Component-Based Method Development: An Experience Report

Kurt Sandkuhl, Hasan Koç

Component-based Method Development: An Experience Report

Kurt Sandkuhl[1,2,3], Hasan Koç[1]

[1]The University of Rostock, Institute of Computer Science
Chair Business Information Systems, Albert-Einstein-Str. 22, 18059 Rostock, Germany
[Kurt.Sandkuhl, Hasan.Koc]@uni-rostock.de

[2]Jönköping University, Box 1026, 55 111 Jönköping, Sweden
[3]ITMO University, St. Petersburg, Russia

Abstract. A method defines a systematic process for problem solving including the required aids and resources. This paper aims at contributing to the area of method development and in particular to practices and experiences in this field by reporting on a case from conceptual modelling and reflecting on lessons learned in it. The contributions of the paper are (1) an application case from method development in a distributed team, (2) the actual method development process integrating work procedure, cooperation principles and notation, and (3) experiences and lessons learned from developing a method component for context modeling.

Keywords: method component, method engineering, method development, enterprise modeling, conceptual modeling.

1 Introduction

In very general terms, a method defines a systematic process for problem solving including the required aids and resources. Many engineering disciplines use methods as means to capture proven practices and to formalize best practices. In computer science and business information systems, methods do not only address solution development processes or parts thereof, but also the construction of specific artefacts, like various kinds of models. The development of methods usually is a complex process since methods have to be grounded in solid experiences, elaborated with an adequate level of detail and ideally validated in many application cases in order to reach a sufficient maturity level. Although there is a rich body of knowledge in the field of method engineering, the number of experience reports from actual method engineering projects is limited (see section 2).

This paper aims at contributing to the area of method development and in particular to practices and experiences in this field by reporting on a case from conceptual modelling and reflecting on lessons learned in it. The case considered is the development of a component-based methodology in the area of information

systems development. The contributions of the paper are (1) an application case from method development in a distributed team, (2) the actual method development process integrating work procedure, cooperation principles and notation, and (3) experiences and lessons learned from developing a method component for context modeling.

The remaining part of the paper is structured as follows: the background for the work from method engineering is briefly introduced in section 2. Section 3 presents the application case constituting the frame for this research. An overview to the method development process is given in section 4. Section 5 discusses the different phases of the process with experiences and lessons learned. Conclusions and future work are discussed in section 6.

2 Background

Work from the areas of method engineering and related work regarding experiences in method development will be summarized in this section.

2.1 Method Engineering

The research area of method engineering offers a rich body of knowledge how to systematically develop, introduce and adapt "methods". Methods often are considered as prescriptive since they are supposed to provide guidance for problem solving or for performing complex tasks. This requires that a method includes what activities to perform, how to perform them (procedure), what results (artefacts) to develop and how to capture these results (notation) [23]. All methods build on perspectives, values, principles, and categories (with definitions), which are expressed in the method and its elements and which show its underlying theories and rationality.

Different conceptualizations of the term "method" and related terms have been proposed. If there is a close link between procedure, notation, and concepts, the term method component is used [13]. The concept of method component is similar to the concept method chunk [14] and [15] and the notion of method fragment [16]. Methods often consist of an integrated set of several method components, which also could be referred to as methodology [17]. These different method components together form a structure called a framework.

In this paper we focus on the process of a component based method development as a part of the EU-FP7 project "Capability as-a-Service in Digital Enterprises" (CaaS). CaaS proposes to design a business service explicitly considering its delivery context and supports modeling both, the service as such and the application context to facilitate service configuration.

An often used and acknowledged approach in method development is situational method engineering (SME), which basically promotes to adapt methods to the project situation at hand [22]. We argue that our efforts in method development in CaaS Project (see section 3) has overlapping aspects with SME since the general phases of an engineering process were adapted to the specific needs of the application case.

Moreover, the initial CaaS methodology, also called the "base methodology", has been constructed for the specific demands and situation of the project [11]. Finally, due to the component based development approach the component relevant for a specific tasks can be selected "on demand" from a repository and applied correspondingly.

2.2 Experience Reports

This section summarizes experience reports in situational method engineering area and experience reports in method engineering in general. Reflecting on the practices of method application and presenting the usefulness of the applied methods in projects is a decisive and necessary activity. In the literature there are only a few publications reporting from the topics of method engineering experiences, such as method application, realized business value, stakeholders of the method as well as the development process of the method itself [10].

Most of the work in the field of practices from method engineering is being carried out in situational method engineering area, which *encompasses all aspects of creating a development method for a specific situation* [11]. In this respect [7] outlines both the theory of situational method engineering as well as its application in terms of industrial case studies and evaluates possibilities of applying SME with method fragments from OPEN Process Framework (OPF) repository. Likewise [8] reports on experience in the application of a method engineering approach in practice by constructing a situation specific method for a small company. The study shows that not only big enterprises need methods but also small companies benefit from method application and model-based documentation. Finally, in [9] different articles report on the works and experiences gathered by applying the situational method engineering.

Apart from the practices of method engineering, literature reveals several experience reports on applying frameworks and methods in projects. [4] aligns two reference architecture frameworks, TOGAF and NATO architecture framework (NAF) and reports from the implementation experiences in Norwegian Army Forces. Based on the i* framework [5] defines a method and presents experiences on the usage of the framework in large projects from the stakeholder and modeler point of view. Finally [6] reports results from the application of Enterprise Knowledge Development (EKD) Method in various domains and discusses next generation method improvements based on the observations [6].

3 Application Case

Work in this paper is based on the methodology development which is a part of the EU-FP7 project "Capability as-a-Service in Digital Enterprises" (CaaS). The main goal of the CaaS project is to facilitate a shift from the service-oriented paradigm to a capability delivery paradigm. The CaaS project aims to facilitate configuration of business services and development of executable software to monitor the fitness of purpose of these services to evolving business contexts. The CaaS project will deliver the Capability Driven Development (CDD) approach which is supposed to include

methodology, tools and runtime environment. In order to ease adaptation of business services to new delivery contexts, changes in customer processes or other legal environments, the CaaS approach is to explicitly define (a) the potential delivery context of a business service (i.e. all contexts in which the business service potentially has to be delivered), (b) the potential variants of the business service for the delivery context and (c) what aspect of the delivery context would require what kind of variation or adaptation of the business service. This requires development of a new methodic framework supporting capability-driven design in the three industrial cases in CaaS. The CaaS methodology for capability-driven design and development will consist of various components addressing different modelling aspects, such as context modelling, business services modelling, pattern modelling and capability modelling.

According to the definition developed in CaaS (see [18]), a capability is the ability and capacity that enable an enterprise to achieve a business goal in a certain context. Thus, a capability always is defined by specific business services, a defined application context for these business services and goals of the enterprise to be reached. Technically, the model of a capability consists of

- strategic objectives or business goals related to the capability or motivating the creation of the capability. These objectives should be specified in a precise, measurable and accepted way, for example by using enterprise modelling techniques and by elaborating a goal model.
- the business service(s) offered to customers within the capability. In CaaS, the business service(s) have to be specified using a model-based approach. Currently, the focus is on process-oriented approaches.
- the specification of the potential application context where the business service is supposed to be deployed. This specification also has to capture at what points in the process what variation will have to happen. The specification of the capability's potential deployment contexts is captured in a context model.
- an IT-based solution for executing the business service in the defined context, including all variations of the solution for different context instances. The context only defines the switching between variants and potential parameterization, but not the generation of new variants.
- patterns specifying reusable design-time or run-time elements for reaching business goals under specific situational contexts. The run-time patterns are also called capability delivery patterns. The CaaS methodology will provide a method component for identification, elicitation and representation of patterns.

The CaaS methodology has to cover development of all the above parts of a capability model. Methodology development in CaaS is performed in a dedicated work package with four academic partners responsible for different methodology parts. The CaaS methodology will be developed in several versions:

- CaaS base methodology: the main purpose of the initial CaaS methodology, also called "base methodology", is to support the industrial use cases in developing initial capability models, i.e. the business services to be considered in the use cases including their context. For this purpose, the base methodology will cover only selected ways of capability modelling and provide method components supporting these selected ways.

- CaaS methodology: the “regular” CaaS methodology will support a wider selection of capability development processes and extend the base methodology also towards capability delivery and runtime adaptation
- CaaS method extensions: each of the industrial cases in CaaS are supposed to develop extensions of the regular CaaS methodology
- Final CaaS methodology: one of the final results of the CaaS project will be a final version of the CDD methodology including the method extensions and packaged for use outside the CaaS project.

The report about the method development process in section 4 and the experiences presented in section 5 originate from work on the base methodology in the CaaS project.

4 Method Development Process

Within the application case described in section 3, the process for development of the CaaS method and its components roughly follows the general phases of an engineering process: scope setting, requirements analysis, design, implementation and test - with several iterations included in these phases. However, all these phases are adapted to the specific needs of the application case and heavily influenced by the method conceptualization used (see section 5.1). Moreover, the scope of the method development so far is limited to the CaaS base methodology, i.e. future work on other method versions (see section 3) will probably lead to more experiences and refinements of the phases. This section gives an overview of the overall method development process, while section 5 describes the most important phases in detail.

The process was started with organizational and technical preparations as depicted in Fig. 1. The organizational preparations had the purpose to initialize the method development and included formation of the development team, defining the responsibilities of the team members, agreeing on schedule and clarifying available resources. Most of these organizational issues were included in the description of the CaaS methodology work package and confirmed in a kick-off meeting for the method development. The technical preparations were directed to identifying and agreeing on frame conditions. For CaaS, this included the purpose of the method, a set of requirements and the capability meta-model, which were a result of the previously completed requirements work package. The requirements work package also defined four principles which the base methodology has to follow [18]:

- The methodology should not be a monolithic block but component-oriented in order to allow for flexible and situative use of selected method components
- Integration of existing methods or method components should be given preference before substituting them
- CaaS should not develop a single methodology mandatory for all business cases but a reference methodology ready-to-use and pathways from this reference methodology to proprietary methodologies
- All types of models, i.e. pattern, context, process and enterprise models, should be based on the same meta-model

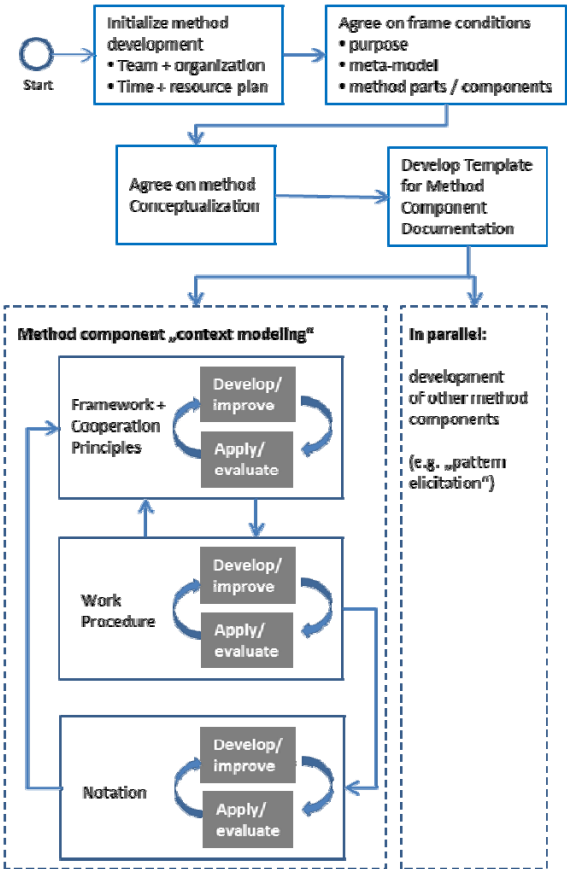

Fig. 1. Method development process

After these preparations, the actual work on the method started with discussing different method conceptualizations and agreeing on one conceptualization to use. The selected conceptualization has a component-oriented method view and is described in section 5.1 in more detail. The decision in favor of a component-oriented method view mainly was motivated by the fact that many different parts has to be accommodated in the CaaS methodology, as described in section 3. In order to provide an aid for developing the different method components, a template for documentation of method components was developed. The template basically is a document with pre-defined chapters and short descriptions of the content supposed to be described in these chapters. The template was supposed to ease the coordination between the different distributed method component developers.

The development process for the different method components happened in different parallel activities performed by different groups from the method development team. This paper will focus on the development process for the context modeling component only. For this component, the framework and the cooperation

principles were developed first. Afterwards, the work procedures for the different steps defined in the framework were elaborated. The last step was to elaborate the notation. All the three steps included several iterations of develop/improve and apply/evaluate activities.

The background for developing the context modeling method component was twofold: on the one side there were the experiences of the development team members in context modeling from previous projects and the knowledge of the state of the art in this area. Unfortunately, the term "context" is used in different meanings in computer science[12]. Hence, there is no single established practice for context modeling in general. On the other side, practical experience in CaaS context modeling was gathered in the CaaS use cases. The first modeling attempts were a more explorative study which concepts from other context modeling areas could be reused; the later attempts were more systematic and resulted in the initial method idea.

5 Development Steps and Experiences

This section will describe the method development steps performed and the experiences gathered during this development.

5.1 Agree on Method Conceptualization

The way methods and method components are described within CaaS is an extension of the method conceptualization proposed by Goldkuhl et al. [19]. Goldkuhl et al. state that a comprehensive method description should describe the perspective, framework, cooperation principles and all method components. Fig. 2 illustrates how these elements of the method conceptualization are related.

- Method components: A method component should consist of concepts, a procedure and a notation. The concepts specify what aspects of reality are regarded as relevant in the modeling process, i.e. what is important and what should be captured a model. These relevant concepts should be named in the method component and explained if necessary. The procedure describes in concrete terms how to identify the relevant concepts in a method component. It may also cover prerequisites and resources. The notation specifies how the result of the procedure should be documented. As a rule, this must provide appropriate expressions for each concept and for the potential relationships between them. In graphic notations, these are the symbols to be used.
- Framework: the method framework describes the relationships between the individual method components, i.e. which components are to be used and under what conditions, as well as the sequence of the method components (if any).
- Forms of cooperation: many modeling tasks require a range of specialist skills or cooperation between different roles. These necessary skills and roles must be described, along with the division of responsibilities between the roles and the form of cooperation. The cooperation form also includes who will take

responsibility for each task or method component, and how the collaboration will be organized.

- Perspective: every method describes the procedure for the modeling process from a particular perspective, which influences what is considered important when developing a model. This perspective often is related to the aims and purpose of the method.

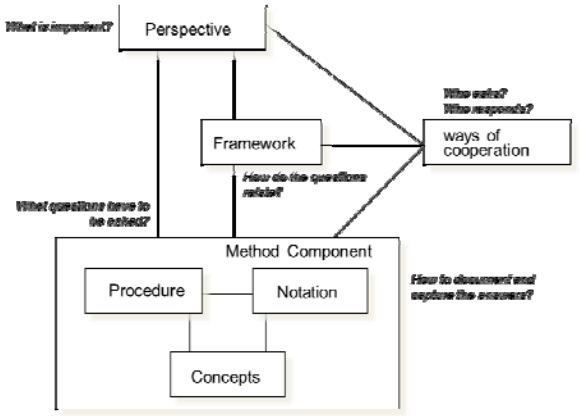

Fig. 2. Method components according to Goldkuhl et al. [19]

The extensions made for CaaS were made with the intention to further operationalize the use of the above method conceptualization. The discussion in the method development team, which included two members familiar with Goldkuhl's conceptualization and five members not familiar with this approach, showed that some terms needed clarification. The following modifications and extensions were made based on the results of these discussions:

- The term „perspective" sometimes confuses method users. This is why the template contains the „purpose" of the method instead.
- Instead of using the term "framework" as proposed by Goldkuhl, we use "overview to method components". The reason for this change is that many of the envisioned method users found the term "framework" misleading.
- The procedure within a method component has been further refined. It contains steps with input, output and tool support
- A method component often includes a number of work steps and sometimes it even contains other method components. Thus, we assume that a method component can include method components, which can be described in the same structure as a complete method: purpose (perspective), framework, cooperation principles and method (sub-)components.

5.2 Define Method Documentation Template

In order to prepare for a uniform or at least similar way of describing the different method components, a template for documenting method components was defined.

The document template is based on the agreed method conceptualization (section 5.2) and defines structure and content of the method documentation. The following structure of the method template was defined:

- Chapter 1 Introduction: Overview to the CaaS methodology in general and brief summary of the method or method component described in the document
- Chapter 2 Purpose and Preconditions: introduction to the purpose of the method and the preconditions required for using it
- Chapter 3 Overview to method components: overview to the different method components and recommended sequence of using them
- Chapter 4 Cooperation principles: Competences, roles and organization structures needed to use the method
- Chapter 5 Method component: one section for each method component describing procedure of working, notation to document the results and important concepts. The procedures consist of steps with inputs, outputs and tool support.
- Chapter 6 Example: an example showing the results of each method component and the overall method

When using the template for documenting a CaaS method all chapters should be with content according to the instructions and guidelines given in these chapters.

5.3 Develop Work Procedures

Development of the work procedures can be classified into three phases. Before the actual development concerning the context modeling began, some effort had been done in the area of modeling context and interpretation of the term within the CaaS Project [12, 21] which basically summarized the relevant work up-to-date. The results of these investigations have been used in the *Preparation* phase for different purposes such as extracting the important terms and concepts in the domain of context and context modeling (see Fig. 3). By conducting this activity the project team realized the first steps towards the identification of the method scope since only the concepts are included, which are closely related to the method application context. Likewise, the excluded concepts helped to limit the method scope.

The important terms have been classified and specified as context dimensions, which later on helped to formulate questions that the method should answer. The core of the development process included three main activities that have been executed recursively in the *Initial Development* phase as illustrated in Fig. 3. The first activity defined the questions that are relevant for method application and grouped them in accordance with their focus, i.e. for designing a context model different question sets are applied. The answers to the questions are transformed into tasks to be executed while grouping the questions has supported us to specify the boundaries of these tasks and to define the components of the context modeling method. The last activity in the initial development phase was to identify how to represent important terms and concepts, i.e. what notation to use when executing the tasks. Results of this phase were presented to the CaaS project team responsible for different methodology parts. After collecting the feedbacks a new iteration has begun and a process oriented

method has evolved, which is depicted in Fig. 4. Fig. 4 shows on the left-hand side the components (find variation, design context, operationalize context, monitor & use context) and on the right-hand side the procedure included in these components.

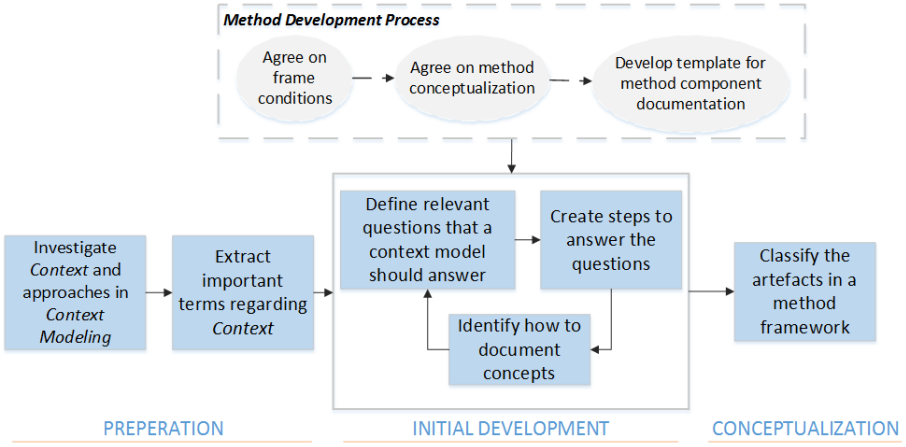

Fig. 3. Phases of Developing the Work Procedure

At that time, the project team agreed on the frame conditions of the method and on the method conceptualization as described in section 5.1. Moreover a method development template has been made available that supports the documentation of different method components in a standardized way. In *Conceptualization* phase the artifacts that have been developed in the last iteration are classified as method components, which comprised of procedures, concepts and notations. Following this the prerequisites and purpose of the method have been formulated. Defining the cooperation principles, i.e. describing the structures and roles within the team using the method, and the enterprise or organization where the method is applied for modeling was a hard task since we had no experience in developing context models.

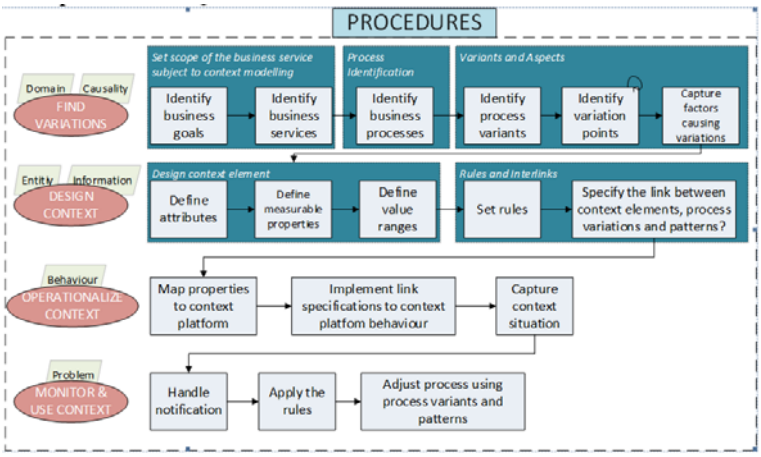

Fig. 4. Important terms, concepts and procedures: A process oriented view

Hence, we were able to describe the competence profile of the context modeler applying the method. The result of this phase was the presentation and discussion of the initial method. An exemplary method component "Find Variations" is illustrated in **Fig. 5**. It should be mentioned that we used placeholders for the method components that need to be further investigated in the following versions of the method. Detailed information on the context modeling method component can be found in [20].

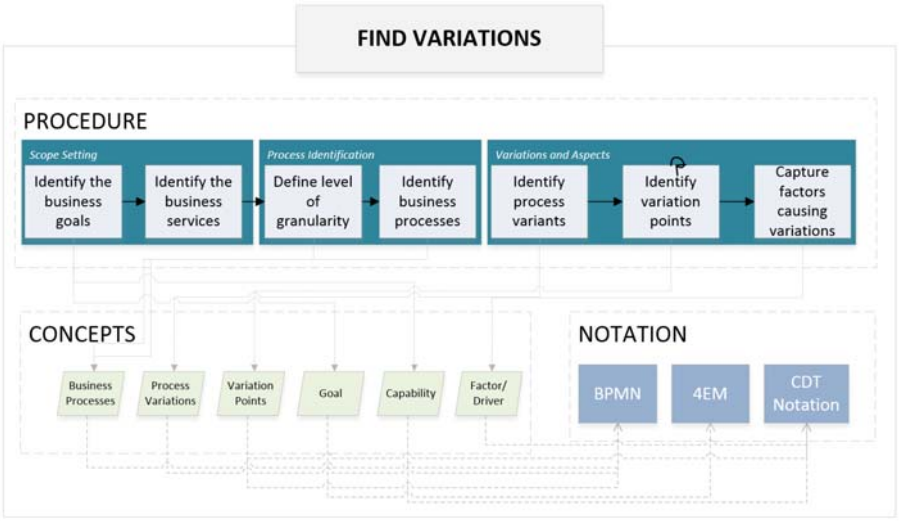

Fig. 5. Initial method: An exemplary method component

5.4 Develop Notation

A visual notation usually consists of a set of graphical symbols, definitions of the meaning of each symbol and compositional rules defining how the symbols may be arranged and diagrams may be composed [2]. The development of the visual notation was an iterative process consisting of alternating elaboration and validation steps. The first elaboration step was to define which concepts and relationships have to be established in the notation. From the first version of the notation, the elaboration process used the principles proposed by Moody [2] for the design of notations. Moody explicitly states that the principles are not only for evaluating and comparing but also for constructing visual notations. The majority of concepts including the relevant relationships could be taken from the meta-model which is presented in [3]: context type, context set, context element, context element range, and measurable property. When developing the context modeling method component (see section 5.3), two additional concepts were defined: variation aspect and variation point. Most relationships between these concepts to be covered by the visual notation also originate from the meta-model:

- Context type <defines> context element
- Context element <is measured by> measurable property

- Context element <has> context element range
- Context set <consists of> context element ranges
- Variation aspects <is related to> variation point
- Context element <causes> variation aspect

The above list of concepts and relationships form the semiotic constructs to be represented in the visual notation. The notation was initially developed by one member of the method development team who is experienced in method development and use of visual modeling languages. This initial development included the creation of shapes for the semiotic constructs and to assign colors and textures to these shapes. Furthermore, the principles proposed by Moody [2] were applied by checking if the initial development was meeting the principles and – if not – how they had to be adapted. The result is summarized in Table 1.

Table 1: Moody's principles and their use during notation development

Moody's principles	**How the principles were taken into account**
1: Semiotic clarity (1:1 correspondence between semantic constructs and graphical symbols)	All concepts have an own symbol; there are no redundant symbols and no symbol overload
2: Perceptual Discriminability (clearly distinguishable symbols)	Different shapes, colors and textures were used to make concepts discriminable
3: Semiotic Transparency (visual representations suggesting their meanings)	Spatial relationships (subset, hierarchy) indicating their meanings were used
4: Include mechanism for dealing with complexity	For context sets and variation aspects, abstraction levels were introduced into the notation
5: Cognitive Integration of information from different diagrams	Variation aspects were integrated into business process models
6: Use full range of visual variables	Besides shape, color and texture also the position (horizontal/vertical) was used for content element range – context element and for variation aspect – variation point relations
7: Dual Coding (use text to complement graphics)	All concepts show the concept name in addition to their specific shape
8: Number of symbols should be cognitively manageable	The number of different semiotic constructs is 7. This is only slightly above the recommended number 6.
9: Different visual dialects for different tasks and audiences	Not supported.

The first validation step was internal validation in the method development team, i.e. the other team members checked the proposed notation using Moody's principles and their own experiences with visual notations. The second validation step involved a practitioner from industry with 20 years of experience in software modelling. The practitioner was familiar with the concepts of the CaaS meta-model, the purpose of context modeling and the work procedure recommended. The result of this step was a minor adjustment in the color of the variation aspect symbol.

Afterwards, the proposed notation was handed over to the developers of the modeling tool, which in the CaaS project includes off-the-shelf components for business process modeling and newly developed functionality for enterprise modeling with the 4EM method and context modeling. The tool developers on the one hand side checked the implementability of the notation and on the other side the compatibility to their understanding of the operational semantics of the CaaS meta-model. The latter resulted in an adaptation of the notation regarding the mechanisms dealing with complexity. Not only context elements but also context element ranges should be part of the context set, since context elements are according to the meta-model only part of the context set since they are only related to context element range.

6 Summary and Future Work

Based on the development of a method component for context modeling in a distributed team, this paper reported on the way of working and experiences collected in method engineering. This section will summarize recommendations and lessons learned.

The first recommendation is related to the overall organization of the method development process. In a distributed team of developers we strongly recommend to treat the method development task like a project and define clear role and task structures. The roles needed are the overall method development responsible and the responsible actors for different method components, which in project management could be considered as project manager and work package managers. Furthermore, there should be an expert for the selected method conceptualization supporting the method component development and the developers as such. To develop a method documentation template and define it as mandatory for all method component also proved valuable. However, we recommend complementing this with additional training for all method developers in how to use the method conceptualization. We expect this to contribute to more consistent component documentation.

Moody's principles for visual notations also proved very valuable but more for the evaluation than for the construction of the notation. The construction of a notation to some extent is a creative process for designing an orchestrated set of semiotic symbols. The process of how to do this is not addressed by Moody's work. The key competence for this creative part from our view is in-depth knowledge of other visual notations and their semiotic symbols, semantics and way of achieving good usability. At least for defining the context modeling notation this was important. One aspect not taken into account by Moody is the implementability of the developed visual notation with the software development environment for the modeling tool. This aspect could be considered as out of scope for the visual notation but nevertheless is crucial for projects attempting to apply the notation in industrial settings. Thus we recommend to use Moody's principles for evaluating and improving visual notations and to add the aspect of implementability.

The method conceptualization by Goldkuhl et al with the extensions and renaming described in section 5.1 proved to be suitable and applicable. The method developers

perceived the conceptualization and its way to decompose a method into different elements as helpful in the overall development process.

The overall method development process described in section 4 worked nicely for our project, but we cannot claim that it is recommendable for all method development projects. The design of this process was based on experiences in previous projects and had rather a pragmatic than a scientific view on method development. We think the approach has the potential to become what in knowledge management is called a good practice. Good practice in this context has the meaning of a proven procedure for reliably completing a defined task [1]. More application cases would be needed to achieve this.

Furthermore, future work will have to focus on improving both, the development process in general and the context method component as such. In the method engineering literature, there is much information on development of method chunks and fragments [24] which most likely also is useful for method components. This aspect needs further investigation. Moreover, the practical use of the CaaS methodology and the context modeling method component will result in a clearer picture about potential other sequences of the method components and the influence of method components on pre/post-conditions of other method components. This will probably lead to adjustment requirements of the approach presented in this paper.

Acknowledgements

We wish to thank the anonymous reviewers for their many comments and recommendations helping us to improve the paper.

This work has been performed as part of the EU-FP7 funded project no: 611351 CaaS – Capability as a Service in Digital Enterprises. This work was also partially supported by the Government of Russian Federation, Grant 074-U01.

References

1. Davenport, T.H. ; Prusak, L.: Working Knowledge. Boston:Harvard Business School Press, 1998.
2. Moody, D. L. (2009) The “Physics” of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering. IEEE Transactions on Software Engineering, Vol. 35, No. 6, pp. 756-779, November/December 2009, IEEE.
3. Zdravkovic J, Stirna J, Henkel M et al. (2013) Modeling Business Capabilities and Context Dependent Delivery by Cloud Services. LNCS vol. 7908. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 369–383.
4. Jørgensen HD, Liland T, Skogvold S (2011) Aligning TOGAF and NAF - Experiences from the Norwegian Armed Forces. PoEM 2011, LNBIP vol. 92. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 131–146
5. Carvallo JP, Franch X (2009) On the Use of i* for Architecting Hybrid Systems: A Method and an Evaluation Report. PoEM 2009, LNBIP vol. 39. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 38–53
6. Stirna J, Persson A (2012) Evolution of an Enterprise Modeling Method – Next Generation Improvements of EKD. PoEM 2012, LNBIP vol. 134. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 1–15

7. Henderson-Sellers, B. (2006) Method Engineering: Theory and Practice. 5th International Conference ISTA'2006, May 2006, Klagenfurt, Austria. GI, pp 13–23
8. Jolita Ralyté (2013) Situational Method Engineering in Practice: A Case Study in a Small Enterprise. CAiSE'13 Forum at the 25th CAiSE, Valencia, Spain, June 20th, 2013. CEUR-WS.org, pp. 17–24
9. Jolita Ralyté, Sjaak Brinkkemper, Brian Henderson-Sellers (eds) (2007) Situational Method Engineering: Fundamentals and Experiences, Proceedings IFIP WG 8.1 Working Conference, 12-14 September 2007, Geneva, Switzerland. IFIP, vol 244. Springer.
10. Hidding GJ, Odell JJ, Parkinson J et al. (1996) Panel: Method Engineering: Experiences in Practice. In: Brinkkemper S, Lyytinen K, Welke RJ (eds) Method Engineering. Springer US, Boston, MA, pp 319–320
11. Brian Henderson-Sellers, Jolita Ralyté (2010) Situational Method Engineering: State-of-the-Art Review. J. UCS 16(3): 424–478
12. Koç H, Hennig E, Jastram S et al. (2014) State of the Art in Context Modelling – A Systematic Literature Review. In: van der Aalst W, Mylopoulos J, Rosemann M et al. (eds) Advanced Information Systems Engineering Workshops, vol 178. Springer International Publishing, Cham, pp 53–64
13. Röstlinger, A. & Goldkuhl, G. (1994) På väg mot en komponentbaserad metodsyn. (in Swedish). Presented at "VITS Höstseminarium 1994", Linköping University, Linköping, Sweden.
14. Ralyté J., Backlund P., Kühn H., Jeusfeld M. A. (2006) Method Chunks for Interoperability. ER 2006, LNCS 4215, pp. 339 – 353, 2006, Springer-Verlag Berlin Heidelberg.
15. Mirbel I., Ralyté J. (2006) Situational method engineering: combining assembly-based and roadmap-driven approaches, Requirements Eng. 11: 58–78.
16. Brinkkemper S. (1995) Method engineering: engineering of information systems development methods and tools, Information and Software Technology, 1995 37.
17. Avison, D. E. & Fitzgerald, G. (1995) Information Systems Development: Methodologies, Techniques and Tools. Berkshire, England: McGraw Hill.
18. S. Bērziša, G. Bravos, T. Gonzalez Cardona, U. Czubayko, S. España, J. Grabis, L. Jokste, J.-C. Kuhr, H. Koc, J. Kampars, C. Llorca, P. Loucopoulos, R. Juanes Pascual, K. Sandkuhl, H. Simic, J. Stirna and J. Zdravkovic , "Deliverable D1.4 - Requirements Specification for CDD. CaaS Deliverable," February 2014.
19. Goldkuhl, G.; Lind, M. and U. Seigerroth (1998) Method integration: the need for a learning Perspective. . IEE Proceedings, Software (Special issue on Information System Methodologies), Vol. 145, Nr 4.
20. H. Koc and K. Sandkuhl , "Task Report 5.2: CaaS Method Component for Context Modeling. CaaS – Capability as a Service for Digital Enterprises, FP7 project no 611351," Rostock University, Germany, 2014.
21. S. Bērziša, S. España, J. Grabis, M. Henkel, L. Jokste, J. Kampars, H. Koç, K. Sandkuhl, J. Stirna, F. Valverde, J. Zdravkovic, "Task Report 5.1: State-of-the-art in Relevant Methodology Areas. CaaS – Capability as a Service for Digital Enterprises, FP7 project no 611351," Rostock University, Germany, 2014.
22. Jolita Ralyté, Rébecca Deneckère, and Colette Rolland (2003). Towards a generic model for situational method engineering. Proceedings CAiSE'03, Springer-Verlag, Berlin, Heidelberg, 95-110.
23. 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.
24. Henderson-Sellers, B., Ralyté, J., Ågerfalk P. and Matti Rossi: (2013). Situational Method Engineering. Springer-Verlag Berlin.