1

Tool-based Support for Organization-specific Enterprise Architecture Management

Sabine Buckl, Thomas Dierl, Florian Matthes, and Christian M. Schweda

Technische Universität München, Institute for Informatics,
Boltzmannstr. 3, 85748 Garching, Germany
{sabine.buckl,dierl,matthes,schweda}@in.tum.de
http://www.systemcartography.info

Summary. The raising complexity and the companies pressure to differentiate can be handled with the help of Enterprise Architecture (EA) management. Therefore, the EA management processes have to be aligned with the specific companies needs and context. To allow a fast and easy introduction of an EA management it is advisable to use known best practices. But as the aforementioned factors have a great impact on the specific EA management of a company, it is important refine best practices according to the organization specifics. This paper describes an approach to support the definition and execution of an organization-specific EA management.

Key words: EA management, Workflow, Process-driven, Tool

1.1 Introduction

Enterprise architecture (EA) management is a commonly accepted instrument for modern organizations to deal with today's challenging environment. Effectively designed an organization-specific EA management function can improve the overall agility of an organization. The design of such management function is a challenging task, in which the different process steps, information flows, and roles that constitute such function have to be shaped and aligned. The complexity of the management subject EA and the high number of involved stakeholders further aggravate the creation of a consistent but organization-specific EA management, and call for tool support during the *design phase*. In this phase, the designer selects the goals to be pursued, the concerns to be addressed, and the roles to be involved in EA management. Based on this selection, the user can be supplied with re-usable and practice-proven building blocks that can be integrated into a tailored EA management function for the using organization. Different sources for such building blocks exist, namely the EA management pattern catalog [1] or TOGAF [2], although the latter does not explicitly state such blocks.

After the organization-specific EA management function has been defined, it must be enacted in the organization. For the *conduction phase*, i.e. for managing the EA, the tool must provide support by initializing the corresponding processes and process steps, informing the relevant actors, and ensuring that their information demands are fulfilled. For the latter, the tool must support the generation of EA views that correspond to well-defined viewpoints [1, 3]. These viewpoints prescribe which architectural information is conveyed to which stakeholder. Building on the prefabricates of [4], where a tool for flexibly visualizing EAs is presented, we subsequently outline the core idea behind an EA management design tool that further allows to enact the designed process, sketch requirements and realization ideas, and give an outlook on future developments.

1.2 Requirements

Central idea of the tool presented in this article is the building-block based configuration of an EA management function. Later this function is enacted by the tool. An outline of the overall structure of the process is shown in Figure 1.1.

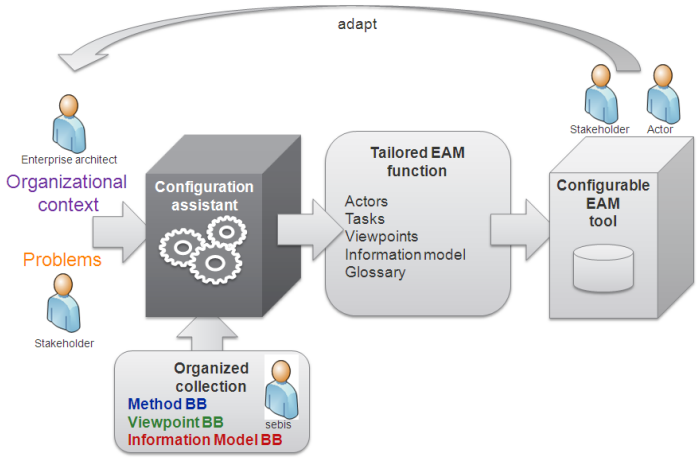

Fig. 1.1. Using building blocks (BB) to configure & adapt an organization-specific EA management function

Put in other words, there is a design phase in which a certain user, as e.g. an Enterprise Architect, defines the organization specific EA management. The second phase is the execution phase in which the defined processes are executed by the tool. Prior to the design phase a sound basis of building blocks has to be established and made available for use in the tool. Although

this phase would also benefit from tool support, we abstain from detailing the required functionality in favor of a concise treatment of design and execution phase, respectively.

All phases require the possibility to store basic information that is needed to execute EAM process, e.g. names of tasks, input and output artifacts, and responsibilities. Therefore, a comprehensive and embracing model for all phases is needed. Additionally, a graphical user interface for all steps is important.

Furthermore an adapt cycle is suggested. This cycle can serve two purposes. First the cycle can *trigger changes* to the EA management, this means, that real changes need a new configuration as e.g. changed goals or organizational contexts. Second, the adapt cycle can *increse maturity* by detailling a concern or advance a methodology.

The specific requirements for the phases design phase and execution phase are further detailed in the following sections.

1.2.1 Design phase

Central idea behind the design phase is the understanding that re-usable building blocks for an EA management function may be extracted from different EA management approaches in literature as well as observed in practical cases. These building blocks are then aligned to a common terminology and their underlying organizational contexts as well as their pursued management goals are elicited. Thereby, the different possibly competing building blocks are interlinked into a *nexus* (cf. Figure 1.2) that backs the design tool.

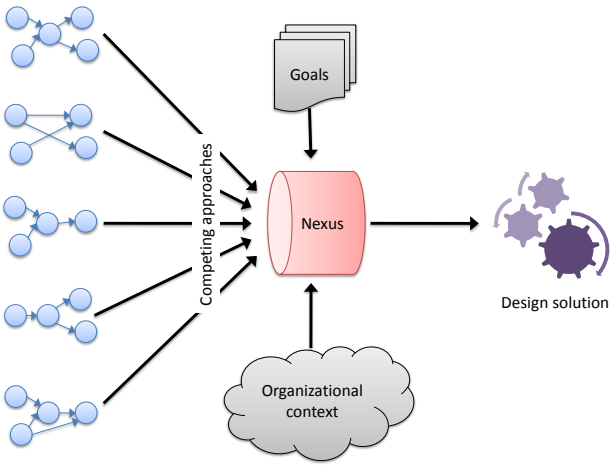

Fig. 1.2. Building block nexus backing the design tool

Based on this nexus, a tool user can design an organization-specific EA management function in seven steps, as follows:

1. Select organizational context
2. Select concern and goal
3. Select question for reifying the goal
4. Add question to concern context
5. Select activity
6. Configure process
7. Adapt process

In step one, the user describes the organizational context into which the EA management function is to be embedded. More precisely, the use ris asked to answer a set of questions (cf. Figure 1.3) that frames the management environment.

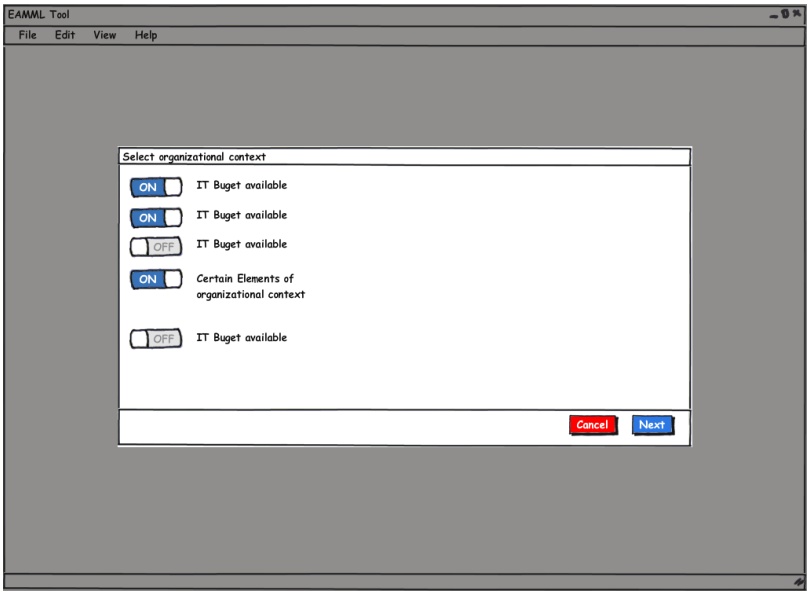

Fig. 1.3. Mock up of the step *Select organizational context* in the design tool

Step two lists the available concerns and goals from the library. The user selects one concern and one goal that will be further detailed and modeled in the following steps.

The step *Select question for reifying the goal* shows the user possible relevant questions for the goal selected in step two. This step further details the goal and defines what the user wants to accomplish.

After selecting the question it is linked to a specific element, such as an attribute, in the concern context.

The last step within the wizard is to select an activity in which the user wants to start modeling. Depending on the already modeled activities different alternatives can be chosen, for example it is necessary to first define how to document the current landscape before the process for developing planned architectures can be defined.

After finishing the wizard steps, step 6 shows the modeling canvas and the user starts to model the process. Therefore, the tool should support several steps detailed further in the following:

Selection of appropriate building blocks, contexts, views, and actors during the design of a process. This includes recommending of complementing building blocks from a best practice library, selection of appropriate organization-specific actors, and recommendation of known best practice viewpoints for the selected building blocks.

Configuration of appropriate triggers and conditions for selected building blocks. That means that the tool should support the user when creating more complicated process triggers and conditions. These can be expressed for example using the Object Constraint Language (see [5]).

Check consistency of the selected, configured, and adapted process to avoid not executable processes. Hereby, the tool should analyze the syntactical and semantical consistency of the process. For example the tool evaluates, if all data needed to create views has been collected before.

Finally, in step seven, the user can adapt the selected and configured process. Although best practice building blocks are used as foundation of the processes there still remains demand for organization-specifics. Therefore, the tool should support the adaptation of the selected building blocks according to the organization's needs. This means that the user should be able to insert, change, or delete process steps. A last consistency check should be done when the user finishes modeling.

After one pair of concern and goal have been finished, the process starts over with step 2. This way the user can always focus on the configuration of a certain part of information for a given goal. The steps 1 to 5 are implemented with the help of a wizard to provide better guidance for the user. The steps 6 and 7 are implemented in a graphical modeling interface that is organized similarly as in different modeling tools, as the Rational Software Architect, with several pallets from which the elements can be drawn on a design canvas. An example mock up is shown in Figure 1.4.

Another important aspect is the constant evolution of an EA management process. Therefore, the tool must be capable to support a multitude of iterations with changes to the existing processes. This includes simple changes to view-points and roles but also applies to complete remodeling of processes according to changed contexts.

While EA management is a collaborative function, the design phase can be considered a single user task. The tool should support the designer by creating a consistent EA management process.

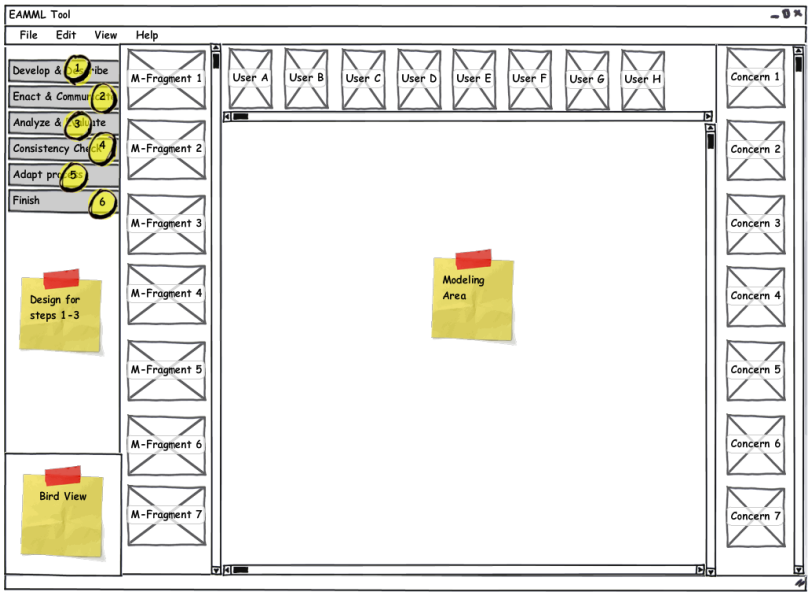

Fig. 1.4. Mock up of the design tool

Recapitulatory the design phase raises the following requirements:

- Single-user application
- Import or interface to a best practice library for building blocks
- Support for composition of building blocks
- Free modeling of processes
- Consistency checking
- Export of modeled processes
- Support for iterations

After modeling the needed processes these are handed to the execution environment and set in action there.

Although only the steps six and seven are directly impacted by the used modeling language, the tool raises several requirements for the modeling language. For example the exchange of information about building blocks from to library to the design tool and the exchange of completed process models to the execution environment. Furthermore, the modeling language should allow consistency checking and it should be easy to use. Because of the strict requirements for the modeling language the next section evaluates different possibilities for modeling languages.

1.2.2 Selection of modeling language

A plethora of process description languages have been developed by various authors. There is no specialized language for EAM processes up to now. There-

fore, two basic options exist: either to develop a specialized modeling language and develop all the necessary tool support, or to use an already existing and defined language with existing tools and engines (see [6]). Because of the enormous effort that is necessary for the development of a specialized language, the preferred way is to find an appropriate existing language. We concentrated on three different languages for the evaluation. They were selected because of their distribution or meaning for existing standards. The selected languages are briefly introduced in the following paragraphs.

Software Process Engineering Metamodel

The first version of the Software Process Engineering Metamodel (SPEM) was released by the Object Management Group (OMG) in 2002. The current version 2.0 was released in 2008. SPEM consists of a process engineering meta-model and a conceptual framework. It is intended to model, document, present, manage, interchange, and enact development methods and processes. The target audiences are process engineers, project leads, as well as project and program managers who are responsible for maintaining and implementing processes for their development organizations or individual projects. [7]

According to the conceptual framework of SPEM it can be used [7]

- to provide a standardized representation and managed libraries of reusable method content,
- to support systematic development, management, and growth of development processes,
- to support deployment of just the method content and process needed by defining configurations of processes and method content, and
- to support the enactment of a process for development projects.

Despite the well-suited intended usage, several drawbacks aggravate the utilization of SPEM fro modeling EA management building blocks. The most important one arises from the fact that SPEM cannot be executed directly. Although the standard document describes, that the execution is possible using a mapping from SPEM to an executable language, such as BPEL, it does not provide further details. Hence, several groups have been working to make SPEM executable.

Within the project SPEM2XPDL[8] a mapping and a transformation algorithm from SPEM to XPDL was developed to make SPEM executable, but up to now this work builds on version 1.0 of SPEM and therefore does not support conecpts like constraints or conditions. This lack of constraints leads to the need to enrich the XPDL models after transformation.

Another project which attempts to make SPEM executable is xSPEM[9]. It extends the OMG SPEM standard so that it is possible to map SPEM models to Petri nets for validation and transformation to BPEL for monitoring. A major drawback of the proposed mapping is the need for completion or

modification to make the generated code executable and these modifications are not traced back to the SPEM model.

Altogether SPEM seems to be not applicable, because rudimentary elements for the execution are missing.

Event driven process chains

In 1992 event driven process chains (EPC) were introduced [10]. They form a widely used semi-formal description language for business processes. An EPC can be modeled with tool support of ARIS Toolset, which is build by IDS Scheer, but also with other tools, which are mainly used in academia.

Because of it being semi-formal, for example the conditions in an conditional branch are not formalized and so the language cannot be executed directly. This was examined in several papers (see [11]). As for SPEM also for EPCs several extensions have been suggested (see [12]), but none of them found general agreement or was implemented in a tool. Another issue of EPC is the non-standardized exchange format.

Business Process Modeling Notation

The Business Process Modeling Notation (BPMN) was published in 2004 in its first version [13]. An updated version 1.1 was released in 2008. As the BPMN specification [14] itself states: "This version does provide a non-normative mapping from BPMN to WSBPEL, but the BPMN specification itself is known to be incomplete with respect to capturing all the required information for WSBPEL. So the mapping is insufficient, in any case.", it is clear, that a direct execution of BPMN is not possible.

In the last few years the University of Potsdam worked in cooperation with SAP Research on an extension for BPMN to make it executable. In 2007 [15] introduced xBPMN, that enriches BMPN with control flow details for execution and in 2008 [16] extended xBPMN with an orthogonal data flow perspective to enable orchestration of processes. But still the language is not fully executable and these extensions are not supported by a tool nor are they standardized.

Important for the tool acceptance is the intuitive and easy of use for the Enterprise Architects. Because of this the modeling language used should be graphical and as far as possible widely known, so users do not have to get to know another modeling language. The special needs arising with this tools is the border less execution of the modeled processes. Because of the special needs BMPN, as a widely known language, was adapted to the specific requirements of the tool.

1.2.3 Execution phase

For the execution of processes different requirements arise. One of the most important is the possibility to communicate needed information. For example,

according to the EAMPC [1] the communication and publishing of information should be done in accordance with viewpoints. For this it is important that views corresponding to these viewpoints can be generated dynamically while execution. Therefore, the tool must support the generation of EA views that correspond to well-defined viewpoints [1, 3], configured in the *design phase*.

Two other important elements of the process execution is the allocation of a user for a specific task who is responsible for the execution and the appropriate persistence of the changed and collected data.

Another aspect the execution tool has to cope with are changing process models. If, as described before, an adaptation of the process model is made and changes to the process model were necessary, the execution tool has to handle these changes in currently running and new process instances. While the handling of new process instances seems to be quite obvious, the treatment of currently running process instances raises some complexities.

It is obvious that these requirements are versatile, reaching from simple requirements up to sophisticated assistance for the user and algorithms for the execution of the processes. The next section sketches implementation possibilities for both phases.

1.3 Use Cases & Implementation

Recent developments in EA management tools show that vendors of these tools try to bring their tools in the web. Although we assume that the design tool is a single user tool there may be companies, where several users are responsible for the design of the EA management processes. Although it is not necessary, because of the non multi-user usage, it seems to be advisable to implement the design tool web-based. Advantages out of this is a fast role-out and fast extension to further users.

1.3.1 Design phase

Modern web-browsers allow to implement user interfaces that are similar to rich client applications with the help of JavaScript. As JavaScript is the only wide spread programming language for browsers it is clearly the language of choice in the browser. On the server side, there are no specific requirements that influence the decision for a server side programming language. Therefore, no preferred language can be set on this side.

The use case diagram in Figure 1.5 shows that the information transfer from the library is an important aspect of the design tool. Because the language used by the tool for modeling is not standardized, there is no exchange format existing. But the used modeling language is based on BPMN and the upcoming version 2.0 of the BPMN standard includes in its beta version [17] two possible exchange formats. One is the BPMN XML Schema Definition (XSD) for the exchange of standardized XML files and another is the BPMN

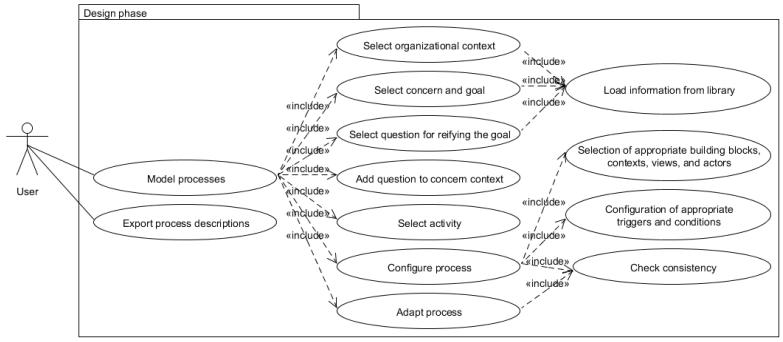

Fig. 1.5. Use cases for the design phase

XMI for transferring object information based on MOF [18]. These two format can be adapted and used for exchanging information.

1.3.2 Execution phase

Because of interaction with so many users, the GUI of the execution tool should implemented as a web application. This can be done for example with the help of the Eclipse RAP Project (see [19]). A simple mock up is shown in Figure 1.6. The tool works as follows: The user visits the website and sees on the left hand side a list of pending tasks he/she has to complete. When the user selects a task, details are shown in the bottom section and views as well as answer panel on the right hand side. When the user selects an answer the tool marks the current task as finished and initiates the next tasks, according to the selected answer.

If the user experiences problems while executing a task the tool gives support in form of an *Escalate* button (see Figure 1.6), which can be used by the user to escalate the process to the process owner, who then tries to put the hurdle aside.

Also for the execution phase an implementation concept was created. This was done with the help of a use case diagram, which is shown in Figure 1.7.

1.4 Outlook

Building block-based modeling of EA management processes is in many ways different from typical process modeling. The tool described in this article can therefore greatly benefit from using a modeling language appropriate for this purpose, i.e. a language especially supporting the composition of process building blocks. Further challenges arise from integrating the process with

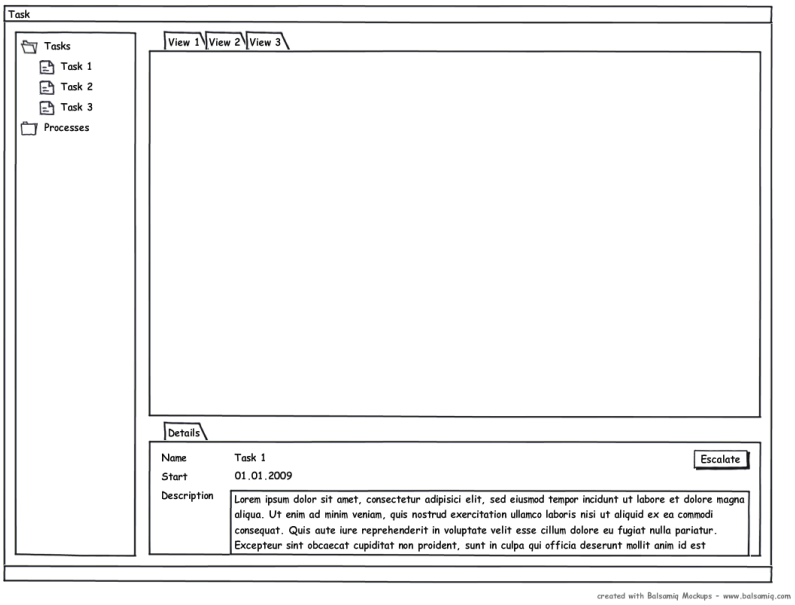

Fig. 1.6. Mock up of the user interface for the execution tool

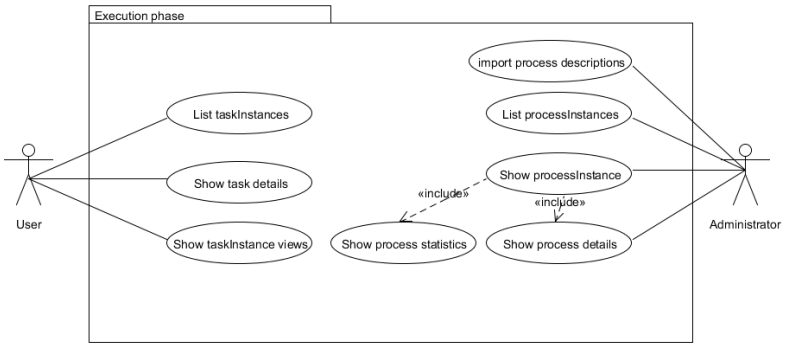

Fig. 1.7. Use cases for the execution phase

other building blocks, e.g. the viewpoints. In each integration, consistency must be verified and minor user-specific adaptations should be supported.

Anoher important part of this approach is the constant evolution of the building block foundation (Organized Collection). Different possibilities for this may appear. For example are the customizations of the users a good source to enhance building blocks. Another option is improving evaluation

methods for selection of building blocks according to collected Best-Practices with their contexts to quantifyable rating.

Further topics in this research area could be investigations of the process for creation of building blocks. Algorithms for the automated composition of patterns would also be very interesting as they define how the tool makes suggestions to the user for composed processes. Important for the tools are furthermore the algorithms for the consistency checking of modeled processes.

References

1. Buckl, S., Ernst, A.M., Lankes, J., Matthes, F.: Enterprise Architecture Management Pattern Catalog (Version 1.0, February 2008). Technical report, Chair for Informatics 19 (sebis), Technische Universität München, Munich, Germany (2008)
2. The Open Group: TOGAF "Enterprise Edition" Version 9. `http://www.togaf.org` (cited 2010-02-25) (2009)
3. Kurpjuweit, S., Winter, R.: Viewpoint-based meta model engineering. In Reichert, M., Strecker, S., Turowski, K., eds.: Enterprise Modelling and Information Systems Architectures – Concepts and Applications , Proceedings of the 2nd International Workshop on Enterprise Modelling and Information Systems Architectures (EMISA'07), St. Goar, Germany, October 8-9, 2007. LNI, Bonn, Germany, Gesellschaft für Informatik (2007) 143–161
4. Buckl, S., Ernst, A.M., Lankes, J., Matthes, F., Schweda, C., Wittenburg, A.: Generating visualizations of enterprise architectures using model transformation (extended version). Enterprise Modelling and Information Systems Architectures – An International Journal **2**(2) (2007) 3–13
5. OMG: Object constraint language (ocl) available specification, version 2.2 (formal/2010-02-01) (2010)
6. Palluch, J., Wentzel, P.R.: Prozess-modellierungssprachen: Eine Übersicht. http://www.methodpark.de/ressourcen/download/white-paper/prozess-modellierungssprachen-eine-uebersicht/show-details-for-download/ (cited 2010-01-20) (2009)
7. OMG: Software & systems process engineering meta-model specification (formal/2008-04-01) (2008)
8. Feng, Y., Mingshu, L., Zhigang, W.: Spem2xpdl: Towards spem model enactment (2008)
9. Bendraou, R., Combemale, B., Cregut, X., Gervais, M.P.: Definition of an executable spem 2.0. In: APSEC '07: Proceedings of the 14th Asia-Pacific Software Engineering Conference, Washington, DC, USA, IEEE Computer Society (2007) 390–397
10. Keller, G., Nüttgens, M., Scheer, A.W.: Semantische prozessmodellierung auf der grundlage ereignisgesteuerter prozessketten (epk). Veröffentlichungen des Instituts für Wirtschaftsinformatik (IWi), Heft 89, Universität des Saarlandes, January 1992 (1992)
11. van der Aalst, W., Desel, J., Kindler., E.: On the semantics of epcs: A vicious circle. In: Proceedings of the EPK 2002: Business Process Management using EPCs, Trier, Germany, Gesellschaft für Informatik (2002) 71–80

12. Mendling, J., Neumann, G., Nüttgens, M.: Yet another event-driven process chain. In: and Information Systems Architectures - an International Journal 1, Springer (2005) 3–13
13. Initiative, B.P.M.: Business process modeling notation (bpmn) - version 1.0 (2004)
14. Group, O.M.: Business process modeling notation, v1.1 - formal/2008-01-17 (2008)
15. Grosskopf, A.: xBPMN. Master's thesis, Hasso Plattner Institut, Universität Potsdam (2007)
16. Schreiter, T.: xBPMN++. Master's thesis, Hasso Plattner Institut, Universität Potsdam (2008)
17. Group, O.M.: Business process modeling notation, ftf beta 1 for version 2.0 - dtc/2009-08-14 (2009)
18. OMG: Meta object facility (mof) core specification, version 2.0 (formal/06-01-01) (2006)
19. The Eclipse Foundation: Rich ajax platform (rap). http://www.eclipse.org/rap/ (cited 2010-01-28) (2010)