it cannot be a reference/link for a dimension. In turn, a role-playing dimension allows the creation of different views of the same dimension. Finally, a bridge table (or many-to-many relationship) allows to create a third table with two one-to-many relationships, where we can specify the name of this table and some additional attributes.

A lot of data stored in a DW has some spatial context (e.g., city, state and country). This means that if one intends to properly use this data in decision support systems, it is necessary to consider the use of a Spatial DW (SDW). A SDW is an extension of the traditional DW. It has an additional spatial component (a spatial feature type) that we define from a position (a geometric attribute) more a location (a descriptive attribute), where the location is optional. Basically, a SDW extends the star model through the inclusion of this spatial component in dimensions or in fact tables. Much research has focused in SDW modeling (see Section 4). However, we identified that most of these works defines metamodels that mix concepts for dimensional modeling with concepts for cube modeling, which we disagree, because, as already stated, a DW (conventional or spatial) can be analyzed/queried by any data analysis tool (not only OLAP). With aim of overcoming such limitation, we propose the Spatial Data Warehouse Metamodel (SDWM), which is presented using UML metaclasses (see Section 2).

The remaining of this paper is organized as follows. In Section 2 we propose the SDWM Metamodel and present its definitions. Next, in Section 3 we give an overview about our CASE tool for helping in the SDW modeling tasks and present a practical application of our metamodel and CASE tool. Then, in Section 4 we make a brief discussion about some existing works for SDW metamodel/CASE tool. Finally, in Section 5 we present some conclusions and indications for future work.

2 SDWM: A Spatial Data Warehouse Metamodel

SDWM is a metamodel that embeds the following significant features: (*i*) disassociating DW modeling from OLAP cube modeling; (*ii*) representing the spatiality in a SDW simply stereotyping attributes/measures as spatial, rather than stereotyping dimension/fact table as spatial or hybrid; (*iii*) capturing whether the geometry of a spatial attribute/measure can be normalized and/or shared; (*iv*) supporting the following DW modeling techniques: degenerated dimension, many-to-many relationship (bridge table) and role-playing dimensions; (*v*) providing a set of stereotypes with pictograms that aims to be concise and friendly; (*vi*) being used as a basic metamodel for a CASE tool that aims to model logical schemas of SDW, as well as to check whether these schemas are syntactically valid. In Figure 1 we introduce SDWM using the UML class diagram.

In Figure 1, we have three enumerations, which cover one of the possible values for an attribute. The Cardinality enumeration represents whether the relationship is many-to-one, one-to-many or many-to-many. This enumeration is important to define the primary/foreign key (like in R-DBMS) or OID/REF (like in OR-DBMS). That is, the table on the "many" side has a foreign key (or a REF) to the table of the "one" side. With respect to many-to-many cardinality, we apply the bridge table technique, which creates a third table with two one-to-many relationships. In turn, the *DataType* and *GeometricType* enumerations represent the primitive or spatial data types supported by SDWM, respectively. We highlight that these enumerations are just data type indications, which will be translated for specific data types of a DBMS. Furthermore, we also point out that the spatial data types are conform to the *Simple Feature Access* (SFA) specification of *Open Geospatial Consortium* (OGC).

Our metamodel has five main metaclasses, namely: *Schema*, *Relationship*, *Table*, *DimensionColumn* and *FactColumn*. *Schema* is the root metaclass that corresponds to the drawing area for a SDW schema. For this reason, *Schema* is a composition of zero or more *Table* and zero or more *Relationship*. At last, *FactColumn* and *DimensionColumn* are just a set of different types of columns. Besides the main metaclasses, our metamodel has eight specialized metaclasses, namely: *Fact*, *Dimension*, *Bridge*, *SpatialMeasure*, *DegenerateDimension*, *ConventionalMeasure*, *SpatialAttribute* and *ConventionalAttribute*. That is, a *Table* is specialized in *Fact*, *Dimension* or *Bridge*, which capture the concepts of fact table, dimension table and a bridge table, respectively. A *FactColumn* is specialized in *SpatialMeasure*, *DegenerateDimension* and *ConventionalMeasure*, which correspond to a spatial feature type, a descriptive attribute and a measurable attribute, respectively. Finally, a *DimensionColumn* is specialized in *SpatialAttribute* and *ConventionalAttribute*, which represent a spatial feature type and a descriptive attribute. Note that, *Fact* is a composition of zero or more *FactColumn* and zero or more *ConventionalAttribute* and each *Dimension* or each *Bridge* is a composition of zero or more *DimensionColumn*. We highlight that a *Dimension* table differs from a *Bridge* table because they have different stereotypes (i.e. they represent different concepts), a *SpatialMeasure* differs from a *SpatialAttribute* because they have different stereotypes and a *SpatialAttribute* is a feature type that always has its position (or geometric information) plus its location (or descriptive information) to represent the spatiality in a SDW, while a *SpatialMeasure* may have only its geometric information, since it can be stored without its descriptive information (*hasDescription* = false). That is, on the one hand, whether a feature type is defined as a *SpatialMeasure*, it can store only its position (e.g. geometries of farms); on the other hand, whether this feature type is defined as a *SpatialAttribute*, it has to store its position and location (e.g. geometries and descriptions of farms). In turn, a *DegenerateDimension* differs from a *ConventionalAttribute*, because they have different stereotypes and only the *DegenerateDimension* can be part of fact table identifier, as well as only the *DegenerateDimension* can be defined in a fact table.

In order to capture the tables that are source and target in a relationship, we have the associations named *Source* and *Target* between *Table* and *Relationship*. The metaclass *Relationship* has *cardinality* and can have a *role* for expressing, respectively, the maximum number of instances of the relationship and a particular view of a dimension associated with a fact table (i.e. a role-playing dimension). Another important attribute is *name*. This attribute stores a label that identifies a metaclass.

In order to define whether the position of spatial measure/attribute must be normalized in a different table from its location, the *isNormalized* attribute is defined as a *Boolean*. That is, whether this attribute is defined as true, the geometric information is normalized in a separate table from the table that stores the descriptive information of the spatial attribute/measure. Otherwise (*isNormalized* = false), the geometric information is defined in the same table that stores the descriptive information of spatial attribute/measure. SDWM also allows to define whether the spatial (or geometric) information can be shared among several spatial attributes/measures. To accomplish this, it is necessary to define the same name and the same geometric type. Furthermore, for each spatial attribute/measure that will share its geometry, the attributes *isNormalized* and *isShared* must be defined as true. The default value for *isNormalized* and *isShared* is false and, in this case, there is not a special notation (i.e. the attribute/measure is written in regular font). However, when *isNormalized* or *isShared* are defined as true the attribute/measure appears in bold and/or italic font, respectively.

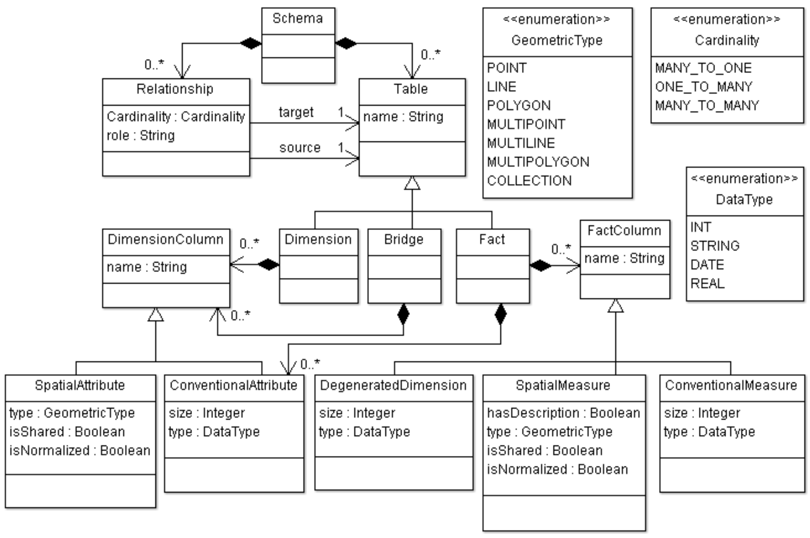


Fig. 1. SDWM Metamodel.

Spatial measures have the attribute *hasDescription*, which allows to define whether the spatial measure has a description (*hasDescription* = true) and a geometry or, otherwise (*hasDescription* = false), whether the spatial measure has only a geometry. *DegeneratedDimension, ConventionalMeasure* and *ConventionalAttribute* may have a size and each specialization of *DimensionColumn* and *FactColumn* has an associated type from our allowed data types (i.e. *DataType* and *GeometricType* enumerations). SDWM uses stereotypes and pictograms to increase its expressiveness (see Figure 2) and to represent primitive types and spatial types (see Figure 3).

3 A Real-Life SDW

In order to evaluate the correctness and usefulness of our metamodel, we developed a CASE tool, called SDWCASE, that was used to design a SDW with meteorological data from the *Laboratory of Meteorology of Pernambuco* (LAMEPE). This laboratory has a net of meteorological *Data Collection Platform* (DCP) for monitoring atmospheric conditions. SDWCASE is a CASE tool that offers a concise and friendly GUI that is based on the set of stereotypes with pictograms presented in Figures 2 and 3. With our CASE tool, the designer can interact with the SDW schema by inserting, excluding, editing, visualizing at different zoom levels, exporting a figure (e.g. JPG, GIF, PNG) or XMI (*XML Metadata Interchange*) file. Moreover, SDWCASE also allows the validation of the modeled schema. For example, (*i*) two tables (dimension or fact) or two attributes (in the same table) cannot have the same name; (*ii*) a table cannot be associated with itself; (*iii*) measures and degenerated dimensions can only exist in a fact table; (*iv*) dimension tables and bridged tables can only have attributes. The first and the second validations are ensured by programming, but the third and the fourth are intrinsically/automatically ensured by our metamodel (see Figure 1). SDWCASE is implemented in Java using the *Eclipse Graphical Modeling Framework* (GMF) plus the *Eclipse Modeling Framework* (EMF) and, in its current version, it generates code only for PostgreSQL with PostGIS. However, it can be done for any spatial DBMS.

Stereotype	Pictogram	Description
Fact	Ft	Fact Table
Dimension	Dt	Dimension Table
Bridge	Bt	Bridge Table
ConventionalAttribute	C	Conventional Attribute
ConventionalMeasure	C	Conventional Measure
DegeneratedDimension	d	Degenerated Dimension
SpatialAttribute	S	Spatial Attribute
SpatialMeasure	S	Spatial Measure
Relation	╱	Relation

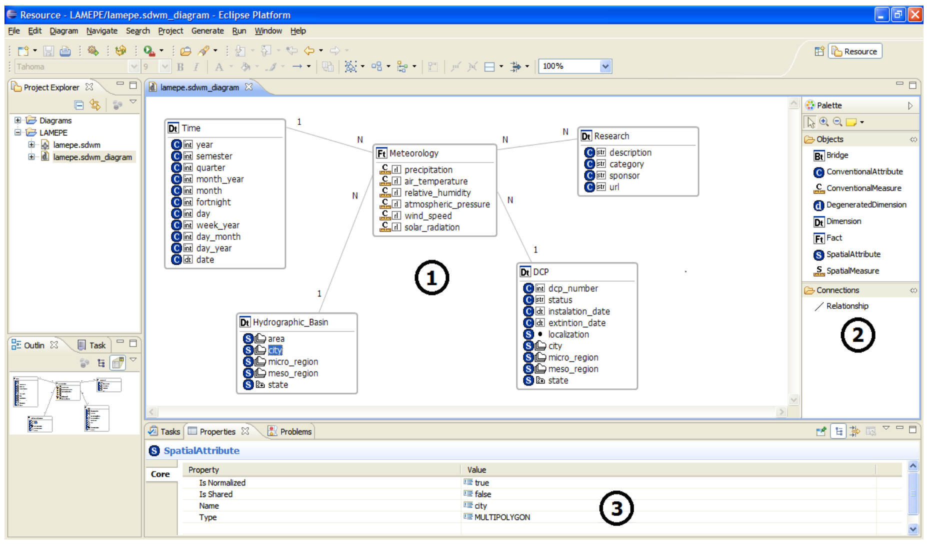
Fig. 2. SDWM Metamodel Stereotypes.

Stereotype	Pictogram	Description
INT	int	Integer type
STRING	str	String type
DATE	dt	Date type
REAL	rl	Real type

Stereotype	Pictogram	Description
POINT	•	Point geometry
LINE	⌇	Line geometry
POLYGON	⏢	Polygon geometry
MULTIPOINT	∴	Multiple Points geometry
MULTILINE	⌇⌇⌇	Multiple Lines geometry
MULTIPOLYGON	⏢	Multiple Polygons geometry
COLLECTION	⏢	Geometry Collection geometry

Fig. 3. SDWM Primitive Type Stereotypes and Geometry Type Stereotypes.

In Figure 4 we show the SDWCASE GUI with the LAMEPE SDW using many-to-many relationship. The SDWCASE GUI has a palette (area 2 in Figure 4) with all elements (defined in SDWM) that the designer needs to model a SDW. The modeling tasks starts with a click on the desired element in the palette and place it in the drawing area (area 1 in Figure 4). Next, the designer may edit the properties of the element (area 3 in Figure 4), and add new elements or relationships. Note that (*i*) each element is easily identified by its pictogram and (*ii*) the SDW schema is concise. That is, only using spatial attributes/measures, we can represent the spatiality in a SDW with a short notation. In Figure 4, we have one fact table, four dimension tables, and conventional and spatial attributes, which are stereotyped according to SDWM pictogram. In this figure you can note that there is a many-to-many relationship between the fact table *Meteorology* and the dimension *Research*. In this case, our CASE tool abstracts the creation of a third table to implement this relationship. However, an explicit bridge table also can be defined in SDWCASE. Another schema using bridge table, role-playing dimensions, spatial measure, degenerated dimension and conventional attribute can be seen in [2].



Property	Value
Is Normalized	true
Is Shared	false
Name	city
Type	MULTIPOLYGON

Fig. 4. SDWCASE GUI with LAMEPE SDW using many-to-many relationship.

4 Related Work

In order to do a systematic evaluation of these works, we are using the following features to compare, which we retain critical for modeling SDW:

1. disassociating DW modeling from OLAP Cube modeling;
2. making a CASE tool available to users;
3. supporting the following DW modeling techniques: degenerated dimensions, many-to-many relationships and role-playing dimensions;
4. supporting spatial attributes rather than spatial or hybrid dimension;
5. supporting spatial measures;
6. providing a set of stereotypes with pictograms that aim to be concise – i.e., it provides a short notation;
7. capturing whether the geometry of a spatial attribute/measure can be normalized and/or shared.

Bédard et al. [1, 9] define three types of spatial dimensions: the non-geometric spatial dimensions (all level are conventional data), the geometric spatial dimensions (all level are spatial data) and the mixed spatial dimensions (it has conventional and spatial data in the same dimension). The authors also differentiate numerical and spatial measures, where the spatial measures are considered as a collection of geometries.

Fidalgo et al. [3, 10] define a metamodel and a CASE tool for SDW modeling. Similarly the previous work, the authors specify concepts of measures (conventional or spatial) and dimensions (conventional, spatial or hybrid). Moreover, the metamodel and the CASE tool provide a set of stereotypes and pictograms, which are for SDW modeling.

However, both, metamodel and CASE tool, although support the technique degenerated dimension, they do not support many-to-many relationships neither role playing dimensions techniques nor spatial attributes.

Malinowski and Zimányi [7, 8] define an extension of ER model to represent dimensions, hierarchies and spatial measures/levels. The extension makes use of classes and relationships, both stereotyped with spatial pictograms, to model the geometry of spatial levels and the topological relationships between these levels.

Glorio and Trujillo [4, 5] define an UML profile and a CASE tool that use a set of stereotypes and pictograms for dimensions, hierarchies and spatial measures/levels. Although this work supports the technique degenerated dimension, it does not support many-to-many relationships neither role playing dimensions.

In short, all works support spatial measure, but no work supports spatial attributes. Consequently, no work captures whether the geometry of a spatial attribute must be normalized and/or shared. Moreover, only Fidalgo et al. [3, 10] do not mix DW modeling with OLAP modeling, as well as, only Fidalgo et al. [3, 10] and Glorio and Trujillo [4, 5] support degenerated dimension technique, but these works do not support many-to-many relationships neither role playing dimensions techniques. Finally, although most of these works provides a set of spatial stereotypes with pictograms, these works represent the spatial information as a stereotyped class, which does not provide a concise/short notation, because it pollutes the SDW schema whether it has much spatial information. In Table 1 we compare our work with the related works discussed here.

Table 1. Analysis of related works and our proposal.

	Bédard et al.	Fidalgo et al	Malinowski and Zimányi	Glorio and Trujillo	Our Proposal
DW vs. OLAP Modeling	NO	YES	NO	NO	YES
CASE Tool	NO	YES	NO	YES	YES
Degenerated Dimensions	NO	YES	NO	YES	YES
M-N Relationships	NO	NO	NO	NO	YES
Role-Playing Dimensions	NO	NO	NO	NO	YES
Spatial Attributes	NO	NO	NO	NO	YES
Spatial Measures	YES	YES	YES	YES	YES
Short notation	NO	NO	NO	NO	YES
Normalized/Shared Geo.	NO	NO	NO	NO	YES

5 Final Remarks

Many proposals have focused in metamodel and/or CASE tool for SDW. However, most of these works defines metamodels that (*i*) mix concepts of DW modeling with concepts of the OLAP modeling; (*ii*) does not support important techniques of DW modeling, (*iii*) represents the spatiality in a SDW stereotyping the dimensions and fact table as spatial or hybrid, rather than stereotyping the attributes/measures as spatial; (*iv*) defines a complex taxonomy of spatial dimensions and measures, (*v*) does not provide a concise and friendly set of stereotypes with pictograms; and/or (*vi*) is not used as a basic metamodel for a CASE tool. In order to give a contribution to solve the previous problems, we have proposed the *Spatial Data Warehouse Metamodel* (SDWM), which defines the constructors and the restrictions needed to design SDW schemas that are consistent and unambiguous. Our metamodel is more straightforward and more expressive than its related works, because it (*i*) represents the spatiality in a SDW assigning spatial stereotypes in attributes and measures, (*ii*) disassociates the DW modeling from the OLAP cube modeling, (*iii*) captures whether the geometry of a spatial attribute/measure can be normalized and/or shared, (*iv*) proposes a set of stereotypes with pictograms that aims to provide a short/concise notation, and (*iv*) supports the following DW modeling techniques: degenerated dimension, bridge table and role-playing dimensions. For this, SDWM can be used as a basic metamodel for a CASE tool that aims to make the design of invalid SDW schema much harder, as well as to make the automatic SQL/DDL code generation from these schemas.

To evaluate our proposal, SDWM has been implemented in a CASE tool and tested with a case study that illustrates a use of our metamodel/CASE tool, demonstrating that the semantic and syntax of our metamodel are modeled correctly, and its notation is unambiguous. The CASE tool is named SDWCASE. It is implemented in Java and in its current version, generates SQL/DDL code for PostgreSQL/PostGIS. In future work, other spatial DBMS will also be covered. Other direction for future work is the: definition of a metamodel and CASE tool to model and query a Spatial OLAP cube.

Acknowledgments This work was partially supported by the process BEX 5026/11 CAPES/DGU from the CAPES Brazilian foundation.

References

[1] Bédard, Y., Merrett, T. and Han, J.: Fundamentals of spatial data warehousing for geographic knowledge discovery. In: Geographic Data Mining and Knowledge Discovery 2001: 53-73.

[2] Del Aguila, P. S. R., Fidalgo, R. N., Mota, A.: Towards a more straightforward and more expressive metamodel for SDW modeling. DOLAP 2011: 31-36

[3] Fidalgo, R. N., Times, V. C., Silva, J. Souza, F. F.: GeoDWFrame: A Framework for Guiding the Design of Geographical Dimensional Schemas. In: DaWaK 2004: 26-37.

[4] Glorio, O., and Trujillo, J.: An MDA Approach for the Development of Spatial Data Warehouses. In: DaWaK 2008: 23-32.

[5] Glorio, O., and Trujillo, J.: Designing Data Warehouses for Geographic OLAP Querying by Using MDA. In: ICCSA (1) 2009: 505-519.

[6] Kimball, R. and Ross, M.: The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling, second ed., Wiley, NewYork, 2002.

[7] Malinowski, E. and Zimányi, E.: Representing spatiality in a conceptual multidimensional model. In: GIS 2004: 12-22.

[8] Malinowski, E., and Zimányi, E.: Advanced Data Warehouse Design From Conventional to Spatial and Temporal Applications, Springer, 1st ed. 2008.

[9] Pestana, G., Silva, M. M. da, & Bédard, Y.: Spatial OLAP modeling: an overview base on spatial objects changing over time. In ICCC 2005: 149-154.

[10] Silva, J., Oliveira, A. G., Fidalgo, R. N., Salgado, A. C. and Times, V. C.: Modelling and querying geographical data warehouses. In: Inf. Syst. 2010 35(5): 592-614

[11] Thomsen, E.: OLAP Solutions: Building Multidimensional Information Systems. John Wiley and Sons, 2nd ed. 2002

[12] Witten, I. H. and Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, Morgan Kaufmann , 2nd ed. 2005.

[13] Worboys, M., and Duckham, M. 2004. GIS: A Computing Perspective, Taylor & Francis. 2nd ed. 2004.

Risk Identification of Tailorable Context-aware Systems: a Case Study and Lessons Learned[⋆]

Mohammad Zarifi Eslami, Brahmananda Sapkota, Alireza Zarghami, Eelco Vriezekolk, Marten van Sinderen, Roel Wieringa

Department of Electrical Engineering, Mathematics and Computer Science
University of Twente - The Netherlands
{m.zarifi,b.sapkota,a.zarghami,e.vriezekolk,m.j.vansinderen,r.j.wieringa}@utwente.nl

Abstract. In this paper, we discuss possible risks posed by the application of tailorable context-aware systems in real-life practices. We use a tailorable context-aware system in the homecare domain as a case study to identify and analyse such risks. Next, we discuss which of these risks can be generalized to the use of tailorable context-aware system in other contexts than homecare. This would help the users of such systems to prevent the risks and guide the design and implementation of them.

Keywords: Risks, Tailorable context-aware systems, Homecare.

1 Introduction

A context-aware system adapts its behavior based on a model of the user's current context [2]. Such a context model is usuallyinferred from data of sensors in the environment of the user. If the adaptation is not only based on the context model, but also on user-defined preferences and requirements, we call this a Tailorable Context-aware (TC) System [19].

The use of TC systems can bring important benefits in many domains. Such benefits include easier to use, more useful and personalized services, as a consequence of proper consideration of the user's context and preferences. However, TC systems can also introduce new or increased risks for the person or organization using these systems. Such risks arise from assumptions that are made during the design, and which are typical for this type of systems, namely: the context model properly reflects reality, the tailoring is done correctly, and the provided service (e.g., in the form of advice or instructions) is used as intended.

It is therefore important to do a risk assessment of a TC system in relation to the environment in which it will be used. Such an assessment may deliver useful results for the design of the system and its introduction in the environment. So far, risk assessment of context-aware systems has mainly focused on privacy and security aspects [7], whereas other aspects such as availability and accountability have received much less attention [8, 4].

[⋆] This work is part of the IOP GenCom U-Care project(http://ucare.ewi.utwente.nl), sponsored by the Dutch Ministry of Economic Affairs under contract IGC0816.

The goal of this paper is to make a first step towards introducing risk assessment concerning the availability and accountability aspects, as part of a requirements engineering approach for TC systems. As a case study, we use the design of a TC system in the homecare domain, where the system aims at supporting independent living of elderly people in their private environment.

There is an emerging trend in industrialised countries for using IT-based care services such as health monitoring and coaching and medication reminder to support independent living of elderly [1, 3, 11, 16]. The use of these services can have several benefits such as improving the quality of care, quality of life of elderly, saving time of healthcare professionals and responding to the shortage of qualified staff. The European Council recognises improvement of patient safety as one of the benefits of using eHealth systems [14]. However, IT-based care services can also introduce new types of risks.

The concept of risk has been defined differently in different domains [15, 10, 5]. However, there is a common understanding that risk is a combination of the likelihood that an incident will occur and the impact of that incident. In our work, we are not concerned with the quantification of likehood nor of impact. Therefore, based on this common understanding, we define risk for TC systems as: the possibility of an undesirable outcome of an incident (related to the operation and/or use of the system). More specifically, we define availability risk as: the possibility of an undesirable out due to the unavailability of the system or its services; and accountability risk as: the possibility of an undesirable outcome due to the fact that no accountable actor can be found. Since risk is defined in terms of undesirable outcomes, we assume that there are stakeholders which suffer the undesirable outcomes. What constitutes a risk is therefore stakeholder-dependent. If stakeholder goals and requirements would change, we may have to repeat the risk assessment.

The rest of the paper is structured as follows. In Section 2, we describe our research methodology and define the scope of the paper. In Section 3, we briefly describe the case study with a TC system applied in homecare. In Section 4, we present the results regarding the identified risk. Finally, in Section 5, we discuss the lessons learned and conclude the paper.

2 Research Methodology

The purpose of risk assessment is to gather necessary information so that subsequent risk treatment decisions can be taken that are both effective and efficient. According to the ISO framework (ISO-31000), risk assessment consists of risk identification, analysis and evaluation [9].

An essential step in risk identification is the identification of all stakeholders, their goals and their interactions with the system. Any interaction (either tailoring or using the system) that may lead to undesirable consequences for that or another stakeholder will be listed as a risk. Thus, we execute the following to reach the goal of the paper:

– Identifying all stakeholders that use and interact with the system;

– Describing the goals of stakeholders and their interactions with the system;
– Identifying possible undesirable outcomes (risks) of these interactions;
– Draw lessons learned for the general case of TC systems.

One important assumption restricts the scope of the paper. We assume that the TC system technically works as designed, i.e., risks due to malfunctioning of internal components of the system (the risks caused by interactive complexity [12]) are out of the scope of this paper. More specifically, we are interested only in risks arising from the interaction of stakeholders with the system.

3 Description of the Case

We performed our case study in a care-institution in the Netherlands. This institution consists of residential blocks where elderly can live and receive care services. We developed a tailorable IT-based homecare service platform to be evaluated in this care institution. To analyse the existing situation, we interviewed professional nurses who provide care services in this institution. The purposes of the interviews was to gain insight in the commonly performed tasks and the possible risks associated with these tasks.

Providing Tailorable Context-Aware Homecare (TCH) services is one of the required features of successful introduction of IT-based care services [6, 20]. Fig. 1 depicts a simple version of a TCH system. The motivation behind such system is to support a care-giver to create a user-specific *service plan* by using a *tailoring platform*, which can be executed by a *provisioning platform* and satisfy the individual needs and preference of a care-receiver. The detailed information on creating the service plan using the tailoring platform and executing these service plans by the provisioning platform are reported in our earlier works [19, 18].

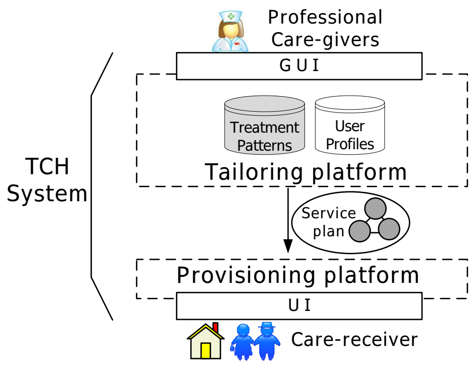


Fig. 1. A TCH system

By interviewing professional nurses we identified that high blood pressure is a common problem among elderly people. Thus we use the blood pressure example, to illustrate the functionality of TCH systems.

– Care-receivers should be reminded to attach the blood pressure measurement tool and measure the blood pressure themselves.

- If the care-receivers ignore the reminder for attaching the measurement tool, the second reminder should be sent after half an hour.
- The message, number of its repetition and the modality can be personalized based on individual requirements.
- If the measured value is high/low, the care-giver should be informed.

4 Risk Assessment

The type of the risks we are interested is caused by different stakeholders participating in the homecare domain. To analyse these risks, we begin with the identification of stakeholders in a homecare domain where care services are provided both with and without using the TCH system. Then we identify a list of possible risks in both situations, and their sources.

4.1 Identification of Stakeholders

The homecare domain is complex and involves various stakeholders with diverse interests (e.g., insurance companies, government, etc.). Excluding the stakeholders that fall outside the scope of the TCH system, we identified *care-givers*, *care-receivers* and *care centers* as three main types of stakeholders.

Anyone involved in providing care to the elderly (*care-receivers*), is considered a *care-giver*. Those who can provide care or interact with elderly include: professional nurses, informal care-givers and physicians. In this work, we consider only professional nurses as the care-givers, because: a) care-receivers spend most of their time with processional nurses while receiving care services in comparison to other care-givers, and b) Professional nurses are the main care-givers who interact with the TCH system to define service plans for care-receivers.

Care centers are institutions who pay for the care-givers and provide facilities to take care of the care-receivers. Care centers define medical protocols providing guidelines for taking care of care-receivers, which should comply with the national medical protocols defined by the government. When carrying out homecare tasks, care-givers must follow these medical protocols.

As shown in Fig. 2, after introduction of the TCH system, four new types of stakeholders appear in the homecare domain. These new types of stakeholders are *IT specialists*, *third-party service providers*, *infrastructure providers* and *hackers*.

An *IT specialist* is a person who can install, test, operate and maintain the TCH system. IT specialists are responsible for defining the treatment patterns based on existing medical protocols and care-givers recommendations and refining them based on operational experiences and test results. *Third-party service providers* own and manage services (such as blood pressure measurement, location determination and medication dispensing services) which can be used and composed by the TCH system to provide desired services to the care-receivers. These providers are located outside the care center and their services are accessible to other services through the Internet. *Infrastructure providers* are responsible for providing the necessary infra services to realise the TCH system such as the Internet and power supply. *Hackers* are individuals or organizations who break into the TCH system and its network and violate its function.

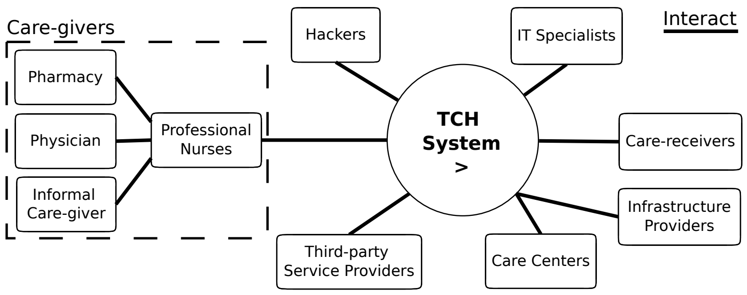


Fig. 2. A context diagram of a care center which is equipped with TCH system

4.2 List of Possible Risks

One of the benefits of a TCH system, compared to the current way of providing services, is the mitigation of existing risks that stakeholders, mainly care-receivers and care-givers, are already dealing with. We identify these existing risks and discuss whether the proposed TCH system can indeed mitigate them. We also identify new risks that might be introduced due to use of a TCH system.

Existing Risks We interviewed care-givers to identify the common tasks generally performed in the homecare domain. Based on the result of these interviews, we identified the risks in the current situation and determined whether these can possibly be decreased using a TCH system.

– **Forgetting to Treat Patients:** The care-givers usually follow a routine schedule in providing care services based on the medical protocols and doctors advice, for example measuring blood pressure every morning right after the care-receiver wakes up. However, it is likely that a care-giver forgets to measure blood pressure for a specific care-receiver or measures it late which might result in unreliable readings. A TCH system can be used to remind a care-receiver to attach the measurement tool whenever it is required.
– **Observation Error**: The care-givers give reports about the situation of the care-receivers to the doctors, family members, pharmacy, etc., based on which the authorised stakeholders take appropriate actions. For example, a doctor can prescribe new medication based on the current situation of the care-receivers, then the pharmacy can provide it, and the care-giver can give the medicine to the care-receivers. However, while measuring the care-receivers vital signs such as blood pressure, the care-givers can make mistakes in reading/writing of values, which can affect the diagnosis made by the doctor. A TCH system can automatically create a correct report based on the measured data which is accessible by a doctor.
– **Action Error**: The care-givers can make a mistake in providing care services to the care-receivers. For example, a care-giver can provide a wrong medicine to a care-receiver. A TCH system can be used to provide medication through a digital dispenser filled with medicine by a pharmacy based on doctor's prescription. We should take to account that there is still the possibility of the pharmacy making a mistake when filling the medicine dispenser.
– **Overlooking the Medical Protocols**: Nurses should follow the medical protocols in providing care services, and are examined yearly to prove that

they still recall them. However, a care-giver may overlook these protocols and make a wrong decision. A TCH system ensures the conformance of medical protocols, because those protocols are embodied in the treatment patterns.

- **Conflict in Medication/Treatment**: Care-receivers typically use multiple medications which are prescribed by different specialists. A doctor may prescribe a medicine without considering other prescribed medications that can have negative effects at other diseases/medicines. It is also common that for a specific disease, a doctors prescribes a new medicine, which has better effect, but without stopping the previously prescribed medicines. It is also possible to have conflict in treatment. For example, a care-receiver can use advise or treatment from different professionals, such as a family doctor, hospital doctor and physiotherapist, which may in itself be correct, but not optimal if used in combination. This risk can be easily detected by a TCH system, if there are predefined rules for conflicting medicines/treatments.

Risks of Using the TCH System One of the key motivations for replacing manual activities with automatic IT-based systems is *human error reduction*. However, many practical experiences shows that in reality, automation may produce new sources and types of errors [17]. This is also true when a TCH system is used in providing personalized IT-based homecare services. We identify the following risks for each stakeholder that interacts with the TCH system.

- **Care-givers:**

  Wrong Configuration: Setting the wrong values for the configuration parameters, e.g., setting higher/lower values for the threshold of blood pressure.

  Conflicting Service Plans: Creating conflicting service plans. An elderly usually suffer from a combination of diseases and hence, a care-giver may ignore the suggestion from the system and potentially create conflicting service plans for the same care-receiver. For example, in a service plan a care-receiver is asked to take his blood pressure at 8:00 AM while in another service plan he is asked to take a walk at the same time.

  Missing Service Plan: Forgetting to create a service plan for a care-receiver.

  Too Little Information: The face-to-face communication will be decreased, so care-giver's knowledge about care-receivers' situation will be limited.

- **Care-receivers:**

  Cheating with the System: Lying to the system whenever a confirmation is needed, for example, a care-receiver may lie about taking the medicine.

  Increased Loneliness: Feeling lonely is a common issue among elderly people. This could get worse by using a TCH system.

- **IT Specialists:**

  Inappropriate Treatment Patterns: Translating the medical protocols to treatment patterns incorrectly or providing incomplete patterns. The possibility of occurring this risk is very low, because this a one-time activity (with minor changes per year) and the incompleteness or incorrectness of the patterns can be detected easily during a testing phase.

- **Third-party Service Providers:**

Service Failures: Services provided by the third-party service providers may not function or function improperly. These services such as the blood pressure measurement service are outside the control of TCH system.

– **Infrastructure Providers:**

Data/Power Network Failures: The data/power network may go down.

– **Hackers:**

Malicious Action: The care-receiver data or configured service plans may be altered/stolen. In fact, we do not consider hackers as new source of risk, since thieves can do the similar damage in the existing situation.

5 Discussion and Conclusions

Context-awareness is becoming an important aspect of any information system. We all can imagine what new features such a system can have: selection of fitting services and adapting system behavior and providing services tailored to users' needs and preferences. However, such a context-aware system can raise new risks because of unpredictable behavior. To the best of our knowledge, there is a limited amount of research and information regarding what new risks and challenges such a system can pose to its users. In this paper, we assume that the context-aware system can perceive and process context information accurately and in-time. In other words, we are only interested in undesired outcomes arising from the interaction of the user/services with a context-aware system.

In order to discuss the risks of TC systems, we need to clarify our understanding of context and context-awareness. In the literature, there are a number of definitions for context, however it is still difficult to say what information is context information and what is not. In this paper, we do not provide a new definition of context-awareness, but analyse context-aware systems and identify their potential risks. We consider context as any information that can be used to adapt the response of a system and add value to provided services for target users. A context-aware system mainly performs context-triggered actions in the form of 'If-then' rules to specify how the system should be adapted [13]. We discuss further the facts that can be generalised to any TC system.

5.1 What Can Go Wrong in TC Systems?

Since we are interested only in risks arising from the interaction with the tailorable context-aware system, to be able to generalize the identified risks, first we should identify the main type of stakeholders who interact with the system. We also discuss why they are the source of risks. Generalizing from our case study, four types of stakeholders interact with the tailorable context-aware systems:

Developer (IT specialists in the case study): Designs and implements the system. Because of lack of domain knowledge, the developer can be the source of risks. However, since a developer's action is not a frequent one, the impact of such a risk is relatively low. Once the risk is identified, it can be fixed and the probability of occurring the same risk again remains low.

Configurator (Care-givers in the case study): Has enough domain knowledge and configures the system parameters and creates new services to satisfy individual needs. Because of lack of IT knowledge, the configurator can be the source of risks. Since he uses the system frequently, the possibility of same risks occurring repetitively is high.

End-user (Care-receivers in the case study): Is a target user and benefits from the output of the system. Because of the lack of knowledge or interest about the system, the end-user can be the source of risks. Even though an end-user has higher rate of interaction with the system, since he usually performs the same type of actions (such as confirmation of receiving a message/service), possibility of same risks occurring repetitively is relatively low.

Third-party Service Provider (Third-party providers in the case study): Provides third-party services (services outside of the control of the system). Because of the lack of amenability, the third-party service provider can be a source of risks. Since his services have a higher rate of interaction with the system, the possibility of same risks occurring repetitively is relatively high.

Based on the fact that the *configurator* and *third-party service provider* are the main sources of risks, we limit the types of risks which are caused by them. Looking at the list of risks we identified for the homecare domain, in the following we generalize risks which can occur in any tailorable context-aware systems.

Wrong Configuration Values: This type of risks occurs due to the tailorability of the system. It is important to consider what can be tailored and what not. This risk happens when a user puts a configuration value which is not in the context model. For example, suppose that the location of a care-receiver is modeled as only *inside home* and *outside home*. A care-giver may insert a value *at park* as the location value. This type of risks can easily be prevented by limiting the possible configuration values, e.g., by checking the value against the model, decreasing the likelihood of this type of risk.

Conflict in a Task with Multiple Context Information: This type of risks occurs due to bad reasoning of the system regarding a task with multiple context information. The system should decide what to do based on context information, however there will be a conflict if there are two different actions for different context information. For example, a reminder should be sent to a care-receiver mobile phone when he is outside home and based on another reasoning he should not receive any reminder message on his phone when he is with somebody. So if he is outside home and is accompanied by somebody, there will be a conflict on sending the reminder message. Since this conflict occurs in one task (for example sending a reminder), a context-aware system can be designed in a way to detect such a conflicts and inform user in advance. This type of risks can be prevented by prioritizing the rules.

Conflict of Different Tasks: This type of risks occur because of performing different tasks which are in conflict. For example, one task is to attach a blood pressure measurement tool and the other task is to take a walk outside. Since this risk occurs in different tasks, it is difficult to detect and prevent it.

Third-parties Service Failure: Service Oriented Architecture (SOA) is becoming popular for designing IT-based systems. A SOA-based context-aware system may utilize services offered by different providers. There are usually Service Level Agreements (SLAs) among the partners to assure the availability of the services. However, in some domains like homecare, which is a safety critical domain, relying on the SLAs may not fully compensate the risks that occur.

The accountability aspects are mainly concerned with identifying the source of the risks and ultimately making that source responsible. Unlike in existing definitions, we treated the accountability as a means of identifying the source of risks (identifying where and how it can be fixed). This treatment is realistic because in the homecare domain, regardless of who is making a mistake, the care center is accountable for any risks that arise to its customers.

The risks due to wrong *configuration values* may lead to unavailability of desired services because the wrong configuration values may cause the system to behave differently. This kind of risk can also be classified as the accountability aspects, because the source of the risk can be traced back.

The risks due to *conflict of different tasks* and the *conflict in a task with multiple context information* may lead to providing undesired services, which can be considered as unavailability of the desired services. It might be difficult to exactly identify the source of risk and hence the accountability, because risk occurs when a newly created task conflicts with the existing one which might have been configured by a different configurator.

The risk due to *third-parties service failure* will lead to unavailability of desired services. The source of the risk can be identified only in terms of service providers, and accountability aspect should be considered in SLAs.

5.2 What More is Needed?

We have performed a first assessment of risks of TC systems in general, and of homecare systems in particular. This has led to a classification of *availability* and *accountability* risks. This is new regarding the current risk assessment literature of context-aware systems, which is mainly about security and privacy risks.

There are some limitations to this study. We have done only one case study and even in this study, we may not have found all available risks. We have interviewed the care-givers about the tasks they perform, and then identified the list of possible risks while they perform their tasks. However, there is a possibility that the interviewees have forgotten important tasks and accordingly important risks. Based on their explanation, we have listed possible risks and there is a possibility that those risks are not real risks. In other cases, other risks may exist too, that are not present in our investigated case. Despite these limitations, we can still claim that we have found a list of *possible* risks. We intend to do more case studies in the near future to identify the importance of the risks as well as the completeness and correctness of the list of identified risks.

Another aspect of future work is to further confirm and elaborate the risks that we found for homecare systems. In addition, we would like to extend this assessment towards requirements engineering, by incorporating the risk assessment as a first step in a requirements engineering process for homecare systems.

References

1. Amigo: Ambient intelligence for the networked home environment project (2008), available at: http://www.hitechprojects.com/euprojects/amigo
2. Baldauf, M., Dustdar, S., Rosenberg, F.: A Survey on Context-aware Systems. IJAHUC 2(4), 263–277 (2007)
3. Batet, M., et al.: Knowledge-driven Delivery of Home Care Services. JIIS pp. 1–36 (2010)
4. Bellotti, V., Edwards, K.: Intelligibility and Accountability: Human Considerations in Context Aware Systems. HCI 16, 193–212 (2001)
5. Duffus, J., Brown, S., Fernicola, N.: Glossary for Chemists of Terms Used in Toxicology. Intl. Union of Pure and Applied Chemistry 65, 2003–2122 (1993)
6. European Commission: Ageing well in the information society - an i2010 initiative - action plan on info. and comm. tech. and ageing. Tech. rep., EU (Jun 2007)
7. Hong, J., Suh, E., Kim, S.J.: Context-aware Systems: A Literature Review and Classification. Expert Syst. Appl. 36(4), 8509–8522 (2009)
8. Hussein, M., Han, J., Colman, A.: Context-Aware Adaptive Software Systems: A System-Context Relationships Oriented Survey. Tech. rep. (2010)
9. ISO: Risk management – principles and guidelines. Intl. Std. 31000 (2009)
10. ISO/IEC: IT – Security Techniques – Guidelines for the Management of IT Security – Part 1: Concepts and Models for IT Security. Intl. Std. 13335-1 (2004)
11. Korhonen, I., Parkka, J., Van Gils, M.: Health Monitoring in the Home of the Future. Engineering in Medicine and Biology Magazine, IEEE 22(3), 66 – 73 (2003)
12. Leveson, N.G.: Engineering a Safer World: Systems Thinking Applied to Safety, p. 4. To be published by MIT Press in Fall (2011)
13. Schilit, B., Adams, N., Want, R.: Context-aware Computing Applications. In: Workshop on Mobile Computing Systems and Applications. pp. 85 –90 (1994)
14. The Council of The European Union: Council Conclusions on a Safe and Efficient Healthcare through eHealth. In: Official Journal of EU (December 2009)
15. United Nations International Strategy for Disaster Reduction UN-ISDR: Terminology on disaster risk reduction. Geneva (May 2009)
16. White, C., et al.: Improving Healthcare Quality through Distributed Diagnosis and Home Healthcare. In: Transdisciplinary Conference on D2H2. pp. 168 –172 (2006)
17. Wiener, E.L.: Cockpit Automation. In: Human factors in aviation, Academic Press series in cognition and perception. pp. 433–461 (1988)
18. Zarghami, A., et al.: Dynamic Homecare Service Provisioning Architecture. In: IEEE Conf. on SOCA. pp. 213–220 (2011)
19. Zarifi Eslami, M., et al.: Flexible Homecare Application Personalization and Integration Using Pattern-based Service Tailoring. In: CIT. pp. 467 – 474 (2011)
20. Zarifi Eslami, M., van Sinderen, M.: Flexible Home Care Automation. In: IEEE 3rd Intl. Conf. on PervasiveHealth (2009)

Social Forking in Open Source Software: An Empirical Study

Kam Hay Fung[1], Aybüke Aurum[1], David Tang[1]

[1]School of Information Systems, Technology and Management,
The University of New South Wales, Australia

kamhayfung@ieee.org, aybuke@unsw.edu.au, dtang@atlassian.com

Abstract. Forking is the creation of a new software project by making a copy of artefacts from another project. Forking is gaining traction in industry because of the maturity of distributed version control systems and the abundance of open source software (OSS) and hosting platforms that support forking. However, forking in OSS is a poorly understood practice in research, often assumed to be damaging to the open source community. This research aims to explore social forking. It uses a conceptual model for forking centring on three key concepts - forks (i.e. created projects), communities (i.e. groups of forks) and contributions (i.e. changes contributed from a forked project to the project from which its artefacts were copied) - to empirically analyse nine public domain JavaScript development communities in GitHub, a web site for hosting social coding. The analysis examined the relationships of these communities, the nature of forking, and the way in which forking and contributions were used in a social setting.

Keywords: forking, cloning, open source software, social coding

1 Introduction

The open source software (OSS) initiative has provided an invaluable source of learning for the software industry, running counter to many existing theories and explanations [20], and offering practices and characteristics in contrast to commercial software development. While much of the open source landscape has been explored – in terms of participants' motivations [8], social and technical characteristics of projects, management issues, legal issues, adoption and business models – a changing technological milieu is likely to offer new opportunities for learning, as it nudges the open source community in subtle but interesting ways. This is because OSS development, as a geographically distributed and asynchronous process, is largely mediated by Internet technologies such as mailing lists, wikis, bug trackers and version control systems [1] – which, as with any technology, evolve over time.

Distributed revision control systems (DRCS) [13] provide the infrastructure and platforms for hosting OSS projects. DRCS, as distinct from the traditional, centralised revision control systems, enable the sharing of code between peers without communicating through a central server [9]. These platforms encourage developers to *fork* each other's projects [4] – that is, to *copy* and *publish* an OSS's code from repositories owned and maintained by others to make changes. These changes can be promoted to the original OSS (e.g. as enhancements) or used to manifest it into a different OSS. The social phenomenon of forking refers to the heedful interaction of members in an OSS community, via forking, in driving the advancement of the OSS.

We do believe that forking has its importance and practicality in OSS development. It, however, has received mixed reception in the literature and which is often contingent on anecdotes and conjecture [4]. The lack of interest in and in-depth research on this topic also stems from the fact that forking in OSS is considered to be unlikely to occur [15]. This brings us to our research question:

How is forking utilised to facilitate OSS development?

This research serves as a first step towards improving the understanding of forking in OSS and hopefully fuelling research in this direction for the benefit of OSS development by online communities. This paper is organised as follows. A review of the literature is given in Section 2. In Section 3, we describe our proposed conceptual model for social forking as a way to explain its key concepts. Section 4 presents our research methodology to empirically examine a web site that facilitates forking and OSS development, using our conceptual model as a guide. Section 5 provides our results and Section 6 presents our conclusions and a discussion of future work.

2 Literature Review

Forking is the creation of a new software project from the same code base as another [4] by anyone, with or without the knowledge or consent of the creator of the original project, as unlike commercial software licences OSS grants the right for anyone to access, modify and redistribute source code [14]. The definitions of forking vary subtly in emphasis. It can be merely seen as the splitting up of an OSS project into two or more projects which are then developed separately. [17]. Sometimes forking is weighed in with dire consequences. It is regarded as the splitting up of an OSS project into competitive [2] and incompatible [5] strands of OSS. Nevertheless, forking fosters the code base of an existing OSS to be moved in a different direction than that of its erstwhile project leadership [4].

It has been claimed there is a strong taboo against forking [5]. It is believed that forking divides an OSS community by weakening both user and developer involvement [11]. Projects forked from another dilute the attention of end users of OSS, the user base of the original project and the support network around it. They also starve the original project of developers, who need to split their effort for different forked projects [11]. Forks are also seen as a band aid solution for technical disagreements and interpersonal conflicts that cannot be resolved in the forking project [6]. Thus, leaders within an OSS community strive to directly resolve those issues within the forking project so as to reduce the need for forks [11].

Forking is seen to distance developers of an OSS community from the original project from which forks were created (i.e. the forking project) and make them lose interest in the project [11]. There is also the potential for duplication of effort in different forks, and rivalry among developers (e.g. claiming their forks are superior) [3]. A developer might face the loss of their reputation since (s)he is unlikely to contribute to multiple forks as well as the forking project and claim the credit for all his/her effort [5, 15]. Even when a developer can make modifications in a fork for a good cause, (s)he can feel a sense of rejection when modifications are but declined(?) by administrators of the forking project [7].

Forking has been touted as a danger to OSS development and its adoption [12] since forking has the potential for fragmenting the design into competing [2] and incompatible versions [5]. Whilst it is easy to create forks, the number of forks for a project, and hence variants of it, can explode, leading to a loss of commonality [7], the original intent and main characteristics of the software produced from the forking project.

On the positive side, forking permits specialisation of OSS for different needs [10]. For example, parties in an OSS community may use forking to extend or customise a standard implemented in the OSS to their advantage [19]. The OSS in a forked project can also evolve separately from the project from which it was forked to satisfy new requirements [4]. New features are experimentally developed in forks independently of the OSS from the original project. Variants of the OSS (e.g. for Apple iPhone vs. for Android) from the original project can also be developed in a fork. Hence, forking is also regarded as a source of innovation [2].

In an OSS development environment, an individual has some freedom to fork a project and choose to work on whatever part of the OSS suits his/her interest, agendas and approaches [11, 16], irrespective of time and geographical constraints. Forking also provides developers leeway in exploring alternatives for OSS [16]. Developers can also work at a fast pace without being bogged down by the bureaucracy of consensus-driven processes that manage changes [10]. When developers work on different forks, they have the opportunity to compete with one another by developing the OSS solution of best interest to their OSS community.

3 A Conceptual Model for Social Forking

To facilitate our investigation, we firstly define a conceptual model for social forking and its terminology. Forking is the task of creating a new software project from existing artefacts of another software project. Artefacts are not limited to source code. They can be anything related to the software that a project aims to develop, such as documentation, examples, test harnesses and third party libraries. The forked project (or simply the fork) can be used for enhancement, bug fixes, innovation and so on. A fork and the one from which it is forked forms a *successor-predecessor* relationship. A developer of a fork has a vested interest in the original project from which it was forked but the owner of the original project may not necessarily be aware of the forks created and their development activities.

When changes are made to the original project, the underlying DRCS notifies coders of its forked projects about the changes. They may then review the changes and incorporate them as an *update* to their forks as at their own pace, without any involvement of the owner of the original project. The social aspect of forking comes into play when social interaction occurs in order for a successor fork to make *contributions* [3] to a predecessor fork. In a simple case, a coder who has forked a project makes changes to it, notifies the owner of the predecessor project of the changes and interacts with the owner by exchanging comments about the changes. The owner then incorporates the changes into the predecessor project.

Forking is utilised at two layers of abstraction: endogenous and exogenous. *Endogenous forking* occurs within a community of social developers who work on forks for the same software product line. For instance, one creates a fork to enhance an existing feature of a software product. A fork tree for forks in a community can be represented as a tree structure. At the top of the tree is the *master* fork of the community from which all forks are created. *Primary* forks are those directly created from the master fork. Likewise, *secondary* forks are those created for primary forks.

Exogenous forking refers to the creation of forks across communities of social developers. Through exogenous forking, import and export relationships are established across community borders. In the former, a copy of a master fork in one community is *imported* into a master fork in another community. It can be used for bringing a copy of a third-party library into another project for creating a new software product. This import operation decouples activities of coders in one community from the other. The coders in the new community work with a baseline version or snapshot of the master fork, while those in the original community continue to evolve its master fork. Note also that the import operation only brings in whatever artefacts are required for the importing community. A more complex form of exogenous forking is to import forks from multiple communities into another community.

In an export relationship, artefacts in community B are purported to be an add-on or plug-in feature to artefacts in community A. Artefacts in B are essentially decoupled from those in B; the runtime code produced from artefacts in B can be run independently of those produced from A and vice versa. This independence also distinguishes between export and import relationships.

A fork is to a fork tree what a branch is to a version tree for version control systems in which branches are created for auxiliary tasks to be carried out [18]. Auxiliary tasks include performing fixes, distributed development, custom modification, dealing with conflicting updates [18]. Although both a fork tree and a version tree (as well as their artefacts) are structured similarly and managed under version control, a fork tree is differentiated from a version tree in a few ways:

- A fork can be used to initiate a separate strand of independent development for a new OSS. Unlike those in a branch, changes made in the fork need not be promoted to its predecessor;
- A fork can be created from only a *subset* of the artefacts from its predecessor whereas in a branching operation, a whole copy of a project's artefacts is made into its branch; and

- A fork can be created from *two* or *more* predecessors, thereby bringing features from different predecessors. A branch can only have one predecessor.

4 Research Methodology

A three-step empirical study was carried out to examine the phenomenon of social forking using our conceptual model as an analysis framework. In step 1, we chose and examined Github (https://github.com/) since it is one of the top websites based on the number of OSS development projects hosted[1]. We started our search for communities (i.e. software repositories in Github) using JavaScript programming language since JavaScript was the most popular in GitHub (20%) and it might also increase our chance of finding exogenous forking. We then narrowed our selection to those with the highest number of forks which was indicative of a high level of social activities and a rich source of forking data. In step 2, we developed and ran a tool against each of the software repositories identified to extract data from them via GitHub's public APIs (which are RESTful Web Services); to transform extracted data into a format based on our conceptual model; and to load the transformed data into a relational database. In step 3, the empirical data loaded in the database were explored to identify patterns of social forking. Our analysis of the results is presented next.

5 Results

The top nine JavaScript development communities hosted by Github having the highest numbers of forks were selected and used in our empirical analysis in November 2011. We investigated the relationships of the communities by examining their documentation and the files in their repositories. The communities exhibit import and export relationships and there are two variants in the former. In one case a full copy of the artefacts from community A is imported into community B whereas in the other case only a subset of the artefacts from A were imported into B. **Fig. 1** depicts the analysed communities and their relationships. The number labelled alongside each relationship instance represents the number of integration contributions made to a community as a result of maintaining the import/export relationships. This is further elaborated, together with the analysis of results for contributions, in Section 5.1.

5.1 Forks

All the communities we investigated had forks created at the primary and secondary levels, with two having forks at the tertiary level. Of the (7789) primary forks examined, 3.2% of them were used to create secondary forks and about the same ratio were used for the secondary forks. These figures are indicative that sub-communities were formed within the communities, which means developers participated in developing

[1] http://en.wikipedia.org/wiki/Comparison_of_open_source_software_hosting_facilities

features in the primary forks through their secondary forks. We observed that some primary forks were created to develop brand new innovations in the community based on their respective master forks, but were not necessarily intended to be incorporated into their masters. For instance, “nodejsjp/node” is a primary fork for “joyent/node” for the purpose of developing a Japanese version of “joyent/node”. Indeed, these forks in conjunction with their successor forks formed sub-communities.

Of all the primary forks created, only 14% offered contributions to their respective master forks (11% for secondary to primary forks). A possible explanation for the high ratio of non-contributing forks (86% for primary and 89% for secondary) is that their developers were using forks to learn or experiment with the OSS’s features only.

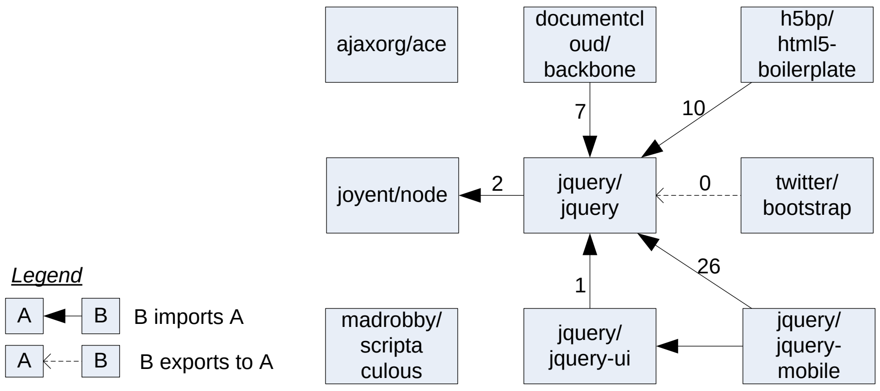


Fig. 1. Reviewed communities and their relationships

5.2 Contributions

The kinds of contributions made to each fork in the fork tree can be seen as a reflection of the ways in which their successor forks were used in each community. We analysed the titles and descriptions of over two thousand contributions and identified keywords which we subsequently used to categorise the contributions as defect fixes(43%), code enhancement(12%), documentation(7%), integration(2%), example enhancement(2%) and changes to test code and documentation(1%). The rest could not be categorised because of insufficient information. The integration category is triggered by the import relationship between two communities. For instance, when new features are added to artefacts in one community, they may also require import to the community that imported the original version of these artefacts. The artefacts in the latter community may also require changes to its artefacts in order to utilise the new features imported from the former community.

Of the contributions analysed from over one thousand contributing forks, the contribution rates (i.e. number of contributions divided by number of contributing forks) were 2.3 and 2.5 for contributing primary and secondary forks respectively. Note that contributions have the potential to be propagated further up the fork tree. This was evident from one case found during our examination of the contribution logs. A secondary fork (“flavaflav/jquery-mobile”) produced a contribution to its primary fork (“arsduo/jquery-mobile”) and the changes associated with that contribution were packaged as yet another contribution which was subsequently incorporated into the master fork (“jquery/jquery-mobile”).

5.3 Users

Close to seven thousand developers created about eight thousand forks in the OSS communities analysed. We discovered some interesting behaviour of forking. In an extreme case one developer created forks in eight out of the nine communities analysed (i.e. working on projects in multi-communities). Six developers individually created more than one fork in the same community. This could be because each of these developers was using several forks to work on different parts of an OSS. 16% of developers made contributions to their OSS communities and from which we also identified some interesting behaviour of contribution. 5% of these developers made contributions to more than one community, with the top eight having made contributions to three communities. The most active developer made one hundred and forty-three contributions! The OSS community most attracted to contributions was jquery/jquery in which each developer created 3.3 contributions on average.

6 Conclusions, Limitations and Future Work

There has been a lack of critical studies into forking in OSS. In the literature there is a debate about whether or not forking should be used; some contend it as damaging to OSS development whilst some advocate its benefits to OSS communities and the OSS developed. To improve on this lack of knowledge about forking, we empirically investigated nine OSS communities from GitHub to gain an initial understanding of how it was used to facilitate OSS development in a social setting. It shows that forking was actively used by community participants for tasks such as fixing defects and creating innovations. “Forks of forks” were also utilised to form sub-communities within which specific aspects of an OSS product line were nurtured.

Since our study is in its infancy, we limited our analysis to one OSS hosting website, one programming language and nine communities, which poses validity threats to our findings. We thus plan to expand our study with additional web sites hosting OSS development (e.g. SourceForge, BitBucket, Gitorious) and interviews with OSS communities’ members to gain better understanding of their perspectives on social forking. The excitement around social forking combined with our research to date will hopefully invigorate future research in this area and increase its uptake in practice.

7 References

1. Barcellini F, DÈtienne F, Burkhardt J-M, Sack W (2008) A socio-cognitive analysis of online design discussions in an open source software community. Interacting with Computers 20(1):141-165
2. Bitzer J, Schröder PJH (2006) The impact of entry and competition by open source software on innovation activity. In: J. Bitzer aPS (ed.): The Economics of Open Source Software Development 219-246
3. Cheliotis G (2009) From open source to open content: Organization, licensing and decision processes in open cultural production. Decision Support Systems 47(3):229-244
4. Ernst NA, Easterbrook SM, Mylopoulos J (2010) Code forking in open-source software: a requirements perspective. CoRR abs/1004.2889
5. Feller J, Fitzgerald B (2000) A framework analysis of the open source software development paradigm. Proc. 21st Int. Conf. Info. Systems 58-69
6. Fogel K (2010) Producing open source software. http://producingoss.com/en/index.html. Accessed on 15 Nov 2011.
7. Glass RL (2003) A sociopolitical look at open source. Communications of the ACM 46(11):21-23
8. Hertel G, Niedner S, Herrmann S (2003) Motivation of software developers in open source projects: an Internet-based survey of contributors to the Linux kernel. Research Policy 32(7):1159-1177
9. Lanubile F, Ebert C, Prikladnicki R, Vizcaíno A (2010) Collaboration tools for global software engineering. IEEE Software 27(2):52-55
10. Moen R (1999) Fear of Forking. http://linuxmafia.com/faq/Licensing_and_Law/forking.html. Accessed on 22 May 2011.
11. Muffatto M, Faldani M (2003) Open source as a complex adaptive system. Emergence 5(3):83-100
12. Nagy D, Yassin AM, Bhattacherjee A (2010) Organizational adoption of open source software: barriers and remedies. Communications of the ACM 53(3):148-151
13. O'Sullivan B (2009) Making sense of revision-control systems. Communications of the ACM 52(9):56-62
14. Open Source Initiative The Open Source Definition (Annotated). http://www.opensource.org/osd.html. Accessed on 01 June 2011.
15. Raymond E (1998) Homesteading the noosphere. First Monday 3(10)
16. Raymond E (1999) The cathedral and the bazaar. Knowledge, Technology & Policy 12(3):23-49
17. Stewart KJ, Gosain S (2006) The impact of ideology on effectiveness in open source software development teams. MIS Quarterly 30:291-314
18. Tichy WF (1985) RCS - a system for version control. Software: Practice and Experience 15(7):637-654
19. Vetter GR (2007) Open source licensing and scattering opportunism in software standards. Boston College Law Review 48(1)
20. von Krogh G, Spaeth S (2007) The open source software phenomenon: Characteristics that promote research. The Journal of Strategic Information Systems 16(3):236-253

ProMetheuS: A Suite for Process Mining Applications

Lucantonio Ghionna[1], Luigi Granata[2], Gianluigi Greco[1], and Massimo Guarascio[3]

[1]Dipartimento di Matematica, Università della Calabria, I-87036 Rende, ITALY
{l.ghionna,ggreco}@mat.unical.it
[2]Exeura SRL, Via Pedro Alvares Cabral - C.da Lecco, I-87036, Rende, ITALY
{luigi.granata@exeura.com}
[3]ICAR-CNR, Via P.Bucci 41C, I-87036, Rende, ITALY
{guarascio@icar.cnr.it}

Abstract. Process mining is an established approach for analyzing and modeling complex business processes. In this paper we showcase ProMetheuS, a flexible and scalable suite for process mining natively designed for industrial applications. Moving from the experience of the *ProM* framework, the state-of-art process mining tool, ProMetheuS introduces three innovative designing elements. Firstly, ProMetheuS defines the concept of flow of mining, which is aimed at supporting the design of complex mining applications, where various mining tasks can be combined and automatically orchestrated at run-time. Secondly, ProMetheuS exports a rich set of facilities to help developers in building interactive applications providing on-the-fly feedback during analysis. Finally, behind the scenes, a powerful stream-based log-handling subsystem ensures scalability in data-intensive applications.

1 Introduction

In the context of enterprise automation, *process mining* is an established approach to support the analysis and the design of complex business processes [1]. In a typical process mining scenario, the goal is to derive a model for a transactional process capable of explaining all activities registered in some log given at hand. Eventually, the "mined" model can be used to design a detailed process schema possibly supporting forthcoming enactments, or to describe its actual behavior.

The *ProM* framework [2] is an open and extendable tool for process mining, which enables users to write and import their own mining algorithms as plug-ins. *ProM* currently supports a wide range of process mining applications (e.g., control-flow mining, decision tree induction, or clustering, to cite a few) and analysis tasks (e.g., validation of process models, performance analysis, or statistical evaluations). Thanks to this valuable packaging, *ProM* represents the state-of-art tool for process mining, and many real-world scenarios exploiting its mining capabilities have been discussed in literature (see e.g., [3]). Despite its success, however, certain issues of flexibility and scalability might arise with the use of the framework, which limit its effectiveness in handling complex industrial applications [2].

In this paper, we describe ProMetheuS[1], a novel suite for process mining introducing innovative designing elements, which are aimed at facing some limitations of *ProM* and at providing new insights on the development of process analysis software.

First, ProMetheuS has been specifically conceived to support *the definition of complex mining applications, where various mining tasks can be combined and automatically orchestrated at run-time*: Process mining applications may involve dozens of different tasks, ranging from data acquisition, to data manipulation, information extraction based on different mining algorithms, recombination of mining results, and visualization. These different kinds of task can be managed in *ProM*, but at the price of requiring human intervention in their coordination. Indeed, constructing complex mining applications requires manually invoking the various tasks by collecting and storing each intermediate result and by reusing them as the input for some further tasks. *ProM* 6.0 has simplified the chaining of intermediate results by letting tasks be aware of the kinds of inputs/outputs they are supporting [2]. In order to automatize and easily deploy mining applications involving different tasks, ProMetheuS introduces instead the concept of "flow of mining", a very natural and manageable way of designing complex mining processes. Indeed, ProMetheuS supports the deployment of mining applications in their entity, by allowing to design mining processes as complex flows of elementary bricks. Each brick produces an output that may be used as input for other bricks in the flow. Consequently, users may incrementally build the desired flow, by connecting existing blocks or adding new ones to manipulate produced outputs. In fact, ProMetheuS comes equipped with a run-time engine that supports and monitors the execution of the mining flow and that orchestrates the compositions of the various elementary bricks. Notably, ProMetheuS allows the definition and the organization of bricks in *workspaces*, grouping resources according their application domain (e.g., text mining, rules learning, etc.). Resources belonging to different workspaces can be transparently connected together to build mixed flows.

Second, ProMetheuS allows users to *build interactive applications providing on-the-fly feedback during analysis*: A plug-in based architecture is a crucial factor to provide flexibility for real-world applications. However, each plug-in is current viewed in the *ProM* framework as a monolithic box, where interaction is limited to the startup phase in which users configure the execution environment of each algorithm by setting all parameters. ProMetheuS extends the flexibility of each plug-in by introducing an "interactive execution" mode (in addition to the standard "batch" one), i.e., it supports an approach to process mining where users may continually interact with the mining algorithms and provide feedbacks trough the graphical user interface.

Finally, ProMetheuS *ensures scalability over large volumes of data*: In real industrial environments, enormous volumes of data are available for mining analysis. Yet, few efforts (see e.g., [4]), have been spent to provide an adequate support for data-intensive applications. *ProM* imports the whole log into the main memory or, if this is not feasible, loads only a batch of data per time and stores remaining batches in disk-

[1]The system has been released by Exeura S.r.l.---under the name of "OKT Process Mining Suite"---as an open source software for the OpenKnowTech project founded by the Italian Ministry of University and Research (MIUR).

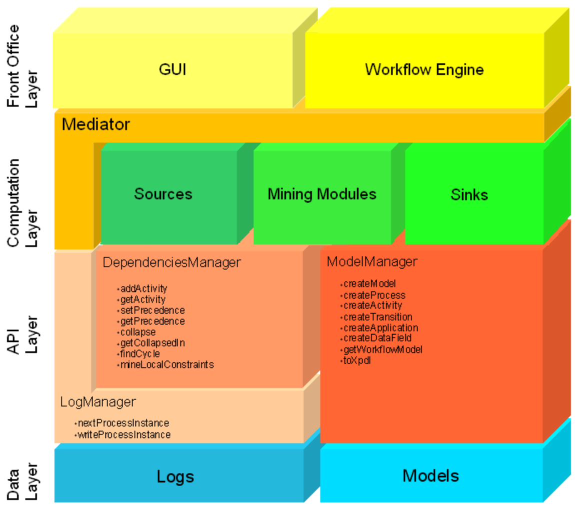


Fig. 1. ProMetheuS's Architecture

resident swap files, at the price of slower access time[2]. To face scalability issues, ProMetheuS adopts instead a data management subsystem based on a stream handling model for data acquisition. Thus, rather than building a complete in-memory representation of data, this model stores statistical sketches only, while supporting on-demand streaming access to the original log (no additional paging files are required).

2 ProMetheuS Architecture

As shown in Fig. **1**, ProMetheuS is implemented over four distinct logic layers. The *data* layer manages physical low-level operations for acquiring and storing elementary data types. The layer defines primitives for the input/output and for the modification of *Log* data, representing log files, of *Model* data, representing the abstraction of a process model, and of *Custom* data, representing user defined data-types. Regarding log representation, ProMetheuS supports the MXML data model [5], while a subset of the standard XPDL 2.0 [6] is used for model representation.

The *API* layer is responsible of two basic functionalities. Firstly, it supports the efficient internal storage of log files. In particular, it handles a main-memory repository storing *dependencies graphs*, i.e., graphs whose nodes represent the activities in the process log and whose edges represent the relationship of precedence among them. In fact, these structures are internally built by scanning once the input log, while further I/O operations may be executed when additional information is needed[3]. Secondly, it allows a transparent access to the data layer through dedicated *managers*. In details, a

[2]*ProM* 6.0 uses the *OpenXES* library---see http://www.xes-standard.org/openxes/start.
[3]*SAX* libraries are used for reading XML streams---see http://www.saxproject.org.

LogManager, a *DependenciesManager*, and a *ModelManager* provide primitives to manipulate logs, dependencies graphs, and models respectively (see Fig. **1**).

Above the API layer, it is placed the *computational* layer. In ProMetheuS, a computational resource is a plug-in component, which performs a specific task in a flow of mining. ProMetheuS provides three main templates for computational resource. A *source* is a template conceived to access the input data on which the mining analysis has to be performed. In particular, a *Log Source*, a *Model Source*, and a *Custom Source* are provided for handling logs, models, and custom data sources respectively. *Mining modules* are responsible of performing mining algorithms and statistical evaluations on the input provided by source modules. In particular, a *Log Miner* template manages a *Log* as input, and produces as output one or more instances of *Log*. A *Model Miner* template works on a *Log* input, and produces a *Model*. ProMetheuS comes equipped with various mining modules, with the default one being the α-miner [1]. A *Custom Module* template is conceived to work on *Custom* types. Finally, *sinks* templates are intended to manage the outputs of the mining process. These templates are useful for visualization, statistical analysis and storage of the computed results.

At the computational layer a *Mediator* manages communications between plug-ins and the *Front Office* layer. In particular, during the batch execution mode, the mediator automatically checks for the dependencies among the involved plug-ins, by tracing the state of the various executions and the execute availability of their input. Indeed, a plug-in may be executed only when all its inputs are available. Importantly, during the interactive mode, the mediator manages the interaction between the various graphical component associated with the different plug-ins. Basically, when the state of a component is modified, an event is generated and sent to the mediator, being then in charge of dispatching it to the other components, which can react accordingly.

The Front Office Layer exports GUI functionalities for the creation of a process mining flow, for the configuration of environment parameters, and for the visualization of results. A workflow engine is provided for the batch execution of the analysis.

3 ProMetheuS in action

We now overview the functionalities of ProMetheuS, by showcasing the complete deployment of a sample flow of mining. We also discuss some scalability results obtained by executing the flow in different configuration scenarios.

3.1 Designing and executing a flow of mining

To design a flow of mining, ProMetheuS provides the user with an intuitive GUI consisting of several graphical elements and facilities. The *Workspace Explorer* shows all available workspaces as navigable entries, in which plug-ins are organized according their type (i.e., source, mining modules, or sinks). The *Workarea* is the design panel on which users can freely customize mining flow properties. Users can quickly add/remove concrete instances of plug-in definitions (by dragging them from

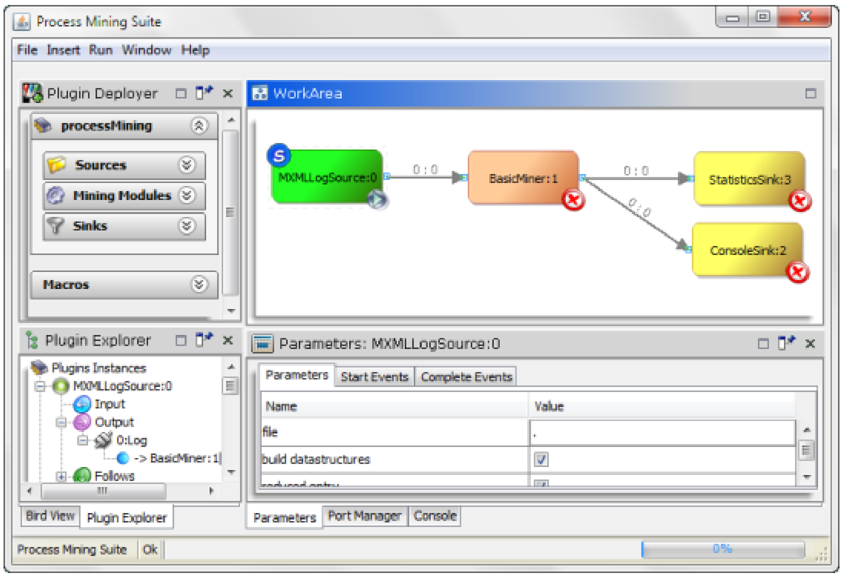
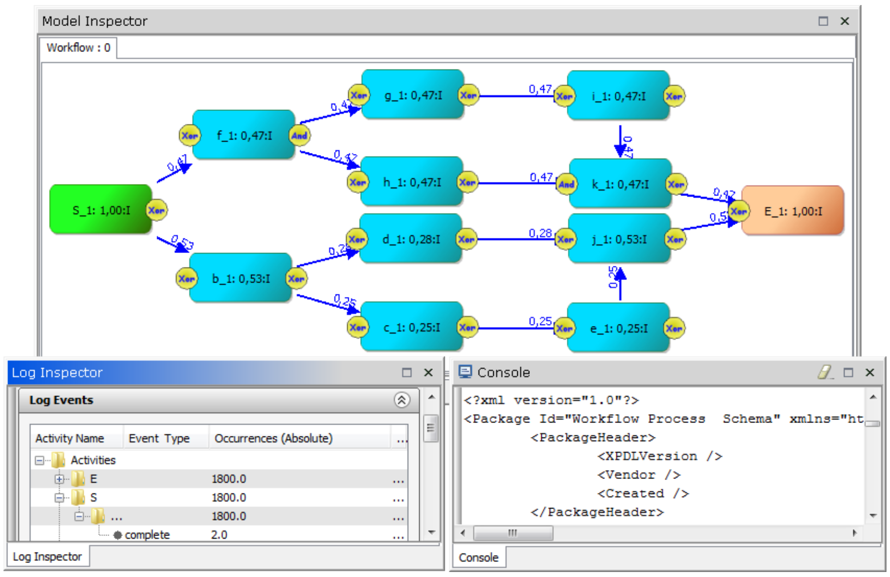


Fig. 2. Flow of mining: A *MXML Log* is connected to a *Basic α-Miner* redirecting computed model to sinks for visualization in XPDL format and statistics evaluations. Plug-ins's parameters can be configured by double-clicking on the plug-in instance.

the workspace explorer), edit connections between plug-ins, combine input/outputs, and control execution flow. Once a plug-in instance is placed, users can configure its execution environment in two steps: *Parameters Configuration*, where if the selected plug-in requires some input parameters, then users can proceed to their configuration, and *Edge Configuration*, where users can insert a new connection between modules, can rearrang a defined connection, or can remove it from the mining flow.

A sample flow of mining ready for the execution is depicted in Fig. **2**. Given a flow of mining, users can run all executable plug-ins at once, or execute only selected ones. Interestingly, users may define different flows in the same work area and run unrelated plug-ins in parallel, reducing the overall execution time of the analysis. After the computation, users can visualize the actual value of input/output data by using ProMetheuS's default *inspectors* graphical components. An inspector is a very generic data explorer, which is able to produce a suitable representation of a specific data type of the flow (see Fig. **3**). Notably, users can program their own inspectors for custom data types or can create multiple views on the same data set, each one depicting some portion of the data information of interest.

Plug-ins can be graphically composed in high-level blocks of components performing user-defined operations. In many occasions, it might be necessary to perform the same operation many times in the same mining flow or in different flows as well. In order to suite this need, ProMetheuS supports the grouping of connected plug-ins into *macros* that can be used as bricks with their own input and outputs (see Fig. **4**).



Activity Name	Event Type	Occurrences (Absolute)	...
Activities			
E		1800.0	...
S		1800.0	...
...		1800.0	...
complete		2.0	...

```
<?xml version="1.0"?>
<Package Id="Workflow Process  Schema" xmlns="ht
        <PackageHeader>
                <XPDLVersion />
                <Vendor />
                <Created />
        </PackageHeader>
```

Fig. 3. Inspecting the flow: The input/output of executed plug-ins can be inspected by clicking on flow arrows or on connection ports. The log inspector shows relevant statistics (e.g., number of activities) on the log, the model inspector draws the process workflow reporting summary information (e.g., frequency of a transition), the output inspector provides the resulting XPDL.

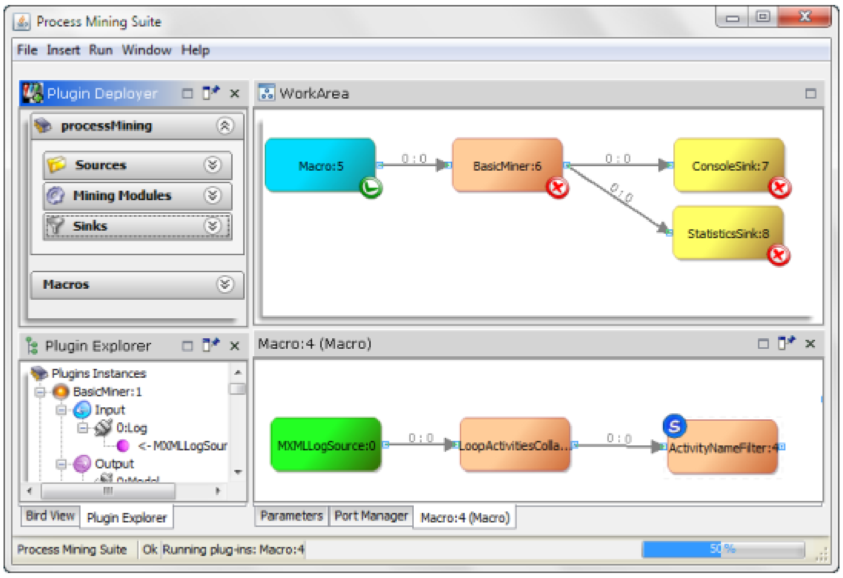


Fig. 4. Defining a macro: The source log is processed by collapsing looping activities, and by filtering activities names. Then, the output of the macro is used as input in the original flow.

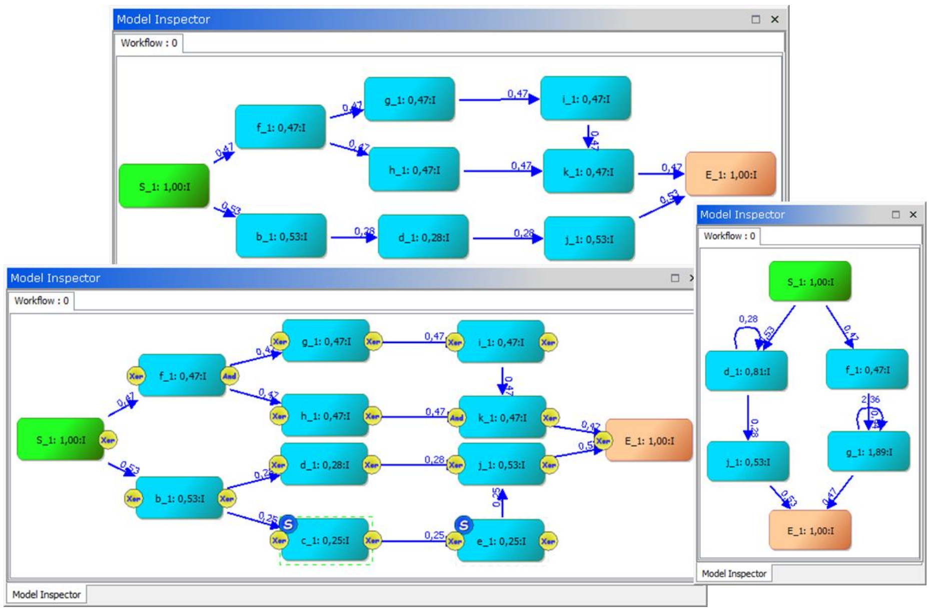


Fig. 5. Model refinement: The user runs the α-miner to compute a preliminary model. Based on summary information, (s)he sets a cut parameter for removing infrequent activities, an runs the α-miner again to compute a pruned model. Finally, (s)he identifies the desired workflow by interactively collapsing some of the intermediate model's activities through the interface.

3.2 Interactive refinement of intermediate results

ProMetheuS allows users to modify at the run-time the parameters of mining algorithms and to manipulate their execution logic on the basis of feedbacks they provide during the current execution. To support interaction, plug-ins can be equipped with customizable graphical components. In particular, a plug-in can be associated with a *menu*, with a *toolbar*, with a *main pane* (i.e., a graphical area for controlling execution), with a *bottom pane* (i.e., a panel for setting parameters), and with the *quick view* (i.e., a panel providing an overview of the plug-in status). Fig. **5** depicts a possible interactive refinement of a model computed by the α-miner of the flow of Fig. **2**.

3.3 Example execution

The execution time of the flow shown in Fig. **2** has been measured in both ProMetheuS and *ProM* considering different log sizes[4] (see Fig. **6**). Notably, the time required by ProMetheuS to complete the whole flow is much lower than the time *ProM* needs to just import the log. Moreover, as shown in Table **1**, the performances of the *ProM* buffered importer deteriorate quickly at the growing of the log size, because of the overhead of writing swap files.

[4] We used a Xeon 4 quad-core, with 8 Gb of Ram and running Ubuntu 11.04 Server.

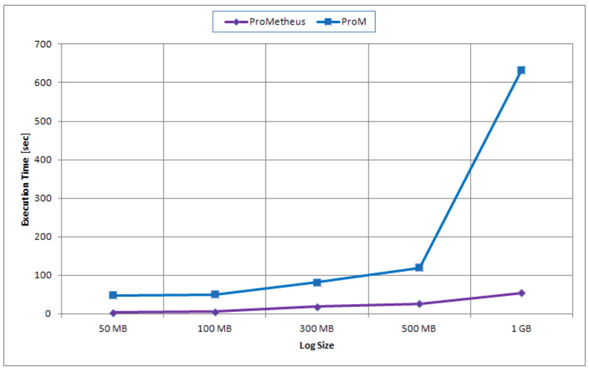


Fig. 6. Analysis Execution Times

Log size	**ProMetheuS**	***ProM***
1 Gb	55	1750
3 Gb	180	1800
5 Gb	225	1800
7 Gb	420	1800
10 Gb	580	1800

Table 1 Log importing time (timeout: 1800 sec)

4 Conclusions

In this paper we presented ProMetheuS, a suite for process mining applications [7] providing novel design elements. ProMetheuS is based on the concept of flow of mining, which enables user to design complex mining processes in which different mining tasks can be combined. Each task can be controlled interactively, and users can exploit run-time feedbacks to improve the quality of their analysis. In the case of mining large logs, the stream-based log handling may also help in achieving good scalability performances by just loading information needed for the analysis.

References

1. W. van der Aalst, A. Weijters and L. Maruster, "Workflow Mining: Discovering Process Models from Event Logs," *IEEE Transactions on Knowledge and Data Engineering,* vol. 16, no. 9, pp. 1128-1142, 2004.
2. H. Verbeek, J. Buijs, B. van Dongen and W. van der Aalst, "ProM 6: The Process Mining Toolkit," in *Proceedings of BPM Demonstration Track*, 2010.
3. N. van Beest and L. Maruster, "A Process Mining Approach to Redesign Business Processes - A Case Study in Gas Industry," in *Proceedings of SYNASC '07*, 2007.
4. J. Ingvaldsen and J. Gulla, "Preprocessing Support for Large Scale Process Mining of SAP transactions," in *Proceedings of Business Process Management Workshops*, 2008.
5. W. van der Aalst and B. van Dongen, "A meta model for process mining data," in *Proceedings of the CAiSE 05 Workshops*, 2005.
6. J. Ping, Q. Mair, and J. Newman, "Using UML to design distributed collaborative workflows: from UML to XPDL," in *Proceedings of IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, 2003.
7. W. van der Aalst, B. van Dongen, and J. Herbs, "Workflow mining: a survey of issues and approaches," *Data Knowledge Engineering,* vol. 47, no. 2, pp. 237-267, 2003.

Virtual Worlds as a Model-View Approach to the Communication of Business Processes Models

Hanwen Guo, Ross Brown, Rune Rasmussen

Information Systems School, Science and Engineering Faculty, QUT, Brisbane, Australia

hanwen.guo@student.qut.edu.au, r.brown@qut.edu.au, r.rasmussen@qut.edu.au

Abstract. Although business process analysis methods are mature today, business analysts and stakeholders are still hampered by communication issues. We argue that using a virtual world to model a business process can benefit communication activities. We believe that virtual worlds can be used as an efficient model-view approach, increasing the cognition of business requirements and analytic results, as well as the possibility of business plan validation. As an exploration paper, we believe that this promising research can encourage people to investigate more research topics in the interdisciplinary area of information system, visualization and multi-user virtual worlds.

Keywords: Virtual World, Business Process Management, Visualization

1 Introduction

An optimization and improvement process for a workflow system involves an intensive communication process between the stakeholder and business analyst [1]. According to communication theory [2], a general communication model adapted in the workflow system optimization and improvement process can be depicted in Fig.1. It is reported that business analysts and stakeholders often have communication problems [2-4]. On the one hand, stakeholders cannot always elaborate their business activities in a well structured way [3]. On the other hand, the visual code used by business analysts inevitably has noise, interfering with cognitive processes in the reader [2].

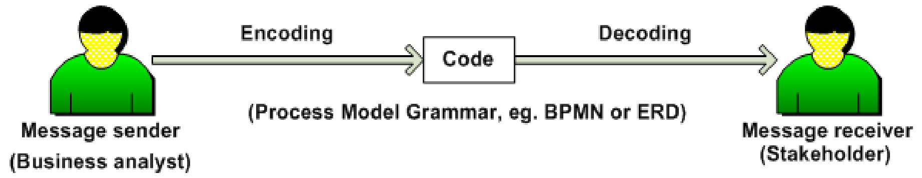


Fig. 1. A general communication model can be applied in the communication process between a business analyst and stakeholder.

Thus, it can be concluded that noise exists in the encoding and decoding process, as well as the visual code, reducing the possibility of stakeholder buy-in to the plan.

As a result, a solution to this problem is to use a semantically transparent code that can assist the reader in inferring the meaning of a code from its appearance [4].

Recently, it has been realized that 3D virtual worlds can be applied in social science [5]. This is because its richer visualization representation abilities enable people to effectively process more information [6]. This strongly suggests that 3D virtual worlds could be a superior process visualization platform, enabling people to recall and cognate about conceptual and non-conceptual content, facilitating the communication process in analyzing, modeling and validating organizational structure and resource behaviors, see Fig. 2.

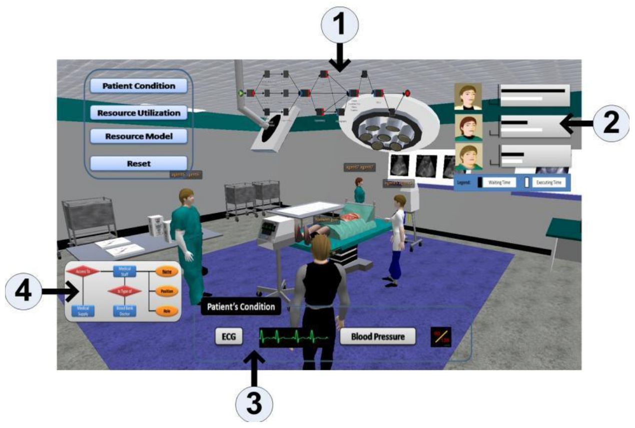


Fig. 2. Snapshots of an emergency treatment workflow visualized in a 3D virtual world, where several avatars are about to revive an injured person. Four HUD images indicate the current state of the visualized workflow system (1) and human resource workload (2), patient's condition (3), and entity relationship (4). This can help virtual world participant (black vest in the middle), whether a business analyst or stakeholder, recall what happens in reality or comment on the conceptual model.

This paper is organized as follows: Section 2 discusses related work. Section 3 explores the rationality of using a virtual world as an alternative communication approach. Section 4 uses a case study in the healthcare domain to demonstrate a set of visualization benefits offered by such a communication approach. At last, Section 5 concludes with a discussion of achievements, and points towards further work.

2 Related Work

The Entity-Relationship Diagram (ERD) can be used as a hands on modeling approach for native stakeholders. However, several researchers [7,8] pointed out the inappropriateness of the representation when an ERD is extend with attributes to represent complex relationships. In addition, Weber [9] concluded that applying an ontology in the modeling process can increase the understandability and perception of the information in a conceptual model.

However, these researchers did not address the issue that sound professional knowledge in information systems plays an important role in understanding modeling results [10], and we cannot guarantee that every stakeholder has such necessary knowledge. Compared with these previous works, this paper intends to provide a new model-view approach that can facilitate communication between stakeholders and business analysts, by allowing participants to observe actual activities at the operational level being juxtaposed with a conceptual model.

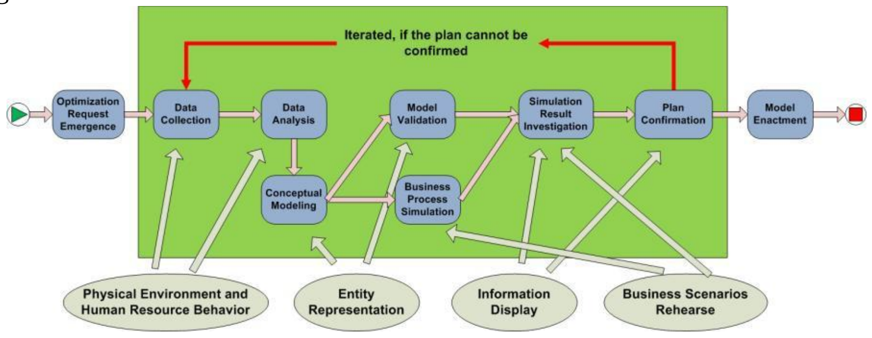
Currently, 3D virtual worlds have become popular research topics in E-commerce domains. Some researchers [11,12] have visualized a process models from the control, resource and data perspectives in a virtual world. Perkins [13] proposed an agent system that plays as an intermediate between a simple workflow engine and a virtual world for representing human resource behavior. Bogdanovych [14] established a methodology called Virtual Institutions (VI) to facilitate the communication between the customer and product sellers, which has a similar purpose to ours.

These works [12,11,13,14] have realized that the virtual world is powerful in demonstrating what is happening in an enterprise. However, they did not address how virtual worlds can be used as an alternative tool for facilitating communication in business process modeling tasks, in particular, how human resource models relate to process models.

3 Virtual World as the Model-View Communication Approach

3.1 Business Improvement and Optimization Activity Review

At the operational level, people are interested in the specific sequence of task events, personnel arrangement and resource behavior [15]. To satisfy these interests, we believe the following list of visualization aspects (but not limited to) should be addressed: Physical Environment and Human Resource Behavior, Entity Representation, Information Display, Business Scenario Rehearsals. These are diagrammatically represented as supporting points in the improvement process life cycle shown in Fig.3.



Iterated, if the plan cannot be confirmed
Optimization Request Emergence
Data Collection
Data Analysis
Model Validation
Simulation Result Investigation
Plan Confirmation
Model Enactment
Conceptual Modeling
Business Process Simulation
Physical Environment and Human Resource Behavior
Entity Representation
Information Display
Business Scenarios Rehearse

Fig. 3. The life cycle of a business process improvement. The phases requiring visual assistances are highlighted in a green rectangle by the ellipses at the bottom of the diagram.

- Physical Environment and Human Resource Behavior --- The states of a physical environment impacts the behavior of a human resource and business task transition [16]. Visualization of this aspect enables insight into relationships between a physical environment and business processes.
- Entity Representation --- Business analysts usually use simple 2D objects to abstractly represent real objects. For example, process models grammars, such as BPMN, are used to describe the state transition of tasks, ER diagram are used to reflect the relationship between human resources and non-human resources.
- Information Display --- Information may be loosely classified as qualitative and quantitative information. Representations of this information will provide people with insight into the workload of human resources, and utilization rates of non-human resources.
- Business Scenario Rehearsals --- Sometimes, business analysts need to use simple visual approaches, such as sliders, to demonstrate the consequence of enacted business models [17]. Such visual assistance is an essential approach in requirements elicitation and analysis.

3.2 Virtual World Introduction

A virtual world is a network-based, computer synthesized dynamic environment, where participants can communicate with each other and observe computer-generated environmental objects [18]. Some selected features are discussed below:

- Geometry Representation. The geometry in a virtual world is composed of a metadata called geometric meshes. The combination of geometric meshes can form the shape of real objects.
- Programming. People can use programming languages to implement system functions, such as the reaction of an object based upon the current state of the virtual world and database connections and/or document printing.
- Avatars. An avatar is a 3D graphical representation of a virtual world participant with a humanoid appearance, it can be used as a vehicle for virtual world creation, exploration and modification, or for presenting an artificial agent.
- Behavior Modeling. A behavioral model is the mathematical formulae of the movement logic implemented by a programming language. Examples of a behavior model can be a picking up goods from the table of an avatar receiving an order.
- Information Visualization, depending on the analysis task, information can be represented in 3D, or can be represented via a 2D representation on a Heads Up Display (HUD).

3.3 Virtual World As Alternative Communication Approach in Business Process Improvement

We now explore the rationale behind using a virtual world as a communication approach in business process modeling. An overview of how the virtual world can satisfy visualization needs during the communication process is described in Table.1.

Table 1. The table shows the relationship between visualization needs and supported virtual world features.

Visualization Needs	*Supported Virtual World Features*
Physical Environment and Human Resource Behavior	Avatars, Geometry Representation, Behavior Modeling
Entity Representation	Geometry Representation
Information Display	Programming, Geometry Representation, Information Display
Business Scenarios Rehearsals	Avatars, Geometry Representation, Behavior Modeling, Programming

In a virtual world, a complex environment can be built up from basic geometries. Creators can twist, squeeze, and stretch basic geometries into the certain shape to satisfy a particular need. The dress and behavior of people can be a form of self expression and social identity [19,20]. People can see the profession of these avatars. In Fig 4, we illustrate such a creation process, as well as appearance and behavior of avatars.

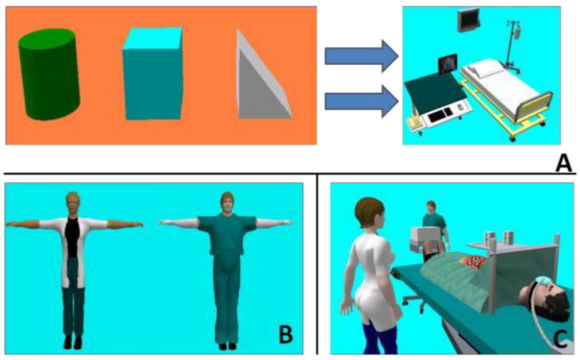


Fig. 4. Illustration of geometry creation process (A), avatar appearance (B), and avatar behavior (C).

Thus, synthetic environments and observable inhabitants of a virtual world, if being correctly translated from reality, enable business analysts and stakeholders to have a concrete observable instance as an object to assess, evaluate, predicate and identify.

4 Case Study

4.1 Visualization Applications

We utilized the YAWL system [21], JADE[1], OpenSim[2], Hippo OpenSim Viewer[3] and OpenMetaverse[4] API to implement our prototype as a proof of concept. The YAWL

1 jade.tilab.com

2 www.opensimulator.org

system is a WfMS that employs a workflow language called YAWL (Yet Another Workflow Language) [22]. JADE (Java Agent DEvelopment Framework) is a JAVA based agent platform, providing developers with an agent system infrastructure platform. These two packages are used to implement our previous agent system [23] that provides underlying agent behaviors, with reference to workflow activity allocation commands. OpenSim, Hippo OpenSim Viewer and OpenMetaverse are 3D application server, 3D application client, and API for behavior modeling of human resource. These three API packages are used to implement our prototype visualization system on top of the agent infrastructure, to produce the images seen in this paper. The architecture of our system is available in Fig.5. We illustrate this architecture with model-view-control (MVC) design pattern.

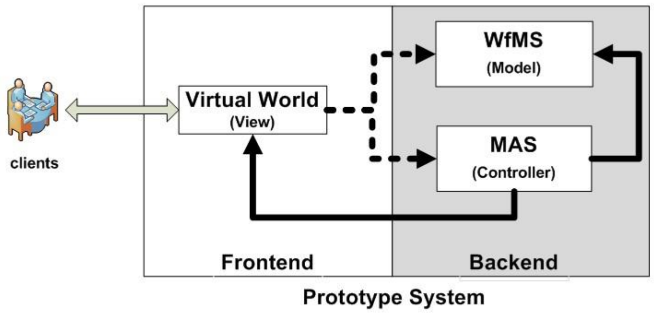


Fig. 5. The architecture of this prototype MVC-based system.

Application in Conceptual Modeling and Modeling Validation.

In a virtual world, people can work together to discuss conceptual models in a direct manner. They can create geometries attached with different textures representing the artifacts used in the reality, and juxtapose these geometries with a modified form of conceptual model. Due to the ability to see the conceptual model in the same space as the person's workplace, business analysts can easily sketch up the components of a model, see Fig.6, and then display them to stakeholders, who can confirm validity of the model.

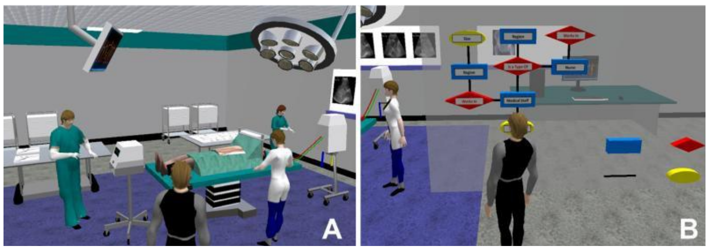


Fig. 6. Conceptual modeling visualization in the virtual world. A business analyst can sketch and observe the behavior of simulated medical staff, see picture A. Based on observations, he is sketching an ER diagram, right picture B.

3 www.mjm-labs.com/viewer

4 www.openmetaverse.org

Application in Business Process Simulation.

In a virtual world, the actual human resource performance and corresponding abstracted information can be simultaneously observed. For example, the concrete and abstract information of task blood transfusion are available to participants, see Fig.9. A local view toward this task such as the responsibility of human resource and non-human resource utilization (the blood bag) can be obtained by native stakeholders. The abstracted information, such as the temporal ordering of task and entity relationship, can be represented through a process model in the HUD (Image D, Fig. 7).

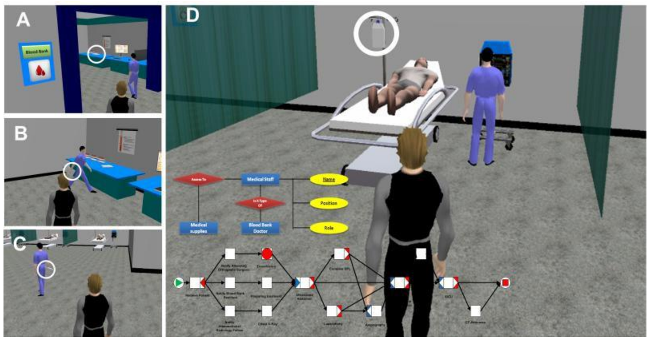


Fig. 7. Illustration of resource behavior observation. The pictures A to D are snapshots from the system. The participant (black vast) can see the actual avatar performance, while executing YAWL model information is displayed in the HUD, (with the red token indicating current workflow state) along with an ER diagram and task description in D.

5 Conclusion

The main purpose of this paper is to provide a visualization approach to strengthen the communication channels between business analysts and stakeholders, before any improvement and optimization activity is conducted. Based on this research, one possible direction forward is the visualization of deviations between two conceptual models that need to be considered. Currently, our system can visualize a workflow system with different simulation configurations by converting the "as-is" result in to 3D visualization of the "to-be". An intuitive indication is still needed for native stakeholders to understand the changes to be introduced by the new model.

Acknowledgement

This research has been financially supported by the Smart Services CRC in Australia - http://www.smartservicescrc.com.au.

Reference

1. Lewis, M., Slack, N.: Operations Management: Critical Perspectives on Business and Management. Routledge (2003)

2. Shnnon, C.E., Weaver, W.: The Mathematical Theory of Communication. University of Illinois Press (1963)
3. Sadiq, S., Indulska, M., Bandara, W., Chong, S.: Major Issues in Business Process Management: A Vendor Perspective. In: the 11th Pacific-Asia Conference on Information Systems:Managing Diversity in Digital Enterprises, pp 40-47 (2007)
4. Moody, D.L.: The "Physics" of Notations: Toward Scientific Basis for Constructing Visual Notations in Software Engineering. J. IEEE Transctions On Software Engineering. 35, 756-779 (2009)
5. Bainbridge, W.S.: The Scientific Research Potential of Virtual Worlds. J. Science, 317, 472-476 (2007)
6. Miller, M.: The magical number seven, plus or minus two. J. Psychological Review 63, 81-97 (1956)
7. Burton-Jones, A., Weber, R.: Understanding relationships with attributes in entity relationshp diagrames. In: 20th international conference on Information Systems, pp 214-228 (1999)
8. Maes, A., Poels, G.: Evaluating quality of conceptual modelling scripts based on user perceptions. Data & Knowledge Engineering 63, pp.701-724 (2007)
9. Weber, R.: Conceptual Modelling and Ontology: Possibilities and Pitfalls. J. Journal of Database Management 14 1-20 (2003)
10. Khatri, V., Vessey, I., Ramesh, V., Clay P., Park S.J.: Understanding Conceptual Schemas: Exploring the Role of Application and IS Domain Knowledge. J. Information Systems Research 17 (1), 81-99 (2006)
11. Eichhorn, D., Koschmider, A., Li, Y., Stürzel, P.: 3D Support for Business Process Simulation. In: Annual IEEE International Computer Software and Applications Conference, pp 73-80, (2009)
12. Betz, S., Eichhorn, D., Hickl, S., Klink, S., Koschmider, A., Li, Y., Oberweis, A., Trunko R 3D Representation of Business Process Models. In: MobIS, pp 73-87, (2008)
13. Perkins, K.S.: Workflow Simulation in a Virtual World. Tranditional Thesis, University of Arkansas, Fayetteville, AR, (2011)
14. Bogdanovych, A.: Virtual Institutions. Tranditional Thesis, UNIVERSITY OF TECHNOLOGY SYDNEY, (2007)
15. Reid, R.D., Sanders, N.R.: Operations Management. Wiley (2009)
16. McCoy, J.M., Evans, G.W.: Chpater 9 Physical Work Environment. In: Handbook of Work Stress, Barling. J., Kelloway, E.K., Frone, M.R. (eds) (2005)
17. Cadle, J., Pual, D., Turner, P.: Business Analysis Techniques 72 Essential Tools for Success. British Informatics Society Ltd, (2010)
18. Burdea, G.C., Coiffet, P.: Virtual Reality. Wiley, (2003)
19. Flugel, J.C.: The psychology of clothes. AMS Press, New York, (1976)
20. Paul, S., Richard, S.: Self-identity and the theory of planned behavior: Assessing the role of identification with "green consumerism". J. Social Psychology Quarterly 55(4), 388-399, (1992)
21. van der Aalst, W.M.P., Aldred, L., Dumas, M., ter Hofstede, A.H.M.: Design and implementation of the YAWL system. In: Proceedings of The 16th International Conference on Advanced Information Systems Engineering (CAiSE 04), (2004)
22. van der Aalst, W.M.P., ter Hofstede, A.H.M.: YAWL: Yet Another Workflow Language. J. Inofrmation Systems 30(4), 245-275, (2005)
23. Guo, H.W., Brown, R., Rasmussen, R.: Human resource behaviour simulation in business processes. In: the International Conference on Information Systems Development, (2011)

On Applying Sonification Methods to Convey Business Process Data

Tobias Hildebrandt[1], Simone Kriglstein[2] and Stefanie Rinderle-Ma[1]

[1] University of Vienna, Austria,
Faculty of Computer Science
{tobias.hildebrandt,stefanie.rinderle-ma}@univie.ac.at

[2] SBA Research, Vienna, Austria
SKriglstein@sba-research.at

Abstract. Visualization of business process models with large numbers of process activities and running instances in different execution states shows various limitations. Hence, using sonification methods becomes a promising idea to enable users to gain insight into process information that cannot be conveyed by using purely visual means. This visionary short paper aims to envision the usage of sonification methods in order to represent business process-related data in all phases of the process life cycle. Sonification methods are presented and analyzed in terms of their potential suitability for representations of process data. Overall, this paper aims at breaking new ground for designing and applying multi-modal approaches for making process information more accessible to users.

Keywords: Business Process Management, Sonification, Process Representation

1 Introduction

During the individual life cycle phases of business processes (design & simulation, operation and analysis), different kinds of process model- and process instance-related data accumulate. For all phases, especially for the design & simulation phase, graphical user interfaces and visualization methods are widespread. However, visualization techniques can reach their limits. As an example, process models in the design and simulation phase can have a huge number of events and activities, which can make it difficult to visually identify deadlocks or weaknesses in process models. In the operation and evaluation phases, processes can have thousands of simultaneously running process instances, which can make it hard to find deviations from regular process execution paths or monitor the health or processes and process instances using only visual means.

The usage of data sonification as an enhancement to process visualizations might be able to tackle some of these challenges. Sonification can be defined as the "presentation of data using sound" [5]. This presentation is usually intended to support the listener or user to gain new insights on the presented data.

Although many reasons appear to apply sonification for representing business process-related data, only very few approaches have addressed this issue so far (e.g., [4]). Gregory Kramer et al. [10] found out, that the auditory perception is especially sensitive to temporal change. Furthermore, sonification, in contrast to a static visualization, can only exist in time. As process instances per definition can only exist in time as well, sonification naturally lends itself to this area. Georg Spehr [16] further concludes, that sonification is more suitable than visualization for complex, irregular or even chaotic data. This promises advances when trying to convey process exceptions and changes to users. Moreover, sonifications can very well be recognized, remembered and recalled later on [5]. Studies, such as the one conducted by Salvador et al. [14], point out that sonification can, under certain conditions, yield better results than visualization in terms of the accuracy and efficiency of data exploration while interfaces that combine both modalities yield significantly better results than each of the modalities alone. Beside the usage of sonification to enhance visual means, it is also applied in situations where the visual focus and attention are needed elsewhere (e.g., in cockpits or operating rooms) or to support blind or visually impaired people.

This visionary short paper provides an analysis of sonification methods with respect to their suitability in the area of business processes. Existing basic sonification methods are discussed and their potential to convey information in consideration of the business process life cycle is analyzed. This is a first step towards a multi-modal approach in order to make information related to business processes more accessible to users.

The paper is structured as follows. Section 2 presents basic sonification methods. Possible solutions regarding how sonification methods can be used to make process-related data more accessible to users are discussed in Section 3. Finally, Section 4 concludes the paper and gives an outlook on future work.

2 Basic Sonification Approaches

Nowadays, there is a growing amount of research in the fields of areas of applications, methods, and perception of sonification. Regarding the application of sonification there is, among others, research in the fields of astronomy, volcano activity, ice glaciers, RNA structures, brain activities or weather data. Other examples are sonifications of software code and sonically enhanced data mining. Furthermore, there are a few examples in the fields of social sciences: sonifications for population developments and election outcomes, sport sciences or economics (e.g., sonification of stock market data [2]). McKinney et al. [12] describe a system that aurally and visually presents peer-to-peer networking traffic. There is several research (e.g., [13,1,6] that investigates methods for interactive sonifications. Other research aims towards multi-modal sonification (systems that combine sonification with graphical user interfaces and visualization techniques) in different application areas (e.g., [9,7]).

After this brief introduction into the field of sonification, in the following the four probably most widely used and researched sonification methods (audification, auditory icons, earcons and parameter mapping) will be introduced.

Audification. An audification is the direct conversion of data points into sound. It interprets data sequences as an audio waveform by mapping the data to sound pressure levels. Therefore, a very high number of data sets is needed in order to produce audible results, which is limiting the field of possible modes of operation. [5]

Auditory Icons. Auditory icons are everyday sounds that directly represent the events that are being sonified [5]. As an example, the sound of a paper basket being emptied can be stated that is played back upon emptying the metaphorical paper basket in the Windows operating system. *Pure* audifications convey only the information that certain events have occurred, not other quantitative data that might be connected to those events.

Earcons. Earcons are non-verbal audio messages consisting of motives, which are short rhythmic sequences of pitched tones with variable timbre, pitch and amplitude. Timbre describes the *basic properties* of sounds and is a subjective characteristic that enables the differentiation of two sounds, even though they might have the same loudness and pitch [5].

The concept of earcons is similar to that of auditory icons with the difference, that auditory icons are everyday sounds that directly represent the event that is being sonified, whereas earcons can be *abstract* symbols that are not similar to the real world sound of the represented event or object.

Parameter Mapping. Parameter mapping is the mapping (either direct or by scaling) of data values to specific attributes of sound. These attributes are typically volume, pitch, panning (the position of a sound in the stereo field) or timbre. Other possibilities to map parameters of sound are repetitions and pauses between distinct sound events in loops. Other approaches are to filter out specific ranges of frequencies according to data. Due to these characteristics parameter mapping is often being said to be the sonic pendant to a scatter plot diagram. [5]

Parameterized Auditory Icons and Earcons. Parameterized auditory icons and earcons are mixtures of parameter mapping and auditory icons/earcons and combine the *simple* event-occurrence method of auditory icons/earcons with parameter mapping. In these, sounds convey the occurrence of events, but at the same time quantitative data can be mapped to these sounds. This mapping is analogous to the parameter mapping way of mapping data to sound attributes. Parameterized earcons usually provide more extensive means to map data to sound attributes in comparison to parameterized auditory icons. [3]

3 Applying Sonification Methods to Business Processes

Even though a substantial amount of sonification research has accumulated, so far there seems to be no research concerning the application of sonification in the different life cycle phases of business processes. One of the few examples of the usage of sonification concerning processes in business environments is Grooving Factory [17]. It explored the sonification of production process-related data in order to find bottlenecks and improve logistics. Evaluation showed, that the developed prototypes fulfilled these requirements. Gaver et al. [4] explore with their "ARKOLA Simulation" the production processes of a bottling plant in a multi-modal representation that combines visual and auditory means. It sonifies events during the production process (such as spills of liquid) using real-world recordings of such events. They concluded that the auditory feedback helped in diagnosing problems in the production process. There are a few publications that explored the sonification of different process data (e.g., [8]). Besides the mentioned projects "Grooving Factory" and the "ARKOLA Simulation" there seem to be however no publications that deal with sonifications of process data in a business environment. This leads to the assumption that there is a substantial amount of untackled research potential in this area.

This section will discuss possible solutions in terms of how sonification can be used to make process-related data more accessible to users and to reduce the potential limitations of process visualizations. Therefore, the respective process life cycles phases are analyzed in terms of which sonification techniques might be suited best to support the tasks users typically have to perform in those phases.

Sonification in Process Design. *Audification* relies on a huge number of quantitative data, which makes this technique seem unsuitable in cases where no or little quantitative data needs to be sonified, but instead merely events. *Auditory icons*, on the other hand, seem very suitable for sonifications in the process design phase: for process instances that are created during the simulation of process models, the sonic pendants of the involved activities and events could be played back upon their incidences. Depending on the industry and the type of processes, there is often a variety of self-explanatory sounds that can be used in order to sonify the respective events and activities. In auditory-based sonifications, the relevant events and activities could be sonified by using sounds that naturally represent these events and activities as accurately as possible. As an example, the sound of a shopkeepers bell could signify the reception of a new order. Analogous, the process event "customer has payed his invoice" could be conveyed by playing the sound of a cash register being opened, while the activity "delivery" could be sonified by applying motor sounds. Fig. 1 shows a schematic overview of how such a sonification of a process instance could be realized. The x-axis is the time axis, whereas each row on the y-axis contains a sound file that represents one activity or event. These sound files are played back sequentially from left to right.

In this example, the sounds that convey events have been assigned a fixed length of 1.5 seconds (as the time axis in the lower part of the figure shows). The

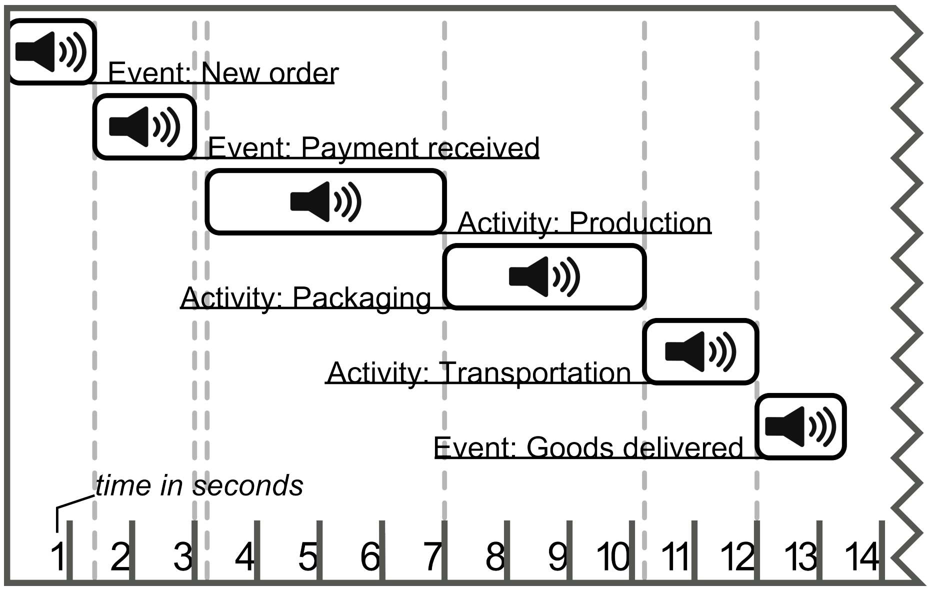


Fig. 1. Schematic overview of an auditory-based sonification of an exemplary process instance

lengths of the audio signals that convey activities, on the other hand, represent the actual duration of the represented activities. Thus, a motor sound with a duration of three seconds could, depending on the scaling, for example signify that the transport took three days. Analogous, a *silence* between activities and events could signify a waiting period. If, for example, there is a gap of two seconds between the playback of the sounds that represent the activities "production" and "packaging", one can conclude that there has been a waiting period of two days between the production of goods and the transport of said. This could be a hint that there are inefficiencies in the process.

Audio files of three different instance sonifications of this example process are available online[1]. The audio file "Example one - average process instance"[2], shows a sonification of an average process instance. The audio file "Example two - no payment"[3] is a sonification of a process instance, in which no incoming payment has been registered. The audio file "Example three - Production and transport delayed"[4] is a sonification of a process instance, in which the activities production and transport have been delayed (which can be recognized by the pauses before the respective audio signals). This simple example tries to show, that auditory icons are able to point out deviations in process instances.

[1] http://soundcloud.com/tobias_hildebrandt/
[2] direct link: http://soundcloud.com/tobias_hildebrandt/business-process-sonification
[3] direct link: http://soundcloud.com/tobias_hildebrandt/business-process
[4] direct link: http://soundcloud.com/tobias_hildebrandt/business-process-1

However, due to its simplicity, this example cannot serve as a comprehensive demonstration of the strengths of sonification (like its ability to convey complex or irregular data). Further prototypes that combine visual and auditory means and base on more complex process models will help in evaluating, if the inherent features of the auditory perception (like the sensitivity to rhythm and its ability to recognize even small changes in sounds over time) can help to convey process instance-related information better than purely visual means.

Earcons are in a similar fashion suitable for process data sonifications but more flexible. For some process events it could prove difficult to find real-world-sonic analogies. For example, it could be a challenge to find sounds that are sonic analogies to the states "customer is already registered" and "new customer". This differentiation would therefore be hard to convey using auditory icons, so the usage of earcons might solve that problem (even though studies suggest that earcons are harder to recognize than auditory icons). By using parameterized auditory icons or earcons, not only the information can be conveyed that a certain event has occurred, but also one or several quantitative data attributes that are connected to that event. For example, one could imagine an auditory icon that conveys the occurrence of an event "incoming payment", while the sum of the payment is mapped to the pitch of that auditory icon.

Parameter mapping might not be the most obvious choice for the process design phase - parameter mapping relies on quantitative data that varies over time, rather than on information on events and their sequences of occurrence, as it is typically the case for process instance-related data.

Sonification in Process Operation and Analysis. Sonifications that aim to assist users during the process operation phase and the process analysis phase probably have to fulfill similar requirements. In both, potential users might want to obtain aggregated information about processes and associated instances and analyze conspicuous phenomena in detail.

Audification does not seem not very suitable for the sonification of data that is related to the process operation and analysis phases, as it seems inflexible in terms of sound design and the structure and format of input data (high amounts of quantitative data that lie within a specific range).

Auditory icons should offer the possibility to recognize deviations of process instances from the process model by the fact that the respective sounds are being played in a different order, or in a different *rhythm* while monitoring and analyzing individual process instances. The same is true for (parameterized) earcons, analogous to what has been said for the process design phase.

Parameter-mapping sonifications might be especially useful during the process operation and the process analysis phases. During these phases, usually quantitative data accumulates that might be mapped to one or several sound streams. These sound streams might then, for example in the process operation phase, be played back continuously which should make it feasible for the user to recognize patterns and modifications as well as to get an overview of the general *health* of individual processes or a complete system. During process operation, an advantage over the usage of purely visual means would be that users would not need to focus their visual attention to specific displays. Thus, they would be able to work on other things while at the same time they could be informed about background activities. Of course, such a sonification would have to be designed in a non-disruptive way.

In the process analysis phase one could imagine a *sonic summary* of a certain time period (for example a shortened sonification of the last 24 hours). In such a sonification it should, after a learning phase, be possible to detect deviations or critical situations during the execution of process instances.

4 Conclusion

Sonification is gaining more and more importance in various disciplines. Due to the existing limitations of visualizations (e.g., keeping track of high numbers of running process instances in different execution states), the authors of this paper propose to introduce it as an enhancement to visualization methods to convey business process information. To achieve this goal it is first necessary to understand how sonification methods can be used to represent business process-related data. The motivation of this paper was to give a first overview of different sonification methods. Possible directions and solutions concerning how sonification can be used to make process-related data more accessible to users were discussed.

Of the presented sonification methods, parameterized earcons and parameterized auditory icons seem to be best suited for sonifications during the process design phase, while the monitoring in the operation phase and the analysis in the evaluation phase seem to be well suited for a parameter mapping sonification.

A multi-modal combination of visualization and sonification should consider the individual strengths and weaknesses of both methods. In general, it should use the abilities of visualization to convey exact information and of sonification to convey changes in temporal developments.

In future work, we will develop and combine sonification methods with visualization methods – particularly for scenarios where visualization techniques come to their limits – that can best be combined into an integrated multi-modal approach for each phase of the business process life cycle. Further, we plan to conduct user studies to evaluate the design of the multi-modal approach and the findings of the evaluations will influence the further design process to improve our approach iteratively.

Acknowledgments

The research was partly funded by COMET K1, FFG - Austrian Research Promotion Agency.

References

1. Beilharz, K., Ferguson, S.: An interface and framework design for interactive aesthetic sonification. In: Jensen, M.A., Kronland-Martinet, R., Ystad, S., Kristoffer

(eds.) Proc. of the 15th International Conference on Auditory Display (ICAD2009). Re:New Digital Arts Forum, Copenhagen, Denmark (2009)

2. Ciardi, C.: sMax: A multimodal toolkit for stock market data sonification. In: Proc. of the 10th International Conference on on Auditory Display. ICAD, International Community for Auditory Display (2004)
3. Gaver, W.W.: Using and creating auditory icons. SFI studies in the sciences of complexity, Addison Wesley Longman (1992)
4. Gaver, W.W., Smith, R.B., O'Shea, T.: Effective sounds in complex systems: the ARKOLA simulation. In: Proc. of the SIGCHI Conference on Human Factors in Computing Systems: Reaching through technology (CHI'91). pp. 85–90. ACM (1991)
5. Hermann, T.: Sonification for Exploratory Data Analysis (PhD Thesis). Universität Bielefeld (2002)
6. Hermann, T.: An introduction to interactive sonification. IEEE MultiMedia 12(2), 2024 (2005)
7. Hermann, T., Hansen, M., Ritter, H.: Combining visual and auditory data exploration for finding structure in high-dimensional data. Technical Report on McMC sonifications (2001)
8. Hermann, T., Niehus, C., Ritter, H.: Interactive visualization and sonification for monitoring complex processes. In: Proc. of the 2003 International Conference on Auditory Display. ICAD, International Community for Auditory Display (2003)
9. Kasakevich, M., Boulanger, P., Bischof, W., Garcia, M.: Augmentation of visualisation using sonification: A case study in computational fluid dynamics. In: Proc. of the IPT-EGVE Symposium. The Eurographics Association (2007)
10. Kramer, G., Walker, B., Bonebright, T., Cook, P., Flowers, J., Miner, N., Neuhoff, J., Bargar, R., Barrass, S., Berger, J., Evreinov, G., Fitch, W., Grohn, M., Handel, S., Kaper, H., Levkowitz, H., Lodha, S., Shinn-Cunningham, B., Simoni, M., Tipei, S.: Sonification report: Status of the field and research agenda - report prepared for the national science foundation by members of the international community for auditory display (1999)
11. Loeb, R.G., Fitch, W.T.: A laboratory evaluation of an auditory display designed to enhance intraoperative monitoring. Anesthesia & Analgesia 94(2), 362–368 (2002)
12. McKinney, C., Renaud, A.: Leech: BitTorrent and music piracy sonification. In: SMC 2011: 8th Sound and Music Computing Conference, 06-09 July 2011, Universita di Padova, Padova, Italy (2011)
13. Pauletto, S., Hunt, A.: A toolkit for interactive sonification. In: Proc. of the 10th International Conference on Auditory Display. International Community for Auditory Display (ICAD) (2004)
14. Salvador, V., Minghim, R., Levkowitz, H.: User evaluations of interactive multimodal data presentation. In: Ninth International Conference on Information Visualisation, 2005. Proceedings. pp. 11– 16. IEEE (2005)
15. Smith, D.R.: Effects of Training and Context on Human Performance in a Point Estimation Sonification Task (Masters Thesis). Georgia Institute of Technology (2003)
16. Spehr, G.: Funktionale Klänge: Mehr als ein Ping. transcript Verlag (2008)
17. Windt, K., Iber, M., Klein, J.: Grooving factory - bottleneck control in production logistics through auditory display. In: Brazil, E. (ed.) Proc. of the 2010 International Conference on Auditory Display (ICAD). International Community for Auditory Display, Washington, D.C., USA (2010)

Toward a Conceptual Comparison Framework between CBSE and SOSE

Anthony Hock-koon and Mourad Oussalah

University of Nantes, LINA
2 rue de la Houssiniere, 44322 NANTES, France
`{anthony.hock-koon,mourad.oussalah}@univ-nantes.fr`

Abstract. In this paper, we discuss the theoretical differences between component-based and service-oriented software engineering (CBSE and SOSE). We present a conceptual comparison framework which confronts their quantitative and qualitative aspects and provides a better understanding of their use. This comparison takes the object orientation (OO) into account to illustrate changing concerns of software engineering between object, component and service.

Key words: SOSE, CBSE, OO, Comparison Framework, Quality Measurement

1 Introduction

Component-based software engineering (CBSE) [1] proposes to reuse existing software entities, called components, to build new applications. Service-oriented software engineering (SOSE) [2, 3] proposes to reuse provided capabilities of existing software entities called services. Both of them have the reusability as theoretical root and rely on the concept of software architecture [4] to describe and manage collaborative software entities. They use numerous similar concepts, approaches, and technologies. Meanwhile, they have continued with their development tracks in parallel and focused on their specific interests. Consequently, there is a mixture of similarities and specialized concepts.

All comparison works between CBSE and SOSE have a bottom-up approach in which they focus on specific technologies to identify the resulted software qualities [5–7] (e.g. comparison of performance between technologies [8]). However, they do not allow a direct comparison at a conceptual level which can offer a global understanding of the differences between the paradigms. To our knowledge, only one other work [9] has a top-down approach which focuses on this conceptual comparison. However, it only tackles some confusions of vocabularies and does not analyze the consequences of these differences on the quality of products and production processes.

In this paper, we propose a conceptual comparison framework which can deduce the resulted software qualities. It is divided into quantitative and qualitative aspects. It takes object orientation (OO) into account to provide a global point of view about the evolution of concerns between object, component and service.

2 Quantitative aspects

The top-level categories of our quantitative part are the product and the process. It does not intend to be exhaustive, but it aims at listing the core concepts of each paradigm to underline their theoretical stance.

2.1 Product and Process

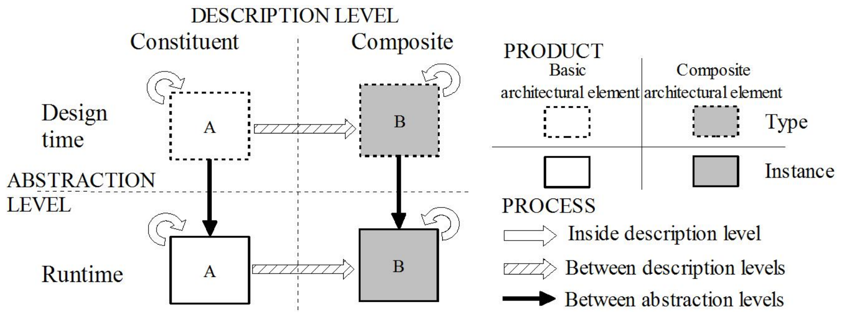


Fig. 1. Abstraction levels and Description levels

A *product* is a software entity or a conceptual entity which is the result of an action or a process. A *process* is an action or a succession of actions which is used to create or modify a product and obtain a specific one.

The product category is divided into two sub categories (Figure 1):

- **Basic architectural element** - basic building blocks of each paradigm;
- **Composite architectural element** - complex products built from existing architectural elements. Their structure clearly identifies the reused architectural elements and their relationships.

Each sub categories is also divided into two groups according to two *abstraction levels*: the *design-time* and the *runtime* (Figure 1).

The process category focuses on the reusability principle shared by OO, CBSE and SOSE, i.e how to reuse existing software entities, the *constituent*, to build new ones, the *composite*. A constituent is a basic or a composite architectural element. These notions of constituent and composite define two *description levels* (Figure 1). The process category is divided into three sub categories according to abstraction and description levels.

- **Inside description level** - gathers processes which target products from the same description level (Figure 1 white arrows) at the two abstraction levels;

– **Between description levels** - gathers processes which target a product from a different description level than the produced one (Figure 1 hatched arrows). This category is divided into design-time and runtime;
– **Between abstraction levels** - gathers processes which target a product from the design-time and then produce a product of the runtime (Figure 1 black arrows).

2.2 Comparison between OO, CBSE and SOSE

We want to emphasize two main differences illustrated by the Table 1. First, SOSE relies on the dynamic service provisioning (discovery and selection) between abstraction levels to produce a concrete service from an abstract service. On the contrary, OO and CBSE rely on the instantiation process between class and object, and component type and component. Then, SOSE relies on additional runtime processes to support the self-adaptation of composite elements. Even if some similar processes are proposed by CBSE, they are not required to specify a component model.

3 Qualitative aspects

Existing works about software quality [10, 11] introduce a huge number of quality criteria based on point of views (developer, user and so forth) or application domains. Our approach is different and proposes a set of core features which are used to express every quality criteria. Users choose a quality criterion they desire to evaluate. Then, they define this quality by combining the measurement of each core feature. This combination is what we call a *qualitative perspective*.

3.1 Core features

We identify six main features:

Loose coupling - measures the dependencies between entities.

Expressiveness - based on the number of concepts and processes provided by the paradigm to specify and manipulate its products.

Abstraction of communication - ability of a paradigm to abstract the communication layer which drives the execution of the application.

Explicit architecture - ability of a paradigm to provide a clear architectural view of the application which follows its principles.

Evolutionary ability - ability of a paradigm to provide a powerful set of concepts and processes to evolve its products.

Ownership - allocation of responsibilities (development, QoS, maintenance, deployment, execution, management, use) among the provider of the reused products and its clients. It expresses the level of liberty granted by the provider to the client.

Table 1. Product-Process: Comparing Object, Component and Service

Product		**OBJECT**	**COMPONENT**	**SERVICE**
Basic architectural elements	**Design-time**	Class	Component type, Connector type	Abstract service
	Runtime	Object	Component, Connector	Concrete Service, Service description
Composite architectural elements	**Design-time**	Composite class	Configuration type, Composite component type	Composition schema type, Composite service type
	Runtime	Composite object	Configuration, Composite component	Composition schema instance, Composite service instance, Composite service description

Process		**OBJECT**	**COMPONENT**	**SERVICE**
Inside description level	**Design-time**	Association Inheritance	Horizontal composition, Interface inheritance Versioning, Refinement	Choreograph, *Inheritance of composition schema type*
	Runtime	Method call	Functionality call,	Choreography, Service provisioning, Service invocation, Service publication, Self-adaptation of composite architectural element
Between description levels	**Design-time**	Composition	Vertical composition	Orchestration,
	Runtime	Method call	Functionality call, Delegation	Orchestration, Service invocation, Service provisioning, *Composition of service descriptions*
Between abstraction levels		Instantiation	Instantiation	Service provisioning, Instantiation of composition schema

3.2 Comparing Object, Component and Service

Figure 2 shows the classification of OO, CBSE and SOSE according to our six features and three levels of importance (low, medium and high). It illustrates an instance of our qualitative comparison. These levels are not a precise measurement but they are used to define a hierarchy between paradigms based on our analyze.

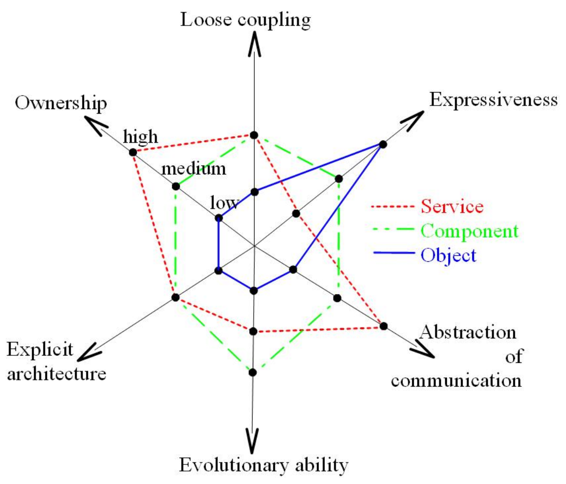


Fig. 2. Comparison of the features

Loose coupling - typically, an Object-based system is built from a set of classes which are tightly coupled while a Component-based or a Service-based system intends to be more loosely coupled. In fact, related topics such as the self-adaptation or the management of heterogeneities are deeply studied by CBSE and SOSE. We set that existing researches reach the same level of maturity. However, our previous work [12] on the definition of the loose coupling notion shows that numerous challenges are still unsolved.

Expressiveness - OO manipulates a huge number of concepts such as granularity, reflexion, template, inheritance, abstraction level, description level and so forth. CBSE's expressiveness mainly relies on OO's researches. However, some advanced concepts such as reflexion or inheritance and polymorphism do not reach the same level of maturity. SOSE has the weaker expressiveness of the three. In fact, it shares the lack of CBSE and adds some others (e.g. the abstraction levels and the distinction between type and instance are still unclear).

Abstraction of communication - SOSE provides the best abstraction of communication. In fact, the global collaboration pattern between constituent services is located inside a single entity: the composition schema which expresses the overall behavior in terms of workflows and dataflows. In CBSE, the communications are located inside the connectors which split the global behavior.

The workflow is not explicit. Therefore, the overall collaboration is harder to understand and manipulate. In OO, the fine granularity of the class and the association link accentuates this drawback of CBSE.

Explicit architecture - typically, an Object-based system lacks an explicit architecture which is easily understandable. CBSE was first introduced to enhance this aspect and develop the concept of software architecture. SOSE directly uses this experience and its difference with CBSE is not significant.

Evolutionary ability - dependent on the architectural graph and the evolution processes which target its nodes, edges, or the overall graph. Typically, OO does not provide an explicit architecture and thus, its community only focuses on the evolution of the nodes and edges. Both CBSE and SOSE handle an explicit architecture. Their communities also study the evolution of the overall graph. However, some works of the CBSE community such as [13, 14] go further and study the evolution process at the meta and meta-meta-architecture levels.

Ownership - SOSE has taken the concept of ownership to the extreme and thus, the provider of services is responsible for the development, the quality of service, the maintenance, the deployment, the execution, the management. On the contrary, CBSE splits the responsibilities at the deployment level. In fact, the client is responsible for the instantiation of the component inside his application and its execution and management. Typically, the object orientation defines the class as a glass box entity and provides some powerful processes to easily manipulate it.

3.3 Qualitative perspectives

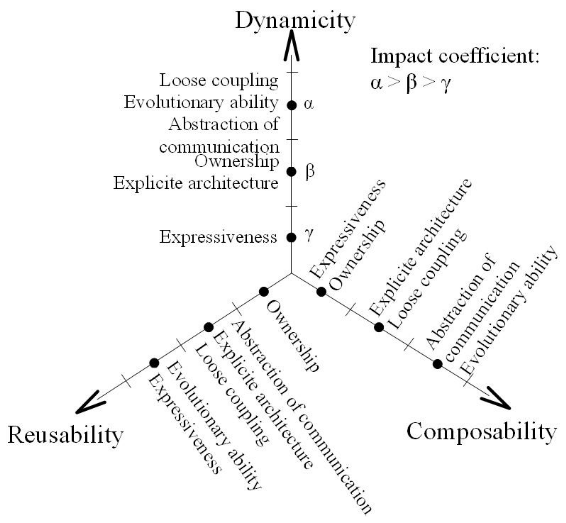


Fig. 3. Quality definition

Our six features represent the main elements of a software development paradigm which impact on the software quality. However, the importance of their impact can vary according to the chosen quality and the user's point of view about this quality. In the figure 3, we act as users and define three quality criteria: reusability, composability and dynamicity. We express our understanding about these qualities and divide the features into three groups, from the γ group which has the weaker impact to the α group which has the stronger impact.

Then, we define a set of formula (e.g. for the reusability(1)) which combine our vision of these qualities dropped to our six features and the previous classification of the three paradigms following these features (Figure 2). Each level (low, medium and high, Figure 3) is associated with a weight (respectively 1, 2 and 3).

$$\begin{aligned} \textbf{Reusability} = \mathit{Object} &: \alpha(4) + \beta(3) + \gamma(1) \\ \mathit{Component} &: \alpha(5) + \beta(6) + \gamma(2) \\ \mathit{Service} &: \alpha(3) + \beta(7) + \gamma(3) \end{aligned} \tag{1}$$

From our qualitative perspective, CBSE has a better reusability than SOSE or OO. The same work can be done for other quality criteria (flexibility, robustness and so forth). The definition of the impacts of each feature on each quality depends on the user's expertise. In fact, each quality represents a particular perspective on our six features and this perspective has to be defined by the user. The following formula $Quality = f(\alpha, \beta, \gamma, \delta, \epsilon, \zeta)$ emphasizes the user role which has to provide:

– the different coefficients (α to ζ) which define the respective importance of each feature according to the chosen quality perspective;
– the function f which defines the way to combine these features.

4 Conclusion

This paper presents a conceptual comparison between CBSE and SOSE divided into quantitative aspects and qualitative aspects. The quantitative aspects classify products and processes which are used to develop some new applications. The qualitative aspects compare the three paradigms following six features which are reused to specified any software quality such as reusability, composability and dynamicity. We show how a user can exploit these features and combine them to evaluate some quality criteria according to their own expertise.

For now, measures of each feature presented in Figure 3 only confront the three paradigms at a conceptual level to provide a hierarchy between them. This hierarchy is based on our own expertise. This approach is sufficient for a direct comparison between theories at a paradigm level, however it is not precise enough to descend to implementation and technological level. In [12], we propose a new definition of the loose coupling which comes along with an objective evaluation formula. Therefore, the same work needs to be done for the five other features. A better measurement of each feature will ensure a better evaluation of the qualitative perspectives defined by users.

References

1. Szyperski, C.: Component Software: Beyond Object-Oriented Programming. Addison-Wesley Professional (2002) isbn 0201745720.
2. Stojanovic, Z., Dahanayake, A.: Service-oriented Software System Engineering Challenges And Practices. IGI Publishing, Hershey, PA, USA (2005) isbn 1591404274.
3. OASIS: Reference architecture for service oriented architecture 1.0. (April 2008) http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/soa-ra-pr-01.pdf.
4. Medvidovic, N., Taylor, R.N.: A classification and comparison framework for software architecture description languages. IEEE Trans. Software Eng. **26**(1) (2000) 70–93
5. Masek, K., Hnetynka, P., Bures, T.: Bridging the component-based and service-oriented worlds. In: Euromicro Conference on Software Engineering and Advanced Applications, SEAA. (2009) 47–54
6. Stojanovic, Z.: A Method for Component-based and Service-Oriented Software Systems Engineering. PhD thesis (2005) Delft University of Technology, isbn 90-9019100-3.
7. Vassilopoulos, D., Pilioura, T., Tsalgatidou, A.: Distributed technologies corba, enterprise javabeans, web services &#8212; a comparative presentation. In: Proceedings of the 14th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing. (2006) 280–284
8. Kim, S., Han, S.Y.: Performance comparison of dcom, corba and web service. In: PDPTA. (2006) 106–112
9. Breivold, H.P., Larsson, M.: Component-based and service-oriented software engineering: Key concepts and principles. In: Euromicro Conference on Software Engineering and Advanced Applications, SEAA. (2007) 13–20
10. Kithchenham, B., Lawrence, S.: Software quality: The elusive target. IEEE Software **13** (1996) 12–21
11. Bianco, P., Kotermanski, R., Merson, P.: Evaluating a service-oriented architecture. Technical Report Software Engineering Institute Carnegie Mellon (2007) http://www.sei.cmu.edu/reports/07tr015.pdf.
12. Hock-koon, A., Oussalah, M.: Defining metrics for loose coupling evaluation in service composition. In: International Conference on Services Computing, IEEE SCC. (2010) 362–369
13. Goaer, O.L., Tamzalit, D., Oussalah, M., Seriai, A.: Evolution shelf: Reusing evolution expertise within component-based software architectures. In: International Computer Software and Applications Conference, COMPSAC. (2008) 311–318
14. Garlan, D., Barnes, J.M., Schmerl, B.R., Celiku, O.: Evolution styles: Foundations and tool support for software architecture evolution. In: Working IEEE/IFIP Conference on Software Architecture, WICSA/ECSA. (2009) 131–140

A UML-based Rich Service Description Language for Automatic Service Discovery of Heterogeneous Service Partners

Zille Huma[1], Christian Gerth[1,2], Gregor Engels[1], and Oliver Juwig[3]

[1] Department of Computer Science, University of Paderborn, Germany[**]
{zille.huma,gerth,engels}@upb.de
[2] s-lab - Software Quality Lab
{cgerth}@s-lab.upb.de
[3] HRS-Hotel Reservation Service, Germany
{Oliver.Juwig}@hrs.de

Abstract. Service-oriented computing (SOC) emerges as a promising trend solving many issues in distributed software development. Following the essence of SOC, service descriptions are defined by the service partners based on current standards, e.g., WSDL [15]. However, these standards are mostly structural and do not provide any behavioral description, which may lead to inaccurate service discovery results. There is a requirement for a *rich service description language* for service partners that encompasses the structural as well as behavioral information in the service description. Furthermore, service discovery based on an automatic matching of these comprehensive service descriptions is a complex task, which is further complicated through the heterogeneity of the service partners' domains in terms of different underlying ontologies. In this paper, we propose a rich service description language based on UML, which allows the specification of structural and behavioral features of a service. In addition, we also briefly discuss how some existing matching approaches can be extended to define an automatic matching mechanism for rich service descriptions resolving the underlying heterogeneity.

1 Introduction

Service-oriented computing (SOC) realizes the idea of reusability through independently developing, automatically discovering and consuming distributed software components (services) on the basis of their interface descriptions.

The automatic discovery of services at design-time as well as at run-time faces certain challenges. The first challenge is that of insufficient information in service descriptions. Standards for *structural* description, such as the Web Service Description Language (WSDL) [15], are unable to comprehensively describe the service *offer* or *request* of service partners. Semantic web service (SWS) approaches [8, 12, 2] come up with notations for rich description of the service

[**] This work was partially supported by the German Research Foundation (DFG) within the Collaborative Research Centre "On-The-Fly Computing" (SFB 901)

offer/request. However, these approaches face certain limitations. For instance, they are not comprehensive enough to cover multiple aspects of service description, such as operation semantics or service protocols, etc. Similarly, [12, 2] are not yet widely accepted in practice because of their diversion from the existing standards and the extra effort required to learn and use their complex notations.

The second challenge is the heterogeneity of service descriptions. The distributed paradigm allows service partners to function in their respective domains and describe their service *offer* and *request* conforming to their own enterprise model (ontology). As a consequence, service descriptions may be semantically similar but are specified differently conforming to their individual ontologies. For automated service discovery such similarities must be identified. Moreover, due to the underlying heterogeneity, the correspondences between the provided and the required operations in the service descriptions may not be limited to 1:1 but there may be more complex types of correspondences, e.g. 1:n, n:1, or n:m, which additionally complicates the matching of service descriptions.

As a solution that addresses these challenges, we propose a UML-based rich service description language in this paper. We give an insight into the features of the proposed language and briefly discuss how the existing matching approaches can be extended to formulate a matching approach that includes heterogeneity resolution features as well. The remainder of this paper is structured as follows: In Section 2, we introduce a real-world scenario, discuss its limitations and outline detailed requirements for a potential solution. Section 3 describes the proposed rich service description language in detail. Section 4 discusses the existing matching approaches with their shortcomings and outlines a potential soultion. In Section 5, we discuss related work. Finally, in Section 6, we discuss future directions of our work.

2 Example Scenario and Detailed Requirements

Figure 1 shows a typical scenario of SOC, where service requestors define their *service request* (SR) and service providers describe their *service offer* (SO) based on their independent local ontologies. These SRs/SOs need to be matched for the purpose of service discovery. As a simplified example scenario, we consider a case study from the e-tourism domain, where tourists can book flights and accommodations online. Figure 2 shows a typical reservation scenario provided by the worldwide accommodation reservation company, Hotel Reservation Service (HRS)[4]. Users connect with HRS through a variety of interfaces like web browser or smart phone. On the other end, HRS as a *service requestor*

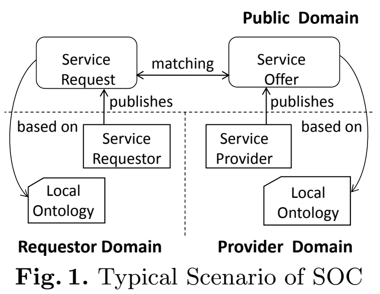


Fig. 1. Typical Scenario of SOC

[4] http://www.hrs.com

connects with services of their partner hotels that act as *service providers*, to carry out the booking of accommodations.

Whenever HRS wants to extend the business by connecting to a new hotel or hotel chain, HRS's web application has to connect to the provided services of new partner hotels through manual matching of SR of HRS and SOs of the hotel/hotel chains.

Currently, the SR of HRS is based on structural features, i.e., an operation signature description conforming to the tourism ontology by the Open Travel Alliance (OTA) [5]. Figure 3 (a) shows an excerpt of this local ontology. An example SR may contain the specification of the following operation signatures: *checkAvailability()*, *getDetails()*, *makeReservation()*, whose input and output parameters are typed over the concepts contained in this local ontology.

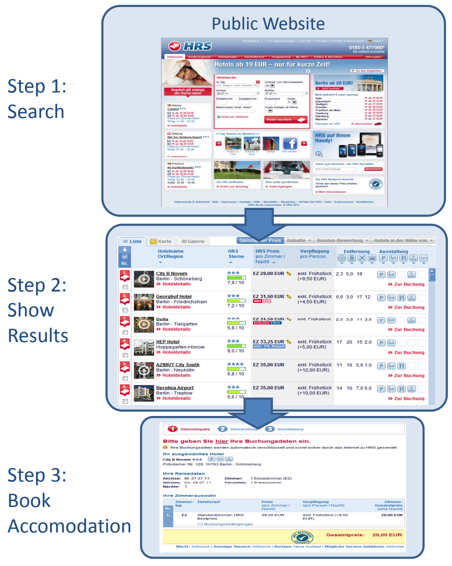


Fig. 2. Reservation Scenario at HRS

To fulfill the requirements of the SR, we assume that there exist two potential service providers *HotelX* and *HotelY*, whose service offers (SOs) are based on the HarmoNET[6] tourism ontology, which covers major sub-domains in e-tourism similar to the ontology by the OTA. Figure 3 (b) shows the HarmoNET-based local ontology for *HotelX* and *HotelY* extended with booking-related concepts. The structural SO of *HotelX* comprises the following operation signatures: *getAvailableRoom()*, *makeABooking()*, and *generateReceipt()*. Similarly, structural SO of *HotelY* has the following operation signatures: *searchRoom()*, *getRoomDetails()*, *bookRoom()*, and *getReceipt()*.

These SOs of *HotelX* and *HotelY* specifying the structural features are manually matched to the structural SR of HRS for service discovery. The service matching is further supported through personal communication between the service partners based on natural language and UML-based diagrams. Such a manual matching based on structural service descriptions makes the service discovery process time-consuming and expensive in terms of time and resources. Additionally, it is also error-prone due to the fact that the SR and SOs are matched on the basis of structural features only because these service description do not contain the behavioral description, such as, operation semantics and

[5] http://www.opentravel.org
[6] http://www.harmonet.org

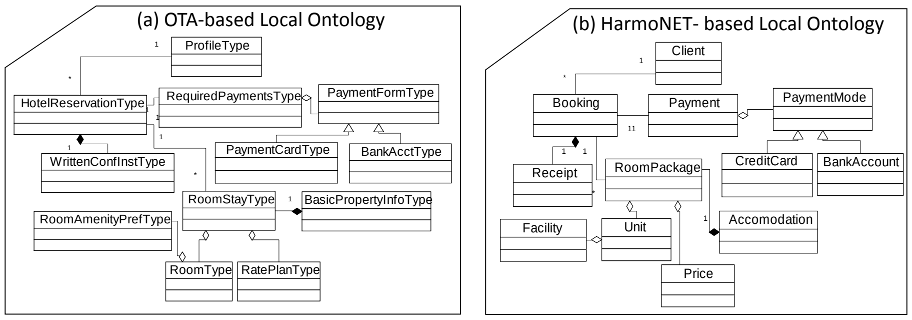


Fig. 3. (a) OTA-based local ontology for HRS and (b) HarmoNET-based local Ontology of HotelX and HotelY

the service protocols of the requested and provided services. A solution to overcome these limitations has to fulfill the following requirements:

R1: A *rich service description language* is required for a comprehensive description of required and provided services containing behavioral information, such as, operation semantics and service protocols in addition to the structural information, i.e., operation signatures. The proposed language should be applicable in practical scenarios as the one described above.

R2: A matching mechanism with heterogeneity resolution features is required for the *rich service descriptions* to enable automatic and accurate service discovery.

3 Rich Service Description Language

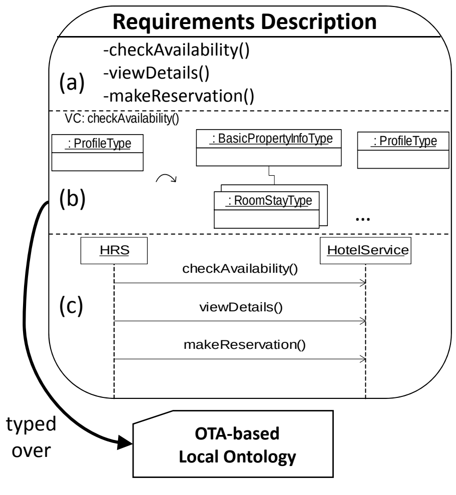


Fig. 4. UML-based Rich Service Description of HRS

In general, a variety of notations and languages for *rich service descriptions* already exists, such as, WSDL-S [8], Web Ontology Language for services (OWL-S) [12], and WSML[2] by Web Services Modeling Architecture (WSMX) [5], etc. However, these languages have certain limitations. For example, WSDL-S [8], which is an extension to the WSDL standard provides notations for operation syntax and semantics only and do not come up with notations for service protocols. On the other hand, languages like WSML [2] and OWL-S [12] provide concrete notations for operation syntax as well as service protocol. In case of operation semantics, these languages do not come up with a concrete notation to specify the pre- and post-conditions of operations and leave the choice of such a rule language to the user. They are still limited to academia because these task-specific languages diverge from existing service description standards making their acceptability difficult in practice. For example, OWL-S leads to long and complex textual descriptions [14] and therefore is reported to require extra effort to learn and use.

We propose a *rich service description language* based on UML [11], which is already a de-facto industrial standard in the area of software engineering, e.g., our industrial partners HRS is already relying on UML notations and diagrams to model and describe certain service features, such as, the required workflow. Reuse of the existing UML artifacts and knowledge makes it easier for the SE community in general and service partners in our case study in particular to adapt to our proposed *rich service description* language.

We propose the following artifacts for rich service descriptions:

(a) A description of operation signatures.
(b) UML-based visual contracts (VC) [6, 4] for semantic description of individual operations.
(c) UML sequence diagrams and UML statechart diagrams for requester's and provider's service protocols respectively.

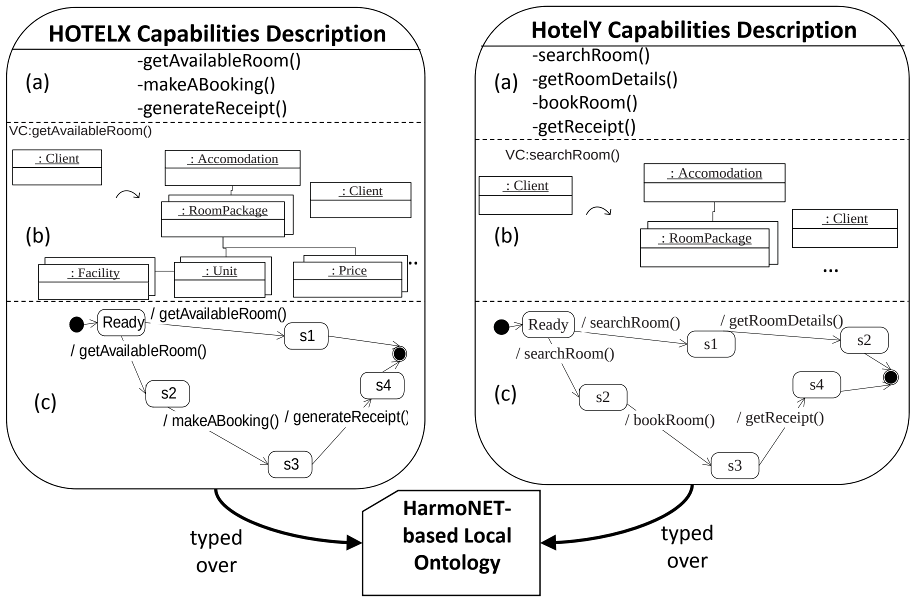


Fig. 5. UML-based Rich Service Description for HotelX and HotelY Services

The SR of HRS and SOs of *HotelX* and *HotelY* based on the proposed language are shown in Figure 4 and Figure 5.

The Web Service Description Language (WSDL) [15] as an existing standard can be used for the syntactic description of operation signatures. A VC specifies the behavior of each operation in terms of pre- and post-conditions based on UML object diagrams as shown in Figure 5. For instance, *searchRoom()* of *HotelY* requires that a *Client* object exists *before* the operation invocation and *Accomodation*, *RoomPackage* and *Unit* objects and their associations are created *after* the operation invocation.

With respect to service protocols, a requestor is interested in specifying a *desired* operation sequence invoked on a provided service. For this purpose, we use UML sequence diagrams to specify the service protocol of a requestor. However, in the case of the service provider, it is important to specify all *allowed* operation invocation sequences for a provided service. Therefore, UML statechart diagrams are the selected notation for provider's service protocol.

In the next section, we discuss the existing approaches to match service descriptions and discuss how they need to be extended to have a comprehensive matching mechanism for rich service descriptions.

4 Towards the Matching of Rich Service Descriptions

To enable automatic service discovery, an automatic matching mechanism for rich service descriptions is required.

In the context of our proposed language, there are already some existing VC matching approaches [6, 10]. [6] proposes a mechanism for matching an operation in the provider's SO to an operation in the requestor's SR. However, this matching mechanism has certain shortcomings. Firstly, it does not deal with the underlying heterogeneity and assumes that the service partners share a common underlying ontology. Such an assumption makes the application of this matching mechanism in practical scenarios difficult, e.g., in our example scenario, service parnters conform to different ontologies from the tourism domain, i.e., OTA and HarmoNET ontologies. A potential matching mechanism has to overcome this ontological heterogeneity to match the service descriptions. Secondly, the approach is limited to 1:1 mappings between operations in the service descriptions and does not consider complex operation mappings, such as, 1:n, n:1, and n:m.

A step further in this direction is the VC matching mechanism proposed in [10]. According to this approach, a 1:n VC matching is often required in a realistic scenario, where multiple operations in provider's SO have to be invoked to fulfill the requirements specified in a single operation in requestor's SR. However, it also ignores the possiblity of underlying heterogeneity of the service partner domains. Additionally, while matching, it does not consider service protocols, i.e., the intended or allowed invocation sequence of required or provided operations, respectively.

Considering these shortcomings of the existing approaches for VC matching, an elaborate and comprehensive matching mechanism is required, which enables an accurate service discovery by considering all aspects of rich service descriptions on one hand and by overcoming the underlying heterogeneity of service partners on the other hand. In future, we aim to come up with such an automatic matching mechanism for service descriptions based on our proposed language.

5 Related Work

One area of related work is concerned with rich service interface descriptions. Apart from the syntactic standard WSDL [15], there are research works, such as, Visual Contracts [6], WSDL-S [8], Web Ontology Language for services (OWL-S) [12], and WSML [2] by Web Services Modeling Architecture (WSMX) [5] for semantic description in terms of their operations' pre- and post-conditions. Most of these approaches use specialized languages that are difficult to learn and use and limited in expressiveness and hence not widely used in practice yet. On the contrary, [6] is a UML-based approach [11], which is already a de-facto standard in software engineering domain.

Another important area of related work is concerned with service description matching. For this purpose, matching mechanisms [10, 7, 1] have been proposed for the service descriptions based on the languages discussed earlier. For instance, [10] proposes a matching mechanism for service descriptions based on VC-based service descriptions leaving some important issues unsolved, such as, dealing with the underlying heterogeneity, performing n:1 operation matching between service partners, and service protocol matching. Similarly, service matching approaches like [7, 1] for service descriptions based on WSML and OWL-S, respectively, do not consider service protocols while service description matching. Other approaches [9, 13, 3] also propose mechanisms for service protocol matching. However, these approaches are either not comprehensive enough in terms of interface description or ignore the underlying heterogeneity.

WSMX [5] propose a comprehensive mediator-based approach to match user goals and service capabilities for accurate service discovery while considering heterogeneity. Even though we share the same aims, our approach differs from the WSMX on the fundamental issue of using a de facto standard like UML [11].

6 Conclusion and Future Work

In this paper, we have identified the shortcomings of the existing service description and discovery scenario in SOC on the basis of a realistic case study of our industrial partner HRS. To overcome these shortcomings, we proposed a UML-based *rich service description language* that encompasses structural features, i.e., operation signatures as well as behavioral features, i.e., operation semantics and service protocols of requested/provided service. Such rich service descriptions comprising structural and behavioral features are a prerequisite for accurate and automatic service discovery. We briefly discussed how existing VC-based service matching approaches can be extended to define an automatic service discovery mechanism for rich service descriptions. In future, we intend to define an automatic matching mechanism for the rich service descriptions overcoming the shortcomings of the existing service matching approaches.

References

1. A. Mukhija, A.D.S., Rosenblum, D.S.: Dino: Dynamic and Adaptive Composition of Autonomous Services. White paper, Department of Computer Science, University College London, London (2007)
2. de Bruijn, J., Lausen, H., Polleres, A., Fensel, D.: The Web Service Modeling Language WSML: An Overview. In: Sure, Y., Domingue, J. (eds.) ESWC. LNCS, vol. 4011, pp. 590–604. Springer (2006)
3. Cavallaro, L., Nitto, E., Pradella, M.: An Automatic Approach to Enable Replacement of Conversational Services. In: Baresi, L., Chi, C.H., Suzuki, J. (eds.) ICSOC-ServiceWave '09. LNCS, vol. 5900, pp. 159–174. Springer (2009)
4. Engels, G., Güldali, B., Soltenborn, C., Wehrheim, H.: Assuring consistency of business process models and web services using visual contracts. In: AGTIVE. pp. 17–31 (2007)
5. Haller, A., Cimpian, E., Mocan, A., Oren, E., Bussler, C.: WSMX - A Semantic Service-Oriented Architecture. In: ICWS '05:. pp. 321–328. IEEE Computer Society (2005)
6. Hausmann, J.H., Heckel, R., Lohmann, M.: Model-based Development of Web Service Descriptions Enabling a Precise Matching Concept. International Journal of Web Services Research 2(2), 67–85 (2005)
7. Klusch, M., Kaufer, F.: WSMO-MX: A hybrid Semantic Web service matchmaker. Web Intelli. and Agent Sys. 7, 23–42 (2009)
8. LSDIS Lab: Web Service Semantics. http://lsdis.cs.uga.edu/projects/WSDL-S/wsdl-s.pdf
9. Motahari, N., Reza, H., Benatallah, B., Martens, A., Curbera, F., Casati, F.: Semi-automated Adaptation of Service Interactions. In: Williamson, C.L., Zurko, M.E., Patel-Schneider, P.F., Shenoy, P.J. (eds.) WWW '07. pp. 993–1002. ACM Press (2007)
10. Naeem, M., Heckel, R., Orejas, F., Hermann, F.: Incremental Service Composition based on Partial Matching of Visual Contracts. In: Rosenblum, D.S., Taentzer, G. (eds.) FASE 2010. LNCS, vol. 6013, pp. 123–138. Springer (2010)
11. Object Management Group (OMG): Unified Modeling Language (UML) – Superstructure, Version 2.3. http://www.omg.org/spec/UML/2.3/Infrastructure (2009)
12. OWL-S Coalition: OWL-based Web Service Ontology. http://www.ai.sri.com/daml/services/owl-s/1.2/ (2006)
13. Poizat, R.M.P., Salaün, G.: Adaptation of Service Protocols Using Process Algebra and On-the-Fly Reduction Techniques. In: Bouguettaya, A., Krüger, I., Margaria, T. (eds.) ICSOC'08. LNCS, vol. 5364, pp. 84–99. Springer (2008)
14. Sabou, M., Richards, D., van Splunter, S.: An experience report on using daml-s (2003)
15. W3C: Web Service Description Language(WSDL). http://www.w3.org/TR/wsdl20/ (2007)

AMES: Towards an Agile Method for ERP Selection

Gustaf Juell-Skielse[1], Anders G. Nilsson[1], Andreas Nordqvist[2] and Mattias Westergren[3]

[1] Stockholm University
{gjs, agn}@dsv.su.se
[2] IBM Global Business Services
andreas.nordqvist@se.ibm.com
[3] Accenture
mattias.westergren@accenture.com

Abstract. Conventional on-premise installations of ERP are now rapidly being replaced by ERP as service. Although ERP becomes more accessible and no longer requires local infrastructure, current selection methods do not take full advantage of the provided agility. In this paper we present AMES (Agile Method for ERP Selection), a novel method for ERP selection which better utilizes the strengths of service oriented ERP. AMES is designed to shorten lead time for selection, support identification of essential system requirements, increase learning during the selection process and increase control over the subsequent ERP implementation. These properties of the method will help user organizations to make better and faster decisions when selecting ERP.

Keywords: ERP, ERP-as-a-service, software selection, agile methods, service orientation.

1 Introduction

In light of service orientation of packaged software, long-standing sequential practices for selection will now give room to more agile methods. Although software development has radically changed due to the emergence of agile development methods such as Agile Model Driven Development [1] and Scrum [2], prevailing methods for selecting and implementing packaged software, such as ERP, have so far remained untouched [3]. But recently, service orientation has emerged as an important change driver of how packaged software is built and delivered. Through ERP-as-a-service, suppliers have the opportunity to decrease up-front investments and reduce implementation costs and risks for ERP [4]. The conventional ERP business model is product centric and revolves around the ERP system which is implemented on the premises of the using organization [5], while ERP-as-a-service follows a service dominant logic [6] where users consume services bundled as offerings delivered over the Internet by a supply chain of service providers. Even more interestingly, from the perspective of selection, user organizations can instantly get access to and try-out a number of ERP services without having to pass through a complex installation and a first round of supplier negotiations. However, the sequential methods for selecting an ERP remains and available methods do not take into account the opportunities of service orientation. Therefore, there is a need for new ERP selection methods that utilizes the flexibility of service oriented business models for ERP. For example, it takes only a minute to get access to a full version of 24Sevenoffice[1], an ERP-service for small and medium sized organizations. While the complexity of installing a conventional ERP package just for demo purposes justified user organizations to apply a sequential mind-set, the swiftness of modern service based ERP encourages a flexible and more agile mindset. The reason for the lack of such a method ought to be that the prime focus of suppliers is the implementation phase. There is no interest among suppliers, apart from users selecting them, for developing selection methods.

In this short paper our goal is to sketch out an agile method for ERP selection. We will use design science to design and evaluate our method. The method is evaluated using action research at a small entrepreneurial firm. The benefits of such method would be to shorten lead time for selection, increase knowledge building about requirements and system capabilities during the selection phase, decrease supplier dependency during selection and to initiate data migration during selection. The prime beneficiaries of such a method are user organizations, and organizations representing users, such as Business Application Software Developers Association[2].

The article is organized as follows. In the next chapter we will present methodological and empirical basis for AMES. In chapter three we will describe the method and use running examples to illustrate it. In chapter four, we assess AMES using informed arguments and finally, in chapter five, conclusions are made together with suggestions for future research.

2 Methodological and Empirical Basis for AMES

This section describes the research strategy used, the objectives of AMES, and present the user organization where the method was designed and applied. The action research conducted at the user organization is presented as a running example in Chapter 3.

2.1 Design Science

For developing the method, we have used design science [7, 8] as a research strategy, in particular Peffers et al.'s model for design research [8], which consists of six steps:

1. Identify problem and motivate
2. Define objectives of a solution
3. Design and develop
4. Demonstration
5. Evaluation
6. Communication

[1] http://www.24sevenoffice.com
[2] http://www.basda.org

The problem and its motivation have been discussed in Chapter 1. In the following sections, the remaining steps are addressed.

2.2 Objectives of the Method

The overall objective is to solve the problem, sketched out above, by developing an agile method for selecting service based ERP. User organizations will more quickly identify ERP software with greater organizational fit [13] by applying agile principles [15] which promote customer focus, face-to-face communication and changing requirements as well as cooperation between business and IT personnel[3]. This overall objective is broken down into the following specific objectives for the method:

- Increase requirements quality during the selection phase
- Increase detailed knowledge about system capabilities during the selection phase
- Decrease lead time for selection
- Decrease supplier dependency during selection
- Increase control over subsequent implementation

The objectives of the method are evaluated in Chapter 4.

2.3 User Organization

AMES has been designed during an ERP selection process at a small entrepreneurial firm with 10 employees called Activio [9]. Activio offers and manufactures solutions for physical training and for managing training results digitally. The company is currently in an expansion phase where the number of customers and retailers are increasing rapidly. In order to meet the rate of expansion, the company is looking at possibilities to streamline the order and inventory replenishment processes.

3 Description of AMES

This section describes AMES, which consists of three phases: *Envision, Iterate* and *Decide*, see Figure 1. *Envision* consists of two activities, *Iterate* of three activities and *Decide* of two activities. The boxes behind *Iterate* depict multiple iterations.

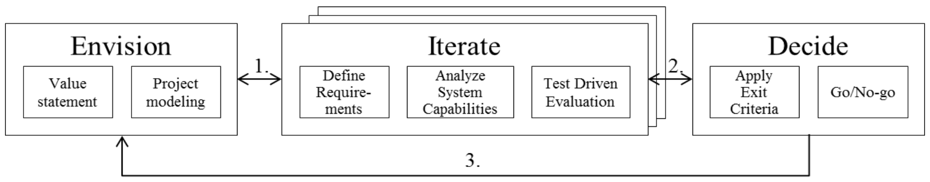


Figure 1. The phases and activities of the AMES method.

[3] Correspond to principles 1, 2, 4, 6, 7 and 10 of the Agile Manifesto [15].

The relationships between the phases are marked 1-3 in Figure 1. The first relationship (1.) between *Envision* and *Iterate* shows that user organizations move from *Envision* to *Iterate* but it is also possible that user organizations return to *Envision* after one or more iterations if needed, e.g. when they have increased their knowledge about system capabilities and want to include more system candidates. The second relationship (2.) between *Iterate* and *Decide* show that user organizations move to a decision when there is no need for further iterations. However, depending on the outcome of the decisions, user organizations may want to return to *Iterate* to evaluate additional requirements or return to *Envision* (3.) to re-start the project.

3.1 Overall Design of the Method

AMES was constructed using Agile Model Driven Development [1] as a starting point. It has inherited much of the characteristics of Agile Model Driven Development: the phases *Envision* and *Iterate*, requirements evolve during project, stakeholder participation, test driven evaluation, short lead times. AMES emphasizes prototyping and iteration to avoid that the selection process stagnates and that time is wasted. Packaged software is ideal for prototyping of user requirements and trying out system capabilities but is seldom used in ERP selection since conventional ERP requires on-premise software installation. There are examples of demonstration, testing and prototyping in previous selection methods [10]. However, in AMES we use iterations actively in order to derive essential requirements. Essential requirements are requirements that need to be met in order for the ERP system to be acceptable to the user organization [11]. Less important types of requirements are often labeled as conditional or optional. Iterations are also used to successively define and clarify requirements. At the beginning of the process, stakeholders define goals with limited knowledge about the benefits of using the system. Therefore, when faced with problems to define certain system requirements, you choose to go on to the next step in the hope of making a better definition in the next iteration. Iterations end when no more essential requirements are identified.

3.2 Envision

The phase *Envision* aims at establishing a preliminary understanding of the goals and scope of using the system. The phase includes two activities: *Initial Value Statement* and *Initial Project Modeling*. The activity *Initial Value Statement* aims at describing the benefits and the value created by the project. Typically ERP is adopted for strategic, operational or technical reasons [12]. The purpose of the activity *Initial Project Modeling* is to define initial requirements on ERP, establish a gross list of candidate systems and to establish a preliminary plan for the project.

Running example
The initial value statement formulated by Activio described the motives for selecting ERP as a combination of strategic and operational. The strategic motive was to establish an organizational structure which enabled Activio's business to continue to

grow. The operational motive was to replace manual procedures for material requirements planning by structured and integrated processes.

The Initial Project Modeling at Activio included a rough understanding of the organization and the critical processes which was based on interviews and discussions with the CEO and the person responsible for logistics. This understanding was used to define initial requirements and a preliminary scope of the project. Moreover, a gross list of candidate systems was complied.

3.3 Iterate

The phase *Iteration* aims at identifying the ERP solution which best satisfies the user organization's requirements. During this phase, requirements are iteratively defined, and evaluated against the candidate systems. Through iterating, essential requirements are identified and system candidates successively removed from the candidate list.

The phase includes three activities: *Define Requirements, Analyze System Capabilities* and *Test Driven Evaluation*. In the activity *Define Requirements* information is collected from the organization and requirements are defined based on this information. In the activity *Analyze System Capabilities* this information is used to make an analysis of the capabilities of the candidate systems. Work is performed in parallel to make accurate analyzes and assumptions against the requirements and involve participants from several functional areas. Based on these analyzes, requirements may be refined. The candidate systems are then evaluated against the requirements in the activity *Test Driven Evaluation*. The issues and problems that arise lay the foundation for new and refined requirements for the next iteration.

Running example
A total of four iterations were conducted at Activio. Candidate systems were evaluated against the requirements. Suppliers were frequently contacted in order to resolve issues that arose and Activio employees were continuously involved in resolving requirements issues. Meetings were arranged at the end of iterations and results from the evaluations were presented. At each meeting the evaluated requirements were demonstrated in the candidate systems. The candidate systems were accessed either as software as services or as downloaded demo versions from internet. During demonstrations new requirements were formulated and issues brought up by Activio's employees were included in the next iteration.

There were three candidate systems left after the second iteration: Mamut, Visma and Specter. During the third iteration the requirements became more detailed and specific which made it possible to achieve greater organizational fit [13] between business requirements and system capabilities. Examples of essential user requirements include full service solution avoiding on-premise software installation, modular service design, integrated customer relationship management and automated financial transactions to creditors. Activio's employees used the meetings to ask questions about system capabilities; identify new requirements and form opinions about the candidate systems.

3.4 Decide

The phase *Decide* aims at coming to a conclusion whether to choose any of the evaluated ERP solutions or not. It consists of two activities: *Apply Exit Critera* and *Go/No-go*. The purpose of the activity *Apply Exit Critera* is to establish guidelines for when the requirements specification and the organizational fit is acceptable. The exit criteria define when the evaluation is completed and when a decision can be made. A general guideline is to stop iterating when the user organization does not identify any more essential requirements and only desirable and optional are formulated. The next activity is to decide whether to continue with a subsequent implementation of any of the candidate systems. The decisions can be of different types:

1. Choose a system and continue with implementation
2. Decide to keep the old system and not continue with implementation
3. Return to *Iterate* to evaluate new requirements or new candidate systems
4. Combine parts of different systems and return to Iterate to evaluate the combined system

Decisions are based on the test results complemented supporting information about total cost of ownership and supplier reliability [10].

Running example
After the fourth iteration at Activio no new essential requirements were identified and it was decided to stop the evaluation and move on to decision. The decision support material was complemented with information about cost estimates and supplier information and reference customers. The full process, from *Envision* to *Decide*, took 10 weeks to perform.

4 Assessment

The method has not yet been empirically evaluated in a thorough way, but we here offer an evaluation in the form of informed argument.

- *Increase requirements quality during the selection phase.* By formulating requirements not only from user expectations but also from increased knowledge about system capabilities it is possible to formulate more essential, detailed and relevant requirements from the perspective of the user organization. Instead of guessing in advance, users can base their requirements on better knowledge about opportunities and limitations of the candidate systems at hand.
- *Increase detailed knowledge about system capabilities during the selection phase.* During a conventional and sequential ERP selection process, it is difficult to fully understand the capabilities of a specific system and user organizations easily become dependent on knowledge transfer from suppliers' sales representatives. Through test driven evaluations, users can practically try-out how their requirements fit with a particular system.

- *Decrease lead time for selection*. AMES emphasizes prototyping and iteration to let user requirements evolve and avoid that the selection process stagnates and that time is wasted. The selection process took 10 weeks at Activio compared to between 21-30 weeks as experienced by user organizations applying the methods developed by the Business Application Software Developers Association [14].
- *Decrease supplier dependency during selection.* By using the increased access to fully functioning ERP compared to conventional ERP packages, user organizations become more independent from suppliers. In addition, by better knowledge about the capabilities of candidate systems in their local setting, they become less dependent on supplier representatives.
- *Increase control over subsequent implementation.* User organizations increase control over the implementation process by a better understanding of the detailed strengths and weaknesses of the candidate systems. Hereby they become less dependent on the relationship with suppliers.

The above objectives are related to each other. Some of the objectives interact positively, e.g. *increase requirements quality during the selection phase* is supported by *increase detailed knowledge about system capabilities during the selection phase*. While other objectives need to be balanced against each other, e.g. time v.s. quality where *decrease lead time for selection* need to be balanced against *increase requirements quality during the selection phase*.

5 Conclusions and Future Work

In this paper, we have proposed a new method for ERP selection. The characteristics of the method include an agile and iterative approach to selecting ERP as service. The method has been successfully used at a small entrepreneurial firm where stakeholders appreciated involvement and early tests and found that AMES supported increased understanding and essential requirements identification. A main advantage of the method is that essential systems requirements evolve iteratively during recurring test driven evaluations, knowledge about detailed system capabilities develop during the selection period and that the decision lead time can be shorter than when using conventional selection methods.

Future work will include a second iteration of method design where aspects such as risk management and more detailed activities for establishing a well-balanced list of candidate systems. We also intend to evaluate the method in more organizations of different size, organizational settings and industries.

In design science, communication is often neglected, and a challenge for future research is to effectively transfer the method to practice. A suggestion is therefore to establish a professional service that supports effective transfer of methods to practice use in organizations.

References

1. Ambler, S.: Agile Model Driven Development Is Good Enough. IEEE Software 20(5), September/October, 71--73 (2003)
2. Schwaber,K. and Beedle,M.: Agile SoftwareDevelopmentwith Scrum. Prentice-Hall. (2002)
3. Sumner, M.: Enterprise Resource Planning. New Jersey, Prentice Hall (2004)
4. Davenport, T. H.: Mission Critical: realizing the promise of enterprise systems. Boston, MA, Harvard Business School Press (2000)
5. Juell-Skielse, G.: Improving Organizational Effectiveness through Standard Application Packages and IT Services. Doctoral Thesis in Computer and Systems Sciences at Stockholm University, Sweden (2011)
6. Vargo, S. L., Lusch, R. F.: Service–dominant logic: continuing the evolution. Journal of the Academy of Marketing Science, 36 (1), 1--10 (2008)
7. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design science in information systems research. MIS Quarterly 28(1), 75--106 (2004)
8. Peffers, K., Tuunanen, T., Rothenberger, M., Chatterjee, S.: A Design Science Research Methodology for Information Systems Research. Journal of Management Information Systems 24(3), 45--77 (2008)
9. Nordqvist, A., Westergren, M.: Design and Evaluation of Agile Method for the Acquisition of ERP Software. Thesis in Computer and Systems Sciences at Royal Institute of Technology, Sweden (2008)
10. Nilsson, A.G.: Anskaffning av standardsystem för att utveckla verksamheter: Utveckling och prövning av SIV-metoden, doktorsavhandling, EFI, Handelshögskolan i Stockholm. In Swedish. (1991)
11. Wiegers, K.E.: First Thing First: Prioritizing Requirements. Software Development, September 1999.
12. Parr, A.N. and Shanks, G. A taxonomy of ERP implementation approaches. In Proceedings of the 33d Hawaii International Conference on System Sciences, (2000).
13. Hong, K.K., Kim, Y.G.: The critical success factors for ERP implementation: An organizational fit perspective. Information & Management 40, 25--40 (2002)
14. Business Application Software Development Association: Selecting a Business System. `http://myfiles.uk-plc.net/c372982/documents/BASDA-Publications/BASDA%20-%20Selecting%20a%20Business%20System%202007%20final.pdf`, last accessed 20.04.2012 (October 2007) BASDA.
15. Fowler, M.; Highsmith, J.: The Agile Manifesto. Software Development, August (2001)

Facilitating Business Improvement by Information Systems using Model Transformation and Metrics

Haruhiko Kaiya[1,2], Shunsuke Morita[1], Kenji Kaijiri[1], Shinpei Hayashi[3] and Motoshi Saeki[3,2]

[1] Dept. of Computer Science, Shinshu University, Nagano 380-8553, Japan
kaiya@shinshu-u.ac.jp
12KA110E@shinshu-u.ac.jp, kaijiri@cs.shinshu-u.ac.jp
[2] National Institute of Informatics (NII), Tokyo 101-8430, Japan
[3] Dept. of Computer Science, Tokyo Institute of Technology, Tokyo 152-8552, Japan
saeki@se.cs.titech.ac.jp, hayashi@se.cs.titech.ac.jp

Abstract. We propose a method to explore how to improve business by introducing information systems. We use a meta-modeling technique to specify the business itself and its metrics. The metrics are defined based on the structural information of the business model, so that they can help us to identify whether the business is good or not with respect to several different aspects. We also use a model transformation technique to specify an idea of the business improvement. The metrics help us to predict whether the improvement idea makes the business better or not. We use strategic dependency (SD) models in i* to specify the business, and attributed graph grammar (AGG) for the model transformation.

Keywords: Requirements Analysis, Strategic Dependency Model, Model Transformation, Metrics

1 Introduction

To develop an information system supporting business, we typically perform the following steps; 1) analyzing current business and construct its model, so called *as-is* model, 2) identifying the problems that lurk in the as-is model, 3) evolving the as-is model to the *to-be* model that can solve the identified problems. In these steps, how to evolve the as-is to the to-be is one of crucial topics because most intellectual activities of human analysts are necessary to create the solutions of the identified problems. Although many techniques and tools are available for supporting these steps, little supporting techniques in the evolution of to-be models exist. For example, idea generation methods such as Brain Storming are considered as a useful technique. Although they can help human analysts to create the idea as solution of the problems, they are too general and weak to support the evolution step more effectively. Activity based cost (ABC) method [1] evaluates as-is activities only from aspects of cost and time spent in executing them, while Balanced scorecard [6] requires well skilled and experienced analysts to set up evaluation criteria. Rather, best practices of the past evolutions allow the ordinary analysts to create the to-be of higher quality with their less efforts and a technique to accumulate and utilized the best practices is necessary. The purpose of this paper is to explore the technique to formalize reusable best practices of how to create to-be models.

Many techniques and tools are related to modeling languages such as BPMN, Workflow Languages, the usages and the extension of UML diagrams, etc. Goal-oriented Requirements Analysis (GORA) can be applied to describe an as-is model and a to-be model of the business, and is useful to clarify its business goals. In [14], the authors developed a language called i* which contains GORA, and used it to represent as-is models and to-be in the organizational context. Almost all of these modeling languages are essentially graphs with types and attributes. Thus, the evolution from an as-is model to a to-be model can be considered as graph transformation and be formalized with graph grammar. This paper proposes a technique to specify the evolution with Attribute Graph Grammar (AGG) [12]. To specify types and attributes on graphs, we use a meta-modeling technique. In addition, we should define metrics to detect the problems in an as-is and the metrics can be defined on the meta-model. Our technique is for formalize best practices in creating to-be models with graph grammar and metrics definitions and accumulating them as reusable assets.

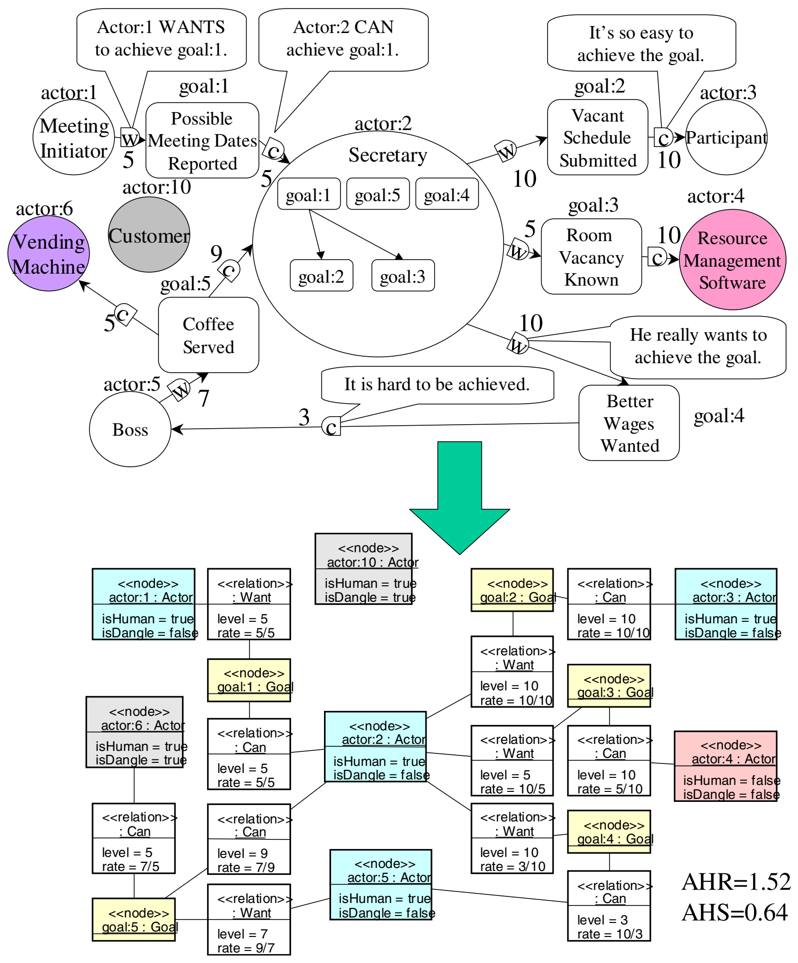
The rest of the paper is organized as follows. Section 2 is for preliminaries and we explain the existing techniques that we use in this paper, i.e. an extended i* and a graph re-writing system based on AGG. We use i* diagrams to represent as-is models and to-be ones as examples. In section 3, we illustrate how to define metrics and graph transformation as the evolution practices from an as-is model into a to-be model. Metrics play an important role because they are used to select applicable and suitable transformation rules and to clarify which aspects on a to-be model can be improved. One of the significant reasons why an information system is developed is to reduce human's responsibilities and efforts in achieving business goals by automating them. A Strategic Dependency (SD) model of i* can represents responsibilities of the stakeholders to goals, and this is a reason why we focus on SD diagram of i* in this paper. However, our technique is not limited to SD diagrams and we can apply other diagrams by defining metrics and graph grammar on its certain meta-model. Section 4 is for listing up related works.

2 Preliminaries

In this section, we briefly introduce i* strategic dependency (SD) modeling and attributed graph grammar (AGG) because our research in this paper uses these two techniques.

2.1 i* SD modeling

Before specifying an information system introduced in business, we have to analyze what kinds of goals human wants the system to achieve instead of human. For example, a human secretary wants a meeting scheduling system to summarize schedules of all staffs because he does not have to do it with the system. A modeling language i* strategic dependency (SD) model [14] is useful for us to analyze it because a dependency about the ownership and the responsibility for each goal is clearly represented. In i*, an as-is model is used for specifying current dependencies in business, and a to-be model is used for specifying expected dependencies in the business normally with information



Actor:1 WANTS to achieve goal:1.
Actor:2 CAN achieve goal:1.
It's so easy to achieve the goal.
actor:1
Meeting Initiator
goal:1
Possible Meeting Dates Reported
actor:2
Secretary
goal:1
goal:5
goal:4
goal:2
goal:3
goal:2
Vacant Schedule Submitted
actor:3
Participant
actor:10
Customer
actor:6
Vending Machine
goal:3
Room Vacancy Known
actor:4
Resource Management Software
goal:5
Coffee Served
He really wants to achieve the goal.
It is hard to be achieved.
actor:5
Boss
Better Wages Wanted
goal:4
<<node>> actor:1 : Actor isHuman = true isDangle = false
<<relation>> : Want level = 5 rate = 5/5
<<node>> actor:10 : Actor isHuman = true isDangle = true
<<node>> goal:2 : Goal
<<relation>> : Can level = 10 rate = 10/10
<<node>> actor:3 : Actor isHuman = true isDangle = false
<<node>> goal:1 : Goal
<<relation>> : Want level = 10 rate = 10/10
<<node>> goal:3 : Goal
<<node>> actor:6 : Actor isHuman = true isDangle = true
<<relation>> : Can level = 5 rate = 5/5
<<node>> actor:2 : Actor isHuman = true isDangle = false
<<relation>> : Want level = 5 rate = 10/5
<<relation>> : Can level = 10 rate = 5/10
<<node>> actor:4 : Actor isHuman = false isDangle = false
<<relation>> : Can level = 5 rate = 7/5
<<relation>> : Can level = 9 rate = 7/9
<<relation>> : Want level = 10 rate = 3/10
<<node>> goal:4 : Goal
<<node>> actor:5 : Actor isHuman = true isDangle = false
<<relation>> : Can level = 3 rate = 10/3
<<node>> goal:5 : Goal
<<relation>> : Want level = 7 rate = 9/7
AHR=1.52
AHS=0.64

Fig. 1. An Example of an extended i* Model (top) and its Internal Representation (bottom). The internal representation is based on the meta-model in Figure 2.

systems. The to-be model should be thus better than the as-is model with respect to human actors in the to-be model.

To enable the model transformation in a SD model effectively, we extend the syntax of the SD model, and the extended models partially violate syntax of the original SD model [13]. The extended syntax will be explained in detail in the next section. Here we explain how and why we extend the SD model by using an example in Figure 1 in addition to the original syntax of the SD model.

A SD model consists of several pairs of an actor (called a *depender*), a goal and another actor (called a *dependee*). In each pair, these two issues are modeled; an actor wants to achieve a goal, and another actor can achieve the goal. Actor:1, goal:1 and actor:2 at the top-left side in Figure 1 shows such a pair. We call a relationship between the goal and an actor who wants to achieve the goal as a *want-relation*, and another relationship between the goal and another actor who can achieve the goal as a *can-relation*. We attach an attribute called "level" to the want-relation and the can-relation respectively. The attribute takes a value from 1 to 10. When an actor really wants to achieve a goal, the level of its want-relation takes large value. Otherwise, the level takes small one. When an actor can achieve a goal almost completely, the level of its can-relation takes large value. For example in Figure 1, the level of a want-relation between actor:2 and goal:4 takes ten because the secretary (actor:2) really want to get better wages (goal:4). However, the level of a can-relation between actor:5 and goal:4 takes 3 because it is hard for his boss (actor:5) to give better wages. Can-relations and want-relations have another attribute called "rate". The attribute will be explained in the next section because the attribute is an intermediate attribute to calculate metrics of a SD model.

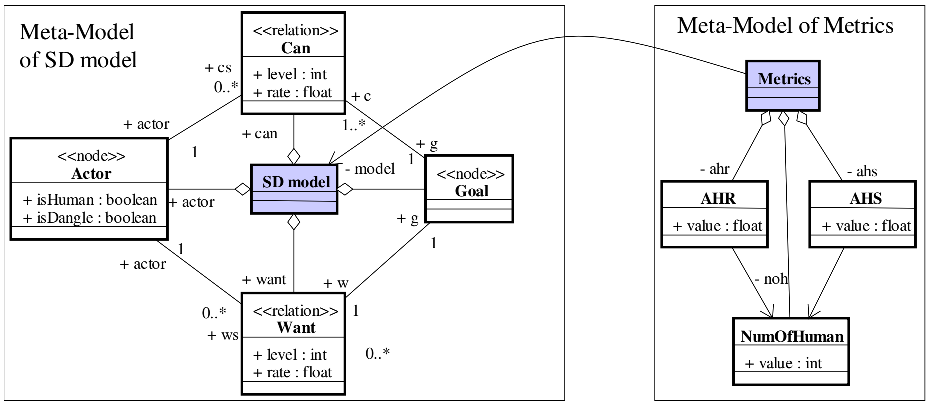
We explain two attributes on actors: *isHuman* and *isDangle*. Both attributes take Boolean value. Distinguishing human actors from other actors is important because one of the policies of our model transformation is to minimize the responsibility of human and to maximize the satisfaction of human. An attribute isHuman is used for this purpose. In original i*, completely isolated actors such as actor:10 in Figure 1 are called dangling actors, and such actors should not be contained in a model [13]. In our extension, we call any actors with true value in isDangle attribute as dangling actors such as actor:10 and actor:6 in Figure 1. We then permit a model to contain such dangling actors because we want to record potential alternatives of dependers and dependees in the model so that we can easily explore another possibilities of actor dependencies. In addition, we permit a goal to have more than one candidate of dependees in the same reason. However, only one dependee has to have isDangle attribute as false and the others has to have isDangle attribute as true even if a goal has more than dependees. This constraint is defined as an OCL expression (context Goal) of the meta-model in Figure 2. For example in Figure 1, goal:5 has two dependees actor:6 and actor:2, but only actor:2 takes false value in isDangle.

Original i* has several types of goals (more precisely intentions) such as goals, soft-goals, tasks and resources. However, we only use goals in our SD model. Goals in our SD model can be used as goals and soft-goals in original i* because goals between a can-relation with level 10 and a want-relation with level 10 can be regarded as goals in original i* and others can be soft-goals. We mainly focus on very early stages of requirements definition, but tasks and resources seem to be related to architecture and/or implementation issues. We thus do not use tasks and resources in our model.

2.2 Graph Rewriting System

In Model Driven Development (MDD), one of the technically essential points is the model transformation. Since we use a class diagram to represent a meta-model, a model, i.e. an instance of the meta-model can be considered as a graph, whose nodes have types and attributes, and whose edges have types, so called attributed typed graph. In this paper, the model transformation is thus defined as a graph rewriting system, and graph-rewriting rules dominate allowable transformations. Here we briefly introduce a graph rewriting system.

A graph rewriting system converts a graph into another graph or a set of graphs following pre-defined rewriting rules. There are several graph rewriting systems such



```
context Can
inv: rate == g.w.level/level

context Want
inv: rate == g.c->select(actor.isDangle==false)->collect(level)->sum() / level

context Goal
inv: c->select(actor.isDangle==false)->size()==1

context Metrics
inv:
noh.value == model.actor->select(isHuman && isDangle==false)->size()
ahr.value ==
  model.can->select(actor.isHuman && actor.isDangle==false)
  ->collect(rate)->sum() / noh.value
ahs.value ==
  model.want->select(actor.isHuman && actor.isDangle==false)
  ->collect(rate)->sum() / noh.value
```

Fig. 2. Meta-model of a SD model and its Metrics

as PROGRESS [10] and AGG [12]. Since we should deal with the attribute values attached to nodes in a graph, we adopt the definition of the AGG system in this paper. An example of AGG can be found in Figure 3 explained in the next section.

3 Metrics and Transformation

We define a meta-model (grammar) of both a SD model and its metrics in Figure 2 to facilitate effective model transformation on a SD model. An instance of a SD model is shown at the bottom of Figure 1. Currently, we use two metrics called average human responsibility (AHR) and average human satisfaction (AHS) with the help of a complementary metric called number of human actors (NumOfHuman) as shown in the figure. The formal definitions of these metrics are shown as the OCL invariants in Figure 2.

We will show how to calculate these metrics by using a SD model in Figure 1 and the intermediate values in Table 1. In the table, the values of Can.rate sum up for each non-dangling actor respectively. For example, 5/5 and 7/9 are summed up for actor:2 because actor:2 can achieve two goals of goal:1 and goal:5 and Can.rate in can-relations between actor:2 and each goal takes 5/5 and 7/9 respectively. The value of Can.rate

Table 1. Intermediate values for calculating AHR and AHS in a SD model in Figure 1. NumOfHuman.value is 4 in this model.

	sum. of Can.rate	sum. of Want.rate
actor:1	0	$\frac{5}{5}$
actor:2	$\frac{5}{5}+\frac{7}{9}$	$\frac{10}{10}+\frac{5}{10}+\frac{3}{10}$
actor:3	$\frac{10}{10}$	0
actor:5	$\frac{10}{3}$	$\frac{9}{7}$
row total	$\frac{5}{5}+\frac{7}{9}+\frac{10}{10}+\frac{10}{3}$	$\frac{5}{5}+\frac{10}{10}+\frac{5}{10}+\frac{3}{10}+\frac{9}{7}$
row total / 4	1.52 (= AHR.value)	1.02 (= AHS.value)

shows relative weight of responsibility with respect to expectation of a depender. For example in Figure 1, actor:5 has to achieve goal:4, and actor:2 wants to achieve the goal. In this case, Can.rate between actor:5 and goal:4 takes 10/3 (= 3.3). We may regard actor:5 takes a really heavy responsibility about goal:4 because this value means actor:2's expectation is about three times as actor:5's ability. On the other hand, actor:3 takes a reasonable responsibility about goal:2 because Can.rate between actor:3 and goal:2 is 10/10 (= 1.0). The row total of "sum. of Can.rate" shows the total of such responsibility. We finally divide the value of row total by the number of non-dangling human actors, and we can get the AHR.value, 1.52 in this case.

How to get AHS.value is almost the symmetrical way above. The value of Want.rate shows relative weight of satisfaction with respect to ability of a dependee. In our extended SD model, we permit a goal to have more than one dependee such as actor:6 and actor:2 of goal:5 in Figure 1. However, we only use a non-dangling actor in such a case, and our OCL (context Goal) guarantees there is only one non-dangling actor.

Because metrics AHR and AHS can be calculated from any SD model in conformance with the meta-model in Figure 2, we can observe changes of these metrics during any model transformation. We regard a model transformation is good when AHR decreases because systems make the responsibility of human to be decreased. We also regard a model transformation is good when AHS increases because systems have to increase satisfaction of human. Currently, we only have two metrics but we may append any metrics that are useful to evaluate model transformation.

Figure 3 shows an example of a model transformation using AGG. The as-is model in the figure is a part of the SD model in Figure 1. Three patterns of graphs such as NAC, LHS and RHS in AGG correspond to a model transformation rule. When a part of graph is matched to LHS and the part is not matched to NAC, the part is transformed into RHS. Because each node in a graph pattern may contain variables and variables in RHS can be defined based on variables in LHS, metrics and its changes can be naturally represented in a model transformation rule written in AGG. In this example, AHR varies from 0.388 (=0.7777/2) to 0, and AHS varies from 0.64 (=1.28/2) to 0.71. This model transformation thus causes the reduction of the human responsibility and the gain of the human satisfaction.

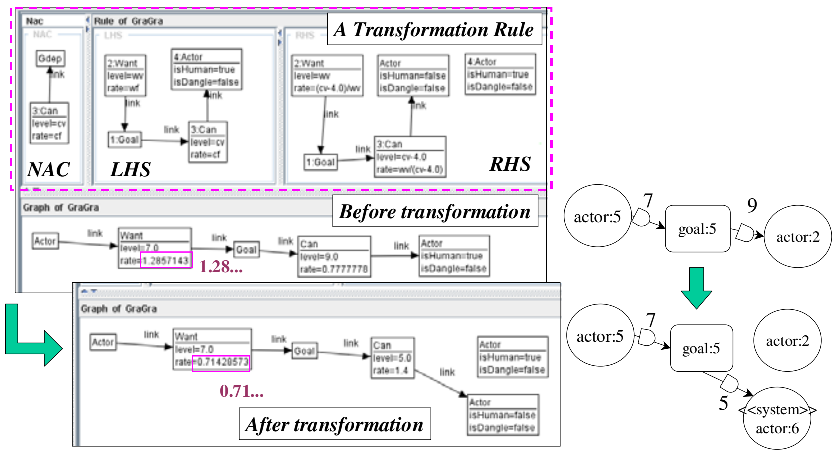


Fig. 3. Example of a model transformation using AGG. An as-is model in this figure is a part of the SD model in Figure 1.

4 Related Work

Activity-based Costing (ABC) [1] and Balanced Score Card (BSC) [6] are famous methodologies to explore better to-be models of business, and they focus on several attributes such as costs, performance, time, knowledge and so on. Such kinds of attributes can be introduced in our SD meta-model, or our meta-models can be explicitly related to other meta-models such as business process with such attributes. These methodologies are useful to explore problems of an as-is model, but to-be models are not explicitly specified. In our method, the model transformation technique explicitly specifies such to-be models.

We can find several researches for transforming as-is business process to to-be one [9, 8, 4], and some of them use metrics for facilitating better transformation. Our research mainly focuses on strategic dependencies that are earlier than processes with respect to clarifying requirements. As mentioned in the previous section, the meta-modeling technique enables us to make explicit relationships between early requirements (SD model) and other concerns such as process, architecture and implementation. This explicit relationships and the separation of concerns help us to integrated and rational decision for business improvement. Because transformation among such different concerns is already proposed [2], it is not so difficult to define such relationships.

Original i* [14] has a lot of vocabularies such as a goal, a soft-goal, a task, a resource, an actor, an agent, a role, a position and so on. We however use only a goal and an actor because these two are the fundamental elements of a SD model. There are a lot of extensions of i* [5, 7, 11] and they have more vocabularies than original one. Our method in this paper can be extended naturally according to each extension because such extended vocabularies can be formalized by attributes attached to goals, actors, can and want-relations. There are some researches about i* model using metrics (summary can be found in [3]), but there are few ones focusing on changes of the metrics.

5 Conclusion

In this paper, we proposed a method to improve an as-is model of business by using a model transformation technique and metrics on the model. We use i* SD model for representing business models and AGG for model transformation. Currently, we only focus on strategic dependencies among actors in business, but we have to take into account more information such as business process, architecture, implementation and so on. Our method can easily take into account such additional information because it uses meta-modeling technique useful for make explicit relationships among different aspects of the business. We want to extend our current meta-model including metrics in such a way.

Acknowledgement This research is supported by Takahashi Industrial and Economic Research Foundation, Japan, 02-001-A24-1009 and MEXT/JSPS KAKENHI Grant Number 23500042.

References

1. Robin Cooper, Robert S. Kaplan, and Lawrence S. Maisel. *Implementing Activity-Based Cost Management: Moving from Analysis to Action*. Inst of Management Accountants, Oct. 1993.
2. Ken Decreus, Monique Snoeck, and Geert Poels. Practical challenges for methods transforming i* goal models into business process models. In *RE*, pages 15–23, 2009.
3. Xavier Franch and Gemma Grau. Towards a catalogue of patterns for defining metrics over i*models. In *CAiSE*, pages 197–212, 2008.
4. Kaori Fujiwara, Bala Ramachandran, Akio Koide, and Jay Benayon. Business process transformation wizard: a bridge between business analysts and business process transformation technology. In *IEEE SCC*, pages 83–90, 2007.
5. Paolo Giorgini, Fabio Massacci, John Mylopoulos, and Nicola Zannone. Modeling security requirements through ownership, permission and delegation. In *RE*, pages 167–176, 2005.
6. Robert S. Kaplan and David P. Norton. *The Balanced Scorecard: Translating Strategy into Action*. Harvard Business School Press, Sep. 1996. `http://www.balancedscorecard.org`.
7. Haralambos Mouratidis and Paolo Giorgini. Secure tropos: a security-oriented extension of the tropos methodology. *International Journal of Software Engineering and Knowledge Engineering*, 17(2):285–309, 2007.
8. Marion Murzek, Gerhard Kramler, and Elke Michlmayr. Structural patterns for the transformation of business process models. In *EDOC Workshops*, page 18, 2006.
9. Mariska Netjes, Hajo A. Reijers, and Wil M. P. van der Aalst. On the formal generation of process redesigns. In *Business Process Management Workshops*, pages 224–235, 2008.
10. Andy Schürr. Developing graphical (software engineering) tools with progres. In *ICSE*, pages 618–619, 1997.
11. Alistair G. Sutcliffe. Trust: From cognition to conceptual models and design. In *CAiSE*, pages 3–17, 2006.
12. Gabriele Taentzer. Agg: A graph transformation environment for modeling and validation of software. In *AGTIVE*, pages 446–453, 2003. `http://user.cs.tu-berlin.de/~gragra/agg/`.
13. Eric Yu, Jaelson Castro, and Anna Perini. Strategic Actors Modeling with i*, RE 2008 - Tutorial, Aug. 2008.
14. Eric S. K. Yu. Towards modeling and reasoning support for early-phase requirements engineering. In *RE*, pages 226–235, 1997.

Transforming Transaction Models into ArchiMate

Sybren de Kinderen[1], Khaled Gaaloul[1], and H.A. (Erik) Proper[1,2]

[1] CRP Henri Tudor
L-1855 Luxembourg-Kirchberg, Luxembourg
`sybren.dekinderen, khaled.gaaloul, erik.proper@tudor.lu`
[2] ICIS, Radboud University Nijmegen
P. O. BOX 9010 6500, GL Nijmegen, The Netherlands

Abstract. ArchiMate, a language for modelling an organisation from a holistic perspective, lacks guidelines and techniques for exploring each of its perspectives in depth. To address this issue, we propose to use the DEMO modelling technique and toolset as a front-end for ArchiMate. In particular, DEMO adds to ArchiMate a conceptual clarity, as well as tools and techniques for modelling business processes.
Specifically, in this paper we contribute a formal model transformation from DEMO to ArchiMate, and show how this model transformation can be used to transform DEMO models into ArchiMate models. Our model transformation approach is illustrated by a fictitious but realistic case study from the insurance domain.
Keywords: ArchiMate, DEMO, meta model, model transformation

1 Introduction

ArchiMate, is an Open Group standard [1, 2] for the modelling of enterprise architectures[3]. Being designed as a general purpose modelling language for enterprise architecture [1], it allows architects to model an enterprise from a holistic perspective, showing amongst others, an organization's products and services, how these products and services are realized/delivered by business processes, and how in turn these processes are supported by information systems and their underlying IT infrastructure. This holistic perspective on an enterprise helps to guide change processes [3], provides insight into cost structures [4], and more [1].

Because of the inherent holistic nature, ArchiMate lacks specificity on how to model the different perspectives *in-depth*. For example, ArchiMate lacks guidelines for process modelling, and lacks expressivity for modelling an enterprise from a value exchange perspective [5]. Moreover, as claimed by [6] ArchiMate lacks conceptual clarity and precision. This "lack" is, however, a direct consequence of the coarse-grained, and holistic, nature of ArchiMate. In that sense this *freedom of interpretation* has been designed into the language on purpose [1]. Nevertheless, as a result, different modellers do indeed create different models.

[3] http://www.opengroup.org/archimate/

To address the above issues, it has already been suggested that ArchiMate could benefit from the integration with a method such as DEMO to provide it with a more explicit way of working, supporting architects in the creation of models [6]. In this paper, we focus on bridging between DEMO and ArchiMate.

DEMO, short for Design and Engineering Methodology for Organizations, is a method comprising of a comprehensive set of conceptual modelling techniques, in combination with a theory based a way of thinking and associated way of working, focused on modelling/analysing/designing the *essential* aspects of an organization [7, 8]. DEMO uses the word *essential* here to refer to the implementation-independent aspects of an organization. As such, DEMO aims to abstracts away from implementation-specific details, such as the information systems present in business collaboration. Linking DEMO and ArchiMate would enable architects to use the semantically rich way of thinking of DEMO to create ArchiMate models. These models would then primarily be ArchiMate models providing an essential view of the business processes (business layer) and the information processing (application layer) in the enterprise. Further benefits of linking DEMO and ArchiMate are discussed in [9].

The core contribution of this paper is two-fold: (1) a formal mapping of the meta models underlying DEMO and ArchiMate, accompanied by a rationale of why such a mapping is beneficial (2) a systematic application of the DEMO and ArchiMate meta models to map a model created in DEMO to a model of an enterprise architecture in ArchiMate. We use a running example of an insurance scenario to illustrate our ideas.

The remainder of this paper is structured as follows. Sect. 2 illustrates, by means of the case of an insurance company, how we intend to use DEMO as a front-end to ArchiMate. Therafter, we show how to formally transform a DEMO model into an ArchiMate model (Sect. 3).

2 Modelling insurance processes in DEMO

2.1 Archinsurance: selling car insurance via insurance brokers

We use the insurance company Archinsurance, as documented in [3, 10], as a fictitious but realistic use case. We focus on car insurance, an insurance product that Archinsurance sells via *insurance brokers*. The main reason for selling insurance via brokers is to reduce the risk of *adverse risk profiles* [11], incomplete or faulty risk profiles of customers that lead insurance companies to sell inappropriate insurance packages.

2.2 Using DEMO transaction patterns for process modelling

We use DEMO to model the sale of car insurance by Archinsurance. The DEMO meta model is depicted in Fig. 2, with an instantiation of this model, called the DEMO process model, in Fig. 1. Chief to the creation of this process model are DEMO *transaction patterns*.

A DEMO transaction pattern is a process-based pattern of (instantiations of) DEMO meta model concepts, showing the sequence of acts that *always* needs to be executed to realise a social interaction between two actors (this interaction being called a DEMO 'transaction'). So, here we see DEMO's emphasis on the essential aspect of an organisation, as mentioned in the introduction: no matter what the domain, if we perceive of an organisation as a social entity, then we see a pattern of generic acts that *always* occurs in carrying out a transaction [8]. So, for example, one actor always has to initiate a transaction by performing the act 'request' (which in the Archinsurance case may translate to the act 'Apply for insurance' as carried out by a customer), while another actor has to always perform the 'execute' act in order to produce the good or service that the initiating actor is interested in (see Fig. 1). In the Archinsurance case, this may translate to the act 'Find matching package' which, as mentioned before, is executed by the insurance broker. Such patterns are particularly useful as they guide us in creating process models, whereas a language on its own, such as ArchiMate, cannot.

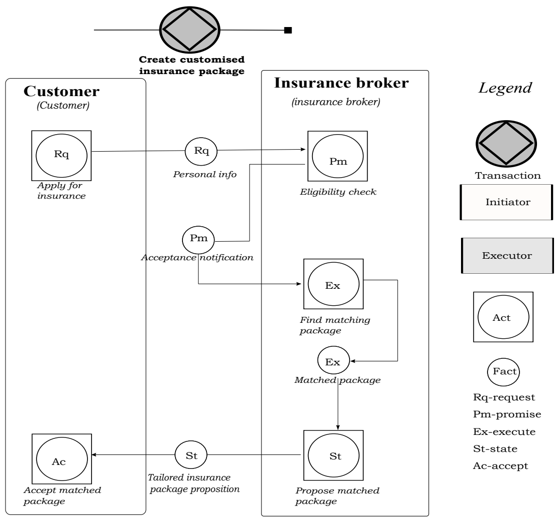


Fig. 1. An excerpt of the DEMO Business process model

3 Translating DEMO process models to ArchiMate

We now show how to create an ArchiMate enterprise architecture model, using the DEMO process model as a baseline. To this end, we first introduce the used mapping technique (in Sect. 3.1), and subsequently apply this technique to transform a DEMO process model to an ArchiMate model (in Sect. 3.2).

3.1 The used mapping technique

For mapping DEMO to ArchiMate, we use the meta model mapping technique described in [12] where authors distinguish different types of mappings, the most relevant for our work being (1) *class-to-class mappings*, which relates a concept from meta model A to a concept from meta model B (e.g., a 'Subject' from DEMO relates to an 'Actor' from ArchiMate). And (2) *relation-to-relation mappings*, which relates concept relationships from meta model A with concept relationships from meta model B (e.g., 'performs_role' between the concepts Subject and Actor from DEMO relates to the ArchiMate relation 'assigned_to' between the concepts Actor and Business role).

3.2 Mapping the DEMO meta model to the ArchiMate meta model

Now we translate a DEMO process model for Archinsurance into an ArchiMate business layer model. We do this in two main steps: (1) Translate the concepts from a DEMO process model to an ArchiMate process model, which we can do given that, looking at the concept definitions, the holistic ArchiMate language subsumes DEMO's social perspective, (2) Define a (partial) enterprise architecture model from a business perspective that focuses on the DEMO process model. Here, we construct an ArchiMate model from the mapped DEMO concepts. As we now actually construct an ArchiMate model, we take here into consideration (a) the difference in abstraction level between DEMO and ArchiMate, and (b) additional ArchiMate constructs not present in DEMO, for example for depicting an IT perspective on the organisation at hand.

Step 1: Horizontal integration via meta model mapping The first step will apply our mapping between the DEMO meta model and the ArchiMate business layer meta model (see Fig. 2). Here, we make a mapping on a purely *horizontal* level (cf. [12]), meaning that we consider only differences between aspects modelled in DEMO and ArchiMate *on the same abstraction level.* In doing so, we apply the DEMO - ArchiMate meta model mapping from Fig. 2, and the corresponding rationale of our meta model mapping (i.e., Table 1). In Fig. 2, we define a specialisation relation between the mapped concepts from DEMO to ArchiMate concepts.

For Archinsurance, we apply this mapping as follows. For reference, see the Archinsurance ArchiMate model in Fig. 3, and the Archinsurance DEMO process model in Fig. 1. For instance, as explained in Table 1 *subjects from DEMO map to business actors in ArchiMate.* For Archinsurance, we thus map the subject 'customer' to the business actor 'customer' in ArchiMate. Due to space restrictions, we will not exemplify with our scenario the rest of concept mappings explained in Table 1.

In addition, we perform relation-to-relation mappings between DEMO and ArchiMate (see Table 2). As such, we map *the relation (Subject)performs_role(Actor) from DEMO to the relation (Business actor)assigned_to(Business role) from*

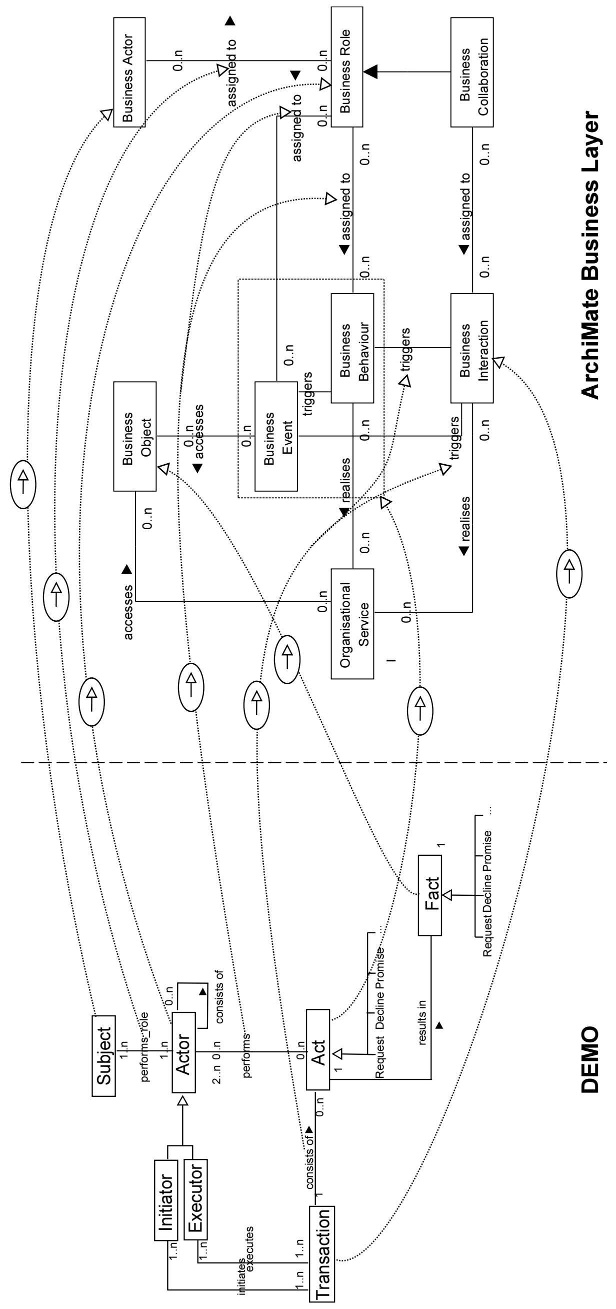


Fig. 2. Mapping of DEMO and ArchiMate meta models

ArchiMate. For example, in both DEMO and ArchiMate, the department 'insurance broker' performs the role of 'insurance broker'. The relation-to-relation mappings are explained in Table 2.

Table 1. DEMO - ArchiMate meta model concept mapping relations

DEMO - ArchiMate Concepts	Mapping rationale
Actor Business role	In DEMO, an actor refers to a social role played by a subject in an organisation. Such a social role corresponds to the definition of a business role in ArchiMate where roles are typically used to distinguish responsibilities.
Subject Business actor	A DEMO subject is an organisational entity - person, department or otherwise - that can fulfil an organisational role. This corresponds to a business actor in ArchiMate, which is an organisational entity that performs some behaviour (cf. [3]), thus it can also fulfil a role.
Act Business behaviour/event	An act is performed by a subject in a social role. Its scope is about contribution/coordination for services. In the ArchiMate context, it corresponds to the realisation of an organisational service via a business process or a function (business behaviour) or a business event (e.g., an external request).
Transaction Business interaction	For DEMO transactions, the initiation and execution are performed by different actors. This emphasises the interaction aspect that we can find in ArchiMate, where a business interaction is carried out by more than one actor.
Fact Business object	A fact is any object that results from performing an act. In ArchiMate, this corresponds to a business object, which 'represent the important concepts in which the business thinks about a domain' [3].

Step 2: Vertical integration: defining an appropriate abstraction level in ArchiMate The second step consists of defining an enterprise architecture model using ArchiMate to represent a DEMO process model.

First, in addition to the horizontal differences in Step 1, we now consider also the *vertical* differences between DEMO and ArchiMate. This means that we remove from the ArchiMate model any elements that are too detailed for depicting a holistic perspective on the organisation at hand. For example: for the Archinsurance case, we thus remove the business object 'acceptance notification' (which ArchiMate inherits from the DEMO process model in Fig. 1), since they are too detailed for the high-level model overview provided by ArchiMate.

Second, we supplement the model elements inherited from DEMO with ArchiMate constructs. This we do to fully express a holistic perspective on the organisation at hand, most prominently in terms of the supporting IT infrastructure. For example, as we can see in Fig. 3, for Archinsurance we model that

Table 2. DEMO - ArchiMate meta model relations mapping

DEMO - ArchiMate relation	Mapping rationale
performs_role assigned_to	In both DEMO and ArchiMate, one relates a real world entity (e.g., Archinsurance) to a role played by that entity (e.g., the role of insurer in the case of Archinsurance).
consists_of triggers	As transactions map to business interactions, and acts map to business events and business behaviour, the relation 'consists_of' between transactions and acts in DEMO maps logically to the relation 'triggers' between business interactions and business events/business behaviour in ArchiMate.
performs assigned_to	While both use different nomenclature, in both DEMO and ArchiMate, a role - not the real-world entity behind it - carries out acts.

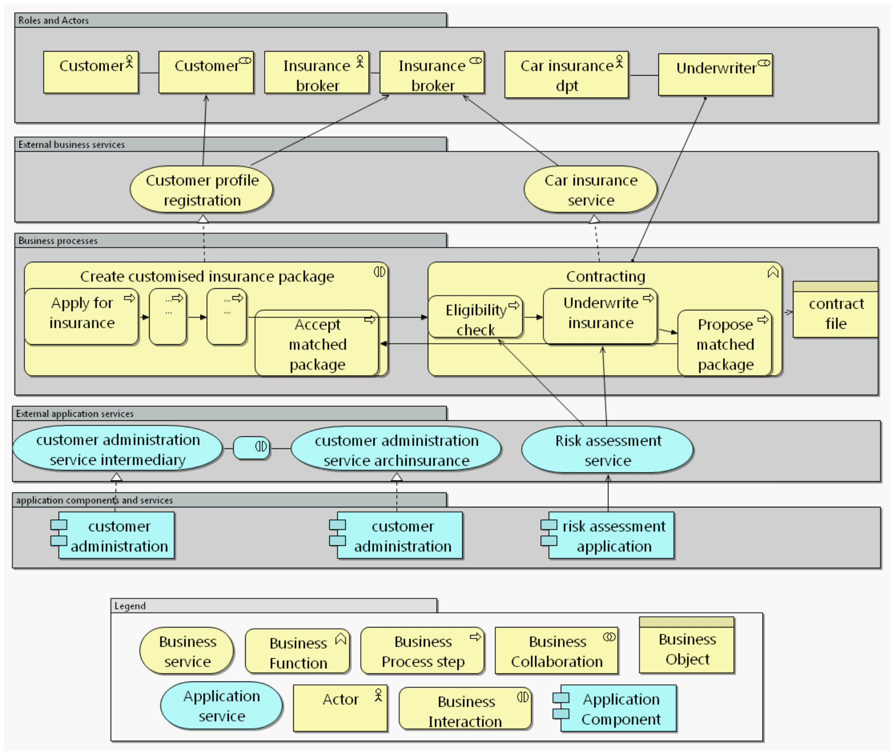


Fig. 3. (Partial) enterprise architecture model based on DEMO process

the business process activities 'eligibility check' and 'underwrite insurance' are supported by a risk assessment application, and that the business collaboration 'create customized insurance package' is supported by administrative applications from both the insurance broker and Archinsurance.

4 Conclusion and Future Work

In this paper, we used DEMO as a front end for ArchiMate to help creating enterprise architecture models. Using a fictitious case from the insurance domain, we introduced a computationally supported mapping between DEMO and ArchiMate, and showed how this mapping can be applied to translate a DEMO model into an ArchiMate model.

For further research, we will look into enriching ArchiMate itself with other techniques, for example to add expressivity from a value perspective. This naturally introduces a number of research challenges, such as how to balance model integration - changing ArchiMate itself - with model transformation - leaving a concern to a specific technique, and import results into ArchiMate.

References

1. M.M. Lankhorst et al. *Enterprise Architecture at Work: Modelling, Communication and Analysis*. Springer, Berlin, Germany, 2005.
2. M.-E. Iacob, H. Jonkers, M.M. Lankhorst, and H.A. Proper. *ArchiMate 1.0 Specification*. The Open Group, 2009.
3. H. Jonkers, M.M. Lankhorst, R. van Buuren, S.J.B.A. Hoppenbrouwers, M. Bonsangue, and L. Van der Torre. Concepts for Modeling Enterprise Architectures. *International Journal of Cooperative Information Systems*, 13(3):257–288, 2004.
4. R. van Buuren, J. Gordijn, and W. Janssen. Business case modelling for e-services. In *18 th Bled eConference eIntegration in Action*, 2005.
5. V. Pijpers, J. Gordijn, and H. Akkermans. e3alignment: Exploring inter-organizational alignment in networked value constellations. *International Journal of Computer Science & Applications*, page 59, 2009.
6. R. Ettema and J.L.G. Dietz. Archimate and demo–mates to date? *Advances in Enterprise Engineering III*, pages 172–186, 2009.
7. J.L.G. Dietz. *Enterprise ontology: theory and methodology*. Springer Verlag, 2006.
8. J.L.G. Dietz. The deep structure of business processes. *Communications of the ACM*, 49(5):58–64, 2006.
9. S. de Kinderen, K. Gaaloul, and E. Proper. On transforming demo models to archimate. In *To appear in the Proceedings of the 2012 EMMSAD/Eurosymposium workshop, Gdansk, Poland.* Springer Verlag, 2012.
10. M.M. Lankhorst, H.A. Proper, and H. Jonkers. The Architecture of the ArchiMate Language. *Enterprise, Business-Process and Information Systems Modeling*, pages 367–380, 2009.
11. J.D. Cummins and N.A. Doherty. The economics of insurance intermediaries. *The Journal of Risk and Insurance*, 73(3):359–396, 2006.
12. S. Zivkovic, H. Kuhn, and D. Karagiannis. Facilitate modelling using method integration: An approach using mappings and integration rules. 2007.

Quality Management Modeling of Business Processes in IS Projects

Michał Kuciapski

University of Gdańsk, Department of Business Informatics,
m.kuciapski@ug.edu.pl

Abstract. Years of evolution in the field of information systems, business processes and universal modeling notations have resulted in the creation of modern modeling languages, such as BPMN or UML. Founded languages concentrate on processes and activity flow without taking into account important management categories like quality or control. These aspects permit the research thesis that there is a room for further modeling improvement. The article concentrates on analyzing the role of quality management modeling for business and project processes. It begins by introducing the current state of modeling languages. It is a starting point for the presentation of elaborated quality-oriented notation for the business and project processes. The concept discussed in the second section is presented as a separate diagram type. A sample model is also included showing notation capabilities and its practical usability. The third part of the article presents verification of proposed quality management modeling approach of processes in IS projects. The article concludes with a summary.

Keywords: quality management, business process modeling, project management, modeling notation

1 Introduction

As opposed to the analysis and design of information systems dominated by UML language, there can be no single denoted key notation in the discipline of process modeling. In many situations, organizations leave methodological and notational aspects to the contractor, even though some methodologies used by companies pre-date the notion of process modeling [8] and most approaches employed are niche in character.

A strong impulse in the development of modeling approaches occurred in the 1990s. This was a natural consequence of organizations' reorienting in the direction of process management and solutions were elaborated for use in business process management. An exception, however, was UML (Unified Modeling Language) as a standard for general use. Modeling approaches connected with IDEF (Integration Definition Methods) [9] were widely practiced, with the most important being IDEF3. Visual modeling approaches like ARIS (Architecture of Integrated Information

Systems), with its characteristic diagram eEPEC, began to have more impact. This popularity was connected with ARIS integration on the part of the company SAP into the SAP R/3 ERP system. Another popular modeling system is BPMS (Business Process Management System) implemented by BOC into ADONIS business process management software.

The first efforts to unify visual modeling techniques were conducted by the OMG (*Object Management Group*) consortium. Research and design was concentrated on development of UML in the field of business process modeling [1] [4]. With this scope in mind, suitable standard extensions were prepared, the most important of which are Eriksson-Penker Business Extensions [3] and Rational UML Profile for Business Modeling [3]. Another modern approach considered as a candidate for wide adoption is BPMN (*Business Process Modeling Notation*) [5], which, unlike UML, has a very specialized character dedicated to business process modeling.

All the above modeling notations concentrate on modeling processes as a flow of activities with distinguishing decision-making aspects. Some integrate notation, allowing modeling document flow (UML and BPMS), workflow (BPMS) and risk management (UML) but they still do not take into account important management categories distinguished by PMI (Project Management Institute) like: quality, control or communication [2]. These are all especially important in processes connected with project management. Integration of risk management into notation in BPMS shows that organizations responsible for business process modeling are aware of current notation restrictions. Nevertheless, new notation can only be exclusively acknowledged as the initial one on a general stage [9]. Risk and quality are suggested as key aspects of project management [9]. Risk management modeling has been presented in separate articles from the perspective of e-learning projects [10] as a general modeling concept in the form of an extension for modern notations like BPMN and BPMS [10]. According to The Bull Survey, one of the major causes of project failure is lack of or poor quality control [6]. In 35% of projects analyzed, weak, improper quality management was identified as a failure criterion. Only bad team communication and lack of planning were rated higher. Even though quality management is treated as integral part of project management it is omitted in modeling of process. Lack of integrating complex specifications of quality management for activities often causes a cursory quality management in projects. This indicates the importance of proper, systematic and unified quality management of business and project processes with appropriate support from modeling notation in designing process specifications. None of the modeling approaches presented here contains diagrams for managing quality of processes. It is an important factor in the lack of integration of quality management in models for business and project processes, not to mention the quality-oriented modeling of processes.

2 Quality management modeling of business process approach in IS projects

Using the quality-oriented modeling approach to manage IS projects, with a strong emphasis on integrating it with business process models, should be an important element in eliminating project failures due to no or poor quality control. Quality management is meant accordingly to PMI organization as degree, in which a set of inherent characteristics provides the imposed requirements. Lack of any notation for complex, universal quality management modeling with a close connection to business models or project processes demanded the elaboration of author notation from scratch. Moreover in future it could be implemented as an additional diagram type in popular modeling approaches like UML or BPML and integrated with diagrams like activities diagram (UML) and business process diagram (BPML).

As a key element of quality management modeling, quality management diagrams were assumed to refer models of processes. To achieve such a result dedicated stereotypes for business processes diagrams were elaborated (table 1).

Table 1. Notation stereotypes used for identifying management categories

Symbol	Description
👷	quality management
⚠	risk management
👥	communication management
$	cost management
⏱	time management
🔍	control management
🛠	resource management
📄↓↑	document flow management

As solution to integrate quality management modeling with processes, stereotypes for activities were used. Such integration of quality management is possible for the BPMS by assigning the correct symbols to activity objects. In this way, the correctly ascribed icons fulfill the role of stereotypes for activity objects thus extending their meaning. Such an approach permitted both the integration of quality management modeling with processes models and quick reference in locating detailed quality control diagrams connected with particular processes. Fig. 1 presents such integration for the process requirement analysis of e-learning course development which is the starting process for e-learning course development and implementation project. Integration of quality management stereotypes also allows for verification whether the number of activities exists that contain actions for ensuring the requirements of intermediate components and the final product. In e-learning projects such elements are: course outline, e-learning course script, e-learning course instructional design, multimedia objects, authoring tool components and, as a final product, e-learning course implementation packages. The elaborated modeling approach thus supports quality-oriented modeling of business and project processes.

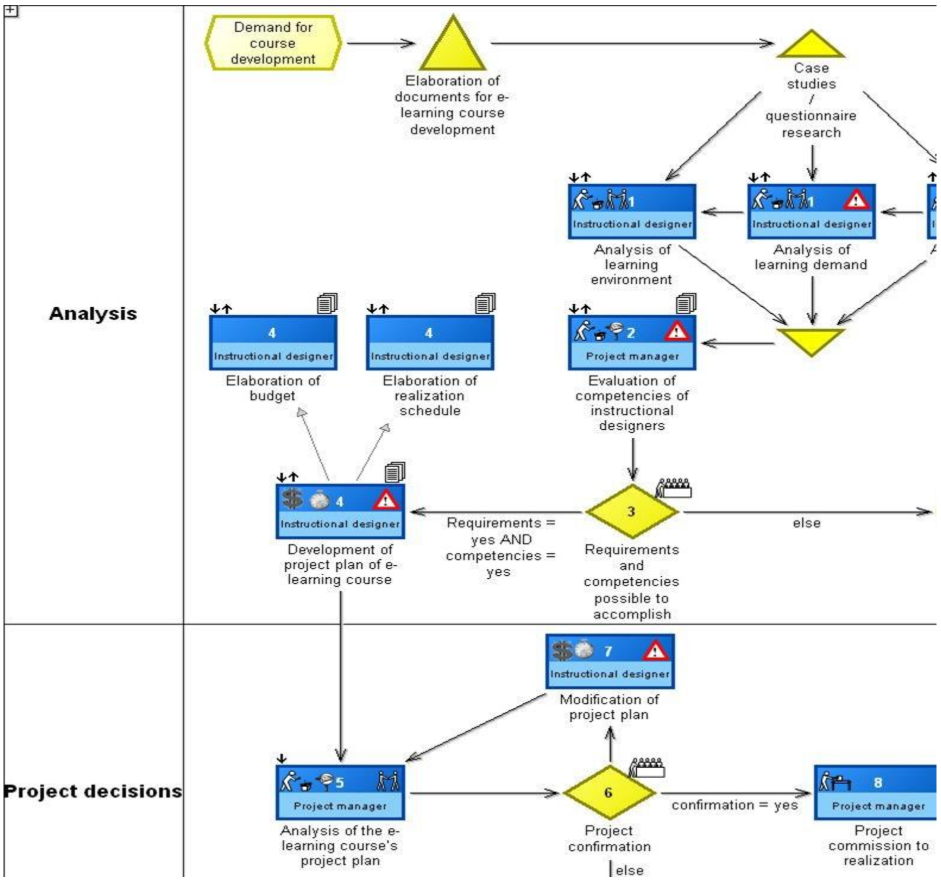
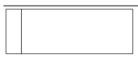
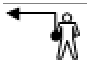
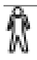
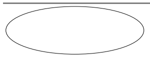
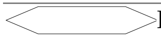
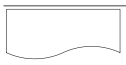
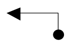
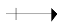
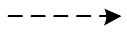

Analysis
Project decisions
Demand for course development
Elaboration of documents for e-learning course development
Case studies / questionnaire research
Instructional designer
Analysis of learning environment
Analysis of learning demand
Elaboration of budget
Elaboration of realization schedule
Project manager
Evaluation of competencies of instructional designers
Development of project plan of e-learning course
Requirements = yes AND competencies = yes
Requirements and competencies possible to accomplish
else
Modification of project plan
Analysis of the e-learning course's project plan
Project confirmation
confirmation = yes
Project commission to realization
else

Fig. 1. Process model with integrated quality management – requirements analysis of e-learning course development

The modeling approach used for the integration of quality management with process models was also used for other important project management categories such as risk, control, costs, time, resources, document flow and communication. The notation was developed based on the present author's experience. As a new concept it required appropriate verification that is presented in the third part of the article. Quality management modeling is based on a dedicated diagram consisting of the visual notation elements presented in table 2.

Table 2. Notation for quality management modeling of business processes

Symbol	Name	Description
	Activity	Activity connected with quality management and related by name to the business or project process model.
	Coordinator	Person responsible for actions of quality management within the framework of the activity.
	Participant	Role responsible for supporting actions of quality management.
	Action	Action undertaken for quality management of the activity.
	Method	Quality management method applied during activity implementation.
	Result	An element, usually a document, which is the result of quality management for the action.
	Manages	Connector that assigns the coordinator to the activity for which tasks related to quality management are executed.
	Participates	Connector that indicates the roles involved in the actions of quality management for the activity.
	Process route	Path of quality management realization for the process.

The aim of the modeling notation showed in table 2 is not only to support specifying quality management for business processes. It concentrates also on stimulating quality management thinking about processes. Thus starting modeling element in notation is activity that forces analysis what operations should have quality management and what activities connected with quality management are missing in processes. Later with the use of notation elements like coordinator and participants proper roles for running quality actions should be specified. Finally formalization has to include methods of quality management execution with distinguishing results. Thus the approach assumes diagram development from bottom (activities and coordinators) to up (methods and results).

Elaborated modeling notation was used during the development of a complex, integrated project management model for the development and implementation of e-learning projects. A sample quality management diagram elaborated with the use of the notation developed for the process of the elaboration of e-learning course script is presented in fig. 2. Quality management is mainly connected with activities related to assessment, training, consultation and verification. Thus, quality management in the process of the elaboration of an e-learning course script refers to activities such as course outline evaluation in accordance with requirement analysis or consultation of methodology for preparing an e-learning course script. The aim of these activities is to provide the script content best suited to the requirements of participants and to the specificity of adopting and running e-learning courses.

Each activity has at least one action identified in the field of evaluation or monitoring, such as comparing the outline of the e-learning course with requirement analysis. For each action, input elements were identified, usually in the form of documents to be reviewed or consulted. For those activities highlighted and dedicated to quality management for realizing e-learning projects, the following are also defined: alternative methods of quality management execution, coordinator as the role responsible for supervising activity implementation and, lastly, participants supporting the coordinator. Actions are accompanied by results, mainly in the form of previously developed and improved components or as a list of changes, such as the requirements analysis for the activity of requirements' analysis of e-learning course.

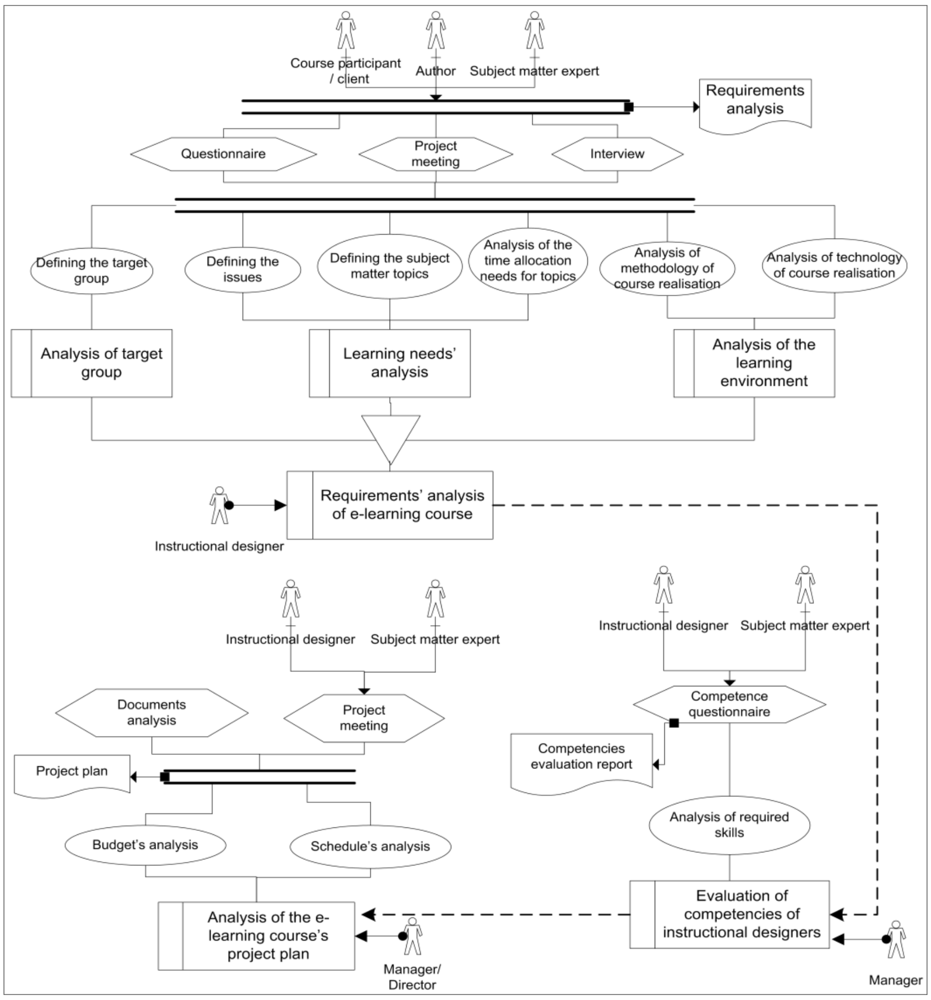

Course participant / client
Author
Subject matter expert
Requirements analysis
Questionnaire
Project meeting
Interview
Defining the target group
Defining the issues
Defining the subject matter topics
Analysis of the time allocation needs for topics
Analysis of methodology of course realisation
Analysis of technology of course realisation
Analysis of target group
Learning needs' analysis
Analysis of the learning environment
Requirements' analysis of e-learning course
Instructional designer
Instructional designer
Subject matter expert
Instructional designer
Subject matter expert
Documents analysis
Project meeting
Competence questionnaire
Competencies evaluation report
Project plan
Analysis of required skills
Budget's analysis
Schedule's analysis
Analysis of the e-learning course's project plan
Manager/ Director
Evaluation of competencies of instructional designers
Manager

Fig. 2. Quality management diagram for the process of requirements analysis of e-learning course development elaboration

Appropriate quality management diagrams were elaborated for all processes outlined in the e-learning course: requirement analysis, elaboration of script, instructional design, production, implementation and evaluation, and revision (periodical).

Elaborated notation allows developing diagrams that visualize, in a complex manner, the specification of quality management for a sequence of activities identified in processes flow diagrams. It needs verification of assumptions that elaborated quality-oriented modeling for business and project processes supports effective IS project realization. Such evaluation is presented in the third part of the article.

3 Evaluation of an elaborated quality process management modeling approach for processes

Proving the usability of an elaborated quality-oriented modeling approach for processes was only possible by implementing it for specifying processes of real-word projects. Such approach was carried for developing model for managing e-learning projects. It occurred that none additional notation elements, neither modifications were required for modeling quality of processes in projects. It indicated the completeness of elaborated quality-oriented modeling approach and its usability.

Second evaluated element of developed quality-oriented modeling approach was whether it assists in distinguishing of important quality management activities during designing project management models. Therefore verification of impact on quality management of developed model for e-learning projects was conducted in two areas:

- compliance of projects with budgets and schedules by comparing actual and planned schedules and budgets;
- time required for correcting errors connected with design and production processes.

Verification was conducted for six projects of e-learning course development and implementation. For three projects, management was based on the elaborated quality-oriented modeling of processes presented in the second part of this paper. Other projects were based on general project management model developed by PEUG.

Verification of project compliance with budgets and schedules was checked by a comparison of planned and real data. For projects based on project manager's experience and the basic model, serious diffractions were noticed both in the initial stage and in the duration of the process. Detailed analysis has revealed that the main reason for project failure was connected with incorrect quality management. For the design process, they related to lack of substantial consultations for designing multimedia objects and to the evaluation of a list of planned multimedia objects with their initial concepts. E-learning courses' projects realized with the use of a quality-oriented modeling of business processes approach had only minor diffractions from the schedule and were nearly executed precisely as planned.

Quality-oriented modeling of business processes was also evaluated in the field of time requirements for correcting errors connected with the design and production processes, through analysis of quantitative data collected during e-learning project realization. Fault monitoring was carried out on multimedia objects as they are the most labor-intensive element. The analysis conducted explicitly indicates that the use of an elaborated model positively impacts on the complexity of executing corrective actions towards eliminating errors in multimedia objects.

Results of verification of e-learning project model based on elaborated modeling approach versus general project management justify its usability as a complex solution for designing quality management of processes. They highlight its positive impact on developing models that improve project management especially in the field of quality management.

4 Summary and conclusions

The present study showed the concept of quality management modeling of business processes in IS projects. As a starting point, a review of modeling notations and systems was offered with an outline of their weaknesses. Analysis proved that none of the modeling approaches presented contained diagrams for managing quality in projects. This meant that the integration of quality management with business processes was also omitted. Lack of available modeling approaches for quality management, taken in conjunction with the failure to include aspects associated with them, has required the development of the author notation presented in the second part of the article. Elaborated notation was used in modeling processes. One of the key factors was appropriate integration of quality management with business processes, which was achieved by using stereotypes for activity objects. An elaborated, integrated modeling approach was used in development of a quality-oriented model for managing the processes of an e-learning project.

The article concluded with an assessment of proposed quality management modeling approach. Model verification was conducted by comparing e-learning projects that used elaborated model with those that did not. The results of the research confirmed completeness of proposed modeling approach. They also confirmed that the use of elaborated quality modeling notation allows for better designing of processes, supporting running projects more accordingly to schedules and budgets. It is necessary to point out that analogous verification should be conducted for other than e-learning projects to fully verify modeling approach usability and capabilities.

5 References

1. Ambler W., *The Elements of UML 2.0 Style*, Cambridge University Press, Cambridge 2005
2. Baca C., *Project Manager's Spotlight on Change Management*, Sybex, Almeda 2005
3. Eriksson H., Penker M., *Business Modeling with UML: Business Patterns at Work*, Wiley, New York 2000
4. Eriksson H., Penker M., Lyons B., Fado D., *UML 2 Toolkit*, OMG Press, Indianapolis 2004
5. Harrison-Broninski K., *The Future of BPM. Part 2*, http://www.bptrends.com/publicationfiles/09-06-ART-FutureBPM20f6-Harrison-Broninski.pdf, 12.08.2011

6. http://www.it-cortex.com/Stat_Failure_Cause.htm#The Bull Survey 1998
7. Johnston S., *Rational UML Profile for Business Modeling*, http://www.ibm.com/developerworks/rational/library/5167.html
8. Kalnins A., Barzdins J., Podnieks K., *Modeling Languages and Tools: State of the Art, Proceedings of 2nd International Conference on Simulation, Gaming, Training and Business Process Reengineering*, 2000
9. Kuciapski M., *Model for Project Management for Development and Implementation of E-Learning Courses*, Forbig P., Gunther H., Perspectives in Business Informatics Research, Proceedings of 9th International Conference, BIR 2010, Springer, Rostock 2010
10. Kuciapski M., *Risk Management in E-learning Projects of Courses Development and Implementation*, Studies and Materials of Polish Society of Knowledge Management, Congress of Young Scientists, ISBN 978-83-7518-245-3, Międzyzdroje 2010

Suitability of Active Rules for Model Transformation

(Demo paper)

Lingzhe Liu* and Manfred A. Jeusfeld+

* lliu@rsm.nl, Erasmus University, Rotterdam, The Netherlands

+ manfred.jeusfeld@acm.org, Tilburg University, The Netherlands

Abstract. Model transformation promises faster and higher-quality system development processes by automating certain development steps. There are numerous proposals such as QVT and triple graph grammars that are applied in practical design environments. To our surprise, active rule systems have not yet been considered as a platform to execute model transformations. We investigate in this paper the suitability of active rule systems via a case study. An empirical analysis of the complexity is provided as well. It turns out that active rules support the basic functional requirements but some extensions to their execution engine and join order optimization would be needed.

Keywords. Model-driven architecture, model transformation, active rule

1 Introduction

The model-driven architecture aims at lifting software development to a higher, more abstract level [KWB2003]. If mappings exists that translate from a more abstract model towards a more concrete one, then the effort would be shifted from coding towards modeling.

To realize the vision, models for all abstractions levels (specification, design, implementation) are represented as instances of meta models, in particular MOF-based. The state of the art of model transformation is materialized in tools based on QVT, and on triple graph grammars (TGGs). Surprisingly, there was so far no attempt to use active rules for the task of model transformation. The models can be represented in an active database, and active rules can encode the transformations between models. In particular, fine-grained incremental updates can be supported.

In this paper, we investigate in more detail the suitability of active rules for the task model transformation. In the subsequent chapter, we study the state of the art in model transformation and establish the requirements that a solution based on active rules should fulfill. Afterwards, a case study on realizing a UML-to-Relational-Database mapping is presented.

2 State of the Art

Model transformation has a language aspect and a tool aspect [CH2003,CH2006]. The *transformation language* encodes the specification of the model transformation, i.e. what elements should be transformed. The *transformation tool* provides the engine interpreting the specification, i.e. realizes how the transformation is performed.

Model-to-code transformation can be regarded as a special case of *model-to-model* transformations, since models can be explicated as a linear textual representation. *Declarative* approaches are contrasted to *imperative* approaches. The latter describe the steps of transforming source models to target models, while the former describe the relations between elements of the source and target models. Imperative approaches hence amalgamate the tooling aspect with the language aspect. We pick two declarative transformation languages to extract concrete requirements for model transformation: QVT (query-view-transformation) and TGG (triple graph grammars). Both languages are rule-oriented, .i.e. the transformation is specified by a set of transformation rules.

2.1 QVT-Core

QVT [QVT2009,Ecl2011,JK2006,XLH*2007,Kur2008] comes in three flavors, QVT-Relations (defining transformations as a set of relations), QVT-Core (defining the semantics of QVT-Relations with a simpler set of language construct), and QVT-Operational (imperative flavor of QVT). We concentrate subsequently on QVT-Core.

QVT-Core is *multi-directional* in nature. The same rule can be read in both directions, but only if the underlying logic of the transformation is the same. Modularity is supported by defining modules containing transformations, which themselves contain the individual mappings. The smaller parts inherit context settings from the larger parts. A transformation rule in QVT-Core distinguishes three areas: the left hand side (source model), the right-hand side (target model), the middle area (traceability objects represented as ordinary model elements). The latter maintain the dependencies between the generated elements of the target model and the elements of the source model(s). The transformation rule makes a test ("guard pattern") on all three areas, checking which elements exists, and demands in its "bottom pattern", which elements in the target and middle areas shall be generated for a given element on the source side ("realized variables"). Variables are bound to elements that may stem from different models.

QVT-Core can specialize transformation rules via a refinement feature. It inherits all mappings from the refines rule that are not overruled or removed. Moreover, there is a nesting mechanism which binds variables at the higher level that are then used at the nested levels.

2.2 TGG

Triple-graph grammars [KS2006] extend graph grammars by a middle graph that basically represents the traceability objects between a left-hand side (LHS) and the right hand side (RHS). Both the LHS and the RHS state dependencies between elements of the source model(s) and elements of the target model(s). The LHS represents the current state of the transformation, i.e. the pre-condition. The RHS declares which elements are present after application of the rule, adding new traceability objects and new objects in the target model, and possibly also in the source model. Besides the test on (non-) existence of nodes and links, TGGs also support cardinality pattern, e.g., that a node has exactly n others nodes linked to it.

Since TGGs are by nature non-deterministic, the actual decision on rule execution is done by the transformation tool [Agra2003]. Round-trip transformation with graph grammars is supported by the Fujuba tool [GZ2005].

3 Case Study: UML-RDBMS

The complexity of the modeling language leads to complex specifications of the model transformation. The purpose of this section is to find out whether ECA rules scale for realistic model transformations, both in terms of specification complexity (here: size of the specification) and the execution time. We use the ECA rule implementation of ConceptBase for both purposes. It is in principle Turing-complete and allows to represent models of many different modeling languages due to its metamodeling facility.

The QVT-Core example transformation UML-RDBMS [OMG 2009] is the benchmark transformation. It consists of 366 lines of QVT-Core code to transform a (simplified) UML class diagram to a relational schema including primary and foreign key specifications. As QVT also employs mapping objects, they form the basis for defining the ECA rules[1]. For example, consider the QVT rule `packageToSchema`:

```
map packageToSchema in umlRdbms {
        uml ()   { p:Package }
        rdbms () { s:Schema  }
        where () {
            p2s:PackageToSchema|
            p2s.umlPackage = p;
```

1 A detailed presentation of the representation of the QVT-style mapping with ECA rules is given in [Liu2010]. The full example including all ECA rules is online at http://merkur.informatik.rwth-aachen.de/pub/bscw.cgi/3015942.

```
                p2s.schema = s; }
            map { where () {
                  p2s.name := p.name;
                  p2s.name := s.name;
                  p.name := p2s.name;
                  s.name := p2s.name; } } }
```

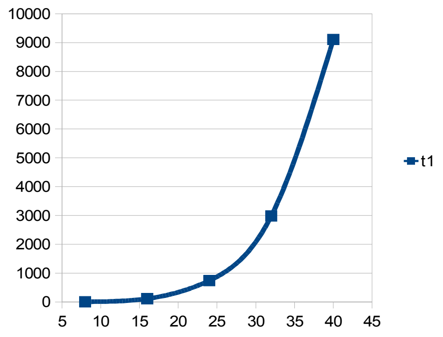
Its representation as an ECA rules for the direction towards RDMS is:

```
UnmatchedPackage in QueryClass isA UPackage with
  constraint
    c1: $ exists name1/String
    (~this name name1) and
    (not (~this matchable FALSE)) and
        (not exists p2s1/PackageToSchema
           (p2s1 umlPackage ~this) and
           (p2s1 name name1)
        )$
end

create_p2s_tr_s in ECArule with
  mode m: Deferred
  inTrans intrans : umlRdbms
  nonMappingRuleFlag mf: FALSE
  exedirn exedir2: SimpleRdbms {* Mapping Direction *}
  ecarule
    mr_p2s_s : $ p/UPackage
                 name1/String p2s1/PackageToSchema
                 tr/Transformation
    ON Ask exeTrans[tr/trans] {* top level mapping *}
    IF NEW (p in UPackage) and
       (p in UnmatchedPackage) and
       (p name name1)
    DO CALL CreateIndividual(P2S,p2s1),
       Tell (p2s1 in PackageToSchema),
       Tell (p2s1 name name1),
       Tell (p2s1 umlPackage p)
    $
end
```

The first clause `UnmatchedPackage` defines an auxiliary query checking whether the source model has not yet been mapped. The ECA rule `create_p2s_tr_s` uses the tag '`IF NEW`' to indicate that the condition is tested against the new database state, i.e. including the updates done by previous ECA rule executions. The translation of the QVT-Core specification UML-RDBMS to ECA rules required 1504 lines of code. This includes about 400 lines of code for the specification of the meta models of UML class diagrams and RDBMS. It should be noted that the QVT-Core rules are bi-directional. Hence, at least two ECA rules had to be coded per QVT-Core rule. Still, the ECA rule coding is about 4 times longer. Of the 34 ECA rules, 16 are in mode 'Deferred' and 18 in mode 'Immediate'. Additionally, 38 query classes are defined to check the structure of the source model and the current state of the model transformation. In combination, the ECA rules support both mapping directions.

To check the performance of the ECA rule representation, we created five UML class diagrams with 8, 16, 24, 32, and 40 types (classes and associations) and measured the transformation times. In the first run t1 (see chart on the right), the conditions of the ECA rules where evaluated in the order in which they occurred.



The X- axis represents the input model size, the Y-axis is the transformation time in seconds. This is almost an intractable performance with a polynomial regression function close to O(n^4). The reason is the lack of query optimization on the conditions of the ECA rules. Hence, we included several optimizations strategies in the ECA system of ConceptBase leading to the following execution times:

The best variant t6 is almost linear and orders of magnitude better than variant t1! It has a small quadratic factor that is due to the 'IF NEW' construct. ConceptBase uses tabling for evaluating derived predicates and queries. The tabled extensions of the predicates speed up the computation. However, when the condition is evaluated against the new database state, then the old extension is no longer valid and has to be re-computed. This re-computation happens for each ECA rule execution. One can expect that an incremental update of the tabled predicate extensions would remove the small quadratic factor from the transformation time. The transformation of an input model of size 40 requires about 7.5 seconds on a 2.4 GHz CPU. A considerable portion is due to the relatively expensive Tell operation of ConceptBase. It maintains several indexes to store facts and each Tell includes transformations from names to identifiers, and from identifiers to memory addresses.

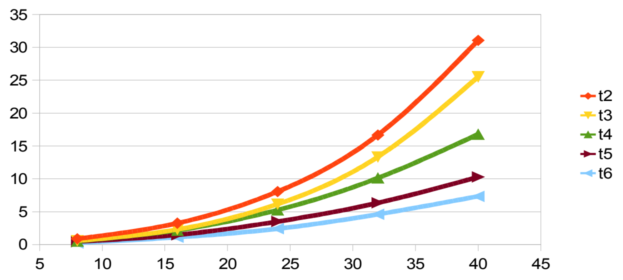


Table 1: Comparison of the three approaches

Category	Criteria	Transformation Approach		
		QVT-Core	**TGGs**	**ECA rules**
Syntax & Expressiveness (Language aspect)	scoping	guard patterns	LHS	On-part and IF-part
	pattern for source domains	bottom patterns of source domains	LHS	IF-part
	pattern for checking target domains	bottom patterns of target domains	RHS (read w/ attribute constraints)	IF-part, QueryClass
	pattern for enforcing target domains	bottom patterns of target domains	RHS (write w/ attribute assignments)	DO-part
	specification of constraint and assignment	logical spec., assignments as constraint in check mode	graphical spec., w/ simple constraints and assignments	IF-part, assignments in DO-part
	directions	bi-directional	bi-directional	unidirectional
	modularity	module, transformation and map	N/A	limited, meta model
	reuse	refinement of mappings	reusable node (in Fujaba)	reuse of post-condition
	composition	nested mapping	graphic composition	reuse of post-condition
	n:1(wrt. elements)	supported	supported	supported
	n:m(wrt. elements)	supported	supported	simulate
	N:1 (wrt. models)	supported	supported	meta model
	N:M(wrt. models)	simulate	supported	meta model
	logical constraints	supported	partially supported	supported
Execution semantics (Tooling aspect)	execution condition	N/A, mappings with valid matches are always executed	block (in [AKS2003])	1. IF-part 2. `active` attribute of `ECArule`
	location determinism	non-deterministic, need tool support	non-deterministic, need tool support	deterministic, pattern matching begins with the triggering element

4 Conclusions

The purpose of this paper was not to present yet another model transformation approach. We rather explored the suitability of existing active rule systems to *implement* model transformations. This was not investigated before. We argue that active rules are a quite natural platform for model transformation. The main result is that active rules are suitable for the task. The coding overhead is manageable, and there is practically no performance penalty. From the viewpoint of active rules, the modularity and bi-directionality are shortcomings that need to be addressed. We believe that modularity can be defined with a suitable meta model. Bi-directionality requires a code generation approach, e.g. generating ECA rules from a QVT-Core specification. This is in close reach as the case study indicates.

Active rules provide more expressive power than the other approaches, e.g. for analyzing the source/target models and the current state of the transformation. The generation of ECA rules from the more abstract QVT or TGG specification is subject to future work. Further research shall also focus on properties like termination and confluence of the ECA rules, and the use of metrics in mapping rules.

References

[AH1988] S. Abiteboul, R. Hull. Data Functions, Datalog and Negation (Extended Abstract). Proc. ACM SIGMOD, Chicago, USA, 143-153.

[AKS2003] A. Agrawal, G. Karsai, G., F. Shi. Graph Transformations on Domain-Specific Models. Technical Report, Vanderbilt Univ., Nov. 2003.

[AWH1995] A. Aiken, J. Widom, M. Hellerstein. Static analysis techniques for predicting the behavior of active database rules. *ACM TODS* 20(1):3-4, 1995.

[AFN2010] M. Arenas, R. Fagin, A. Nash. Composition with target constraints. Proc. ICDT'2010, Lausanne, Switzerland, March 20-22, 2010, 129-142.

[CLST2010] D. Calegari, C. Luna, N. Szasz, A. Tasistro. A Type-Theoretic Framework for Certified Model Transformations. Proceedings of SBMF'2010, Springer LNCS 6527, 112-127, 2010.

[CH2003] S. Czarnecki, S. Helsen. Classification of Model Transformation Approaches OOPSLA'03 Workshop on Generative Techniques in the Context of Model-Driven Architecture.

[CH2006] S. Czarnecki, S. Helsen. Feature-based survey of model transformation approaches. *IBM Syst. J.*, 45(3):621-645, 2006.

[DGG1995] K.R. Dittrich, S. Gatziu, A. Geppert: The Active Database Management System Manifesto: A Rulebase of ADBMS Features. Springer, *LNCS* 985, Springer 1995, ISBN 3-540-60365-4, 3-20.

[Ecl2011] Eclipse Foundation. Model 2 Model (M2M). Online http://www.eclipse.org/m2m/.

[GZ2005] H. Giese, A. Zündorf (eds.). Fujuba Days 2005. Proceedings, Paderborn, Germany, 2005.

[KWB2003] A. Kleppe, J. Warmer, W. Bast. MDA Explained: The Model Driven Architecture: Practice and Promise. Addison-Wesley:Boston, 2003.

[Jeu2009] M.A, Jeusfeld. Metamodeling and method engineering with ConceptBase. In Jeusfeld, M.A., Jarke, M., Mylopoulos, J. (eds): Metamodeling for Method Engineering, 89-168. The MIT Press, 2009.

[JK2006] F. Jouault, I. Kurtev. On the architectural alignment of ATL and QVT. Proceedings ACM Symposium on Applied Computing, Dijon, France, April 23-27, 2006, 1188-1195.

[Kur2008] I. Kurtev. State of the Art of QVT : A Model Transformation Language Standard. *Data Engineering* 5088(ii): 377-393.

[Liu2010] L. Liu. Active Rules for Model Transformation. Master thesis. Tilburg University, The Netherlands, 2010.

[Nic1982] J.-M. Nicolas. Logic for Improving Integrity Checking in Relational Data Bases. Acta Inf. 18:227-253, 1982.

[QVT2009] Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification Version 1.1, Object Management Group, 2009.

[XLH*2007] Y. Xiong, D. Liu, Z. Hu, H. Zhao, M. Takeichi, H. Mei. Towards automatic model synchronization from model transformations. Proceedings 22nd IEEE/ACM Intl. Conf. on Automated Software Engineering., 2007.

- - -

This paper is a companion paper for the demonstration of the active-rule based model transformation at the CAiSE 2012 Forum. The active rules of the case study are available from

http://merkur.informatik.rwth-aachen.de/pub/bscw.cgi/3015942

They can be executed with the ConceptBase system available from

http://conceptbase.cc.

Enterprise Modelling in Distributed Teams – Lessons Learned from Information Demand Modelling

Magnus Lundqvist[1], Kurt Sandkuhl[1,2], Ulf Seigerroth[1]

[1]School of Engineering at Jönköping University,
P.O. Box 1026, 55111 Jönköping, Sweden
[Magnus.Lundqvist, Ulf.Seigerroth]@jth.hj.se

[2]Rostock University, Institute of Computer Science
Albert-Einstein-Str. 22, 18059 Rostock, Germany
Kurt.Sandkuhl@uni-rostock.de

Abstract. Traditionally, enterprise modelling has one of its application areas in the context of improving business practice and management. In such improvement situations an important dimension is optimized information flow in organisations, i.e. to be able to provide the information required to complete organisational tasks. In order to systematically capture and analyse information demand in enterprises, a method for information demand analysis has been developed. The subject of this paper is the use of this method in distributed teams of modellers. This requires transfer of method knowledge to the modellers, coordination of its application, systematic evaluation of lessons learned, and collection of change proposals for the method. The aim is to report on the process of method knowledge transfer and usage including the lessons learned and implications on the used method.

Keywords: Enterprise modelling, information demand modelling, method knowledge transfer, method development, practice of modelling

1 Introduction

Enterprise modelling, enterprise architecture, and business process management are three areas that for a long time have been part of a tradition where the mission is to improve business practice and management [1]. There are close relations between these areas and the information systems field in the aim to improve organisations [2]. This improvement process often involves activities such as understanding and evaluating the current situation of a business and then developing and implementing new ways of working [3]. In this context, business process management and enterprise modelling often are used and applied as techniques to perform a business diagnosis, often referred to as modelling and analysing the AS-IS situation, and to develop and implement improvements, often referred to as the TO-BE situation [3, 8]. An important aspect of business diagnosis is the information flow within organisations, i.e. to analyse whether all organisational roles receive the information required for performing their work tasks and fulfilling their responsibilities.

One of the prerequisites for efficient business diagnosis and enterprise modelling projects is the use of a well-defined method [5], which guides the modelling procedure, defines the notation to be used for capturing modelling results, helps to identify important concepts and viewpoints, and supports identifying relevant stakeholders [6.7]. This paper addresses method use in distributed teams and in particular it focuses on experiences from the transfer of method knowledge. The experiences presented originate from the use of a method for information demand analysis (IDA), i.e. a method for analysing the information demand of organisational roles as a part of the information flow analysis [4]. This method was used in distributed teams of modellers, which required transfer of method knowledge to the modellers, coordination of its application, systematic evaluation of lessons learned, and collection of change proposals for the method. The aim of this paper is to report on the process of method knowledge transfer and usage including the lessons learned and method implications. The contributions of this paper are (1) an industrial case illustrating method use in distributed teams, (2) lessons learned from the process of transferring method knowledge, and (3) implications for the method as such regarding alignment between different models-on-plastic and electronic models.

The remaining part of the paper is structured as follows: Section 2 introduces the process and industrial cases of information demand modelling in distributed teams. Section 3 describes and discusses the lessons learned and method implications derived. Section 4 summarizes our work and describes future activities.

2 Information Demand Modelling in Distributed Teams

This section describes the context of information demand modelling in distributed teams forming the basis for lessons learned and experiences presented in this paper. This context includes an industrial case (section 2.2) and the process coordinating the modelling work (section 2.1).

2.1 Process of Modelling in Distributed Teams

The context for using the IDA method was the infoFLOW-2 project, which aims at improving information flow in small and medium-sized enterprises (SME) and has a runtime from 2010 - 2012. One of the main intentions of the project is to investigate, whether information demand-centric thinking can have advantages compared to process-centric thinking. When solving organizational problems, infoFLOW-2 starts from understanding and modelling the information demands in an organization, instead of modelling the work processes. The project includes two partners from automotive supplier industries, a system integrator specialized on IT-solutions for SME, a public-private partnership in information logistics research, a research institute and a university, responsible for the project management. Additional enterprises are involved on a case basis.

In the preceding project, infoFLOW-1 (2006-2009), the method for information demand analysis was developed. The method is documented in an English and a

Swedish handbook. Both handbooks aim at supporting method use by describing each phase of the method with preconditions, steps to be performed, way of working, expected results and aids, if relevant. Since many participants in the infoFLOW-2 project are native Swedish speakers, the Swedish version was considered an important element to ease method application.

Work in the infoFLOW-2 project, including modelling in distributed teams, was coordinated by infoFLOW-2 project meetings with all project members attending. During the meetings, upcoming cases for information demand modelling were briefly introduced and the decision was made, who should perform and how the case should be performed. During the first part of infoFLOW-2, the basic strategy was to transfer method knowledge by always involving at least one modeller in the case who was part of the method development team, i. e. the handbook was basically considered as accompanying material for the cases in the first infoFLOW-2 phase. The other modellers involved in the cases were supposed to learn the method by observing the experienced modeller and by stepwise getting more responsibility for the case.

Later in the project, we added cases where only the method handbook served as means to provide the method knowledge or where the modelling teams did no longer include one of the initial method developers. In total, we so far performed 4 cases with involvement of method developers, 4 cases without method developers but with modellers who were involved in at least one of the first 4 cases, and 3 cases completely based on handbook use only. These 3 cases were outside the infoFLOW-2 project and using the English handbook version.

For all cases and during all phases of infoFLOW-2, the project meetings served as central coordination unit, i. e. the modelling results, experiences when performing the modelling, and improvement or change requests for the method and the method handbook were discussed during the infoFLOW-2 meeting in the project team and documented in the minutes. The infoFLOW-2 project includes 4 industrial and 2 academic partners. On average the meetings had 10 participants (5 from industry, 5 from academia). Among these 10 were 5 who were involved in the method development and the main method engineer.

2.2 Industrial Case

The research work presented in this paper is motivated by a number of real-world cases, one of them was selected for brief presentation in this section: the SAPSA case.

The SAP Swedish User Association (SAPSA) is a non-profit association for organizations that use the enterprise system SAP. The main purpose with SAPSA is to provide an arena for exchange of knowledge and experiences and networking for SAP stakeholders. SAPSA also aims at taking care of the member's demands for development of SAP software and services and third part products certified by SAPSA. The members have unlimited access to SAPSA´s focus groups (groups with expertise within certain areas) and the annual SAPSA conference. The background for doing enterprise modelling and information demand modelling at SAPSA was an articulated need from SAPSA to elucidate their interaction with the focus groups, members, and other stakeholders. At the time of this case SAPSA experienced some difficulties in how to increase the activity and exchange between SAPSA central, the focus groups, the members, and other stakeholders. The core area for the modelling session therefore addressed the central roles; SAPSA, Focus groups, User companies, SAP consultancies, SAP Sweden, and SAP International. The people that were present at the information demand modelling seminar were, from SAPSA: the CEO, the Event coordinator, the economy administrator, and the secretary from the SAPSA board. From the research project we participated with one researcher. One representative also participated from the industry. The industrial project representative is also the secretary in the SAPSA board.

The actual modelling seminar lasted for five hours including a scoping discussions (framing and setting the scene), the actual modelling, and validating discussions in the end of the seminar. The actual modelling session was divided into two phases. First we modelled the actual situation and how the different roles were interacting today (AS-IS), see fig. 1 below.

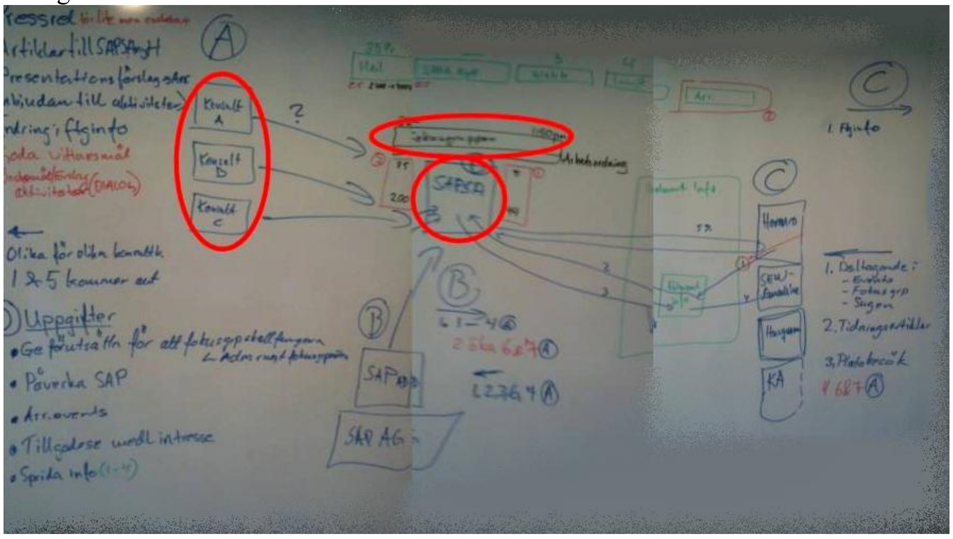


Figure 1: Example of model from the first modelling phase

During this stage we also had an evaluating discussion about the current practice concerning the interaction between the involved roles. This evaluation resulted in a couple of core problems in relation to the exchange and interaction between the specified roles. Based on these problems we started to design a future interaction schema that could solve these problems. One important solution was to employ a new role at SAPSA, a Focus Group Coordinator. After the modelling seminar at SAPSA the models were then transformed into electronic versions, see fig. 2 below.

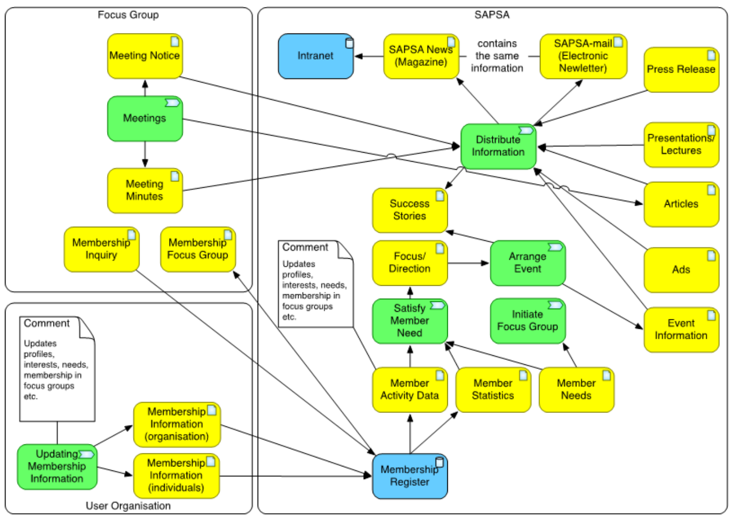


Figure 2: Information demand model of the SAPSA case

As in a couple of other cases, we observed that there are somewhat troubling differences between the models that were developed on site, usually on plastic sheets or on whiteboard, and the models when they are transformed into electronic versions. It seems that the notation rules that are specified in the method for information demand analysis are a bit tricky to translate into the models during the actual modelling sessions. It seems, as there is a need for structural planning of the roles in the models, which is hard to do initially when the model starts to evolve. In some sense you need to have the whole picture before this type of structuring is possible.

3 Lessons Learned and Method Implications

Lessons learned from distributed application of the information demand analysis method are presented in this section. Due to space limitations, we selected just two lessons for discussion: experiences from modelling in distributed teams (3.1), and the necessity to avoid gaps between the electronic and pre-electronic models (3.2).

3.1 Experiences from modelling in distributed teams

The most important lesson learned from the process and coordination of distributed modelling is that more tools and aids supporting the use of the method are required. Although we were in the very fortunate situation to have 6 modellers in the project team who were quite deeply involved in the method development, the transfer of this method to other modellers by joining the team and learning from the experienced ones had a number of problems. The experienced modellers all had the same IDA method steps and a joint perspective on how to perform the modelling in common, but they performed the actual modelling slightly differently, i. e. they did not share the practices of IDA originating from their own backgrounds. Examples of these differences are

- how to perform the scoping (start from an organizational unit or start from a work process?),
- how to layout the models,
- how detailed to document terms and concepts (e. g. the responsibilities of roles), or
- how to schedule the modelling process (e. g. plan, interviews, modelling session and feedback workshop at the beginning of the process or one after the other)

These differences in practice basically are a consequence of the nature of methods, which are supposed to be support and a guideline for action, not a rigid and precise algorithm, since adaptation to situational requirements is considered an element of method success. However, when teaching a new method to modellers, these differences in practice often are perceived as deviations from the recommended method use or as inconsistencies in the method then as supportive practices. For relatively newly developed methods, like in our case the IDA method, variations in the practices should be made explicit by either offering alternative paths in the method descriptions or additional aids, like checklists or textual practice description, complementing the method handbook.

Our recommendation is to develop a training course for the method to be transferred (in our case the IDA method) and to include a detailed example for all steps of the method in the handbook. The training course will help to make the material and aids more detailed, the example will ease understandability of the handbook and help to identity gaps.

3.2 How to avoid gaps between pre electronic and electronic models?

Based on practical experiences from the case presented in section 3, we have recognised what we on a conceptual level would describe as model gaps. These model gaps are closely related to the work procedure that is prescribed in the IDA-method. The normal work procedure in IDA, and many other modelling methods, is that the modelling is performed in two steps. In the first step we do the modelling on big plastic sheets or papers with post-it notes and white-board pens. The reason for this is to be able to do the modelling in an interactive manner together with different stakeholders. The goal is to get the stakeholders to be active in the actual modelling activities in different ways. In the second step the models on the plastic sheet is transformed into digital models in a modelling tool. As a result of this transformation process we have recognised that the differences between the plastic model and the transformed digital model can be of quite some difference. For information demand modelling this transformation process involves a restructuring of the models in terms of role-clustering of tasks and the needed information based on different rules. This clustering process could in this case be regarded as an analysis activity, which probably is not so easy to do during the actual modelling session. This structuring planning of clustered roles in the models is hard to do initially when the model starts to evolve. In some sense you need to have the whole picture (whole models) before this type of structuring is possible. This creates what we have chosen to call conceptual model gaps.

These model gaps are typical examples of alignment deficiencies between different work steps in the method. When we have these types of deficiencies in our models we will have to put down specific efforts into analysing the models as such rather than the specific case, which rather should be the priority. It is therefore important to really address notation issues when we are developing methods for a certain purposes. The notation should therefore be simple enough so that we can devote our modelling efforts to case analysis and not model analysis, i.e. the notation and the notation rules in a method should not require analysing activities in order to produce the model.

In order to identify and implement the implications for the information demand modelling method used in our cases, we have now initiated a method development activity where these alignment issues in the method are treated. Our suggestion for solution, which we already piloted in a number of cases, is to refine the procedural description of how both the initial draft of the model, as produced during seminars and the documentation of those into electronic models are constructed, which is supposed to ensure traceability between the different versions of the model.

More concrete, we started improvement work of the method with the following objectives:

- A more precise correspondence between the "paper-based" symbols used for models-on-plastic and the notation used in the electronic version. For all elements in the notation, a corresponding paper symbol has to be selected and labelled accordingly, e.g. colour, shape and print on the paper symbol have to be defined. This will at least ease the work of translating models-on-plastic to electronic ones.
- A checklist for supporting consistency and completeness of the models-on-plastic. In the modelling team, one member will have the task to assure at the end of the modelling session that the number of quality issues is as low as possible. Aspects to check include relations between model elements or additional textual descriptions of important concepts
- Guidelines for layout of models-on-plastic. Might be useful. However, this aspect has to be investigated carefully because standardizing the layout might hinder the user participation, which would be an unwanted effect.

4 Conclusions and Future Work

Illustrated by an industrial case, this paper presented experiences and lessons learned from transferring method knowledge and performing information demand modelling in distributed teams. These lessons learned mostly concern the practice of demand modelling and of transferring method knowledge. An important lesson learned is that method knowledge transfer "by heads" (i.e. by including persons experienced in the use of the method in a modelling team) does not substitute a good method handbook, since even these "heads" need some sort of normative ground to base their practice on.

Furthermore, a number of implications for the method as such can be derived from the experiences:

- Traceability between models-on-plastic and electronic models has to be improved, e.g. by correspondence between paper-based symbols and method notation, in order to avoid alignment deficiencies between different work steps
- the layout of the information demand models as part of the secondary notation has to be further explored and developed, in order to ease the visualization and an understanding of role – information dependencies.
- more tools and aids supporting the use of the method are required, which make variations in the practice of different modellers explicit by either offering alternative paths in the method descriptions or additional aids, like checklists or textual practice description, complementing the method handbook

Implementation of the above changes and evaluating their effects in the practice of modelling constitutes the future work in this area.

Acknowledgment

The research presented in this paper was supported by the Swedish Knowledge Foundation (KK-Stiftelsen), grant 2009/0257, project "Demand Patterns for Efficient Information Logistics (infoFLOW-2)" and by the Swedish Foundation for Internationalisation in Higher Education and Research (STINT), grant IG2008-2011, project "Development and Evolution of Ontologies".

References

1. Harmon, P. (2010). The Scope and Evolution of Business Process Management. In J. vom Brocke & M. Rosemann (Eds.), Handbook on Business Process Management 1: Introduction, Methods, and Information Systems (pp. 83-106). Berlin and. Heidelberg, Germany: Springer.
2. Seigerroth U. (2011) Enterprise Modelling and Enterprise Architecture – the constituents of transformation and alignment of Business and IT, accepted for publication in International Journal of IT/Business Alignment and Governance (IJITBAG)
3. Hayes, J. (2007). The Theory and Practice of Change Management. Basingstoke, UKNew York, NY: Palgrave Macmillan.
4. Lundqvist M., Sandkuhl K., Seigerroth U. (2011) Modelling Information Demand in an Enterprise Context: Method, Notation, and Lessons Learned. International Journal of Information System Modeling and Design (IJSMD), Vol. 2, Issue 3, pp. 75-94, 2011.
5. Avison, D. E. & Fitzgerald, G. (1995) Information Systems Development: Methodologies, Techniques and Tools. Berkshire, England: McGraw Hill.
6. Ralyté J., Backlund P., Kühn H., Jeusfeld M. A. (2006) Method Chunks for Interoperability, D.W. Embley, A. Olivé, and S. Ram (Eds.): ER 2006, LNCS 4215, pp. 339 – 353, 2006, © Springer-Verlag Berlin Heidelberg.
7. Brinkkemper S. (1995) Method engineering: engineering of information systems development methods and tools, Information and Software Technology, 1995 37.
8. Seigerroth U. (2011) Enterprise Modelling and Enterprise Architecture – the constituents of transformation and alignment of Business and IT, International Journal of IT/Business Alignment and Governance (IJITBAG), Vol. 2, Issue 1, pp 16-34, 2011

VariaMos: a Tool for Product Line Driven Systems Engineering with a Constraint Based Approach

Raúl Mazo[1,2] , Camille Salinesi[1], Daniel Diaz[1]

[1] CRI, Université Paris 1 – Sorbonne, 90, rue de Tolbiac, 75013 Paris, France
[2] Ingeniería de Sistemas, Universidad de Antioquia, Medellín, Colombia

raulmazo@gmail.com, {camille.salinesi, daniel.diaz}@univ-paris1.fr

Abstract. The creation of error-free variability models and their usage in product line analysis and product derivation is central to product line engineering (PLE). The complexity of these tasks makes tool support a success-critical factor. Tools supporting the core activities of PLE are a challenge and a real need for academics, industrial researchers, and practitioners of the PLE domain. In this paper, we present a tool for variability modeling, model integration, verification and analysis, derivation requirements specification and product derivation.

Keywords: Product line engineering, variability, product line models.

1 Introduction

Variability models are used to specify the variability of software product lines. These variability models are represented by means of a modeling formalism. In our literature research, we have found quite a number of variability modeling formalisms, such as FODA (Feature-Oriented Domain Analysis) [9], Orthogonal Variability Models (OVM) [13], UML classes [26], DOPLER [5] and Goals [6]. To represent and reason on these models, a number of approaches and tools exist in the literature. However, there is a lack of methods and tools that can support modeling, integration, reasoning and complex configuration on the Product Line (PL) domain. This lack is more accentuated when the model is composed of a collection of views representing the same product line. In this paper, we present a tool allowing represent, integrate, reason and configure product line models.

The paper is structured as follows: Section 2 gives a brief overview of our tool VariaMos. Section 3 describes some functions of VariaMos. Section 4 presents related tools supporting integration, verification, analysis and configuration of product line models. Section 5 concludes the paper and describes future works.

2 VariaMos Architecture

VariaMos (Variability Models) is an Eclipse plug-in for specification, automatic verification, analysis, configuration and integration of multi-view product line models. From a deployment point of view, VariaMos is an Eclipse plug-in that communicates with our GNU Prolog [3] by means of a socket. The VariaMos tool, its documentation and a video training are available online[1].

3 Functionalities

VariaMos allows working simultaneously on a set of models in multi-formalism mode. There are several activities that VariaMos is intended to support: domain engineering with multiple models, integrated verification of the verification criteria existing in literature [1, 14], analysis [1] and configuration [10, 16]. In additiVn, MariaMos allows creating/editing Product Line Models (PLMs) that have been imported as SPLOT XMI[2] or constraint program text files (cf. Figure 1(a)) and exporting/importing PLMs using a XMI or a constraint program file. This functionality allows communicating models from and to other applications.

3.1 Integration of Variability Models by means of Constraint Programs

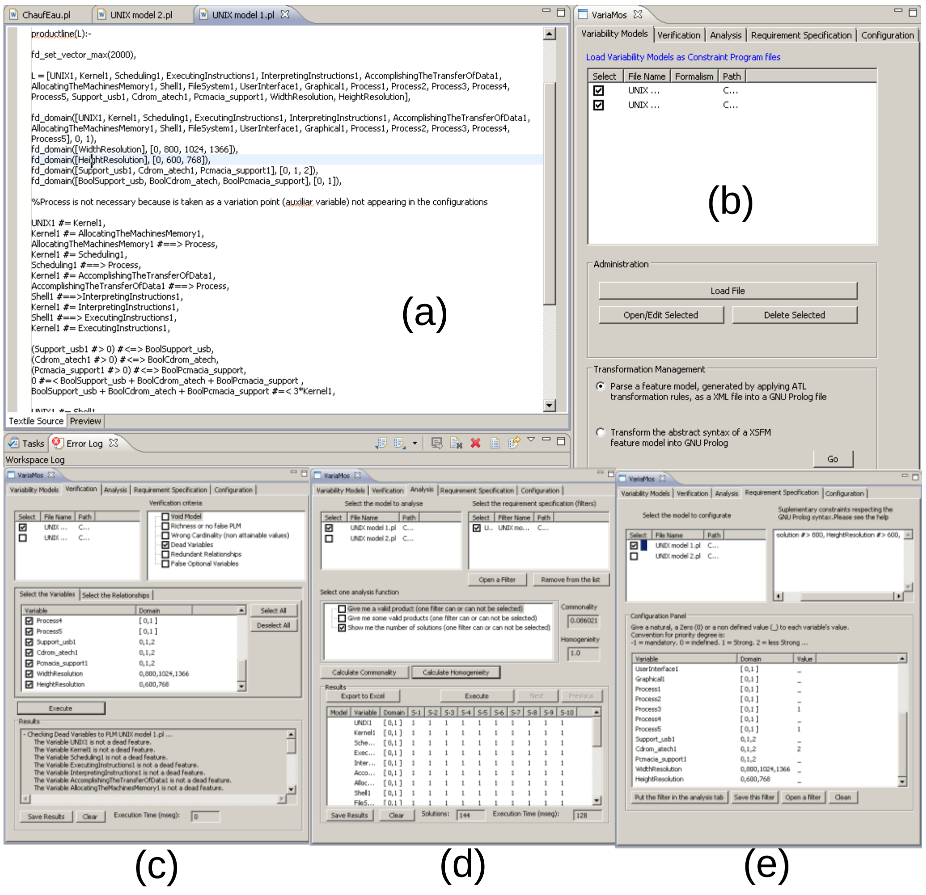
In our approach, each view of the product line system is transformed into a constraint program. A constraint program is a collection of constraints without a specific order. In this way, the constraint programs, representing the different views of the PL system, can be easily integrated into a single constraint program. The resulted constraint program represents the general system and offers a richer view of the PL (than individual views). VariaMos implements the five integration strategies presented by [10]. In our approach, two models' elements referring to the same concept must have the same name; we do not deal with mismatching of names. Mazo et al. [10] offer a list of rules to transform the most popular formalisms to represent variability models into constraint programs. Once each view of the PL system is transformed into CP, they can be integrated in a single constraint program using the graphical user interface presented in Figure 1 (b).

3.2 Verification of Variability Models

VariaMos implements the typology of verification criteria presented in [10]. Using this classification we can detect if the model is void [9], if the model is not a false PLM [1, 14], if the model does not have errors (like dead variables [1, 9, 14] or variables with wrong domains [1, 14], inconsistencies (like full-mandatory features [1] requiring optional features [9]) and redundancies (like full-mandatory variables in-

[1] https://sites.google.com/site/variabilitymodels/home/downloads/PresentationVariaMos2.js
[2] http://www.splot-research.org

cluded by another variable [14] or inclusion of a relative father [14]). A snapshot of the graphical user interface of VariaMos to implement these verification operations is presented in Figure 1 (c).



```
productline(L):-

fd_set_vector_max(2000),

L = [UNIX1, Kernel1, Scheduling1, ExecutingInstructions1, InterpretingInstructions1, AccomplishingTheTransferOfData1,
AllocatingTheMachinesMemory1, Shell1, FileSystem1, UserInterface1, Graphical1, Process1, Process2, Process3, Process4,
Process5, Support_usb1, Cdrom_atech1, Pcmacia_support1, WidthResolution, HeightResolution],

fd_domain([UNIX1, Kernel1, Scheduling1, ExecutingInstructions1, InterpretingInstructions1, AccomplishingTheTransferOfData1,
AllocatingTheMachinesMemory1, Shell1, FileSystem1, UserInterface1, Graphical1, Process1, Process2, Process3, Process4,
Process5], 0, 1),
fd_domain([WidthResolution], [0, 800, 1024, 1366]),
fd_domain([HeightResolution], [0, 600, 768]),
fd_domain([Support_usb1, Cdrom_atech1, Pcmacia_support1], [0, 1, 2]),
fd_domain([BoolSupport_usb, BoolCdrom_atech, BoolPcmacia_support], [0, 1]),

%Process is not necessary because is taken as a variation point (auxiliar variable) not appearing in the configurations

UNIX1 #= Kernel1,
Kernel1 #= AllocatingTheMachinesMemory1,
AllocatingTheMachinesMemory1 #==> Process,
Kernel1 #= Scheduling1,
Scheduling1 #==> Process,
Kernel1 #= AccomplishingTheTransferOfData1,
AccomplishingTheTransferOfData1 #==> Process,
Shell1 #==>InterpretingInstructions1,
Kernel1 #= InterpretingInstructions1,
Shell1 #==> ExecutingInstructions1,
Kernel1 #= ExecutingInstructions1,

(Support_usb1 #> 0) #<=> BoolSupport_usb,
(Cdrom_atech1 #> 0) #<=> BoolCdrom_atech,
(Pcmacia_support1 #> 0) #<=> BoolPcmacia_support,
0 #=< BoolSupport_usb + BoolCdrom_atech + BoolPcmacia_support ,
BoolSupport_usb + BoolCdrom_atech + BoolPcmacia_support #=< 3*Kernel1,
```

Load Variability Models as Constraint Program files

Select	File Name	Path
☑	UNIX ...	C...
☑	UNIX ...	C...

Administration

Load File · Open/Edit Selected · Delete Selected

Transformation Management

[✓ Parse a feature model, generated by applying ATL transformation rules, as a XML file into a GNU Prolog file]

Transform the abstract syntax of a XSFM feature model into GNU Prolog

Go

Verification criteria

Select	File Name	Path
☑	UNIX ...	C...
☐	UNIX ...	C...

- ☐ Void Model
- ☐ Richness or no false PLM
- ☐ Wrong Cardinality (non attainable values)
- ☑ Dead Variables
- ☐ Redundant Relationships
- ☐ False Optional Variables

Select the Variables | Select the Relationships

Variable	Domain
☑ Process4	[0,1]
☑ Process5	[0,1]
☑ Support_usb1	0,1,2
☑ Cdrom_atech1	0,1,2
☑ Pcmacia_support1	0,1,2
☑ WidthResolution	0,800,1024,1366
☑ HeightResolution	0,600,768

Select All · Deselect All · Execute

Results

```
- Checking Dead Variables to PLM UNIX model 1.pl ...
    The Variable UNIX1 is not a dead feature.
    The Variable Kernel1 is not a dead feature.
    The Variable Scheduling1 is not a dead feature.
    The Variable ExecutingInstructions1 is not a dead feature.
    The Variable InterpretingInstructions1 is not a dead feature.
    The Variable AccomplishingTheTransferOfData1 is not a dead feature.
    The Variable AllocatingTheMachinesMemory1 is not a dead feature.
```

Save Results · Clear · Execution Time (mseg): 0

Select the model to analyse

Select	File Name	Path
☑	UNIX model 1.pl	C...
☐	UNIX model 2.pl	C...

Select the requirement specification (filters)

Select	Filter Name	Path
☑ U..	UNIX mo...	C...

Open a Filter · Remove from the list

Select one analysis function

- ☐ Give me a valid product (one filter can or can not be selected)
- ☐ Give me some valid products (one filter can or can not be selected)
- ☑ Show me the number of solutions (one filter can or can not be selected)

Commonality: 0.086021
Homogeneity: 1.0

Calculate Commonality · Calculate Homogeneity

Results

Export to Excel · Execute · Next · Previous

Model	Variable	Domain	S-1	S-2	S-3	S-4	S-5	S-6	S-7	S-8	S-9	S-10
	UNIX1	[0,1]	1	1	1	1	1	1	1	1	1	1
	Kernel1	[0,1]	1	1	1	1	1	1	1	1	1	1
	Sche...	[0,1]	1	1	1	1	1	1	1	1	1	1
	Exec...	[0,1]	1	1	1	1	1	1	1	1	1	1
	Inter...	[0,1]	1	1	1	1	1	1	1	1	1	1
	Acco...	[0,1]	1	1	1	1	1	1	1	1	1	1
	Alloc...	[0,1]	1	1	1	1	1	1	1	1	1	1
	Shell1	[0,1]	1	1	1	1	1	1	1	1	1	1
	FileS...	[0,1]	1	1	1	1	1	1	1	1	1	1

Save Results · Clear · Solutions: 144 · Execution Time (mseg): 128

Select the model to configurate

Select	File Name	Path
☑	UNIX model 1.pl	C...
☐	UNIX model 2.pl	C...

Suplementary constraints respecting the GNU Prolog syntax.Please see the help

```
solution #> 800, HeightResolution #> 600,
```

Configuration Panel

Give a natural, a Zero (0) or a non defined value (_) to each variable's value.
Convention for priority degree is:
-1 = mandatory. 0 = indefined. 1 = Strong. 2 = less Strong ...

Variable	Domain	Value
UserInterface1	[0,1]	_
Graphical1	[0,1]	_
Process1	[0,1]	_
Process2	[0,1]	_
Process3	[0,1]	1
Process4	[0,1]	_
Process5	[0,1]	1
Support_usb1	0,1,2	_
Cdrom_atech1	0,1,2	2
Pcmacia_support1	0,1,2	_
WidthResolution	0,800,1024,1366	_
HeightResolution	0,600,768	_

Put the filter in the analysis tab · Save this filter · Open a filter · Clean

Fig. 1. GUI of VariaMos: (a) Definition/edition of Product Line Models, (b) Integration, (c) verification, (d) analysis and (e) configuration. Fig. 1 in high resolution is available at: https://sites.google.com/site/variabilitymodels/home/downloads/GUIofVariaMos.JPG

3.3 Execution of Analysis Operations

All the analysis operations implemented in VariaMos are taken from literature and from industrial projects with our partners; most of the operations are explained and referenced on the literature review of Benavides et al. [1]. A small description of each analysis operation implemented in VariaMos and how they have been implemented are presented as follows:

1. Calculating the number of valid products represented by the PLM. This operation may be useful for determining the richness of a PLM. VariaMos implements this operation with GNU Prolog in the following way: *g_assign(cpt,0), pl(_), g_inc(cpt), fail;g_read(cpt,N)*, where *pl* is the fact that represents the product line model. With this operationalization we avoid the overload of the RAM with each solution generated and counted by the solver because each time a solution is found, we release the pile of solutions before the generation of a new one.
2. Obtaining the list of *all valid products* represented by the PLM, if any exist. This operation may be useful to compare two product line models. The list of valid product is obtained one by one from the solver by means of the backtracking technique. As the screenshot shows it in Figure 1(d), VariaMos provides users with the possibility to navigate in the list of products using the *Next* and *Previous* buttons.
3. Calculating product line commonality. This is the ratio between the number of products in which the set of variables of the PLM is present and the number of products represented in the PLM. This operation calculates the number of solutions in which all the variables of the PL are present and divides this number with the result obtained with operation 1.
4. Calculating Homogeneity: A more homogeneous PLM would be one with few unique variables in one product (i.e. a unique variable appears only in one product) while a less homogeneous one would be one with a lot of unique variables. By definition `Homogeneity = 1 - (#unicVariables / #products)`. This operation computes the number of variables that appear in only one product by means of a request to the solver and computes the number of products using the operation 1.
5. Calculating variability factor: This operation takes a PLM as input and returns the ratio between the number of products and 2^n where n is the number of variables considered. In particular, 2^n is the potential number of products represented by a PLM, assuming that there are not cross-tree constraints on the model and that all PLM's variables are Boolean. Variability factor = NProd / 2^ NVar. This function uses the solver to compute the number of variables and the number of products in the PLM.
6. Checking validity of a configuration. A configuration is a collection of variables and may be partial or total (e.g., the partial configuration presented in Figure 2(d)). A valid partial configuration is a collection of variables respecting the constraints of the PLM but not necessary representing a valid product. A total configuration is a collection of variables respecting the constraints of a PLM and where no more variables need to be added to form a valid product. This operation may be useful to determine if there are or not contradictions in a collection of variables or to determine whether a given product is available in a product line. To operationalize this function, the configuration to check is considered as a collection of external constraints where each constraint corresponds to the assignation of a particular value to each one of the variables of the PLM. Then, the external constrains and the constraints of the PLM are executed together in the solver to verify if the whole of constraints is consistent (i.e., there is a valid solution satisfying all these constraints).

7. Executing dependency analysis or decision propagation. It looks for all the possible solutions after assigning some fix value to a collection of values and then asking the solver for almost one solution. This operation is very similar to the operation 6, however, with this operation we can check the satisfaction of constraints by means of reification, and not only the satisfaction of variables of the PL as in operation 5.
8. Specifying external requirements specifications for configurations using constraints. This operation allows the specification of constraints that are not constraints of the domain, but configuration constraints. To operationalize this function, external constraints are defined in GNU Prolog and then added to the constraints of the PLM; once added, all the constraints are executed in the solver. See [10] for more details and Figure 1(e) for a snapshot of the implementation of this function in VariaMos.
9. Applying a filter. This operation takes a configuration (i.e., set of variables, each one with a particular value) and a collection of external requirements and returns the set of products which include the input configuration and respect the PLM's constraints and the external constraints. Figure 1(e) presents a snapshot of the GUI of this function in VariaMos.
10. Calculating the number of products after applying a filter. This operation uses the technique presented in operation 1 to compute the number of products that can be configured from a PLM in presence of a filter. A filter is presented as a collection of external constraints and particular assignation of values to the variables of the PL. To operationalize this function, the filter is added to the collection of the PLM's constraints and then executed in the solver. Figure 1(d) presents a snapshot of the GUI of this function in VariaMos.
11. Find an optimal product with respect to a given attribute like cost *(min goal)* and benefit *(max goal).* Detection of "optimal" products is very important for decision makers as presented in [10]. To operationalize this function we use the *fd_maximize* and the *fd_minimize* facts offered by the GNU Prolog solver.

3.4 Other Features

According to [8], a tool for automating reasoning on variability models should be efficient, scalable and with enough expressivity to represent different kinds of variability constraints. These characteristics are evaluated on VariaMos as follows:

Reasoning efficiency. The execution time of each reasoning operation can be calculated by the solver by means of a request for the current time (by means of the prolog function `user_time(T1)`) at the beginning and at the end (by means of the prolog function `user_time(T2)`) of each constraint program. The time spent by the solver to execute the operation at hand, is computed by means of the clause: `T is T2 - T1`. We have showed the reasoning efficiency of VariaMos in several works; for instance: [10, 12, 15] show the efficiency of VariaMos in verification of product line models and [11] shows the efficiency of VariaMos in transforming PLMs.

Scalability. VariaMos scalability has been validated using a corpus of 54 models specified in several languages, representing several domains and with sizes from 9 to 10000 variables. In all these cases, VariaMos shows a promising scalability in the execution of the reasoning operation presented in this paper. The results have been reported in works like [10, 12, 15].

Expressivity. In VariaMos, product line models can be loaded as XMI or text files and then, labeled with it particular notation. VariaMos offers several capabilities to represent and transform different types of product line models into constraint programs. In addition, models can be edited with XML and text editors furnished by Eclipse IDE. The power of expression of VariaMos is compared with the one of constraint programming to specify PLMs [10, 15].

4 Related Works

The most of the tools for supporting product line engineering focus on one or two aspects but not in all of the aspects presented in this paper.

For instance, from the point of view of modeling, there are tools like Feature Plugin[3], XFeature[4], AHEAD Tool Suite[5], Pure::variants[6] and Requiline[7]. The most of these tools were built to graphically construct feature models and to derive products from these models, not to reason on these models.

From the point of view of analysis and verification, most of the tools found in literature are formalism-dependent and they only focus on feature models. In addition, most of them focus on verifying the consistency of a combination of features (a feature configuration) against the feature model. Tools like FAMA[8] and SPLOT[9] consider several analysis and verification operations over feature models; however, they have been targeted in the analysis and verification of models represented by a single view.

From the point of view of expressivity, modeling tools available in the literature are just starting to offer some model-to-model transformation capabilities, but these are still limited and often ad hoc. Some examples of these tools are: Andro-MDA[10], openArchitectureWare[11], Fujaba[12] (From UML to Java And Back Again), Jamda[13] (JAva Model Driven Architecture), JET[14] (Java Emitter Templates), MetaEdit+[15] and Codagen Architect[16]. There are also approaches that do combine multiple variability

[3] http://gp.uwaterloo.ca/fmp
[4] http://www.pnp-software.com/XFeature/
[5] http://www.cs.utexas.edu/~schwartz/ATS/fopdocs/
[6] http://www.software-acumen.com/purevariants/feature-models
[7] http://www-lufgi3.informatik.rwth-aachen.de/TOOLS/requiline
[8] http://www.isa.us.es/fama
[9] http://www.splot-research.org
[10] http://www.andromda.org.
[11] http://www.openarchitectureware.org/
[12] http://www.fujaba.de
[13] http://sourceforge.net/projects/jamda
[14] http://www.eclipse.org/articles/Article-ET/jet_tutorial1.html
[15] http://www.metacase.com/
[16] http://www.codagen.com/products/architect/default.htm

models, e.g., KumbangTools[17] combining the feature and component-based models. However, none of them deals whit transformation of product line models, where the semantic of the model represents not only one but an undefined collection of product models.

From the point of view of configuration, there are several tools in literature that address this topic. For instance, FAMA, SPLOT and FdConfig [16]; however these tools do not support as much reasoning operations over product line models as VariaMos do. In addition, they do not support reasoning operations over multiple PLMs.

5 Conclusions and Future Works

In this paper we introduced the first release of VariaMos which is an Eclipse plug-in for edition, integration, verification, analysis and configuration of PLMs. We introduced the functionalities of the tool and we exposed some of the most relevant design and implementation details. Finally, we showed the differences between VariaMos and other tools found in literature and we concluded that VariaMos supports more variability modeling languages, automatically verifies more criteria than the other tools, and is the first tool to implement reasoning operations over multi-views PLMs. Although VariaMos is not a mature tool yet, its promising capabilities of extensibility, interoperability, scalability, expressivity and efficiency will allow the tool to become accepted and used by the academic and industrial community in the future.

Several challenges remain for our future work. On the one hand, the implementation of more verification and analysis functions. For instance, verification against a meta model defined by users, incorporation of a guided process allowing correcting anomalies and support incorporation for incremental verification are envisaged for future releases. On the other hand, it is planned to incorporate, in our tool, a graphical representation of constraint programs, automation of PLM construction from a collection of products models, multi-stage configuration of products from complex requirements formulated as constraint programs and also connection with other kind of solvers; e.g., SAT (SATisfiability), BDDs (Binary Decision Diagrams) and SMTs (Satisfiability Modulo Theories) in order to improve the efficiency of certain reasoning operations.

Acknowledgments

Many thanks to Diego Quiroz, Sebastian Monsalve, and Jose Ignacio Lopez for their invaluable help with this tool.

[17] http://www.soberit.hut.fi/KumbangTools/

References

1. Benavides D., Segura S., Ruiz-Cortés A. “Automated Analysis of Feature Models 20 Years Later: A Literature Review”. Information Systems. Elsevier, 2010.
2. Czarnecki, K., Helsen, S., Eisenecker, U. “Formalizing cardinality-based feature models and their specialization”. Software Process Improvement and Practice, 10(1):7– 29, 2005.
3. Diaz D., Codognet P. “Design and Implementation of the GNU Prolog System”. Journal of Functional and Logic Programming (JFLP), Vol. 2001, No. 6, October 2001
4. Djebbi O., Salinesi C. “Towards an Automatic PL Requirements Configuration through Constraints Reasoning”. Int. Workshop on Variability Modelling of Software-intensive Systems (VaMoS), Essen, Germany, January 2008.
5. Dhungana D., Grünbacher P., Rabiser R. "The DOPLER Meta-Tool for Decision-Oriented Variability Modeling: A Multiple Case Study," *Automated Software Engineering*, 2010 (in press; doi: 10.1007/s10515-010-0076-6).
6. González-Baixauli B., Laguna M., Sampaio J. “Using Goal-Models to Analyze Variability”. First International Workshop VaMoS, 2007.
7. Griss, M., Favaro, J., Allesandro, M. “Integrating Feature Modeling with RSEB”. Proceedings of the 5th International Conference on Software Reuse, Vancouver, Canada, 1998.
8. Hai H. Wang, Yuan Fang Li, Jing Sun, Hongyu Zhang, Jeff Pan. “Verifying feature models using OWL”. Journal of Web-Semantics (2007) 117–129.
9. Kang K., Cohen S., Hess J., Novak W., Peterson S. “Feature-Oriented Domain Analysis (FODA) Feasibility Study”. Technical Report CMU/SEI-90-TR-21, Software Engineering Institute, Carnegie Mellon University, November 1990.
10. Mazo R., Salinesi C, Djebbi O., Diaz D., Lora-Michiels A. “Constraints: the Heart of Domain and Application Engineering in the Product Lines Engineering Strategy”. International Journal of Information System Modeling and Design IJISMD. ISSN 1947-8186, eISSN 1947-819. April-June 2012, Vol. 3, No. 2.
11. Mazo R., Salinesi C., Diaz D., Lora-Michiels A. “Transforming Attribute and Clone-Enabled Feature Models Into Constraint Programs Over Finite Domains”. 6th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE), Springer Press, Beijing–China, 8-11 June 2011.
12. Mazo R., Lopez-Herrejon R., Salinesi C., Diaz D., Egyed A. “Conformance Checking with Constraint Logic Programming: The Case of Feature Models”. In 35th Annual International Computer Software and Applications Conference (COMPSAC), IEEE Press, Munich-Germany, 18-22 July 2011. Best Paper Award.
13. Pohl K., Böckle G., van der Linden F. “Software Product Line Engineering: Foundations, Principles and Techniques”. In: Springer-Verlag New York, Inc., Secaucus, NJ, 2005.
14. Salinesi C, Mazo R. “Defects in Product Line Models and how to Identify them”. Software Product Line - Advanced Topic, edited by Abdelrahman Elfaki, InTech editions, ISBN 978-953-51-0436-0, April 2012.
15. Salinesi C., Mazo R., Diaz D., Djebbi O. “Solving Integer Constraint in Reuse Based Requirements Engineering”. 18th IEEE International Conference on Requirements Engineering (RE'10). Sydney - Australia. September-October 2010.
16. Schneeweiss D., Hofstedt P. “FdConfig: A constraint-based interactive product configurator”. International Conference on Applications of Declarative Programming and Knowledge Management (INAP) Vienna, Austria. September 28-30, 2011.

Modelling Security Requirements in Socio-Technical Systems with STS-Tool

Elda Paja, Fabiano Dalpiaz, Mauro Poggianella, Pierluigi Roberti and Paolo Giorgini

Department of Information Engineering and Computer Science,
University of Trento, Italy
{paja,dalpiaz,poggianella,roberti,giorgini}@disi.unitn.it

Abstract. Security Requirements Engineering (SRE) deals with the specification of security requirements for the system-to-be starting with the analysis of security issues as soon as in the early requirements phase. STS-ml is an actor- and goal-oriented requirements modelling language for Socio-Technical Systems (STSs), which represents the security needs the stakeholders express as constraints over the interactions between actors. In this paper, we present STS-Tool, the security requirements engineering tool that supports STS-ml. STS-Tool allows for modelling a socio-technical system at a high level of abstraction, expressing constraints (security needs) over the interactions between the actors in the STS, and deriving security requirements in terms of social commitments (promises with contractual validity). It offers multi-view modelling, allowing designers to focus on a different perspective at a time, while promoting modularity.

1 Introduction

Socio-Technical Systems (STSs) are complex systems in which social actors interact with one another and with technical components to fulfil their goals. Each participant is autonomous, and the system is defined in terms of the interactions among actors, which may be: *social reliance*, actors rely on others to achieve their goals, and *information exchange*, actors exchange relevant information. In such systems, many security issues arise from the interaction between actors, and on how the exchanged information is manipulated. Therefore, *social* aspects are a main concern when analysing security.

The importance of considering security from a social and organisational perspective is widely recognised in literature [4,6,7,11]. However, such approaches either rely on high-level concepts that are hard to map to technical requirements (e.g. [4,7]), or suggest purely technical security mechanisms (e.g. [3]). In our view, SRE should start from high-level concerns and refine them into requirements for the system-to-be.

Goal-oriented approaches to security requirements engineering seem to be appropriate for designing secure STSs, since they build upon the concepts of intentional and social actors, who have objectives to achieve and interact with others to achieve them. Existing approaches, such as Tropos [1], Secure Tropos [8], and SI* [5], enable representing actors and their dependencies, in an organisational perspective, but they make the assumption that actors will behave as depicted in the model. Given that the participating actors in an STS are mutually independent—thus, their behaviour is not disclosed to others and they cannot be controlled—, we cannot make such assumption. Instead, the best a designer can do is to allow actors to specify security constraints over their interactions. We refer to these constraints as *security needs* to distinguish from the general security requirements of the system-to-be.

Based upon these principles, we have previously proposed STS-ml (Socio-Technical Security modelling language) [2], an actor- and goal-oriented modelling language that supports the modelling and analysis of security requirements for STSs. In this paper, we present STS-Tool [1], a security requirements engineering tool for STS-ml. The tool offers a graphical modelling environment to allow the definition of the system in terms of actors and their interactions.

The rest of the paper is organised as follows. Sec. 2 briefly outlines the STS-ml language. Sec. 3 presents the main features of STS-Tool. Sec. 4 describes a possible usage scenario. Sec. 5 presents conclusions and future work.

2 STS-ml

STS-ml builds on top of Tropos [1] and its security-oriented extension [5]. It revises the high-level organisational concepts from Tropos, maintaining a minimal set of concepts including actor, goal, delegation, etc., and uses the concept of *social commitment* among actors, to specify security requirements.

The particularity of STS-ml is that it allows actors to express *security needs* over interactions to constrain the way interaction is to take place. This is important, because the actors are mutually independent, and it is when they enter interactions that they might want to express their concerns regarding security. For instance, in e-commerce, a buyer would want a seller not to disclose its credit card details to other parties, and to use this information strictly to perform the payment of the acquired goods.

Social commitments [9] are promises with contractual validity that actors make and get from one another, to achieve their objectives. Formally, commitments are a quaternary relation *C(debtor, creditor, antecedent, consequent)* between a debtor and a creditor (both being actors), in which the debtor commits to the creditor that, if the antecedent is brought about, the consequent will be brought about. In STS-ml, we consider commitments about security-related properties. This concept is used to offer a guarantee that the debtor acknowledges the specified security need by making a commitment, and will behave as required by the security need by bringing about the commitment. For this, whenever a security need is specified from one actor to the other, a commitment on the other direction is expected from the second actor to satisfy the security need. For instance, in e-commerce, the provider commits to prospective buyers that their credit card details will not be disclosed to other parties, and will be used only for the payment of their acquired goods.

The outcome of STS-ml is a security requirements specification expressed in terms of commitments, in which the debtor actor is *responsible* for the satisfaction of the security requirement, whereas the creditor actor is the *requestor*. Fig. 1 outlines STS-ml: the specifications of security requirements for the system-to-be are derived once the modelling is done and the security needs imposed by the actors are expressed. STS-ml

[1] STS-Tool is available for download at `http://www.sts-tool.eu`

supports multi-view modelling: interactions among actors can be represented by focusing on orthogonal views. As shown in Fig. 1, STS-ml consists of three different views: *social*, *authorisation*, and *information*. The security needs are expressed in the operational view (Fig. 1), which consists of the three aforementioned views. The operational view is automatically mapped to the specification *security requirements* for the system-to-be, which supports the security needs expressed in the operational view.

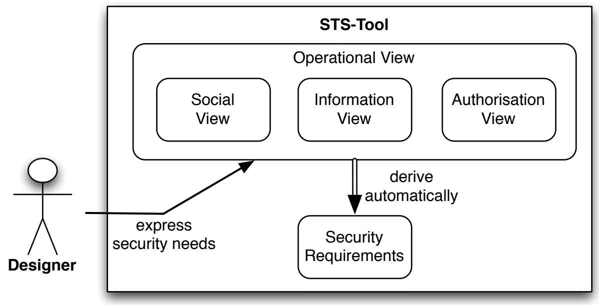


Fig. 1: From the operational view to security requirements

The *social view* represents actors as intentional and social entities. Actors are intentional as they have goals they want to achieve, and they are social, because they interact with others to get things done, mainly by *delegating goals*. Actors may possess documents, they may *use*, *modify*, or *produce* documents while achieving their goals, and they may *distribute* documents through *document provision* to other actors.

The *information view* gives a structured representation of the information and documents in the given setting. Information can be represented by one or more documents (made tangible by), and on the other hand one or more informations can be part of some document. It is important to keep track of how information and documents are interconnected, to be able to identify which information actors manipulate, while using, modifying, producing, or distributing documents for achieving their goals.

The *authorisation view* shows the permission flow from actor to actor, that is, the authorisations actors grant to others about information, specifying the operations actors can perform on the given information, namely *use*, *modify*, *produce*, and *distribute*. Apart from granting authority on performing operations, we consider also whether authority to further give authorisations is granted.

Following our intuition of relating security to interactions, we allow stakeholders to express their security needs over *goal delegations* and *authorisations* regarding information. Once the modelling is done, and all the security needs are specified, the list of of security requirements can be automatically derived from the operational view.

3 STS-Tool

STS-Tool is a modelling tool for STS-ml. It is a standalone application written in Java, and its core is based on Eclipse RCP Engine. It is distributed as a compressed archive for multiple platforms (Windows 32 and 64 bits, Mac OS X, Linux), and is freely available for download. STS-Tool has the following features:

– *Supports specification of projects*: the socio-technical security models are created within the scope of project containers. A project contains a set of models. Each project refers to a certain scenario. Typical operations on projects are supported: create, save, load, modify, rename.
– *Diagrammatic*: the tool enables the creation (drawing) of diagrams. Diagrams are created only within a project. Apart from typical create/modify/save/load operations, the following is also supported:
  • Export diagram to different file formats (png, pdf, etc.);
  • Provide *different views* on a diagram, specifically: *social view*, *information view*, *authorisation view*. Each view shows specific elements and hides others, while keeping always visible elements that serve as connection points between the views (e.g. roles and agents). Inter-view consistency is ensured by for instance propagating insertion/deletion of certain elements to all views.
– *Consistency checking*: the tool helps to create diagrams that follow the semantics of the modelling language, thus improving consistency and validity.
– *Generating requirements documents*: the tool allows the generation of requirements documents that contain the list of security requirements derived from the model in terms of social commitments. Moreover, this document contains information describing the models, which is customisable by the designer. The designer can select which concepts or relations he wants more information about.

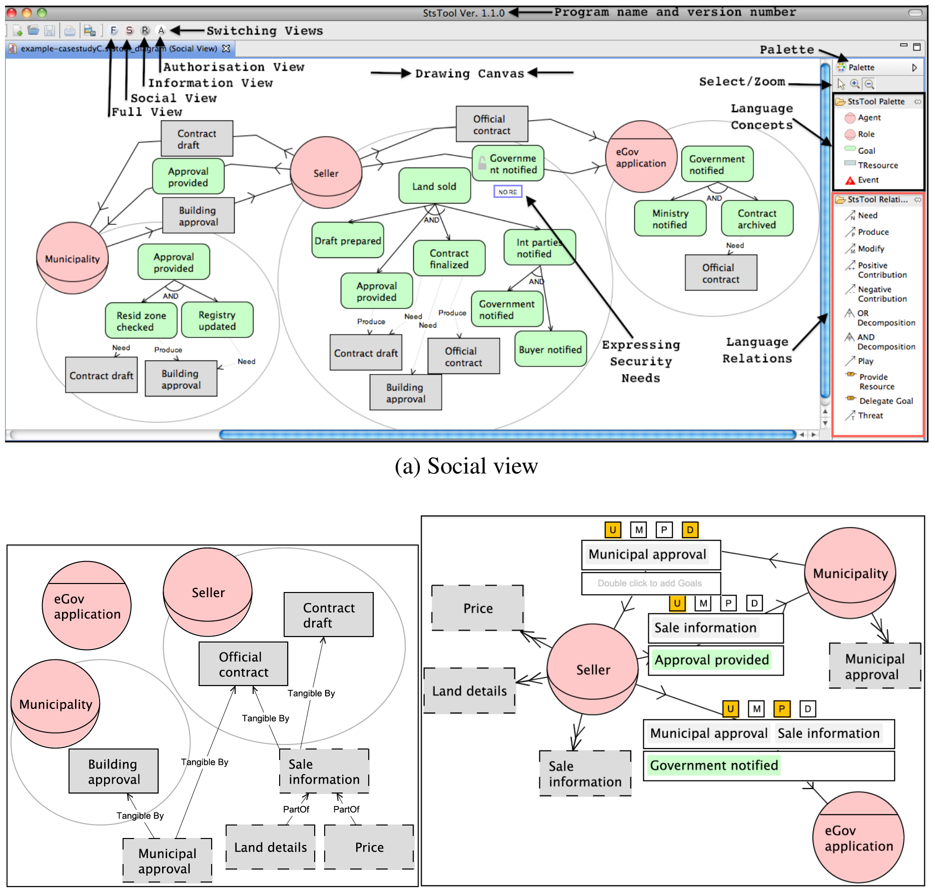
4 Modelling with STS-Tool

We will demonstrate the features of STS-Tool by modelling an illustrative example from a case study on e-Government.

Example 1 (e-Government). Land selling involves not only finding a trustworthy buyer, but also exchanging several documents with various governmental bodies. The seller needs the municipality to certify that the land is residential zoning. The land selling process we consider is supported by an eGov application, through which the official contract (including the municipalitys certification) is sent to the ministry (who has the right to object) and is archived.

Fig. 2 shows the three orthogonal views supported by the tool, namely *social*, *information*, and *authorisation* view, together with the list of derived security requirements (*commitments view*). We present here the steps to follow for performing the modelling of our e-Government example to show how the tool facilitates and supports the modelling process:

1. *Building the Social View*: we start the modelling with the representation of the roles and agents present in the scenario. For this, we switch to *Social View* (Fig. 2a), and select these concepts from the Palette. In our example, we represent the *Municipality*, and the *Seller* as roles, whereas the *eGov application* as agent. When first created, roles and agents come together with their rationale (open compartment), so



(a) Social view

(b) Information view

(c) Authorisation view

Fig. 2: Multi-view modelling for the eGovernment scenario

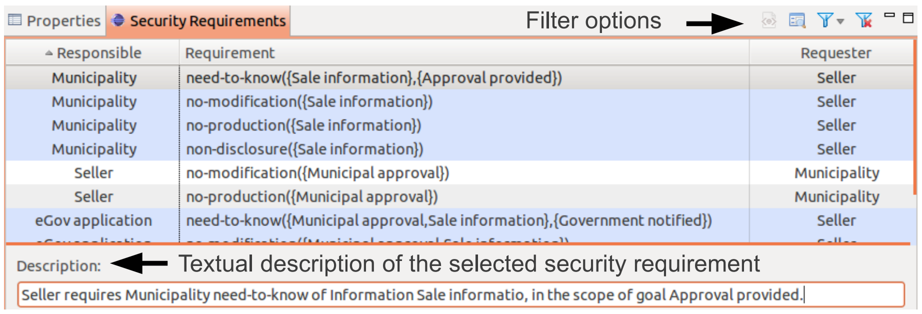
that we can specify goals or documents they have. The rationales can be hidden or expanded, to give the possibility to focus on some role/agent at a time. Actors want to achieve one or more goals. We place actor goals within their rationale: the seller has goal *Land sold*. Goals are refined by *AND/OR-decompositions* obtaining goal trees: *Land sold* is the root goal to be fulfilled. The tool facilitates a correct modelling of goal trees, by not allowing goal cycles.For some goals, actors need to rely on others through goal delegation. When drawing a delegation, the tool makes sure that the actor does have a goal before allowing to draw the goal delegation relationship. Then, the delegated goal is automatically created within the compartment of the delegatee. If a role/agent is delegated the same goal from different roles/actors, the tool maintains one copy of the goal within the delegatee's rationale. Following the semantics of the language, once a goal delegation is drawn from a delegator to a delegatee, the tool does not allow a delegation (or delegation chain) that ends up to the delegator, that is, delegation cycles are also not allowed in the tool.

2. *Are there any Security Needs?*: the designer analyses delegations, to see if any of the supported security needs applies over goal delegations. In Fig. 2a, the *Seller* requests *eGov application* not to repudiate the delegation of goal *Government notified*. To specify this using the tool, the designer clicks on the delegated goal, to have a drop down list of security needs and selects the desired ones. Some of the security needs are mutually exclusive; for these, the tool allows the selection of only one security need. Once the security need is selected, a locker appears on the goal to show that security needs have been specified, and the list of specified security needs appears below the goal, represented with distinguishable labels and different colours.
3. *Refining the Social View*: to achieve their goals, actors need, modify, and produce documents. For instance, the *Seller* needs document *Contract draft* to achieve goal *Contract finalised* (Fig. 2a). To model this, we choose the concept Document from the palette, name it *Contract draft* and then select the relation *Need* from the Palette and connect the goal with the document. The tool helps the designer by allowing this relation to be drawn only starting from the goal to the resource, not vice-versa.
4. *Analyse information*: we switch to *Information View* and *represent informations* and *documents*, relating them together. The tool inherits the roles/agents together with the documents from the social view, so the designer needs just specify how the different documents are interconnected (PartOf) and what information they represent (TangibleBy). For instance, *Official contract* and *Contract draft* contain (make tangible) *Sale information* (Fig. 2b). The tool allows TangibleBy to be drawn only from informations to documents, whereas PartOf to be drawn only between informations or documents respectively. Cycles of PartOfs are not allowed by the tool.
5. *Further refine the Social View*: the designer switches back to the *Social View* to represent information exchange. The tool allows to draw *document provisions* starting only from an actor that produces the document or is in possession of that document. In Fig. 2a, the *Seller* produces document *Official contract* and provides it to the *eGov application*, which needs this document to achieve goal *Contract archived*. Similarly, the designer represents the other interactions with *Municipality*.
6. *Model ownerships*: switch to the *Authorisation View* and define who are the owners of the different informations. The tool inherits roles/agents from the other views and the informations from the information view, so the designer just needs to link the roles/agents with the information, using the *Own* relation from the Palette. In our example, the *Seller* is the owner of *Sale information*.
7. *Model authorisations*: starting from information owners, we draw the authorisations they grant to other actors. For this, the relation Authorisation is selected from the Palette and is drawn starting from one actor to another. This action creates on the canvas an authorisation box that includes labels for the four supported operations (use-U, modify-M, produce-P,distribute-D), which the designer can select by clicking on the label. Below, there are two boxes, which specify that the designer should double click to respectively add a set of informations, and a set of goals. In our example, the *Seller* authorises the *Municipality* to use *Sale information* in the scope of goal *Approval provided* (Fig. 2c). Security needs over authorisations

are specified implicitly from the granted authorisation, so the designer needs not do anything, apart from specifying authorisations. For instance, the *Seller* requires the *Municipality* not to disclose *Sale information*, since the label 'D' for the operation distribute is not selected.

This modelling process (steps 1–7) is iterative. The views can be further refined, depending on the level of detail that is needed. The changes in one view have effects on other views. As described above, the different roles/agents are maintained throughout the views, so the addition/deletion of some role/agent would affect the other views. However, even in these cases, the tool provides support by checking that a role/agent is deleted only when it does not have any interactions with other roles/agents.

Once the modelling is done, and all security needs have been expressed, the tool allows the *automatic derivation* of security requirements. The security requirements are listed and they can be sorted or filtered according to their different attributes: Responsible, Requirement, and Requestor (Fig. 3). For instance, filtering the security requirements with respect to the Responsible actor, gives an idea of who are the actors responsible to satisfy the requirements, while filtering them according to the Requirement, groups together requirements that refer to the same type of security need. Finally, a textual *Description* is provided for every selected security requirement.



Responsible	Requirement	Requester
Municipality	need-to-know({Sale information},{Approval provided})	Seller
Municipality	no-modification({Sale information})	Seller
Municipality	no-production({Sale information})	Seller
Municipality	non-disclosure({Sale information})	Seller
Seller	no-modification({Municipal approval})	Municipality
Seller	no-production({Municipal approval})	Municipality
eGov application	need-to-know({Municipal approval,Sale information},{Government notified})	Seller

Description: Textual description of the selected security requirement

Seller requires Municipality need-to-know of Information Sale informatio, in the scope of goal Approval provided.

Fig. 3: Security requirements via commitments

At the end of this process, the tool allows designers to export models and generate automatically a *security requirements document*, which helps them communicate with stakeholders. This document is customisable: designers can choose among a number of model features to include in the report (e.g., including only subset of the actors).

5 Conclusion and future work

Our work on the STS-ml and tool is ongoing as part of the European research project Aniketos[2]. We are iteratively evaluating our language and tool on case studies from different domains, namely, telecommunications, air traffic management control, and

[2] http://www.aniketos.eu/

e-Government. These case studies offer different complexities, sizes and operational environments, so they prove suitable for our needs. The current version of the tool is a result of an iterative development process, where the release of internal versions of the tool has been followed by evaluation activities [10].

Future work about STS-Tool includes (i) embedding automated reasoning capabilities to identify inconsistencies and conflicts between requirements; and (ii) implementing a plugin management system that allows for adding functionalities to STS-Tool.

Acknowledgments

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant no 257930 (Aniketos) and 256980 (NESSoS).

References

1. Paolo Bresciani, Anna Perini, Paolo Giorgini, Fausto Giunchiglia, and John Mylopoulos. Tropos: An Agent-Oriented Software Development Methodology. *Autonomous Agents and Multi-Agent Systems*, 8(3):203–236, 2004.
2. Fabiano Dalpiaz, Elda Paja, and Paolo Giorgini. Security Requirements Engineering via Commitments. In *Proceedings of the First Workshop on Socio-Technical Aspects in Security and Trust (STAST'11)*, pages 1–8, 2011.
3. Donald G. Firesmith. Security Use Cases. *Journal of Object Technology*, 2(3):53–64, 2003.
4. Paolo Giorgini, Fabio Massacci, and John Mylopoulos. Requirement Engineering meets Security: A Case Study on Modelling Secure Electronic Transactions by VISA and Mastercard. In *Proceedings of the 22nd International Conference on Conceptual Modeling (ER 2003)*, volume 2813 of *LNCS*, pages 263–276. Springer, 2003.
5. Paolo Giorgini, Fabio Massacci, John Mylopoulos, and Nicola Zannone. Modeling Security Requirements through Ownership, Permission and Delegation. In *Proceedings of the 13th IEEE International Conference on Requirements Engineering (RE 2005)*, pages 167–176. IEEE Computer Society, 2005.
6. C.B. Haley, R. Laney, J.D. Moffett, and B. Nuseibeh. Security Requirements Engineering: A Framework for Representation and Analysis. *IEEE Transactions on Software Engineering*, 34(1):133–153, 2008.
7. Lin Liu, Eric Yu, and John Mylopoulos. Security and Privacy Requirements Analysis within a Social Setting. In *Proceedings of the 11th IEEE International Conference on Requirements Engineering (RE 2003)*, pages 151–161. IEEE Computer Society, 2003.
8. Haralambos Mouratidis and Paolo Giorgini. Secure Tropos: A Security-Oriented Extension of the Tropos methodology. *International Journal of Software Engineering and Knowledge Engineering*, 17(2):285–309, 2007.
9. Munindar P. Singh. An Ontology for Commitments in Multiagent Systems: Toward a Unification of Normative Concepts. *Artificial Intelligence and Law*, 7:97–113, 1999.
10. Sandra Trösterer, Elke Beck, Fabiano Dalpiaz, Elda Paja, Paolo Giorgini, and Manfred Tscheligi. Formative User-Centered Evaluation of Security Modeling: Results from a Case Study. *International Journal of Secure Software Engineering*, 3(1):1–19, 2012.
11. Axel van Lamsweerde. Elaborating Security Requirements by Construction of Intentional Anti-Models. In *Proceedings of the 26th International Conference on Software Engineering (ICSE 2004)*, pages 148–157. IEEE Computer Society, 2004.

LiProMo—Literate Process Modeling

Jakob Pinggera, Thomas Porcham, Stefan Zugal, and Barbara Weber

University of Innsbruck, Austria
{jakob.pinggera, thomas.porcham, stefan.zugal, barbara.weber}@uibk.ac.at

Abstract. Recently, research on quality issues of business process models has begun to investigate the process of process modeling, i.e., the process of creating process models. In particular, it has been recognized that during this process, well-functioning communication between domain experts and system analysts is essential for understandable process models. This paper proposes the LiProMo approach to foster communication among system analysts and domain experts by flexibly interlinking textual descriptions and formal process models. The feasibility of LiProMo is shown by a prototypical implementation as well as a visionary scenario that illustrates the usage and benefits of LiProMo. The adoption of Cheetah Experimental Platform as basis for the prototye will support empirical evaluation of LiProMo, as planned for future work.

Key words: business process modeling, process of process modeling, literate process modeling

1 Introduction

Business process models play an important role for managing business processes [1]. Business process models, or process models for short, are for example used to support the analysis and design of process-aware information systems, service-oriented architectures, and web services. In addition, they help to obtain a common understanding of core processes of a business [2] and enable us to identify problems and to discover opportunities for improvement [3].

The process of creating process models, denoted as *process of process modeling* [4], can be characterized as an iterative and collaborative process which typically involves several stakeholders like domain experts and system analysts [5]. It has been recognized that this process of process modeling influences the quality of the resulting process model [4–6]. During this process, information about the domain to be modeled is transferred from the domain experts, who have the knowledge about the domain, but usually lack formal modeling skills, to the system analysts, who select appropriate modeling constructs and formalize this information. This communication, however, is often hampered by the fact that different vocabularies are used, leading to misunderstandings and faulty process models. Similarly, without information from a domain expert, a system analyst may find it difficult to infer the business rules behind modeling constructs [7], bearing a potential source of error. Given the fact that a considerable percentage of lifecycle costs are related to maintenance [8], there is strong demand for better understandable process models reducing the time needed for conducting changes and decreasing the risk of introducing errors.

In this paper, we introduce a technique called Literate Process Modeling (LiProMo), which aims to improve communication during the process of process modeling as well as the maintainability of resulting process models. To this end, LiProMo interweaves the textual descriptions of business processes and their formal business process models. In this way, arbitrary formal process models can be annotated with text fragments, presumably providing a discussion basis for domain experts and system analysts. To put the concepts of LiProMo into practice, we developed a prototypical implementation of an editor for LiProMo. Future validation of the LiProMo approach will be based on this editor.

The paper is structured as follows. Section 2 introduces LiProMo. Section 3 presents the LiProMo prototype. Section 4 describes how we envision the usage of LiProMo. The paper is concluded with related work in Section 5 and a summary and future work in Section 6.

2 Literate Process Modeling

LiProMo is based on the idea of combining graphical process models and informal textual descriptions (cf. Fig. 1A). This combination is motivated by dual channel theory [9], which states that pictorial and textual representations are processed differently by the human mind. Pictorial information is processed in parallel by the visual system whereas textual representations are processed serially by the auditory system [10]. This fact is exploited by dual coding theory, suggesting that conveying information is more efficient when images and text are combined [11]. [12] goes even further by claiming that *"textual encoding is most effective when used in a supportive role: to supplement rather than to substitute for graphics"*. This is underpinned by the findings presented in [13] stating that the understanding of a business problem is significantly increased when reading a BPMN model and the corresponding written use case description.

State of the art process modeling environments like Signavio[1] or IBM Websphere[2] provide means for adding textual descriptions to activities and including comments either directly in the process model or on separate pages. Similarly, wiki-based process modeling systems allow users to link parts of the process model to wiki pages (for an overview see [14]). The major disadvantage of this approach is that by incorporating comments directly into the process model *"visual clutter"* is added, which might *"confound their interpretation by making it more likely they will be interpreted as constructs"* [12]. On the contrary, attaching comments to activities or adding them on separate pages makes them more difficult to access and therefore prone to split-attention effect [15], hampering the understanding of process models [16].

LiProMo follows the idea of Literate Programming [17], which was designed to foster program comprehension. It was later introduced in UML modeling, combining textual descriptions and UML models to create more comprehensive documentations [7]. We adopt this idea for business process modeling to support domain experts and system analysts when collaboratively creating process

[1] www.signavio.com

[2] www.ibm.com/software/websphere

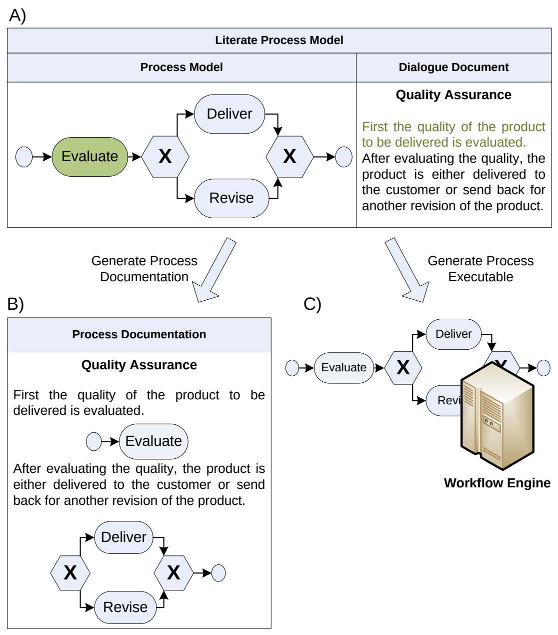
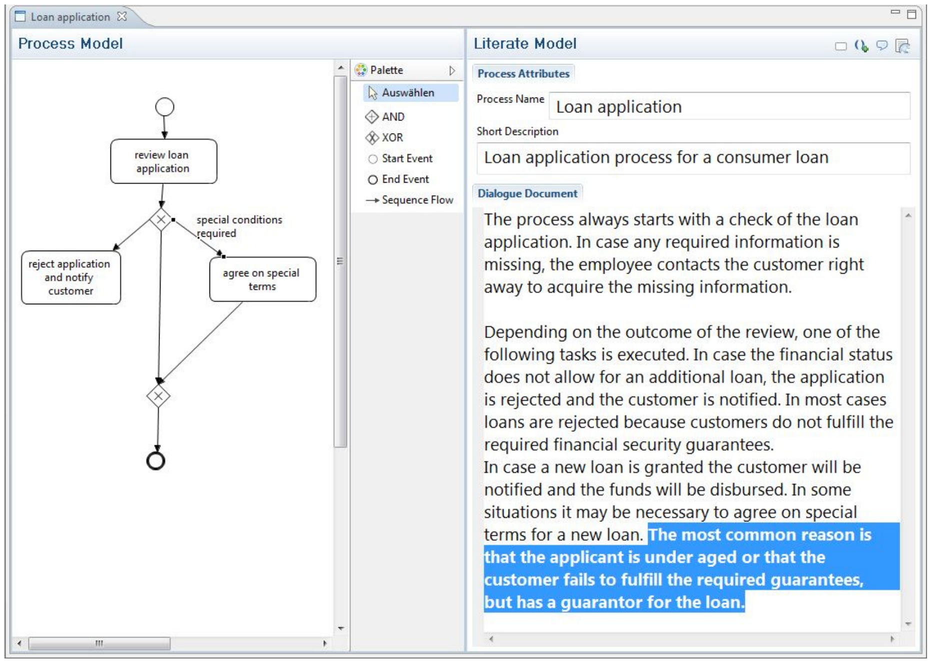

A)
Literate Process Model
Process Model
Dialogue Document
Evaluate
Deliver
Revise
X
Quality Assurance
First the quality of the product to be delivered is evaluated.
After evaluating the quality, the product is either delivered to the customer or send back for another revision of the product.
Generate Process Documentation
Generate Process Executable
B)
Process Documentation
Quality Assurance
First the quality of the product to be delivered is evaluated.
Evaluate
After evaluating the quality, the product is either delivered to the customer or send back for another revision of the product.
Deliver
Revise
C)
Evaluate
Deliver
Workflow Engine

Fig. 1. Literate Process Modeling

models and to improve process model maintainability. Fig. 1 sketches the basic idea of LiProMo. The process modeling editor depicted in Fig. 1A consists of two separate areas holding the process model and the corresponding textual description. To avoid split attention effect, the LiProMo approach juxtaposes the complete textual description and the graphical process model and links model elements with the corresponding task descriptions. The explicit links between process model and textual description can also be exploited for generating process documentations by interweaving the process model and the corresponding parts of the informal specification in a single document (cf. Fig. 1B). When revisiting the process model for conducting changes, the process documentation provides valuable knowledge about modeling choices made in the past, since not only the information contained in elements of the formal process model, but also additional information is readily available and linked to the corresponding model elements, e.g., explanatory examples. The additional information provided by the documentation reduces the risk of introducing errors. Efforts needed for creating additional documentation are expected to be regained as less time is spent on fixing errors [17]. LiProMo models can also serve as executable specifications, i.e., the process model can be fed into a workflow engine providing execution support for process models (cf. Fig. 1C).

3 Literate Process Modeling Editor

We developed a prototypical editor for LiProMo to assess its feasibility. As illustrated in Fig. 2, in a LiProMo model, the textual description of the whole process model and the actual process model are juxtaposed. During the creation of the process model, system analysts and domain experts work together in creating a process description tightly interconnected with the graphical process model (for details on how we envision such a scenario, see Section 4). The LiProMo editor allows for explicitly linking model elements, i.e., single activities or edges, but also whole process fragments that should be documented, to arbitrary passages in the textual description. No restrictions are imposed on either the size and shape of the process fragments or the linked text fragments. The editor automatically highlights the associated textual descriptions for selected modeling elements and vice versa, e.g., in Fig. 2 the edge labeled "special conditions required" is selected, resulting in the highlighted passage of the textual description on the right. The text gives examples for special conditions that would result in taking this execution path.



Process Name: Loan application

Short Description: Loan application process for a consumer loan

The process always starts with a check of the loan application. In case any required information is missing, the employee contacts the customer right away to acquire the missing information.

Depending on the outcome of the review, one of the following tasks is executed. In case the financial status does not allow for an additional loan, the application is rejected and the customer is notified. In most cases loans are rejected because customers do not fulfill the required financial security guarantees.
In case a new loan is granted the customer will be notified and the funds will be disbursed. In some situations it may be necessary to agree on special terms for a new loan. The most common reason is that the applicant is under aged or that the customer fails to fulfill the required guarantees, but has a guarantor for the loan.

Fig. 2. Literate Process Modeling Editor

Another benefit of making associations between modeling elements and textual descriptions explicit is the possibility of generating structured documentation. For every association, the textual description and the process fragment are displayed next to each other, resulting in a step-by-step explanation of the process model. The textual information documents the modeling elements and gives additional information that cannot be derived from activity names or edge descriptions.

In order to be able to evaluate the potential benefits of the LiProMo approach, we built the editor on top of Cheetah Experimental Platform [18]. By logging all interactions, i.e., changes to textual descriptions and graphical process models with the modeling environment to a central database, we are able perform a step-by-step replay of the process underlying the creation of the LiProMo model at any point in time, enabling us to perform analysis similar to [4].

4 Our Vision: Literate Process Modeling in Use

This section describes how we envision the interactions among system analysts, domain experts and the LiProMo editor. For this purpose, we provide a short example describing the collaborative creation of a LiProMo model. In particular, assume that a domain expert and a business analyst document the process of consumer loan applications in a banking institution.

Initially, the domain expert indicates that *"the process always starts with a check of the loan application. In case any required information is missing, the employee contacts the customer right away to acquire the missing information"*. So the system analyst enters the textual description and starts to create the graphical process model. To this end, the business analyst creates the first activity *"review loan application"* by selecting the relevant piece of textual information and choosing the activity name. Based on this information, the LiProMo editor creates the activity and links it to the relevant piece of text. This way, the activity name can be kept short, still providing additional information for future users, e.g., *"in case any required information is missing, the employee contacts the customer right away to acquire the missing information"*. The domain experts sees the changes in the process models and the highlighted textual description and explains that *"depending on the outcome of the review, one of the following tasks is executed. In case the financial status does not allow for an additional loan, the application is rejected and the customer is notified"*. The system analyst updates the textual description and decides to model the described scenario using the exclusive choice pattern. Hence, the business analyst creates an XOR split and an activity for rejecting the application and informing the customer, which is linked to the respective passage in the textual description. The domain expert adds *"in most cases we reject loans because customers do not fulfill the required financial security guarantees"*. The system analysts adds an additional comment to the reject activity and the edge connecting the XOR split and the activity including the examples given by the domain expert. The domain expert continues his explanations and states that *"in case a new loan is granted the customer will be notified and the funds will be disbursed. In some situations it may be necessary to agree on special terms for a new loan"*. As before, the business analyst first updates the textual information and adds an activity for agreeing on special conditions. The business analysts asks for examples of these special conditions. The domain experts answers *"there might be several reasons, but the most common are that the applicant is underage or that the customer fails to fulfill the required guarantees but has a guarantor for the loan"*. The business analysts links given examples to the edge between the XOR split and the special terms activity, only adding the phrase *"special conditions required"* to the edge instead of the lengthy examples given by the domain expert. Then the business analyst completes the process model by creating an XOR join and the end event (cf. Fig. 2). The business analyst and the domain expert go through the process model step by step. Since the domain expert is not familiar with BPMN, the system analyst selects the modeling elements they are currently talking about. The LiProMo editor highlights the corresponding textual descriptions, helping the domain expert in understanding the process model and enabling him to identify potential errors.

5 Related Work

We relate our work to four streams of research: research on understandability and maintainability of process models, the process of process modeling, the automatic generation of process models from natural language and research targeting communication between domain experts and system analysts.

Understandability and Maintainability of Process Models. The impact on model understanding and model maintenance has already been examined from various angles. For instance, [19] looks into the effect of modeling expertise, [20] discusses the influence of domain information, [16] investigates hierarchy, whereas [21] describes the impact of activity names. Similarly, [12, 22] discuss the relation between cognitive aspects and the understanding of process models. Like LiProMo, all these works deal with understandability and maintainability of process models, however, LiProMo rather focuses on the *process of process modeling* than on the outcome of process modeling, i.e., the resulting process model.

The Process of Process Modeling. The LiProMo approach focuses on improving the process of process modeling. Similarly, [23, 24] discuss the interaction of system analysts and domain experts. However, these works focus rather on the negotiation than on the creation of the process model, as done in LiProMo. The process of process modeling was investigated in [6, 18]. In contrast to LiProMo, they embrace a descriptive point of view on the process of process modeling, rather than trying to improve it.

Process Model Generation from Natural Language. There has been considerable research in the area of automatically deriving process models from natural language. [25] proposes a technique to automatically create BPMN models from natural language. [26] describes the creation of BPMN models based on group stories. Even though the automated generation of process models seems promising it is not clear—as argued in [27]—in how far these process models are well understandable. Moreover, several approaches impose restrictions on textual descriptions to provide sufficient structure to be able to derive a process model. Hence, we do not aim to automatically create process models from natural language, but rather support modelers in creating well documented process models in an iterative process involving domain experts and system analysts.

Improving Communication between Domain Expert and System Analyst. As motivated in this work, communication between domain experts and system analysts is often impaired by a different set of skills and vocabulary. For declarative process models, this problem has been tackled by the Test Driven Modeling (TDM) methodology [28, 29]. TDM combines test cases and process model to provide a common vocabulary for domain experts and system analysts. Besides improving communication such a combination improves the maintainability of declarative process models, as shown in [30]. In contrast, LiProMo focuses on supporting the creation and maintainability of imperative process models. In the upcoming evaluation we will strive for similar effects when utilizing LiProMo.

6 Summary and Outlook

In this paper we presented LiProMo—a technique that tightly interweaves graphical process models with their textual description. Presumably, the advantage of such an integration is threefold. First, according to dual coding theory, it allows for more efficient processing of information, hence directly supporting system analysts in creating the process models. Second, the tight integration of the textual description provides a common vocabulary, hence improving communication between domain experts and system analysts. Third, during model evolution, the interweaved availability of visual process model and textual description presumably lowers the chance of misinterpretation, hence improving process model maintenance.

So far, however, these conjectures are based on theoretical considerations only. To corroborate them, we are currently planning an empirical evaluation, in which the prototypical LiProMo editor will be used in real-world modeling sessions. Therein, we will specifically investigate the communication patterns between domain experts and system analyst, allowing us to taylor the editor towards specific usage scenarios. Additionally, we will conduct controlled experiments to assess the impact of LiProMo on the maintainability of process models.

Acknowledgements. Gefördert aus Mitteln des vom Land Tirol eingerichteten Wissenschaftsfonds.
This research was funded by the Austrian Science Fund (FWF): P23699-N23.

References

1. Becker, J., Rosemann, M., Uthmann, C.: Guidelines of Business Process Modeling. In: Business Process Management, Models, Techniques, and Empirical Studies, Springer-Verlag (2000) 30–49
2. Rittgen, P.: Quality and Perceived Usefulness of Process Models. In: Proc. SAC '10. (2010)
3. Scheer, A.W.: ARIS - Business Process Modeling. Springer (2000)
4. Pinggera, J., Zugal, S., Weidlich, M., Fahland, D., Weber, B., Mendling, J., Reijers, H.A.: Tracing the process of process modeling with modeling phase diagrams. In: Proc. ER-BPM '11. (2012) 370–382
5. Hoppenbrouwers, S.J., Proper, E.H., van der Weide, T.P.: Formal Modelling as a Grounded Conversation. In: Proc. LAP'05. (2005) 139–155
6. Soffer, P., Kaner, M., Wand, Y.: Towards Understanding the Process of Process Modeling: Theoretical and Empirical Considerations. In: Proc. ER-BPM '11. (2011) 357–369

7. Arlow, J., Emmerich, W., Quinn, J.: Literate Modelling - Capturing Business Knowledge with the UML. In: Proc. UML '98. (1998) 189–199
8. Cordes, D., Brown, M.: The Literate-Programming Paradigm. Computer **24** (1991) 52–61
9. Mayer, R.E., Moreno, R.: Nine Ways to Reduce Cognitive Load in Multimedia Learning. Educational Psychologist **38** (2003) 43–52
10. Bertin, J.: Semiology of Graphics: Diagrams, Networks, Maps. Univ. of Wisconsin Press (1983)
11. Paivio, A.: Mental Representations: A Dual Coding Approach. Oxford Univ. Press (1986)
12. Moody, D.L.: The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering. IEEE Transactions on Software Engineering **35** (2009) 756–779
13. Ottensooser, A., Fekete, A., Reijers, H.A., Mendling, J., Menictas, C.: Making sense of business process descriptions: An experimental comparison of graphical and textual notations. Journal of Systems and Software **85** (2012) 596–606
14. Dengler, F., Vrandecic, D.: Comparison of wiki-based process modeling systems. In: Proc. i-KNOW '11. (2011) 31–34
15. Sweller, J., Chandler, P.: Why Some Material Is Difficult to Learn. Cognition and Instruction **12** (1994) 185–233
16. Zugal, S., Pinggera, J., Mendling, J., Reijers, H., Weber, B.: Assessing the Impact of Hierarchy on Model Understandability-A Cognitive Perspective. In: Proc. EESSMod '11. (2011) 18–27
17. Knuth, D.: Literate Programming. The Computer Journal **27** (1984) 97–111
18. Pinggera, J., Zugal, S., Weber, B.: Investigating the Process of Process Modeling with Cheetah Experimental Platform. In: Proc. ER-POIS'10. (2010) 13–18
19. Mendling, J., Reijers, H.A., Cardoso, J.: What Makes Process Models Understandable? In: Proc. BPM '07. (2007) 48–63
20. Mendling, J., Strembeck, M.: Influence Factors of Understanding Business Process Models. In: Proc. BIS '08. (2008) 142–153
21. Mendling, J., Reijers, H.A., Recker, J.: Activity Labeling in Process Modeling: Empirical Insights and Recommendations. IS **35** (2010) 467–482
22. Zugal, S., Pinggera, J., Weber, B.: Assessing Process Models with Cognitive Psychology. In: Proc. EMISA '11. (2011) 177–182
23. Constantine, L., Lockwood, L.: Structure and style in use cases for user interface design. Object Modeling and User Interface Design (2001) 245–280
24. Rittgen, P.: Negotiating Models. In: Proc. CAiSE '07. (2007) 561–573
25. Friedrich, F., Mendling, J., Puhlmann, F.: Process model generation from natural language text. In: Proc. CAiSE '11. (2011) 482–496
26. de A. R. Goncalves, J.C., Santoro, F.M., Baiao, F.A.: A case study on designing business processes based on collaborative and mining approaches. In: Proc. CSCWD '10. (2010) 611 –616
27. Glinz, M., Seybold, C., Meier, S.: Simulation-Driven Creation, Validation and Evolution of Behavioral Requirements Models. In: Proc. MBEES '07. (2007) 103–112
28. Zugal, S., Pinggera, J., Weber, B.: Toward Enhanced Life-Cycle Support for Declarative Processes. Journal of Software: Evolution and Process **24** (2012) 285–302
29. Zugal, S., Pinggera, J., Weber, B.: Creating Declarative Process Models Using Test Driven Modeling Suite. In: Proc. CAiSE Forum '11. (2011) 1–8
30. Zugal, S., Pinggera, J., Weber, B.: The Impact of Testcases on the Maintainability of Declarative Process Models. In: Proc. BPMDS '11. (2011) 163–177

Workflow Partitioning for Offline Distributed Execution on Mobile Devices

Peter Wakholi and Weiqin Chen

University of Bergen, POB 7802, N-5020, Bergen, Norway
peterokhisa@gmail.com

Abstract. Traditionally, workflow systems are built on the client/server architecture, in which a single workflow server takes the responsibility for the operation of the whole process, thereby requiring connections each time a task is completed. In cases where connection between client and server is not readily available - like in mobile environments, such an approach proves infeasible. Enabling the execution of a group of tasks by mobile clients in distributed and disconnected environments has been proposed as a possible solution. However, the partitioning of a workflow into groups of tasks for offline execution has not been adequately explored. This paper proposes an approach for workflow partitioning and an algorithm that enables automatic discovery of such partitions from a process model as a vital step in assigning grouped tasks. We have implemented the algorithm, evaluated and validated it and proposed ways in which it could be implemented in a real workflow environment.

Keywords: workflow models, partitioning, offline behaviour

1 Introduction

Traditionally, workflow systems are built on the client/server architecture, in which a single workflow server takes the responsibility for the operation of the whole process, thereby requiring connections each time a task is completed. In environments where connections cannot be sustained e.g. devices using mobile networks in rural areas, such an approach would be infeasible. The practice is to allow for offline data collection and require the client device to make connections once network is stable [1]. Solutions that use distributed Workflow Systems have been proposed[2], making it necessary to partition a process model into a group of tasks that can be executed offline.

A number of methods for partitioning workflows for distributed execution have been proposed [3, 4]. However many of the approaches while addressing the need for distributed offline execution of work, do not cater for the need to provide an option that enables the server to maintain control. The partitions created should be simple enough to be executed by a light-weight mobile client and should not in any way alter the underlying logic of the original model maintained on the server. This enables distributed execution to be just an option and not a requirement. In this paper we have proposed a Petri-net based approach to partitioning workflows, based on structural and behavioural aspects of the original process model. We present a method for partitioning that enables work to be dynamically assigned at different stages of the execution process as long as they can be carried out independent of the original model.

This paper proceeds as follows. In section 2 we explore the requirements and provide rules for partitioning. Section 3 provides an algorithm for automatic discovery of such Partitions. Section 4 provides a theoretical and experimental evaluation of the rules and propositions provided. Related work and the contribution of the paper are discussed in section 5. Finally, section 6 provides future work and further discussions.

2 Workflow Partitioning

2.1 Application Example

Before partitioning a workflow model, one needs to consider the goals and operating environment of the system. We therefore provide a case study of the Promise-Pep Clinical Trial[5] that is considering adding mobile phones as one of the platforms used to collect information. The aim of using workflows is to automate the process of data collection and ensure adherence to pre-defined procedures in the trial protocol. The use of mobile phones for data collection would be when community-based visits are undertaken. We take an instance of a field activity (which is part of larger process) that involves a doctor conducting clinics for HIV+ mothers. First, he looks up the client information (loaded on the mobile device) and records additional information about the visit. Then he does two lab tests (Plasma HIV-1 RNA and Stored plasma) and records the information. Finally, he provides a prescription and records this on his mobile device. At the end of the day, when network connection is established, he uploads this information to the server and the process continues. This portion of work assigned to the doctor is a sub-workflow process that is an independent and complete workflow activity, and does not depend on any outside intervention in order to complete execution.

Conceptually, one can therefore argue that partitioning of workflows needs to ensure that the resulting partitions form some logical workflow process and that there are no data, resource and control flow dependencies outside the partition. [2] provides a set of conditions that workflow partition needs to address, which include that, it must be possible to partition a process model and to allocate the resulting fragments on mobile devices as well as stationary computers; the soundness of the process needs to be ensured; both the overall process model as well as its fragments might have to be adapted during runtime, e.g. to deal with exceptional situations. Based on this case study and related literature, the following are necessary for partitioning process models:

1. A partition should not alter the underlying logic of a process model. Any combination of tasks should preserve the order of execution as defined by the process modeller.

2. All tasks from the partition can be assigned to one resource and can be executed by the service(s) on the client device.
3. All data dependencies for the tasks to be executed are contained in a partition and hence there is no need to connect to the server during the lifetime of its execution.
4. User can define the optimum number of tasks to be assigned or select from a set of possible partitions.

2.2 Workflow Partitioning Rules

We use an example of a workflow net in figure 1, and the unfolded net in figure 2. Unfolding the model as proposed by [6] enables us to determine the dependencies of the tasks in an execution sequence. In the unfolded model, *T9* and *P9* refer to the transition $T8$ .and place $P10$. It can be observed that the possible partitions for this model consists of the transitions; $(T4, T5)$, $(T2, T4, T5, T6, T7)$, $(T3, T8)$, $(T2, T4, T5, T6, T7, T8)$. The following do not form valid partitions $(T2, T4)$, $(T5, T7)$, $(T2, T6)$, $(T6, T7)$, $(T1, T3)$, $(T1, T2)$. These only serve as examples since the number of possible combinations are much higher. The following observations can be made about the partitions generated.

1. For all partitions there is one incoming arc one outgoing arc.
2. For all partitions it is possible to move from one transition to another without requiring input or output to the rest of the process model.
3. The groupings that do not form valid partitions either depend on outside conditions or provide output before fully executing.
4. None of the partitions modifies the execution sequence provided in the in figure 1.

Based on these observations, we generate the following rules for partitioning workflow models: Given an unfolding of a workflow net $W = \langle \mathrm{S};\ \mathrm{T};\ \alpha; \beta, \mathrm{is}\rangle$ is, such that $\mathrm{S} = \{\, \mathrm{s}_0, \mathrm{s}_1, \ldots\ldots, \mathrm{s}_\mathrm{m}\}$ is a set of states and $\mathrm{T} = \{\mathrm{t}_0, \mathrm{t}_1, \ldots\ldots, \mathrm{t}_\mathrm{n}\}$ is a set of transitions; the relation $\alpha\!: \mathrm{T}\ \rightarrow \mathrm{S}$ associates to each transition its source state; $\beta\!: \mathrm{T} \rightarrow \mathrm{S}$ associates to each transition its target state. A partition $\mathrm{W}' = \langle \mathrm{S}', \mathrm{T}'\alpha, \beta, \mathrm{is}\rangle$ where $T' = \{t_i, t_{i+1}, \ldots, t_{i+m}\} \subset T$ and $S' = \{s_i, s_{i+1}, \ldots, s_{i+n}\} \subset S$ is possible if:

1. Rule 1: $\forall W' \exists \{\alpha(t_i) = is, \beta(t_i) = s_i, \ldots \beta(t_n) = s_{i+n}\}$ (causal and connected)
2. Rule 2: $\forall W' \neq \{W \notin W'\}$ (No contradiction)

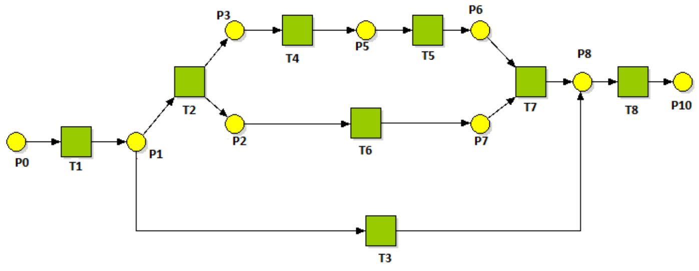


Fig. 1. Example workflow net

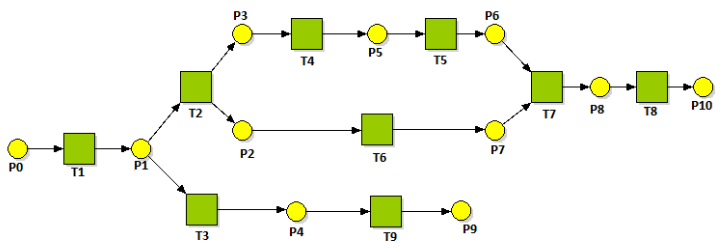


Fig. 2. Workflow net unfolded

3 Automatic Partitioning Algorithm

The relations in the unfolded net can be represented as an incidence matrix that defines the causal relations between the transitions and places as shown in table 1. For a transition column a value of 1 represents an arc entering from a place in the corresponding row, while -1 is for an outgoing arc. The incidence matrix of a directed graph $\mathrm{M} = (\mathrm{P}, \mathrm{T})$ is a $\mathrm{P} \times \mathrm{T}$ matrix $\mathrm{B} = (\mathrm{b_{ij}})$ such that

$$b_{ij} = f(x) = \begin{cases} -1, & \text{if edge } j \text{ leaves transition } j \ (\textit{the relation } \beta(t_j)) \\ 1, & \text{if edge } j \text{ enters transition } j \ (\textit{the relation } \alpha(t_j)) \\ 0, & \text{otherwise} \end{cases}$$

Table 1. Incidence Matrix for unfolded net

	T1	T2	T3	T4	T5	T6	T7	T8	T9	total
P0	-1	0	0	0	0	0	0	0	0	-
P1	1	-1	-1	0	0	0	0	0	0	-
P2	0	1	0	0	0	-1	0	0	0	0
P3	0	1	0	-1	0	0	0	0	0	-
P4	0	0	1	0	0	0	0	0	-1	0
P5	0	0	0	1	-1	0	0	0	0	0
P6	0	0	0	0	1	0	-1	0	0	0
P7	0	0	0	0	0	1	-1	0	0	0
P8	0	0	0	0	0	0	1	-1	0	-
P9	0	0	0	0	0	0	0	0	1	-
P10	0	0	0	0	0	0	0	1	0	-
total	-	1	-	0	0	0	-1	-	-	0

First we apply rule 2: since there must be only one incoming and one outgoing arc, the sum of all arcs within the Partition must be zero. Therefore any arc that is created within the Partition must eventually come to a close. It can be observed that a node (place or transition) with one incoming arc and one outgoing arc will always have Transition column and Place row sum as 0. Rule 2 therefore proposes that for a Partition, the cumulative sum of all the places and transitions must be zero. Considering one of the Partitions $W_1 = (\mathrm{T2}, \mathrm{T6}, \mathrm{T4}, \mathrm{T5}, \mathrm{T7})$ the columns (transitions) and the rows (places) as shaded in table 1 are summed. It can be observed that the cumulative sum of rows and columns is zero. Rule 1 requires that Partitions should be composed of connected and causally related transitions. We therefore institute a search algorithm that starts with the first transition of the intended Partition and adds adjustment transitions (connectedness) while checking for causality. For every transition added, rule 2 is applied and if the cumulative sum of columns and rows is 0, then a Partition is discovered.

The algorithm for Partitions is given below. A mathematical evaluation proves its validity. Since the net is unfolded, the direction of an arc implies that the place/transition can only be visited once in any execution sequence. A column sum in the incidence matrix represents difference in the number of incoming and outgoing arcs while a row represents those of a place. If a Partition has one input place and one output place, then every new arc leaving a Transition creates an execution path that must always close, thus giving a net sum of zero.

```
Require: $A = \langle S; T; \alpha; \beta; is \rangle$            <Unfolded Workflow net>
    $M = (P, T)$                                  <Incidence Matrix>
    $T_j = (b_{ij})$                        <Initial transition column>
    n = 1
    While (n < Number of transitions) do
      If $T_j \Rightarrow T_{j+n}$ then                              <Rule 1>
        add $T_{j+n}$                        <Adjacent transition>
        SumTransitions $\sum_j^{j+n} T$
        sumplaces $\sum_i^{i+n} P$
        If (sumTransitions+SumPlaces=0) then        <Rule 2>
          Partition = $(\mathrm{T_j}, \mathrm{T_{j+1}}, \ldots.. \mathrm{T_{j+n}}$
        End If
      End If
    End While
```

4 Evaluation

This algorithm was implemented by creating a module in the PIPE framework[7]. Based on the unfolded model, Partitions for the first Transition were searched, then the next, until all transitions in the sequence had been visited. The result of running the algorithm gave possible Partitions as, $(\mathrm{T2}, \mathrm{T6}, \mathrm{T4}, \mathrm{T5}, \mathrm{T7})$, $(\mathrm{T2}, \mathrm{T6}, \mathrm{T4}, \mathrm{T5}, \mathrm{T7}, \mathrm{T8})$, $(\mathrm{T3}, \mathrm{T9})$ and $(\mathrm{T4}, \mathrm{T5})$. These match with the observation that were made in section 2.2. The algorithm was evaluated by experimenting on a number of process models whose choice was based on complexity [8] of the underlying constructs by considering the number of Workflow patterns and tasks. Table 2 shows the results of the experiments. A total of five process models were tested, one of which is based on the YAWL for Film [9] process model. For each model, we observed the number of Workflow patterns and the number of tasks. After running the Partition algorithm using the software developed, we counted the number of Partitions discovered and evaluated their correctness and possible omissions.

The hypothesis was that there should be no incorrect or omitted Partitions in order for the rules generated to be valid. This hypothesis was proved to hold for sound Workflow nets, except for state-based patterns like Interleaved Parallel Routing, Milestone, Critical Section, and Interleaved Routing. One common characteristic of all these patterns is that they take a token and return to a place thus giving a net cancellation effect on the Partition. In order to address this problem, the unfolding considered a unidirectional arrow to the transition - based on the fact that our interest is only aimed at ascertaining the state. In addition, Advanced Synchronisation and Multiple instance patterns like the Multi-choice and Multi-merge create large numbers of unfolding, which become complex for analysis.

5 Related Work

There have been attempts to develop a framework that delivers workflow definitions to mobile devices in disconnected environments[10]. The IBM FlowMark [11] is an example of a meta-model that seeks to address the constraints of deploying mobile workflows. Work is loaded to a mobile with the hope that a user is committed to do them. Exotica is an example of a distributed workflow platform where processes are transferred to sites thereby eliminating the need of a centralised server [11].

[12] uses workflow partitioning in BPEL to structurally provide rules based on graph transformations. Transformations are based on rules that ensure that the system exposes the functional behaviour and the flow of the original workflow is observed. The partitioned BPEL processes is executed onto a network of mobile phones. Through this approach, they are able to produce an overall execution model that is equivalent to a centralized one, implemented using disconnected components and independent workflow engines. [2] presents the MARPLE architecture that enables the execution of processes on mobile devices. Through their system, they realize generic process management. The architecture meets the performance requirements of mobile scenarios to cope with specific requirements like broken connections and limited GUIs. They provide a set of requirements for the architecture to work which form part of the basis for the partitioning algorithms proposed. In their work, conceptual issues regarding the partitioning of processes are not provided.

[4] provides a Petri-net based approach for fragmenting workflows for distributed execution. The fragments created can migrate to servers where tasks are performed and new fragments are created. Through this approach a case can be executed on several servers in succession thus enabling the outsourcing of business functionalities.

[3] presents an approach for the distributed execution of workflows based on the fragmentation of high-level Petri-nets. The Petri-nets are fragmented horizontally, vertically and diagonally, and fulfil the necessary requirements for formal workflow behaviour like completeness, minimality and disjointedness. Conceptually, formal methods presented in [3, 4] for distributed workflow differ from our approach due to the problem addressed and deployment environment. Our approach seeks to move work to a client with a light-weight workflow engine for offline execution by combining two or more tasks while maintaining semantics of the original model.

Table 2. Evaluation

Inherent Patterns	**No. of tasks**	**No. of Partitions**	**Incorrect**	**Omissions**
Synchronization, Choice, Sequence, Simple merge, Parallel split	8	4	0	0
Synchronization (nested), Choice, Sequence(x2), Simple merge, Parallel split	9	3	0	0
Synchronization(x3), Simple choice(x6), sequence(x1), Simple merge(x7), Parallel split(2), iterations (x7), synchronising merge (x1)	20	13	0	0
Synchronization, Sequence(x2), Parallel split, Milestone	6	3	2	0
Multichoice, Multimerge	5	0	0	0

6 Discussion and Future Work

In this paper we present an approach to address the problem of partitioning workflows for offline execution in mobile environments and automatically discovering of groups of tasks (Partitions). We present a scenario for such a need and provide a set of requirements for partitioning workflows. Additionally an algorithm for discovering such partitions based on model unfolding and checking is presented. It is important to note that model unfolding represents the full reachability graph using partial orders that preserve the relations between transition occurrences [6]. All reachable markings are therefore represented in a Petri-net unfolding. This enables us to determine the relationships between occurrences thereby making the algorithm proposed sound. The approach proposed therefore considers the static and dynamic behaviour of workflow models when developing partitions.

The algorithm provided in this paper does function correctly for a range of patterns and therefore provides a viable approach to addressing the problem identified. Because an unfolding of a net produces the structural and behavioural aspects of the net, it can be extended to other process modelling languages to enable the discovery of Partitions. Pragmatism in unfolding is necessary to provide workable solutions. So rather than unfolding models to their basic Petri-net based representations, it would be prudent to extend the principles enshrined in this paper in implementing the algorithms on real process modelling environments. Future work will involve implementing this approach in a real workflow environment, developing a light weight workflow engine that is able to execute Partitions on a mobile phone and devising mechanisms for synchronising Partitions with the entire workflow.

7 References

1. Wakholi, P., Chen, W., Klungsøyr, J.: Workflow Support for Mobile Data Collection. Enterprise, Business-Process and Information Systems Modeling 299-313 (2011)
2. Pryss, R., Tiedeken, J., Kreher, U., Reichert, M.: Towards flexible process support on mobile devices. Information Systems Evolution 150-165 (2011)
3. Guth, V., Lenz, K., Oberweis, A.: Distributed workflow execution based on fragmentation of petri nets. pp. 114-125. Citeseer, (Year)
4. Tan, W., Fan, Y.: Dynamic workflow model fragmentation for distributed execution. Computers in Industry 58, 381-391 (2007)
5. http://clinicaltrials.gov/ct2/show/NCT00640263
6. Esparza, J., Heljanko, K.: Implementing LTL model checking with net unfoldings. Model Checking Software 37-56 (2001)
7. Akharware, N., Miee, M.: Pipe2: Platform independent petri net editor. (2005)
8. Lassen, K.B., van der Aalst, W.M.P.: Complexity metrics for Workflow nets. Information and Software Technology 51, 610-626 (2009)
9. Ouyang, C., La Rosa, M., ter Hofstede, A.H.M., Dumas, M., Shortland, K.: Toward web-scale workflows for film production. Internet Computing, IEEE 12, 53-61 (2008)
10. Bahrami, A., Wang, C., Yuan, J., Hunt, A.: The workflow based architecture for mobile information access in occasionally connected computing. pp. 406-413. IEEE, (Year)
11. Alonso, G., Gunthor, R., Kamath, M., Agwaral, D., Mohan, C.: Exotica/FMDC: A Workflow Management System for Mobile and Disconnected Clients. Distributed and Paralell databases 4, 229-247 (1996)
12. Baresi, L., Maurino, A., Modafferi, S.: Workflow partitioning in mobile information systems. Mobile information systems 93-106 (2005)

Towards a Formal Process-driven Framework for Streamlining Patient-centric Care

Wen Yao[1], Akhil Kumar[2], Jerome Rolia[3], Sujoy Basu[3], Sharad Singhal[3]

[1]College of Information Sciences and Technology, Penn State University
[2]Smeal College of Business, Penn State University, University Park, PA 16802
{wxy119,akhilkumar}@psu.edu
[3]Services Research Lab, Hewlett-Packard Laboratories, Palo Alto, CA 94304, USA
{jerry.rolia, sujoy.basu, sharad.singhal}@hp.com

Abstract. Rapidly growing patient interest in enhanced engagement in care processes has motivated health organizations to provide patient-centric care delivery both in clinical and homecare settings. With the goal of giving each patient a more proactive role in their care, we motivate and propose a formal process-driven framework for streamlining patient-centric care and improving patient-provider communication. It will lead to patients having better access to health services and taking more responsibility in their health management. At the same time the burden on healthcare professionals is reduced, while enabling greater efficiency, improved safety and higher quality.

Keywords: patient-centric care, process-driven, clinical pathway, medical guideline, healthcare, patient-provider communication

1 Introduction and Motivation

Despite advances in life expectancy and quality of life, the current healthcare delivery system faces significant challenges in terms of cost, accessibility and quality. One of the goals established by the Institute of Medicine in 2001 is that healthcare delivery should be *patient-centric* [1], which means it should provide care that is respectful of and responsive to individual patient *preferences*, *needs* and *values*. As mobile devices become pervasive, and access to health information becomes easier, patients are becoming more informed. So, it is reasonable to assume that they will play a more interactive role in decision making about their health matters. Hence, there is a need to develop a formal methodology to foster patient-centric care service delivery.

Fig. 1 shows a clinical workflow that delineates the path of a patient who interacts with healthcare teams such as clinics, labs, and pharmacies. In this care process, the patient is the *only* constant who is involved in all the steps and communications among the large number of participants in the healthcare ecosystem. For example, when a patient schedules an appointment, or is discharged from a hospital, the patient communicates with *administrative staff*. At other points of care the patient undergoes clinical activities such as *detection* and *treatment*, which involves various entities such as *departments*, *staff*, *resources*, etc. In this setting, it is important to consider a *process-oriented perspective* that *coordinates and maintains the flow of information between the patient and other entities to ensure an optimal outcome*.

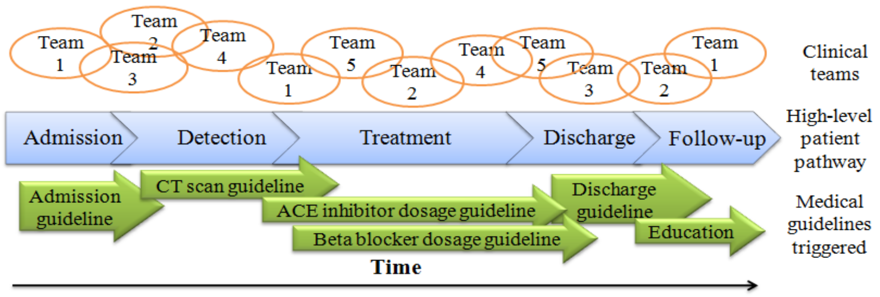


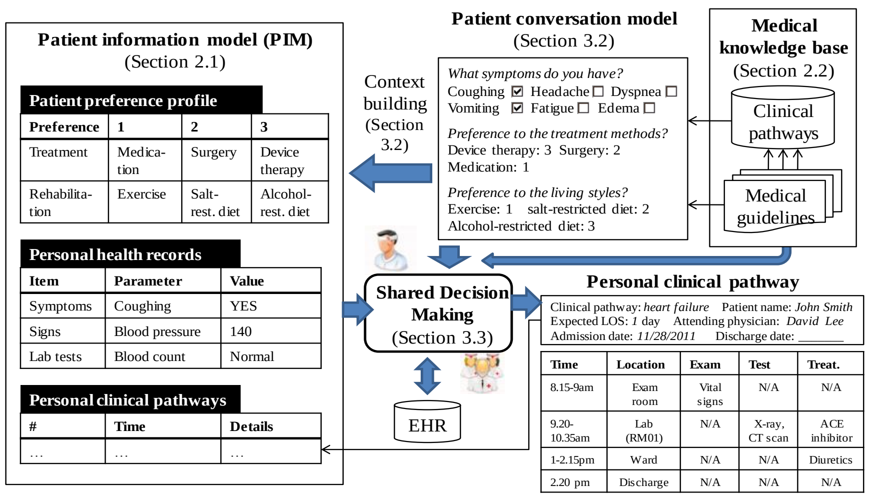
Fig. 1. The healthcare ecosystem

Recent years have seen an increasing interest in IT-based systems that support care delivery. *Although many process-driven approaches have been proposed to support clinical workflow, most are from the care providers' point of view.* Computer-Interpretable Guidelines (CIGs) formalize medical guidelines that were originally in the form of free-format text as computer executable languages, such as Asbru, EON, GLIF, PROforma, and SAGE [2]. The focus of CIGs is on supporting decision making based on best practice to improve the compliance of clinical practice and reduce variations. Thus, these methods are primarily designed for clinicians. Another stream of research uses workflow management systems (WFMSs) to automate and monitor patient pathways, with a focus on addressing specific healthcare challenges. For example, ADEPT$_{flex}$ [3] offers greater workflow flexibility to handle exceptional events; Proclets [4] succeeds in handling weakly-connected interacting workflows with different levels of granularity; Careflow [5] achieves an efficient implementation of clinical practice guidelines; etc. These WFMS systems address the logistics of patient flow from an organizational perspective, but hardly consider patient preferences.

More recently, as the focus of care providers shifts towards patient-centric care, a first step has been to develop applications that support patient access to their own health data and facilitate patient communication with providers (e.g., schedule appointments). A selection of web-based personal health record (PHR) systems, such as WebMD, is reviewed in [6]. Other efforts are devoted towards patient participation and decision making. For example, Porter et al. [7] designed an asthma kiosk application that captures critical information to drive guideline-based care for pediatric asthma. These patient-oriented systems have greatly improved patient communication with providers and their accessibility to health data. However, for the most part they fail to recognize the underlying process a patient undergoes in receiving medical care. We only found a few studies (e.g., Alberta's system [8]) that plan patient pathways for patient self-management. Hence, there is a need to *integrate the process perspective into patient-centric care and make it visible to patients*, to *facilitate patient-provider interaction* in a structured manner and to *give patients a more proactive role*.

This paper proposes a *process-driven approach* to streamline patient-centric care. We formalize clinical pathways based on guidelines and propose a patient information model that incorporates patient needs and preferences. Thus, this framework aims to allow patients to: (1) access their health data and gain insights into the whole process; (2) express choices and take more responsibility; and (3) get a more personalized and coordinated continuum of care. Our approach also benefits the providers since it transfers patient communication workload from medical staff to the system, and tracks patient flows so that process improvement can occur. This paper is organized as follows. In Section 2, we propose a formal framework and describe the patient information model, followed by the decision making process in Section 3. Finally, we discuss future work and conclude the paper in Section 4.

2 A Process-driven Framework for Patient-centric Care



Patient preference profile

Preference	1	2	3
Treatment	Medica-tion	Surgery	Device therapy
Rehabilita-tion	Exercise	Salt-rest. diet	Alcohol-rest. diet

Personal health records

Item	Parameter	Value
Symptoms	Coughing	YES
Signs	Blood pressure	140
Lab tests	Blood count	Normal

Personal clinical pathways

#	Time	Details
…	…	…

What symptoms do you have?
Coughing ☑ Headache ☐ Dyspnea ☐
Vomiting ☑ Fatigue ☐ Edema ☐

Preference to the treatment methods?
Device therapy: 3 Surgery: 2
Medication: 1

Preference to the living styles?
Exercise: 1 salt-restricted diet: 2
Alcohol-restricted diet: 3

Clinical pathway: *heart failure* Patient name: *John Smith*
Expected LOS: *1* day Attending physician: *David Lee*
Admission date: *11/28/2011* Discharge date: ________

Time	Location	Exam	Test	Treat.
8.15-9am	Exam room	Vital signs	N/A	N/A
9.20-10.35am	Lab (RM01)	N/A	X-ray, CT scan	ACE inhibitor
1-2.15pm	Ward	N/A	N/A	Diuretics
2.20 pm	Discharge	N/A	N/A	N/A

Fig. 2. Overview of process-driven framework for patient-centric care

Fig. 2 presents an overall framework for *process-driven*, *patient-centric* care delivery. All steps and decisions are driven by medical guidelines, patient preferences, and needs and values as captured in the *Patient Information Model (PIM)*. *Context-building* is the process of obtaining patient information by conversing with patients in a structured way, e.g., when they are at home or waiting in a clinic. Then, the framework integrates guidelines, aggregate information from past executions stored in the clinical pathways repository, patient needs and preferences, and invokes *the shared decision making* module to suggest options to the patient. The patient can review the options to learn about the issues that pertain to the care process. Finally, the doctor reviews these options, and possibly others also under consideration, with the patient. An action is determined based on their discussion and agreement. All medical decisions, actions, and outcomes for each patient encounter are documented in a *personal clinical pathway* which is recorded in the PIM. The system detects deviations from medical guidelines and requests the doctor to enter reasons for any major deviations, which are logged. The depersonalized process logs collected by our system are analyzed to provide patients with insights about the care of other patients that have experienced similar situations. This framework helps to guide patient conversation by semi-structured process models and coordinates activities among various participants. It helps to reduce the work required by the patient to interact with the system (e.g., reduce duplicate data solicitation, suggests options for patient learning) and enables the patient's participation in various tasks by letting them know what to expect of the care providers and what actions to take at all points of care. We describe the components of Fig. 2 further in the following subsections and in Section 3.

2.1 Patient Information Model

A *patient information model* (PIM) is comprised of three parts: *personal health record* (PHR), *personal preference profile* (PPP), and *personal clinical pathway* (PCP). PHR concerns a patient's lifelong health information that she is allowed to access, coordinate, and share with other parties [6]. It can include patient-reported symptoms, lab results uploaded by patients, or even data from sensors. Usually, it is maintained by patients themselves and can include data from health organizations that they have visited. Here, we assume that PHR is electronic, and is accessible online at any time.

The PPP captures an individual's preferences pertaining to her current situation. We use the rank-ordering method which is a popular comparative scaling technique to evaluate users' preference or liking [9]. For example, the matrix in Fig. 3 shows the preference profiles for patients P1, P2 and P3. It uses a *1 - N* scale, where *N=number of choices in an item category*, for rank ordering the alternative choices within the heart failure guidelines. A larger number indicates a higher rank preference for a choice (1 being least preferred). Thus, the system is aware of patients' preferences of treatment methods, quality-of-life aspects, etc. This profile is acquired or updated from context-building to be discussed further in Section 3.

The PCP documents the actual decisions, actions and outcome organized in chronological order pertaining to a specific episode of care. Deviations from best practice may be necessary to satisfy a patient's needs.

Item	**Applied strategy**		**Treatment method**		**Rehabilitation program**		
Choice	Normal	Aggressive	Medication	Surgery	Exercise	Diet	Physio
P1	1	2	1	2	3	2	1
P2	2	1	2	1	3	1	1
P3	1	1	2	1	1	2	3

Fig. 3. Matrix of patient preference profiles (partial)

2.2 Medical Guidelines and Clinical Pathways

A *medical guideline* is a document that guides decisions and criteria regarding diagnosis, management and treatment in a specific medical discipline (e.g., heart disease). This is naturally aligned with the way they are developed, i.e., by medical staff with different expertise areas. A *clinical pathway* implements medical guidelines after they are *tailored to local and individual circumstances* [10]. In a clinical pathway, different tasks are defined for various roles, and optimized in a logical time sequence. Outcomes are tied to specific interventions, e.g., following a healthy eating pattern for a week might reduce blood pressure. A clinical pathway is basically a template from which concrete patient treatment cases (i.e., process instances) are derived. Fig. 4 depicts an example pathway that associates two medical guidelines from the Agency for Healthcare Research and Quality (AHRQ) for heart failure management [11]. In this way, Fig. 4 guides the evaluation and treatment of patients with heart disease in a structured, *process-driven* manner.

A *personal clinical pathway* (PCP) documents the actual execution for a specific patient. It may correspond to a clinical pathway such as Fig. 4. It keeps track of medical decisions (e.g., prescribe ACE inhibitor which is a pharmaceutical drug used primarily for treating hypertension and congestive heart failure), actions (e.g., dosage for ACE inhibitor), and patient outcomes in chronological order for each patient situation. Each task is associated with its time of occurrence. As noted above, deviations are allowed since humans control the actual execution of the process. A final outcome, e.g., the patient is cured, or ultimately passes away, indicates the end of a PCP. The PCP is a result of clinical decision making which is discussed further in Section 3.

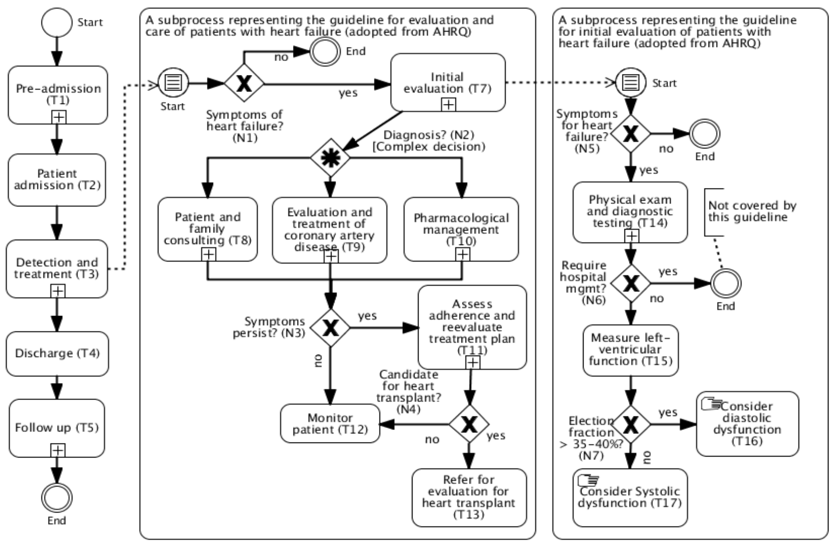

Start
Pre-admission (T1)
Patient admission (T2)
Detection and treatment (T3)
Discharge (T4)
Follow up (T5)
End
A subprocess representing the guideline for evaluation and care of patients with heart failure (adopted from AHRQ)
Start
no
End
yes
Initial evaluation (T7)
Symptoms of heart failure? (N1)
Diagnosis? (N2) [Complex decision)
Patient and family consulting (T8)
Evaluation and treatment of coronary artery disease (T9)
Pharmacological management (T10)
Symptoms persist? (N3)
yes
Assess adherence and reevaluate treatment plan (T11)
no
Candidate for heart transplant? (N4)
Monitor patient (T12)
no
yes
Refer for evaluation for heart transplant (T13)
A subprocess representing the guideline for initial evaluation of patients with heart failure (adopted from AHRQ)
Start
Symptoms for heart failure? (N5)
no
End
yes
Physical exam and diagnostic testing (T14)
Not covered by this guideline
Require hospital mgmt? (N6)
yes
no
End
Measure left-ventricular function (T15)
Election fraction > 35-40%? (N7)
yes
no
Consider diastolic dysfunction (T16)
Consider Systolic dysfunction (T17)

Fig. 4. A clinical pathway for two guidelines from AHRQ [11]

3 The Decision Making Process

In this section, we describe the shared decision making process. Medical knowledge for decision points is formulated as rules that are used to derive recommendations based on best practice and patient preferences.

3.1 Medical Rules

We use rules to embody medical knowledge. The rules help to make complex decisions in clinical pathways through logical reasoning. For example in the clinical pathway of Fig. 4, N2 is a decision node that decides the next step, e.g., treatment or further evaluation, based on patient diagnosis results. A node can be associated with a number of medical rules. Integrating these rules and applying results from rule-based reasoning into a clinical pathway is critical for implementing evidence-based practice. In addition, each rule is associated with a *strength of evidence (SOE)* value to indicate its reliability. The three values for SOE are: A (good evidence), B (fair evidence), and C (expert opinion). They are based on a quality-rating system developed by AHRQ. For example, rule R1 is associated with task T10 and shows recommended medication based on "good evidence" (SOE equals A).

Rule R1 (Node: T10-medication): SOE=A
If a patient's *systolic blood pressure < 90 mmHg* **and**
    there is a *higher risk of complications*
Then prescribe *ACE inhibitors* managed by an *experienced* physician

3.2 Context-building through a Patient Conversation Model (PCM)

A medical decision is context-dependent, where context is patient specific. Our system can facilitate the process of learning about context by asking questions we expect of patients prior to their interaction with the care provider, and recording their responses. For example, in Fig. 4, context that is used at node T3 (detection and treatment) can be collected prior to that point, e.g., at node T1 (pre-admission) or T2 (patient admission). Then, depending on the patient answers, the subsequent questions need to be adjusted. We propose a *patient conversation model* (PCM) that describes the key questions asked at various points of care for a specific clinical pathway.

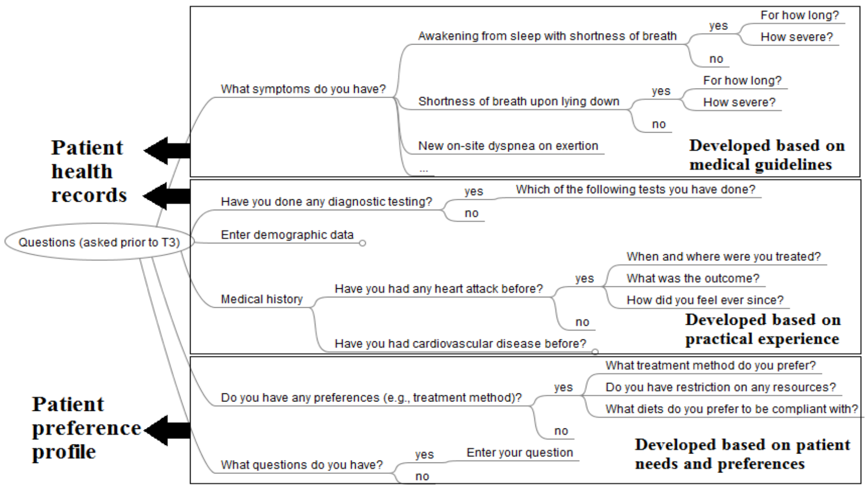
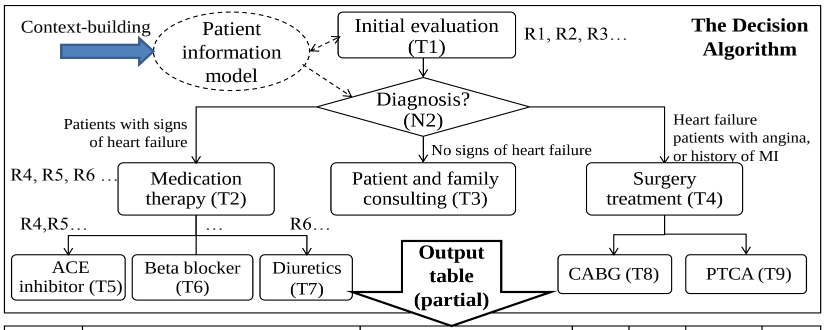

Patient health records
Patient preference profile
Questions (asked prior to T3)
What symptoms do you have?
Awakening from sleep with shortness of breath
yes
For how long?
How severe?
no
Shortness of breath upon lying down
yes
For how long?
How severe?
no
New on-site dyspnea on exertion
...
Developed based on medical guidelines
Have you done any diagnostic testing?
yes
Which of the following tests you have done?
no
Enter demographic data
Medical history
Have you had any heart attack before?
yes
When and where were you treated?
What was the outcome?
How did you feel ever since?
no
Have you had cardiovascular disease before?
Developed based on practical experience
Do you have any preferences (e.g., treatment method)?
yes
What treatment method do you prefer?
Do you have restriction on any resources?
What diets do you prefer to be compliant with?
no
What questions do you have?
yes
Enter your question
no
Developed based on patient needs and preferences

Fig. 5. A partial patient conversation model in a decision tree

Fig. 5 shows an example PCM for heart failure, represented as a decision tree. The top part is derived from medical guidelines and the other two parts are developed based on practical experience and patient needs. These questions are available for patients to answer any time prior to T3 in the clinical pathway of Fig. 4. For example, a patient can enter her answers at home or while waiting for examination. PCM is *process-aware* since context becomes increasingly available as the care process proceeds. Via this model, we also give an opportunity to the patient to access information, e.g., the details of each treatment option (e.g., general success rate, relative cost, and side effects). Thus care providers spend less time on explanation.

3.3 Shared Decision Making

Medical decision making should follow best practice through medical rules and take into account patient information obtained during context-building. The decision algorithm works, briefly, like this: *when a decision node D is reached, we retrieve the rule set RS associated with D, run them against PHR and get evidence-based results*. Other options not triggered by rules may still be presented to patients who know that no guideline supports the options but that may better meet patient preferences. Fig. 6 shows an example of decision making at node N2 (Diagnosis). During initial evaluation, this patient underwent a physical exam and diagnostic testing. Her signs indicate that she might have had heart failure. Her systolic blood pressure is 85 mmHg, and there is a high risk of complications. As a result, medical rules (R1-R6) are triggered at different points of care and produce the results shown in Fig.6. ACE inhibitor (SOE = A) and Diuretics (SOE = C) are recommended based on best practice. Nevertheless, patients and doctors decide which option is chosen.



Options	**Pros**	**Side effect/Risk**	**Prcd.**	**Cost**	**SOE**	**Pref**
ACE inhibitor	Reduce mortality in severe heart failure…	Decrease in blood pressure, increase in potassium …	…	1.5k	R4 (A)	2
Diuretics	Treat various conditions, e.g., high BP	Nausea, dizziness, fatigue...	…	0.8k	R6 (C)	2
Beta-blocker	Reduce risk of recurrent heart attacks…	Side effects similar to heart attack itself	…	1k	N/A	2

Fig. 6. Illustration of the process for recommending decisions

4 Discussion and Future Work

In general, care providers should promote consistency and uniformity in care delivery through implementation of evidence-based practice. The paradox is that, on the one hand, it is desirable to reduce variation by standardizing workflows to conform to best practice; on the other, clinical pathways should be designed to allow flexibility to meet specific needs of patients and resource constraints of a health system. Thus, a formal and radically new approach is required for streamlined communication between patients and providers to deliver evidence-based, yet personalized, care where patients can play a more proactive role in their health care matters.

In this paper, we describe the blueprint for such an approach. We propose a formal process-driven framework to streamline the communication between patients and care providers. Specifically, we introduce a patient conversation model (PCM) that informs the patient and the care provider, and a patient preference profile that informs the care provider. Introducing this information within care processes in a systematic way contributes to patient-centric delivery of care. This approach can benefit patients by allowing them to express their preferences and needs, and play a more active role in their own care. It also transfers a lot of the workload of handling patient communication from the medical staff to the system.

In future, we plan to extend and refine the structure of PCM models based on inputs from health professionals and patients. We also intend to automate the construction of the conversation model. Further, we expect to develop a patient portal based on existing open source tools as an engagement platform for patients and use HL7 messaging protocol [12] to interact with other health organizations to address the data interoperability issues. A prototype system is anticipated and further details of a cloud-based infrastructure to support this model will be described subsequently.

Acknowledgments. Thanks to Dr. Ann Snowdon, ICHIL, for feedback on this work.

References

1. Institute of Medicine: Crossing the Quality Chasm: A New Health System for the 21st Century. National Academies Press, Washington, D.C. (2001)
2. Peleg, M., Tu, S., et al.: Comparing computer-interpretable guideline models: A case-study approach. Journal of the American Medical Informatics Association 10, 52-68 (2003)
3. Reichert, M., Dadam, P.: ADEPT flex—supporting dynamic changes of workflows without losing control. Journal of Intelligent Information Systems 10, 93–129 (1998)
4. Mans, R.S., Russell, N.C., van der Aalst, W.M.P., Bakker, P.J.M., Moleman, A.J., Jaspers, M.W.M.: Proclets in healthcare. Journal of Biomedical Informatics 43, 632-649 (2010)
5. Quaglini, S., Stefanelli, M., Cavallini, A., Micieli, G., Fassino, C., Mossa, C.: Guideline-based careflow systems. Artificial Intelligence in Medicine 20, 5-22 (2000)
6. Kim, M.I., Johnson, K.B.: Personal health records: evaluation of functionality and utility. Journal of the American Medical Informatics Association 9, 171–180 (2002)
7. Porter, S.C., Cai, Z., Gribbons, W., Goldmann, D.A., Kohane, I.S.: The asthma kiosk: a patient-centered technology for collaborative decision support in the emergency department. Journal of the American Medical Informatics Association: JAMIA 11, 458-467 (2004)
8. Delon, S., Mackinnon, B.: Alberta's systems approach to chronic disease management and prevention utilizing the expanded chronic care model. Healthcare Quarterly (Toronto Ont.) 13 Spec No, 98-104 (2009)
9. Brown, T.C., Daniel, T.C.: Scaling of ratings: concepts and methods. Res. Pap.RM-293. Fort Collins, CO: U.S. Department of Agriculture, Forest Service, Rocky Mountain Forest and Range Experiment Station. 24 p. (1990)
10. Lenz, R., Reichert, M.: IT support for healthcare processes-premises, challenges, perspectives. Data & Knowledge Engineering 61, 39–58 (2007)
11. Konstam M, D.K., Baker D. : Heart Failure: Evaluation and Care of Patients with Left-Ventricular Systolic Dysfunction. Rockville (MD): Agency for Health Care Policy and Research (AHCPR) (AHCPR Clinical Practice Guidelines, No. 11.), (1994)
12. HL7: Health Level Seven.http://www.hl7.org/