Towards a CAME Tools for Situational Method Engineering

Nicolas Arni-Bloch

University of Geneva, CUI, 24 rue General Dufour,
CH-1211 Geneva 4, Switzerland
nicolas.arni-bloch@cui.unige.ch

Abstract. While the theory of Situational Method Engineering (SME) becomes increasingly solid, very few engineering tools have been developed to support the application of its research results. In this paper we analyse the requirements for such a tool and detect the capabilities that are not yet provided by existing tools. The paper focuses on the role of the method process enactment mechanism which is generally omitted in such kind of tools. It guides the way to use the method in order to accomplish the development of corresponding schemas.

1 Introduction

Information Systems (IS) are increasingly used in all areas of our society. While the use of IS in organisations grows, the complexity and the diversity of its application domains increases as well. To address this complexity methods are used to guide Information Systems Development (ISD). However, ISD methods are often too generic and cannot be followed literally, they need to be adapted to the specific situation of each ISD project. Situational Method Engineering (SME) aims to resolve this problem and to provide techniques allowing to construct project-specific methods 'on the fly' [1]. It focuses of the formalization of methods in terms of reusable method components and the definition of assembly techniques allowing to construct new methods by reusing these components.

The aim of our work is to propose a tool supporting assembly-based SME and allowing to satisfy method requirements of each specific ISD project. In this paper we present our first steps towards such a tool.

In the next section we present an overview of SME domain and define its basic terms. In section 3 the requirements for method process enactment is discussed, while different tools necessary to support SME are introduced in section 4. The interoperability perspective in SME is described in section 5 and finally section 6 closes this paper with the discussion about our future works.

2 Overview of Situational Methods Engineering Domain

In this section we present an overview of Situational Method Engineering (SME) domain. We define here the main concepts of this domain such as method, situational method, method chunk and assembly based SME. Finally, we present the different level of the method engineering domain.

2.1 Definitions

Method. Generally speaking, a method describes a regular and systematic way how to accomplishing something. In the domain of Information Systems engineering, Brinkkemper defines a method as "an approach to perform a systems development project, based on a specific way of thinking, consisting of directions and rule, structured in a systematic way in development activities with corresponding development product" [2]. In more structured way, a method is made up of a product part and a process part. The product part represents the concepts that are used in the method, relationships between these concepts as well as constraints that they have to satisfy. The process part represents the way to accomplish the development of the corresponding product.

Situational Method. We call situational method the method which is constructed for a specific project in order to perfectly satisfy its specific situation.

Method Chunk. A method chunk is a cohesive, autonomous and interoperable part of method [3]. It is a basic building block in the development of situational methods. Method chunks are stored in a repository also called method base [2]. The notion of a method chunk descriptor is proposed in order to support method chunks selection and retrieval process. The descriptor allows to specify the reuse context of the method chunks and to classify and position them in the repository.

Assembly Technique for SME. It is clear that traditional method construction techniques cannot be applied here as they would be too expensive and time consuming. We need techniques allowing to construct new methods in a fast and effortless manner. One of these techniques is called assembly technique [4]. The method engineering process following this technique consists in three steps: first of all we have to specify the situation of the project at hand and to define its methodological requirements. Second, we have to select method chunks satisfying these requirements and finally, we have to assemble the selected method chunks in order to obtain a coherent situational method**.**

2.2 The Three Levels of Method Engineering

We distinguish three method knowledge levels in the domain method engineering. The highest level represents the *meta-knowledge* of method engineering that is the method meta-model. In this level the product and the process models of methods are specified in terms of product and process meta-models. Depending of the semantic expression capability of this level and is extensibility [5], the meta-model will be able to specify a large or small range of method processes and products. Particularly, the formality level of the process model will define the capability for a tool to realize the enactment of the method. The method enactment is discussed in the next section.

The middle level, called *method knowledge* level, specifies the method product and process perspectives in terms of corresponding models. By instantiating the method meta-model, the process model of a specific method is defined, and the product model used in this method is specified. The construction of the method hold in this level, it can be done from scratch, by assembly of method chunks or by modification of existing methods.

Once a method is specified we can use it; this occurs in the lowest level called *method enactment* level. In this level the method process model is executed and product models are instanced.

3 Enactment of the Method Process

While the product part of a method has been well defined in the literature and had a lot of tools to express it, the process part is less developed [6]. Several languages are available to define the product part of a method. Some of them are specific to a kind of product like UML for object specification, ER for database specification, other are more generic like the meta-language MetaEdit+ [7]. All these languages are supported by tools allowing to manipulate them like CASE tools (Rational rose, PowerDesigner for UML) or metaCASE tools as MetaEdit+. Unfortunately none of these tools can express the process part of a method and support the enactment of the method process model. By ‘express the process part of a method’ we mean the ability to specify, with a *meta-model*, the tasks and procedures and their ordering in order to achieve the method goal. By ‘enactment of the method process model’ we mean the ability of a tool to support the development and elaboration of a corresponding product according to the method process specification. In order to achieve the enactment of a method with a tool, the method meta-model needs to be formal and precise enough; only textual definition of a method is not sufficient to realise the enactment mechanism. The minimal requirements for a tool supporting method enactment are as follows:

Product space. The product space defines all kinds of products that will be used during the method process.

Input product parts. The input product parts represent the objects from the product space that are used by the method in order to realize the process.

Output product parts. The output product parts are the objects from the product space produced by the method by executing its process.

Body of the method. The body of the method is the description of its process; it can be a textual description, an executable action or a more complex formal or semi-formal specification of a process. A textual body describes the process part of a method in natural language. In this case, the tool supporting method enactment can just show the textual information and offer some facilities for the manipulation of the corresponding products. These facilities depend of the product types. It can be a drawing capability for UML modelling, a graphical representation of data, etc. In an executable body the process is expressed as an executable algorithm in a programming language. The method enactment tool has to be able to execute this piece of code. Finally, a complex body is a method process that is composed of other method processes. In this case the tool has to route the enactment in the diferent sub-processes.

4 Tools to Support Situational Method Engineering

Different kinds of tools are needed to support the engineering of situational methods. The first tool is a methods repository also called methods base. In this tool method chunks are stored together with their descriptors. The biggest challenge of this tool is to provide a high level method chunks classification mechanism. Chunks have to be well described in order to know what the method chunk is doing without the need to look inside its specification.

The second tool is the computer-aided methods engineering tool (CAME). This tool is based on the method meta-model and it is responsible for method chunks specification, i.e. their product and process parts definition. Method chunks specification can be done “from scratch”, by assembly or by modification. In the first case product and process models of the method chunk are defined by instantiating the method meta-model used by the tool. In the second case method chunks are assembled in order to satisfy some specific situation. In the third case method chunks are obtained by modification of other method chunks in order to better satisfy the method goal. Depending to the method meta-model, the CAME tool should offer graphical modelling facilities and special features.

The last tool is the method instantiation tool. This tool or this family of tools have to be able to perform the enactment of the constructed method. In other words, the tool has to support the development and elaboration of products according to the method process specification. It has to guide the method user in the application of a selected method chunk and to offer different features for the manipulation of the products to be constructed by using this method chunk. Therefore, the instantiation tool has to understand the process model of the method chunk and be able to execute it. In order to support product construction, this tool has to understand the product model of the corresponding method chunk and to support its instantiation.

As each ISD life-cycle uses a lot of different kind of product types, like products for the analysis phase, design, development, etc., it would be hard to have only one tool for all kinds of products. As we said before, there are a lot of tools to manipulate product parts of methods, but none of them integrates the method enactment facility. One possibility to ensure the method instantiation is to add the enactment capability to these tools. Another way is to add the enactment capability to the metaCase tools like metaEdit+. In this case we have to define the meta-capabilities for method enactment to cover a wide range of product types and activities.

5 Interoperability and Situational Methods

In this section we briefly discuss the notion of interoperability in two perspectives: (1) how situational methods and SME could help to resolve deferent IS interoperability problems and (2) what are the interoperability challenges in SME?

First of all, the repository of method chunks enables the possibility to store method chunks addressing different IS interoperability issues as for example requirements specification for interoperable systems and applications, integration of IS components, etc., These chunks can then be assembled and tuned to respond some particular interoperability situation.

Besides, SME itself has also different kinds of interoperability challenges. To enable method chunks assembly, SME approaches have to deal with interoperable method chunks and have to provide mechanisms to define differences or similarities of method chunks i.e. their product and process models. Therefore, the first challenge of SME is the method chunk definition. As we indicated before, to enable assembly based method engineering we need to have a repository containing a large number of method chunks. One of the possibilities to quickly fill the method repository of an organization is to exchange method chunks between organizations or to by them. But the question is how to ensure the compatibility between different kinds of method chunks, i.e. their process and product models? The incompatibility can be detected in different levels: classification parameters, modeling languages, process or product models specification. But this interoperability challenge is not specific to method engineering domain. It can be compared with other kind of inter-organizational exchanges such as information, data, and applications. We hope that the method chunks for IS interoperability will help to find method solutions to resolve different IS interoperability problems.

The second challenge of SME is called the tool interoperability [2]. As mentioned before in this article, there exist a lot of product types and tools supporting them that cover the ISD life-cycle. Therefore, we are confronted with the integration of these different product types and tools. This problem holds in the overall framework for SME. Different ways exist to address this challenge based on the storage structure like common repository, agent based communication or functional component in a integrate architecture.

6 Future Works

The survey of the literature shows that there are many research works made in the field of SME. Even though the theory of SME becomes more and more solid, there are very few tools available to support the specification of reusable method chunks, their storage in method repositories, their assembly and especially the enactment of methods.

In order to implement all these capabilities we have to resolve a large range of problems such as method chunks definition and classification, assembly-based method construction and method enactment. In our future perspective we consider to focus on one of these challenges, the method enactment mechanism. Therefore, we are consolidating now our method meta-model and we consider the way to implement it in a specific programming environment.

References

[1] Kumar, K. & Welke, R.J. : Method Engineering, A Proposal for Situation-specific Methodology Construction. *In Systems Analysis and Design : A Research Agenda*, Cotterman and Senn (eds), Wiley, (1992) 257-268

[2] Brinkkemper, S.: Method engineering: engineering of information systems development methods and tools. Information and Software Technology, Vol. 38. Elsevier Science B.V. (1996) 275-280

[3] Ralyté, J. & Rolland, C. (2001). An Approach for Method Reengineering. Proceedings of the 20th International Conference on Conceptual Modeling (ER2001), Yokohama, Japan, LNCS 2224, Springer (2001) 471-484

[4] Ralyté, J., Rolland, C.: An Assembly Process Model for Method Engineering. Proceedings of the 13th Conference on Advanced Information Systems Engineering (CAISE'01), Interlaken, Switzerland, LNSC 2068, Springer-Verlag (2001)

[5] Prakash, N.: Towards a Formal Definition of Methods. Requirements Eng, Vol. 2. Springer-Verlag (1997) 23-50

[6] Tolvanen, J.-P., Rossi, M., Liu, H.: Method engineering: Current research directions and implications for future research. In: IFIP TC8 Conference on Method Engineering, Chapman & Hall, 1996

[7] Kelly, S., Lyytinen, K., Rossi, M.: MetaEdit+: A Fully Configurable Multi-User and Multi-Tool CASE and CAME Environment. In: Proceedings of the 8th International Conference on Advanced Information Systems Engineering (CAiSE'96), Heraklion, Crete, Greece, Constantopoulos et al. (eds.), , LNCS 1080, Springer-Verlag (1996) 1-21