Connecting method engineering knowledge: a community based approach

Isabelle Mirbel

I3S Laboratory
UMR 6070 UNSA-CNRS
Les Algorithmes - Route des Lucioles, BP 121
F-06903 Sophia Antipolis Cedex, France
Isabelle.Mirbel@unice.fr

Abstract. Current practices in the field of information system development reveal a crucial need for spreading and sharing methodological knowledge in addition to existing proposals about formalizing, building and tailoring methods. Currently, the methodological knowledge is mostly shared and spreaded inside an organization by organizing training sessions, attending to conferences and reading manuals. Moreover, it is not very interactive and do not provide efficient support for evolution. The methodological knowledge under consideration ranges from very formal descriptions to informal experience report, empirical know-how and best practices. But in reality, feedbacks about methods in practice are most of the time neither captured nor integrated to the corporate knowledge. And finally, method bases which have been developed to store predefined method fragments to support method tailoring inside organizations have not been very successful in the industrial context. For all these reasons we propose an approach to share and spread methodological knowledge based on the concept of community of practice. Our proposal aims at supporting exchange of knowledge outside of the boundaries of the organization and deepens members knowledge and expertise about methodological knowledge by interacting on an ongoing basis. In this paper, we focus on the lightweight top ontology we propose to specify the core concepts required to qualify any piece of knowledge about method.

1 Introduction

Current practices in the field of Information System Development (ISD) show that methods are almost never suited literally and that there is a wide difference between the formalized sequences of steps prescribed by the method and their real application in practice. Indeed, there is a tension between the 'method-in-concept' (the method as formalized in manual) and the 'method-in-action' (as interpreted by practitioners) [4]. And even among the different practical applications of a specific method, differences exist : methods are often uniquely tailored to the project and organizational characteristics [3]. Moreover, practitioners have a negative perception of methods which are seen as too rigid and

Please use the following format when citing this chapter:

Mirbel, I., 2007, in IFIP International Federation for Information Processing, Volume 244, Situational Method Engineering: Fundamentals and Experiences, eds. Ralyté, J., Brinkkemper, S., Henderson-Sellers B., (Boston Springer), pp. 176-192.

too prescriptive [24]. Even if a method is decomposed into fragments, practitioners must apprehend the method as a whole and understand all its concepts in order to use it. It can have negative impact and discourage practitioners from using methods. Methods are often criticized for the emphasized focus on the method artifacts as such, making them look cumbersome. Methods should not be in the forefront during ISD, it should be viewed as heuristic procedures [11]. Most of the provided approaches have been devoted to method engineering (ME) while method use (MU) needs also dedicated approaches [27,7]. Methods need to be maintained based on reflection from practice, transforming tacit knowledge into explicit knowledge. The importance of tacit knowledge partly explains the low acceptance and use of methods [21]. MU is a learning process in which an individual or an organization creates new knowledge about methods and how to apply them [21].

All these tendencies reveal a crucial need for spreading and sharing methodological knowledge (MU) in addition to existing proposals about formalizing, building and tailoring methods (ME).

Currently, the methodological knowledge is mostly shared and spreaded inside an organization by organizing training sessions, attending to conferences and reading manuals. These basic transmission means require that the practitioners are present (to attend to the training or the conference). It is not very interactive and do not provide efficient support for evolution. Therefore, we propose an approach which allows to collect heterogeneous contributions and to make them evolve collaboratively in a more interactive way.

The methodological knowledge under consideration ranges from very formal descriptions to informal experience report, empirical know-how and best practices. But in reality, feedbacks about methods in practice are most of the time neither captured nor integrated to the corporate knowledge. By providing means to exploit different kinds of methodological knowledge in a homogeneous way, we aim at reducing the gap between 'method-in-concept' and 'method-in-action'. As emphasized in [3], although a method focus is important for obtaining coherence in the organization, it is the practice of people that brings the method to life. Integrating the different kinds of methodological knowledge in a common framework will encourage practitioners to be active in documenting, using and keeping the method alive and the related knowledge up to date. It will also contribute to make methods look less cumbersome and less prescriptive. Among our aims, one is to provide means to integrate emerging feedback from practitioner situations to various materials describing methods.

Exisiting works dealing with feedback from practitioner situations (method rationale) capture, represent and analysis this kind of knowledge in order to support method evolution [27]. Therefore they propose to formalize the method knowldge with the help of a meta model and they provide means to relate the method rationale knowledge to the method meta model in order to explicit it [21]. Such a proposal is willing to support evolution inside the frame of a specific method. Our aim is slightly different as we want to facilitate methodological knowledge sharing accross different methods. We want our approach for instance to allow to bring together knowldge about two different object-oriented notations which would be represented by two different meta models but which may have points to share because of their common paradigm. Moreover we focus on MU and therefore we are focusing on MU rationale [27]. We propose an approach based on an ontology which allows to share concepts among methods, models, notations, meta-models or whatever is related to method knowledge. In addition, we exploit the capabilities of a semantic search engine to bring knowledge together. Indeed, this need to share and exchange methodological knowledge goes beyond the scope of a particular method, project or organization. Therefore, our aim is to provide support for collecting contributions from the whole ISD community (researchers, method engineers, developers, ...) : actors from this community may face similar problems independently of their environment (project, team, organizational unit, ...). In addition, method bases which have been developed to store predefined method fragments to support method tailoring inside organizations have not been very successful in the industrial context [23]. Therefore, we do not want to provide a centralized approach based on a repository of resources about methods. On the contrary, we propose a distributed approach based on a shared representation of methodological knowledge and means to annotate resources with regards to this shared representation.

In this context, our aim is twofold: we want to improve the support dedicated to practitioners in order to help them in their daily activities and we also want to improve the corporation support about methodological knowledge by providing means to integrate various kinds of methodological knowledge and to keep the corporate knowledge up to date.

We contribute in supporting the collect of MU experiences, the recording of comments and observations. These experiences may then be for instance analyzed in the way it is proposed in [27] in order to support evolution in ME, but not only. Our focus is mainly on MU. Therefore we aim at capitalizing knowledge to explain and transmit way of working without only focusing on ME evolution aspects. Our proposal aims at giving unexperimented practitioners a way to learn about best practices related to method. We contribute to experience-based learning and to explicitly collect MU rationale across different use contexts and populations.

The paper is organized as follows. Section 2 presents the concept of community of practice our work is based on. The context of our work is presented in Section 3. The concepts of our top ontology, which is the backbone of our approach, are discussed in Section 4. Section 5 presents a motivating example to demonstrate the feasibility of the proposed approach. Section 6 is a conclusion and a summary of the important points dealt with in this paper and introduces perspectives on the future work.

2 Communities of Practice

The concept of a community of practice (abbreviated as CoP in the following) refers to the process of social learning that occurs when people who have a common interest in some subject or problem collaborate over an extended period to share ideas, find solutions, and build innovations [1].

Recently, CoP have become associated with knowledge management as people have begun to see them as ways of developing social capital, nurturing new knowledge, stimulating innovation, or sharing existing tacit knowledge within an organization [9].

These communities aim at capitalizing individual knowledge, increasing the number of exchanges among people and allowing the identification of domain experts. They help in storing and preserving know-hows through a collective distributed and dynamic process. They create connections among people beyond the geographical and organizational structures.

A community is characterized by its domain (ISD methodological knowledge in our case), its practice (methodological pratice through ISD experiences), its members (all the actors participating in the ISD process), its external environment (the organization the actors belong to, the other CoPs and networks the actors are involved in, ...), its resources (documents and tools about methodological aspects of the ISD process) and its history and life. CoPs differ from business or functional units, from teams and networks because people belong to CoPs at the same time as they belong to other organizational structures. An effective organization comprises a constellation of interconnected CoPs, as these are privileged nodes for the exchange and interpretation of information [12]. The concept of CoP seems promising for methodological knowledge management because it is an effective support of transmission of knowledge from expert to novice, especially in terms of practices. It also makes the participating actors more active and implicated, not undergoing the method in our case. It allows exchange of knowledge outside of the boundaries of the organization and deepens members knowledge and expertise in the CoP domain by interacting on an ongoing basis.

Web-based technologies have allowed the emergency of virtual CoPs. Virtual CoPs (VCoPs) are informal networks, existing outside of any one particular organisation, that support professional practionners to develop a shared meaning and engage in knowledge building among their members by providing opportunities for relationship building and interaction through the use of internet based information and communication technology's as well as other methods [28].

A distributed network of practice (DNoPs) consists of a larger, geographically dispersed group of participants engaged in a shared practice or common topic of interest. CoPs and DNoPs share the characteristics of being emergent and self-organizing, and the participants create communication linkages inside and between organizations that represent a kind of "invisible" net existing beside the formal organizational hierarchies [8].

Virtual CoPs, as well as DNoPs, are concepts which allow to share tacit knowledge [28], albeit to a lesser degree than CoPs : they are for instance particularly suitable for transmitting internet specific soft skills; they provide quick and easy comment systems in blogs and interactive environment provided in online forums for instance and they help feedback mechanisms by reducing first the cost of communication and second the cost of storing and retrieving them efficiently.

But efforts have still to be made to fully exploit the potential of web-based technologies. Dedicated solutions have to be provided to answer the needs related to the domain of knowledge under consideration. Tools have been proposed to support CoP [10] (VCoPs and DNoPs). Some of them are generic enough to be suitable whatever the domain of knowledge is, for instance tools ensuring individual participation or tools ensuring community cultivation [10]. Other tools may be customized to match the requirements of the methodological knowledge.

Formality of the ME design is increased rationally in approaches where it is modeled as an argumentation process organized into specific discourse structure [27]. CoP dedicated tools are promising to answer this need. They may provide good support to adequately capture the context in which decisions are made and support method rationale during ME as well as MU.

Whatever the situation, creation, accumulation and diffusion of knowledge cannot be achieved without a good description of this knowledge which is the purpose of this paper.

3 Context of the work

Our aim is to provide means for practitioners to share and exchange knowledge about method engineering. Similar attempts have been made in the field of requirement engineering [23] and in the field of business interoperability [20] to spread and share successful solutions to requirement engineering or interoperability problems.

In [20], a repository of method chunks is suggested as a knowledge management application for projects within the interoperability domain. In this approach the knowledge about interoperability, based on experience and best practices is formalized in the form of reusable method chunks stored in a method chunk repository. A meta-model for interoperability problem classification is provided to support method chunk qualification and ease the indexing and retrieval of the pieces of knowledge stored in the repository.

In [23], the pieces of knowledge are formalized in the form of patterns which are stored in a common repository. Patterns are mainly described through a problem description, a context, references and keywords. Different versions of patterns are managed in association with feedbacks information in terms of comments and evaluations. An original and polymorphic way to link patterns among them is also provided.

On the contrary of the proposals discussed above, we don't want to encapsulate the heterogeneous resources we are dealing with into chunks or patterns to get uniform pieces of knowledge. We rely on annotation mechanisms to qualify them and allow their exploitation.

In addition, we don't propose to keep the knowledge inside a unique repository. We want the resources to be kept at the practitioners side, making them responsible of it.

As in the discussed approaches, we provide a way to classify the pieces of knowledge we are dealing with. We choose to represent this classification as an ontology, as it will be explain in the remaining of this paper, in order to take advantage of the web semantic techniques and tools and fully rely on annotation mechanisms and semantic search engine capabilities.

Our aim is to provide a sound basis to build dedicated knowledge management services (or tools) to support the cultivation of CoPs (or DNoPs) dedicated to methodological knowledge.

In our approach, data is collected from various sources participating in the CoP and stored in a resource description collection belonging to the CoP (semantic storage in Figure 1). It is collected from the practitioners that want to spread their methodological knowledge. As we don't want to centralize the resources but only their descriptions and in order to let all the practitioners be aware about the referenced knowledge, we assume only descriptions of resources are stored in the repository shared by the community.

As our aim is to help practitioners to share and spread methodological knowledge, the descriptions need to be expressed using a commonly agreed set of concepts. It is the purpose of the ME ontology to preserve in the CoP a shared set of terms and concepts about ME. In this lightweight ontology, the shared vocabulary is expressed in terms of concepts, relationships and constraints. When importing new resource descriptions, practitioners browse the ontology and find suitable concepts or enrich the ontology with new concepts or label in order to annotate the resources they are dealing with.

A semantic search engine can easily exploit the ontology to search the resource description collection. Indeed, such an engine supports applications allowing practitioners to exploit the resource description collection and find pointers on methodological knowledge meaningful for them, thus keeping the community aware and alive about each one contribution. The ontology and the search engine constitute the semantic enrichment part of our framework on top of which dedicated knowledge management services may be developed, as summarized in Figure 1. The knowledge management services form the set of tools dedicated to the CoP cultivation. Examples of such knowledge management services are ontology creation and annotation, cooperative knowledge creation to support collaborative problem solving, knowledge retrieval, knowledge dissemination, knowledge visualization, knowledge evaluation, evolution and maintenance.

The architecture we presented has several advantages with regards to our concern:

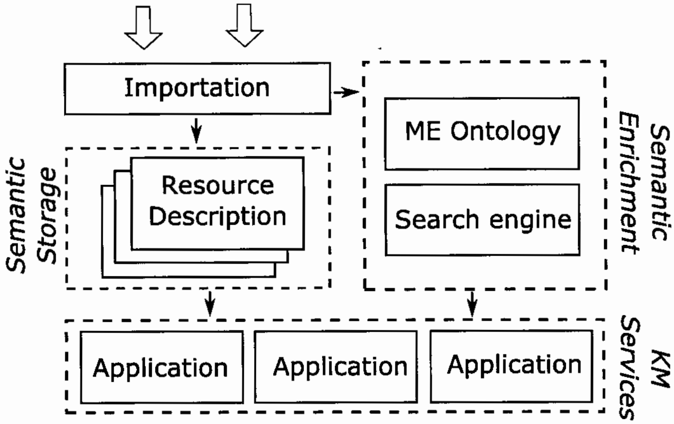


Fig. 1. System Architecture

- It is open and allows easily new practitioners to contribute to the community by exporting the description of their resources in a standard language.
- It is not fully centralized since it does not hold the resources. Resources are kept at the practitioner side, making them responsible of it.
- It allows to reference and qualify in a same way different kinds of knowledge ranging from very formal descriptions to informal experience report, empirical know-how and best practices.

4 A method engineering top ontology

The aim of our work is to provide support to practitioners to help them qualifying the knowledge they own in order to share it with other practitioners. Our approach is based on a lightweight ontology that is shared by all practitioners. We provide a top ontology holding the core concepts we thought required to qualify any piece of knowledge about method. This top ontology will be enrich by the practitioners: they will add new labels for the core concepts and also refine them in order to be able to precisely qualify the knowledge they are dealing with. In this section we discuss the core concepts that constitute the top ontology. In the next section we will show on an example how the top ontology may be refined while importing resource descriptions and how it allows to join heterogeneous resources together.

Our purpose is nor to provide a general ontology about ME or ISD, as it has already been carefully studied in [26,25] for instance, nor to provide a survey of existing approaches about ME or ISD, as good surveys have already been proposed [5,29]. Our aim is to provide the core concepts required in order to support the cultivation of a CoP about ME, that is to say, in a first time, the spreading and sharing about various and heterogeneous pieces of knowledge about methods.

According to the literature about CoP [12], different actors participate in a community which aim is to support the creation, accumulation and diffusion of knowledge *resources* about a domain. The resources or outcomes developed by the CoP (artifacts, stories, routines, documents) constitute the *practice* of the CoP. In the ME domain, we are dealing with *method* of course and *models* which are used to formalize the content of method deliverables as well as to formalize the different steps recommended to built the deliverables. A method may be viewed as a set of loosely coupled method components expressed at different levels of granularity. A *method component* is an autonomous and coherent part of a method supporting the realization of some specific ISD activities. Such a modular view of methods favors the exchange of knowledge about it among different practitioners. The significance of situationality of a method and even more of method components has been clearly enlightened in the literature [30,19]. Therefore, the concept of *context* has to be associated to each practice in order to increase its reusability. These main concepts constitute the backbone of our top ontology. They are summarized in Figure 2 and will be discussed more in detail in the following.

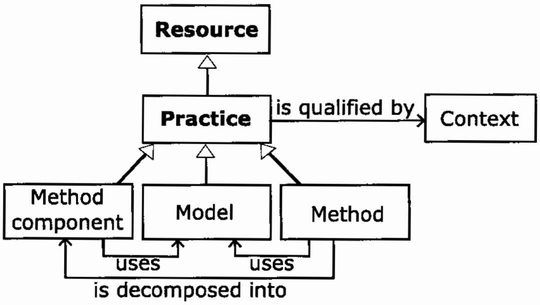


Fig. 2. Top of the ontology

The main information that will help practitioners to understand a *method* usefulness is its *purpose*. One can for instance want to share a feedback about an approach to build a situational method, an experience report about an Xtreme Programming framework or simply advertise a web site about an ISD process. In addition to the purpose of the method, indication about the *manner* the problem is tackled in the method may be interesting to understand the suitability of the piece of knowledge under consideration (while describing it or while searching for methodological knowledge). Method for building methods, for instance, are classified into ad-hoc, by evolution, by extension or by assembly approaches [5]. Requirements engineering methods are usually classified into goal-based, scenario-based or goal-and-scenario based approaches [6]. Similar purposes and manners can be supported by different approaches and different pieces of knowledge. By providing these concepts in the top ontology, as it is shown in Figure 3, it will be possible to join resources together based on these concepts.

A method is considered as a couple of two interrelated models : *product model* and *process model*. The product model of a method defines a set of concepts, relationships between these concepts and constraints for a corresponding schema construction. The process model describes how to construct the corresponding product model. Different notations and languages exist to specify products and processes. Different *paradigms* have been proposed to model products and processes. The relational, functional, intentional and object-oriented paradigms are examples of well-known *product model paradigms*. Activity-oriented, product-oriented, decision-oriented, context-oriented and strategy-oriented paradigms are examples of *process model paradigms*. Different object-oriented notations have been for instance proposed to model products. Different pieces of knowledge, each of them expressed with a different object-oriented model, may be referenced in our CoP. And even if slightly different from the notation point of view, the resources may contain information that could be shared by the different practitioners because of the common paradigm. Figure 3 illustrates the concepts related to the model specification.

A *method fragment* ensures a tight coupling of some process part of a method process model and its related product part [14]. In the product part, also called *product fragment*, the product to be delivered by the method component is captured whereas in the process part, also called *process fragment*, the guidelines allowing to produce the product are given. A method fragment, also called a method bloc or a method chunk, is characterized at least by a name and an intention which specify the goal that the method component achieves. A piece of knowledge may be described as being a product fragment only or a process fragment only or both of them depending on its level of granularity and focus. Indeed, some authors propose two types of method components while others consider only process aspects. Integrated approaches also exist [14]. The resources described with the help of these concepts (method fragment, product fragment and process fragment) consist in guidelines about how to use the meta models (to build models) in addition to references to appropriate parts of product and/or process meta-models. Another way to reuse methodological knowledge is based on generic elements [18]. The *method pattern* concept aims at describing resources which capture generic lows governing the construction and adaptation of methods. Decision-making patterns capturing the best practices in enterprise modeling and domain-specific *process patterns* and *product patterns* [14] are examples of patterns. Fragments as well as patterns may be respectively decomposed into more refined fragments and patterns. Figure 3 summarizes the different concepts suitable for characterizing method components. The relationships between method components and models is summarized in Figure 4.

As our aim is to help practitioners to spread and share methodological knowledge, it is important to provide means for them to express in which context the resource may be useful. Therefore we introduce the concept of *context*.

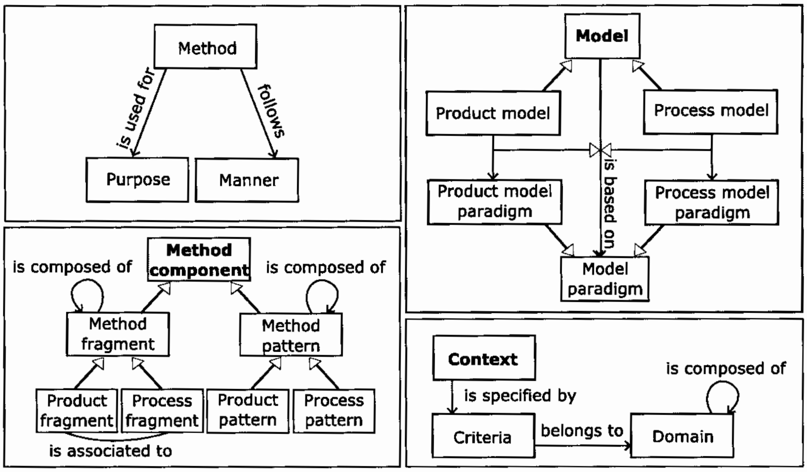


Fig. 3. Content of the ontology

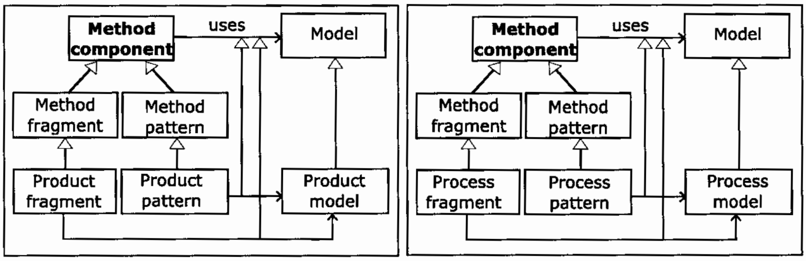


Fig. 4. Relationships among models, fragments and patterns

It is useful for defining potential reuse situations when describing a resource and importing its description into the resource description collection. It is also profitable when searching for resources : it allows to specify the situation in which the resources could be reused. It contributes to better specify what the practitioner is searching for, in addition to the kind of knowledge (model, method, method component, ...) and features about the element under consideration (intention of a method component, paradigm of a model, ...). The context also allows to specify characteristics which are not dependent on the kind of concept under examination and therefore allows to join together heterogeneous knowledge. By heterogeneity we mean resources about different concepts (model, method component, purposes, etc) as well as resources with different levels of formalization ranging from formal descriptions to informal experience report. A context is defined by a set of *criteria* belonging to different *domain* of interest [13]. The organizational perspective (and especially contingency factors [22]) and the human dimension are examples of domain. The practitioner involvement in the ISD project or the time pressure on the project are examples of criterias related to the organizational perspective. The concepts required to specify a context are illustrated in Figure 3.

5 Motivating example

In this section we show on an example how the top ontology we discussed in the previous section can be refined to describe two specific methodological resources in a generic and reusable way and how these resources may be joined together thanks to the ontological knowledge.

Figure 5 shows an example of method chunk extracted from [19,14]. As it is indicated in the descriptor of the chunk, the aim of this chunk is to provide guidelines to build a use-case model from a problem description. This chunk contains a product part and a process part. In the product part, the use-case model the chunk is dealing with is specified. The process part is specified through tactical guidelines. This formalism is part of the NATURE process model and allows to express complex guidelines through a set of steps.

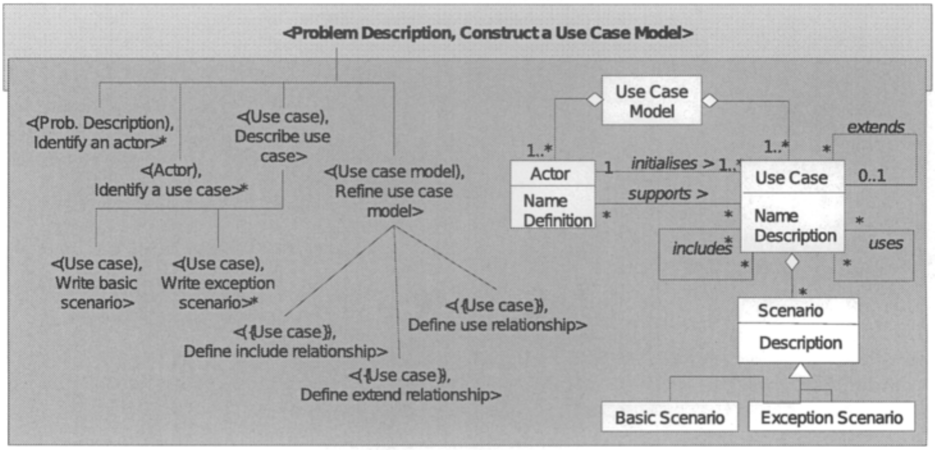

<Problem Description, Construct a Use Case Model>
<(Prob. Description), Identify an actor>*
<(Use case), Describe use case>
<(Actor), Identify a use case>*
<(Use case model), Refine use case model>
<(Use case), Write basic scenario>
<(Use case), Write exception scenario>*
<({Use case}), Define include relationship>
<({Use case}), Define use relationship>
<({Use case}), Define extend relationship>
Use Case Model
1..*
1..*
Actor
Name
Definition
1
initialises >
1..*
supports >
*
*
Use Case
Name
Description
extends
*
0..1
includes
*
*
uses
*
*
*
Scenario
Description
Basic Scenario
Exception Scenario

Fig. 5. A method chunk

The description of this resource in a graphical way is given in Figure 6. The name and intention of the chunk have been represented, as well as the *use-case* and *NATURE* model referred to in the chunk body. A context has been specified to help in understanding the reuse context of the method chunk. It has been described with regards to the *design activities* it is suitable for : *requirement elicitation* and *analysis*. It has also been qualified with regards to the kind of *application domain* it is suitable for: when designing an *information system* or *human computer interfaces*. Of course, in parallel with the specification of this description, the top ontology has to be enriched to take into account the new concepts the chunk contains : *use-case model, NATURE model, Information system, Human computer interfaces, Analysis, Requirement elicitation, design activity* and *Application domain.*

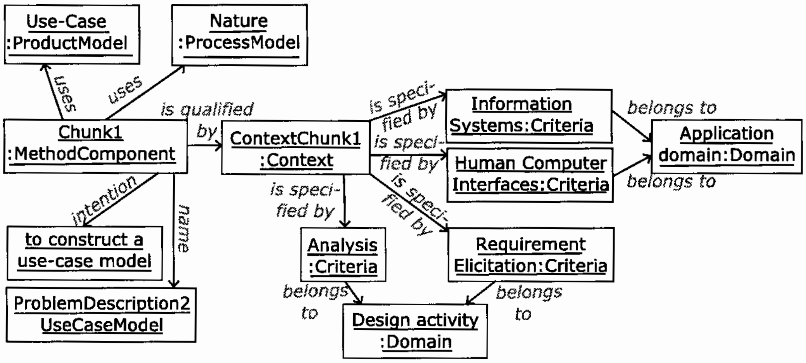
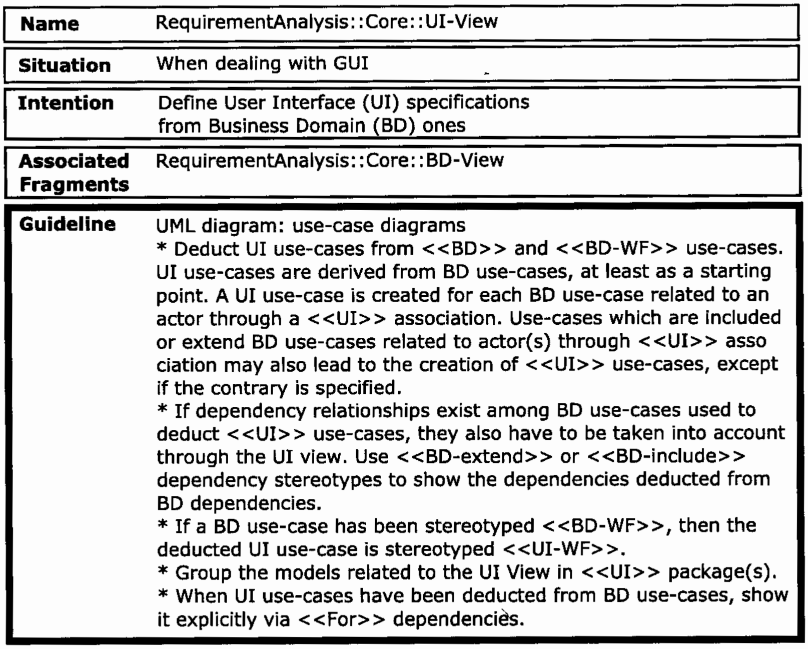


Fig. 6. The chunk description

Figure 7 shows another example of resource about methodological knowledge. It is a method fragment which has been extracted from the JECKO framework [2]. This framework, which is the result of a collaboration between Amadeus SAS and the I3S research laboratory [16], is a context-driven approach to analysis and design to address the problem of adapting methodology to specific development environment. Dedicated method fragments as well as dedicated UML profiles have been proposed to deal with different criteria qualifying the application under development [15,17]. The method fragment shown in Figure 7 is dedicated to applications with a graphical user interface (*situation: when dealing with GUI*) and provides guidelines to help in defining the user interface specification by starting from the business domain specification, as it is explained in the *intention* of the method fragment. The method fragment provides textual guidelines about this specification activity during the *requirement analysis phase* (in this framework fragments are grouped by phases and steps that are recalled in the name of the fragment). It assumes the *use-case model* is used to support this activity.

A graphical view of the description of this resource is given in Figure 8. The name and intention of the fragment have been represented, as well as the *use-case* model referred to in the fragment guidelines. A context has been specified to help in understanding the reuse context of the method fragment. It has been described with regards to the *design activity* it is suitable for : *requirement analysis*. It has also been qualified with regards to the kind of *application domain* it is suitable for : when dealing with a *graphical user interface*.

Again, in parallel with the specification of the method fragment description, the top ontology is enriched to take into account the new concepts the chunk refers to : *Requirement analysis*. The part of the top ontology enriched to take



Name	RequirementAnalysis::Core::UI-View
Situation	When dealing with GUI
Intention	Define User Interface (UI) specifications from Business Domain (BD) ones
Associated Fragments	RequirementAnalysis::Core::BD-View
Guideline	UML diagram: use-case diagrams * Deduct UI use-cases from <<BD>> and <<BD-WF>> use-cases. UI use-cases are derived from BD use-cases, at least as a starting point. A UI use-case is created for each BD use-case related to an actor through a <<UI>> association. Use-cases which are included or extend BD use-cases related to actor(s) through <<UI>> association may also lead to the creation of <<UI>> use-cases, except if the contrary is specified. * If dependency relationships exist among BD use-cases used to deduct <<UI>> use-cases, they also have to be taken into account through the UI view. Use <<BD-extend>> or <<BD-include>> dependency stereotypes to show the dependencies deducted from BD dependencies. * If a BD use-case has been stereotyped <<BD-WF>>, then the deducted UI use-case is stereotyped <<UI-WF>>. * Group the models related to the UI View in <<UI>> package(s). * When UI use-cases have been deducted from BD use-cases, show it explicitly via <<For>> dependencies.

Fig. 7. A method fragment

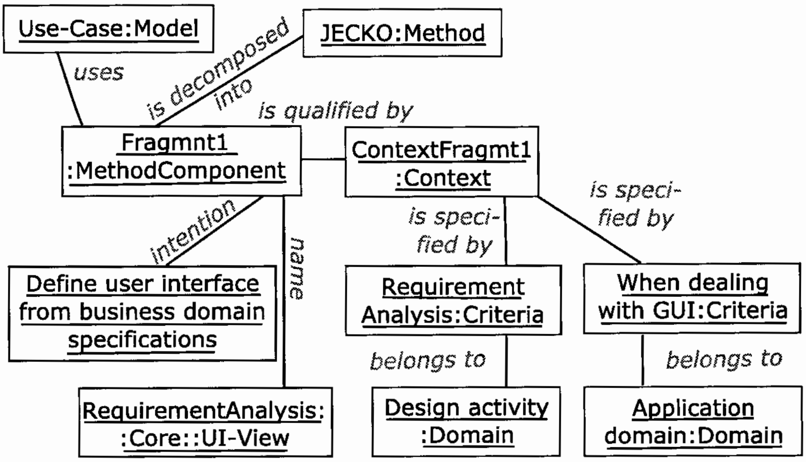

Use-Case:Model
JECKO:Method
uses
is decomposed into
is qualified by
Fragmnt1 :MethodComponent
ContextFragmt1 :Context
is specified by
is specified by
intention
name
Define user interface from business domain specifications
Requirement Analysis:Criteria
When dealing with GUI:Criteria
belongs to
belongs to
RequirementAnalysis::Core::UI-View
Design activity :Domain
Application domain:Domain

Fig. 8. The fragment description

into account the concepts required to qualify the chunk and the fragment under concern is shown in Figure 9.

With the help of a semantic search engine the collection of resource descriptions can be searched and similarities between the two method components

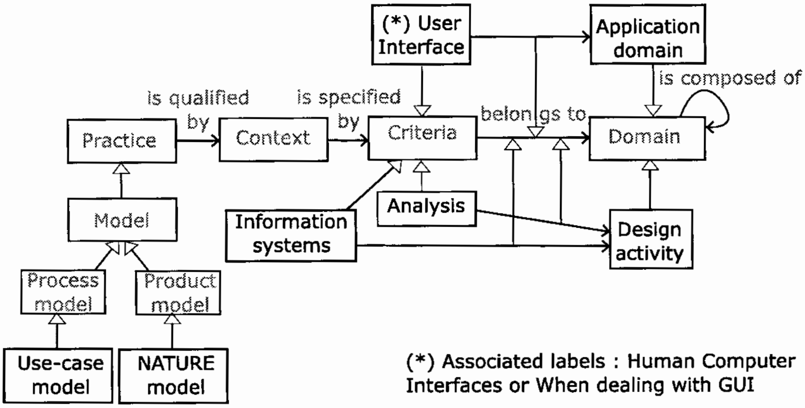


Fig. 9. The extended ontology

under consideration could be found. The chunk and the fragment would be considered as close because of their common product model of course, but also because they share part of their context (the user interface concept) and because they cover design activities which are join together by generalization relationships (analysis in the chunk context and requirement analysis in the fragment context).

The proposed top ontology makes a difference to current practice because pieces of knowledge not expressed with the same formalism (NATURE process model in one case textual description in the other), not belonging to the same method may still be joined together and presented to practitioners as two possible answers to a single query.

Compared to a localized Google search engine, our approach allows experts to provide, through the refinement of the top ontology, a set of meaningful concepts to describe the ME domain. It could be especially useful for novice practitioners when looking for resources about methods. Moreover, as we deal with heterogeneous kinds of resources ranging from very formal descriptions to informal experience report, empirical know-how and best practices, it could be difficult to index all of them in a homogeneous and meaningful way. Keywords extracted from the text body would not lead to good retrieval results. By following an ontology-based approach to capture domain concepts we aim at improving resource indexing and retrieval.

Thanks to our lightweight top ontology, which is very basic and concise, each practitioner participating in the community can contribute to the methodological knowledge description by refining the top ontology when exporting descriptions of resources. Each CoP will then obtain an ontology which reflects the knowledge cultivated in this CoP. It will make possible to reference heterogeneous pieces of knowledge in a common framework and let practitioners exchange pieces of knowledge whatever the formalism, the granularity, the level of detail of what they want to spread and share are. The ontology also constitutes a guidance, especially for novice practitioners, when searching for resources. It provides a set of core elements to start from to express a need and a reuse situation.

6 Conclusion

In this paper we presented an approach to help practitioners in spreading and sharing methodological knowledge. We exploited the concept of CoP, which allows exchange of knowledge outside of the boundaries of the organization and deepens members knowledge and expertise in the CoP domain by interacting on an ongoing basis.

Our concern is more precisely on providing an interactive and evolutive support to integrate heterogeneous contributions about ME to encourage practitioners to be active in documenting, using and keeping the method alive.

In this context, our contribution deals with a lightweight top ontology to specify the core concepts required to qualify any piece of knowledge about method. This top ontology is refined by practitioners when they import resource descriptions which constitute the CoP practice. It allows practitioners to join together resources with different levels of formalization and notations. An access to a large amount of practices, not inevitably partitioned by models, notations or approaches is gained. Moreover, the ontology provides a guidance for defining practitioner need and specifying practitioners situation when searching the resource description collection.

Future works will proceed in both theoretical and practical directions. The theory will focus on providing core concepts to describe actors and ISD activities and techniques in order to extend the scope of our top ontology and better handle all the key element of a CoP [12] devoted to ME. Associated dedicated tools will also be studied. The practical work will consist in testing this approach through several case studies.

References

1. http://fr.wikipedia.org.
2. http://www.i3s.unice.fr/ mirbel/jecko/jecko.html.
3. A. de Moor and H. Delugach (2006) Software Process Validation: Comparing Process and Practice Models. 18th Conference on Advanced Information Systems Engineering - CAISE 2006 Workshop on Exploring Modeling Methods in Systems Analysis and Design - EMMSAD.
4. B. Lings and B. Lundell (2004). Method-in-Action and Method-in-Tool: Some Implications for CASE. 6th International Conference on Enterprise Information Systems - ICEIS 2004, Porto, Portugal.
5. C. Rolland (2005). L'ingénierie des méthodes : une visite guidée. e-TI - la revue électronique des technologies d'information, http://www.revue-eti.netdocument.php?id=726, (1).

6. C. Rolland and C. Salinesi (2001). Ingénierie des systemes d'information. Hermes.
7. A. Järvi and H. Hakonen and T. Mäkilä (2007). Developer driven approach to situational method engineering. IFIP WG8.1 working conference on situational method engineering: fundamentals and experiences (ME07), Geneva, Switzerland.
8. E. Hustad (2007). A conceptual framework for knowledge integration in distributed networks of practice. 40th Hawaii International Conference on System Sciences.
9. E. Wenger (1998). Communities of practice: learning, meaning, and identity. Cambridge University Press, Cambridge, U.K. and New York, N.Y.
10. E. Wenger, N. White, J. Smith, K. Rowe (2005). Technology for communities. CEFRIO.
11. E. Hustad (2007). A conceptual framework for knowledge integration in distributed networks of practice. 40th Hawaii International Conference on System Sciences.
12. E. Wenger (1998). Communities of practice: learning, meaning, and identity. Cambridge University Press, Cambridge, U.K. and New York, N.Y.
13. E. Wenger, N. White, J. Smith, K. Rowe (2005). Technology for communities. CEFRIO.
14. F. Karlsson and K. Wistrand (2006). Combining method engineering with activity theory : theoretical grounding of the method component concept. European Journal of Information Systems, (15), 2006, pp. 82-90.
15. G. Vidou, R. Dieng-Kuntz, A. El Ghali, C. Evangelou, A. Giboin, A. Tifous and S. Jacquemart (2006). Towards an Ontology for Knowledge Management in Communities of Practice.. 6th International Conference on Practical Aspects of Knowledge Management - PAKM 2006, Vienna, Austria, 2006, pp. 303-314.
16. I. Mirbel (2004). A polymorphic context frame to support scalability and evolvability of information system development processes. 6th International Conference on Enterprise Information Systems - ICEIS 2004, Porto, Portugal, April, 2004, pp. 131–138.
17. I. Mirbel and J. Ralyté (2006). Situational method engineering : combining assembly-based and roadmap-driven approaches. Requirement Engineering Journal, 11(1), 2006, pp. 58–78.
18. I. Mirbel and V. De Rivieres (2003). Conciliating User Interface and Business Domain Analysis and Design. 9th International Conference on Object-Oriented Information Systems - OOIS 2003, Geneva, Switzerland, September, 2003, pp. 383-399.
19. I. Mirbel and V. de Rivieres (2002). Adapting Analysis and Design to Software Context: The JECKO Approach.. 8th International Conference on Object-Oriented Information Systems - OOIS 2002, , Montpellier, France, Sept., 2002, pp. 223-228.
20. I. Mirbel and V. de Rivières (2003). UML and the unified process. IRMA Press, 2003.
21. J. Ralyteé and R. Deneckere and C. Rolland (2003). Towards a Generic Model for Situational Method Engineering.. 15th International Conference on Advanced Information Systems Engineering (CAISE 2003), 2003, pp. 95-110.
22. J. Ralyté (2001). Ingénierie des méthodes à base de composants. Université Paris I - Sorbonne, January, 2001.
23. J. Ralyté and P. Backlund and H. Kühn and M. A. Jeusfeld (2006). Method Chunks for Interoperability. 25th International Conference on Conceptual Modeling, ER 2006, 2006, pp. 339-353.

24. J.P. Tolvanen (1998). Incremental Method Engineering with Modeling. Tools: Theoretical Principles and Empirical Evidence. University of Jyvskyl, Finland, 1998.
25. K. van Slooten, B. Hobbes (1998). Characterizing IS development projects. IFIP WG8.1 Working Conference on Method Engineering: Principle s of method construction and tool support, Great Britain, 1996, pp. 29-44.
26. L.H. Jean-Baptiste, C. Salinesi and G. Fanmuy (2005). Sharing Methodological Knowledge with REGAL: "Requirements Engineering Guide for All". 13th IEEE International Conference on Requirements Engineering, Paris, France, 2005, pp. 461-462.
27. M. Bajec, D. Vavpotic and M. Kirsper (2004). The scenario and tool-support for constructing flexible, people-focused system developement methodologies. 13th International Conference on Information Systems Development - ISD 2004, Vilnius, Lituania, September.
28. M. Leppanen (2005). Conceptual Analysis of Current ME Artifacts in Terms of Coverage: A Contextual Approach. 1st International Workshop on Situational Engineering Processes Methods, Techniques and Tools to Support Situation-Specific Requirements Engineering Processes (SREP), in conjunction with 13th IEEE International Requirements Engineering Conference, pp. 75-90.
29. M. Leppanen (2006). Towards an Ontology for Information Systems Development. 18th Conference on Advanced Information Systems Engineering - CAISE 2006 Workshop on Exploring Modeling Methods in Systems Analysis and Design - EMMSAD.
30. M. Rossi and B. Ramesh and K. Lyytinen and J.P. Tolvanen (2004). Managing evolutionary method engineering by method rationale. Journal of the Association for Information Systems, 5(9), pp. 356-391.
31. M.P. Zarb (2006). Modelling participation in Virtual Communities of Practice. London, UK.
32. S. Brinkkemper (1996). Method Engineering: Engineering of Information Systems Development Methods and Tools. Information and Software Technology, 38(4), 1996, pp. 275-280.
33. S. Brinkkemper and M. Saeki and F. Harmsen (1998). Assembly techniques for method engineering. 10th International Conference on Advanced Information Systems Engineering, Pisa, Italy, 1998.