Contextualization of method components

Elena Kornyshova, Rebecca Deneckere, Bruno Claudepierre

Contextualization of Method Components

Elena Kornyshova, Rébecca Deneckère, Bruno Claudepierre
Centre de Recherche en Informatique
Université Paris I – Panthéon Sorbonne
90, rue de Tolbiac, 75013, Paris
{Elena.Kornyshova, Rebecca.Deneckere, Bruno.Claudepierre}@univ-paris1.fr

Abstract— ***Method Engineering (ME) for Information Systems (IS) is a response to the necessity to better fit methods with development activities requirements. Situational method engineering allows defining new methods constructed on the fly following the situation at hand. However, in the reviewed literature, the situation is not always described and there is no proposed approach to handle the specific context of method components. This paper provides a detailed vision of context and a process for contextualizing methods in the IS domain. Our proposal is illustrated with a case study of project portfolio management in the domain of IT governance.***

Keywords: method engineering; method component; contextualization

I. INTRODUCTION

An information systems development methodology (ISDM) is a set of ideas, approaches, techniques and tools which system analysts use to help them integrating organizational needs into an appropriate Information System. However, it is now apparent that no universal method that could be applied to any information system development really exists. Method engineering (ME) represents the effort to improve the usefulness of ISDM by creating an adaptation framework whereby methods are created to match specific organizational situations. ME aims to find solutions to the construction, improvement and modification of the methods used to develop information systems. One of the ME fundamentals for optimizing, reusing, and ensuring flexibility and adaptability of these methods is their decomposition into modular parts (Situational method engineering – SME) [1] [2].

[3] describes the process of SME as composed of three parts, (a) the decomposition of methods into components (Method Component – MC) which are stored in a method repository, (b) the retrieval of components that better match the projects specificities and (c) the construction of a new method with these selected components. The existing SME approaches use a lot of different kinds of queries. Some use similarity measures [4] or semantic similarities by ontology [5] in order to study the matching between components and requirements. Some make their queries on the component endeavour [6] whereas others use the application of requests on the goals [1]. However, the cost of a project can increase as there are more and more artefacts to take into account, which induce a combinatory explosion of all the values to calculate [7]. Finally, the selected components may be quite similar and the method engineer will still have to choose manually between them. To avoid some of these drawbacks, in [7], the component selection is enhanced with multicriteria decision-making techniques. Based on these techniques, [7] offers a prioritization which uses a typology of project characteristics.

Many approaches of SME consider the notion of context in order to guide the selection of a method component from a repository according to a given situation. They deal with different kinds of context factors characterizing situations of IS development projects and offer various methodologies for using context. However, none of these approaches suggest a methodology allowing to define a set of concrete context characteristics for a given method.

Our goal in this paper is to propose (i) a generic model of context based on the state-of-the-art and (ii) an IS development methods contextualization process.

In our view, the context is a set of characteristics which describes situations of a method application. The context is defined for an IS development method and its components. Each method component is then delineated by concrete values of these characteristics.

In this paper, we focus on the contextualization of method components. Therefore, we introduce the frame of contextualization, we present the context model, the context typology and the process to construct the context characteristics set for a given method. We illustrate our proposal with an application which deals with an IT portfolio management guidance.

All processes in this work are formalized with the MAP model which is commonly used in the ME field [8]. In our proposal, this formalism is used to represent the contextualization process in an intentional way. In the case study, it is used to represent the organization of the method components (the links between them).

The notion of method component is described in Section 2. Section 3 proposes a state-of-the-art on the notion of context. Section 4 proposes a conceptualization of this notion and we illustrate our proposal with an example in Section 5. A conclusion and a proposal of further works are done in Section 6.

II. METHOD COMPONENTS

A development process cannot fit all the existing problems and development contexts. This assumption has lead to the development of the ME domain, and more particularly of SME.

The discipline of SME promotes the idea of retrieving, adapting and tailoring modular parts, rather than complete

methodologies, to specific situations. There are various representations of modular parts: fragments [9], chunks [10], components [11], OPF fragments [12] and method services [3] [5] [13]. A comparison of these different kinds of modular parts may be found in [3] and [14].

Method fragment approach [9]. The method component definition consists in encouraging a global analysis of the project while basing itself on contingency criteria. Projects and situations are characterized by means of factors associated with the methods.

Method chunk approach [15]. The chunk approach expresses projects requirements (the context) as a *requirements map*, which is used to test the similarity between requirements and existing components.

Method component [11]. The component description contains its "rationale"; its matching with the context is performed by goal analysis.

OPF fragment [12]. In the OPEN Process Framework (*OPF*), the fragment is generated from an element in a prescribed underpinning meta-model.

Method service [5]. The method service approach uses an identification part that defines the purpose of the service. The component retrieval is thus done by using goal, actor, process, and product ontologies.

Our view of a component has been described in [3]. It includes both the intention oriented approach of the chunks [15] and the notion of method services [5]. We then suggest modelling method components as shown in Fig. 1.

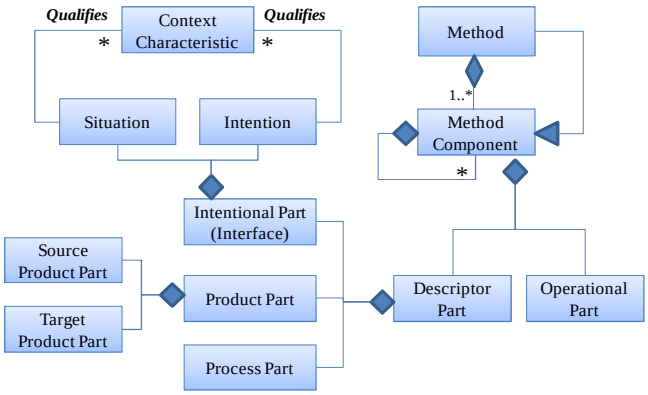

Figure 1. Method component meta-model.

Each *method component* may be a representation of a components composition as they are expressed with different granularity, at various levels of abstraction. For instance, a component may be an entire method that can be decomposed into other less complex components (which, in turn, may also be decomposed into other more simple components).

A *method component* is composed of two parts. The *descriptor part* aims at documenting, retrieving, composing, and invoking the related implementation part.

The *descriptor part* contains three different views: intentional, product and process.

The *intentional part* (sometime called the *interface* of a component) provides information to identify the components. It contains the *intention* to achieve (which describes the general purpose of the component) and the *situation* in which the component may be applied. *Context characteristics* indicate specific conditions to use the component and qualify elements of the intentional part.

The *product part* corresponds to the description of the component input and output product models. The *source product part* defines the product situation before applying the component. The *target product part* defines the result, which must be obtained by the component application.

The *process part* contains guidelines aiming to explain how to apply the component.

III. CONTEXT IN ME: STATE-OF-THE-ART

This section describes a state-of-the-art on context-awareness and on the context of method components.

A. *Cross domains application of Context-awareness*

Context models are multidisciplinary and have been proposed in several areas [16]. The linguistic research is concerned with analyzing the usage context of signs (or words) within a language. Bunt [17] defines five types of context for communication aspects which are respectively:

- *Linguistic*: refers to linguistic material;
- *Semantic:* refers to domain description including objects and properties;
- *Physical*: refers to the environment description in which action or interaction occurs;
- *Social*: refers to the interactive situation which occurs between actors;
- *Cognitive*: refers to the participants' intentions, their evolution relating to perception, production, evaluation and execution.

Context is also formalized using mathematical models. For instance, [18] proposes a cumulative model where the context (*Ctx*) is a timely aggregation of situations. A situation is a state descriptor for a user (*U*) performing a task (*T*) at a time (*t*). The model is depicted by the following formula:

$$Ctx(U,T,t) = \bigcup_{n=1}^{m} (Situation(U,T,t_n))$$

Context awareness is a term originating from pervasive computing, or ubiquitous computing [19]. These systems deal with linking changes in the environment with computer systems, which are otherwise static. Although it is a computer science term, it has also been applied to business theory in relation to business process management issues [20].

There are numerous context-awareness applications when human interactions occur. More related to our study, context models are also proposed for business process reengineering [21], computer science [16], service selection [22] and decision-making within a military situation [23], [24]. In latter

cases, the context model is seen as a way to analyze a given *situation* to guide the way of processing. Thus, context models are mainly used to solve the problem of lacking flexibility and adaptability within processes.

In section IV, we apply this description of the generic concept of context to method components.

B. Method components context

We have identified five main approaches dealing with context in the method engineering field.

Reuse frame. The reuse frame [25] is a framework representing different factors which affect IS development projects. These factors are called *criteria*. Reuse frame allows specifying a context of method fragments reuse, searching method fragments and comparing between them in order to find an alternative fragment to a used one. The reuse frame model includes a reuse situation (which is a set of criteria classified into three dimensions: *organizational*, *technique* and *human*) and reuse *intention*.

Interface. In [26] the method fragment context is defined by its interface which includes a situation and an intention. The situation represents the conditions in which the method fragment can be applied in terms of required inputs product(s). The intention is a goal that the method fragment helps to achieve. Therefore, the interface model includes two elements: the *situation* and the *intention*. These two first approaches have been unified in [27].

Method service context. The method service context [5] aims at describing the situation in project development for which the method service is suitable and defining the purpose of the service. Its model includes *domain* characteristics (project nature, project domain) and *human* (actor), *process* and *product* ontologies.

Contingency factors. Situations (the context) are described by a set of characteristics called contingency factors [28] or project factors [29,30]. These factors are used to define the project situation by assigning values to them. In [28], four categories are given: *domain characteristics* (describing the content of the system), *external factors* (laws and norms), *technical factors* (related to the development platform) and *human factors* (representing the development expertise of people).

Development situation. [31] defines the development situation as an abstraction of one or more existing/future software development projects with common characteristics. This situation is used to characterize the specific projects and to select configuration packages (method fragments). The development situation model includes a characteristics set.

Based on the review of these five approaches, we have identified seven characteristics (context elements) which allow us to compare existing context approaches (See Table I.). This comparison highlights that there is no approach considering all of the possible characteristics.

Moreover, the analysis of these context approaches shows that they do not suggest a way to specify context characteristics.

C. Our proposal

In SME, all approaches are situational, which means they take into account the specific project situation (or *Context*). However, the definition or description of this context is often just superficially addressed.

Our proposal uses the context expressiveness to describe the situation in which a component may be applied. It is then based on the *semantic* type of context previously presented. Moreover, our view of a component includes an intention oriented approach which allows representing the *cognitive* aspect of the context.

The preceding comparative analysis of context approaches shows that they address several aspects of context. However, they do not cover all of them and do not help in the context characteristics specification. Our proposal aims to help the engineer specify these characteristics.

IV. CONTEXTUALIZATION OF METHOD COMPONENTS

A. Enhanced definition of method context

Our goal is to enhance the definition of the context of IS development method for the further selection of components from a repository according to a given situation. In the following we present our vision of context and a process to define the context for a given method.

The context model is presented in Fig. 2. We propose describing the context as a set of characteristics. These characteristics describe situations of a method application. Characteristics are organized into facets for better representation and comprehension. We distinguish two types of characteristics: generic and specific. The first ones are common for most IS engineering projects; the latter ones vary from one project to another. To distinguish between them is important because of their different identification approaches.

TABLE I. COMPARATIVE ANALYSIS OF APPROACHES DEALING WITH CONTEXT IN ME FIELD: CONTEXT ELEMENTS.

Approach	Characteristics							
	Goal/ Intention	***Organiza-tional***	***Technical***	***Human***	***Domain***	***External***	***Process***	***Product***
Reuse Frame	X	X	X	X				
Interface	X							X
Method service context				X	X		X	X
Contingency factors			X	X	X	X		
Development situation	Not specified							

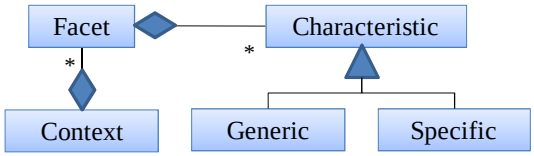

Figure 2. Context model.

Generic characteristics. In order to establish the typology of generic characteristics we have used IS development project characteristics [7]. In this work, a project characteristics typology is proposed in order to guide method components retrieval and to prioritize the selected components.

The suggested typology of context characteristics covers essential aspects of IS engineering projects. Based on [27], [28] and [7], it includes four facets: organizational, human, application domain, and development strategy.

The *organisational* facet (Table II) highlights organisational aspects of IS project development. For instance, the *Management Commitment* characteristic represents the management team involvement in the project. Possible values for this characteristic are Low, Normal and High (i.e. a High value means a high involvement and so on).

TABLE II. ORGANIZATIONAL FACET VALUES

Characteristic	Value domain
Management commitment	{low, normal, high}
Importance	{low, normal, high}
Impact	{low, normal, high}
Time pressure	{low, normal, high}
Shortage of resources	{low, normal, high}
Nature of limited resources	{financial resources, human resources, temporal resources, informational resources}
Size	{low, normal, high}
Cost	{low, normal, high}
Level of innovation	{low, normal, high}
Innovation nature	{business innovation, technology innovation}

The *human* facet (Table III) describes the qualities of persons involved in IS project development. For example, the *User involvement* characteristic represents the kind of participation of the users in the project. Its values may be real or virtual.

TABLE III. HUMAN FACET VALUES

Characteristic	Value domain
Resistance and conflict	{low, normal, high}
Expertise degree	{low, normal, high}
Expert role	{tester, developer, designer, analyst}
Clarity and stability	{low, normal, high}
User involvement	{real, virtual}
Stakeholder number	Num

The *application domain* facet (Table IV) includes indicators characterizing the domain of IS project. For instance, the *Application type* characteristic deals with the different kinds of projects according to the organization structure and can have the following values: intra-organization application, inter-organization application, organization-customer application.

TABLE IV. APPLICATION DOMAIN FACET VALUES

Characteristic	Value domain
Formality	{low, normal, high}
Relationships	{low, normal, high}
Dependency	{low, normal, high}
Complexity	{low, normal, high}
Application type	{intra-organization application, inter-organization application, organization-customer application}
Application technology	{application to develop includes a database, application to develop is distributed, application to develop includes a GUI}
Dividing project	{one single system, establishing system-oriented subprojects, establishing process-oriented subprojects, establishing hybrid subprojects}
Repetitiveness	{low, normal, high}
Variability	{low, normal, high}
Variable artefacts	{organisational, human, application domain, and development strategy}

The *development strategy* facet (Table V) gathers indicators about different characteristics of development strategy. For instance, the *Source system* characteristic represents the origin of the reused elements that may be code, functional domain or interface.

TABLE V. DEVELOPMENT STRATEGY FACET VALUES

Characteristic	Value domain
Source system	{code reuse, functional domain reuse, interface reuse}
Project organization	{standard, adapted}
Development strategy	{outsourcing, iterative, prototyping, phase-wise, tile-wise}
Realization strategy	{at once, incremental, concurrent, overlapping}
Delivery strategy	{at once, incremental, evolutionary}
Tracing project	{weak, strong}
Goal number	{one goal, multi-goals}

Specific characteristics. Their identification is based on the method description. The method engineer defines them by analyzing different aspects which are organized into four facets: intentional, satisfaction, decisional and internal like in [29].

The *intentional* facet concerns the method intentions. The *satisfaction* facet indicates the satisfaction degree that the engineer has about the method application results. The *decisional* facet arises from a decision-making process in the method. The *internal* facet concerns the known criteria associated with the specific project management.

Characteristics typology. Based on this approach we have established the characteristics typology (cf. Fig. 3).

Table VI shows the correspondence between the proposed typology and the existing context elements. We can make some remarks to compare them:

- Our typology covers all existing elements.
- We propose to identify more precisely process characteristics and product part using our approach

instead of using product and process as context characteristics directly.

- We add decisional characteristics which are not presented in the existing typologies.

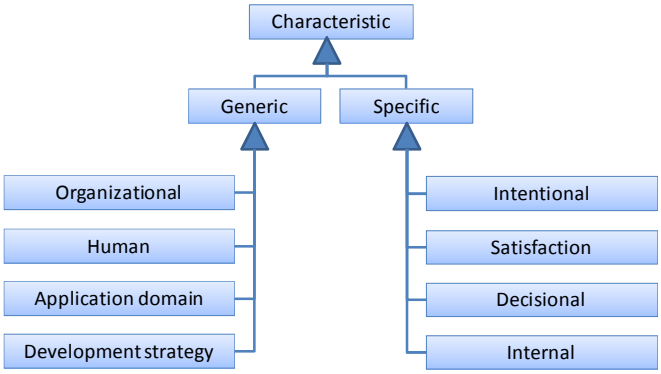

Figure 3. Characteristic typology

TABLE VI. CORRESPONDENCE BETWEEN THE PROPOSED TYPOLOGY AND EXISTING CONTEXT ELEMENTS.

Proposed Typology	Context Elements (cf. Table I)
Organizational	Organizational
Human	Human
Application domain	Domain
Development strategy	Technical
Intentional	Goal/ Intention
Satisfaction	External, Process, Product
Decisional	Process, Product
Internal	Technical, Process, Product

Context granularity. We consider the context granularity at two levels: method context and method component context (see Fig. 4).

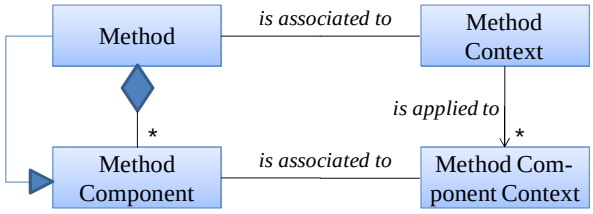

Figure 4. Proposal overview.

The context characteristics set is defined for a method. Then, each method component is described by the valuations of these characteristics. The component's selection is carried out by request on the characteristics values of available method components. If the request does not provide a conclusive result (i.e. there are several or no method that satisfies characteristics), the engineer has to either consider other methods, modify required characteristics values, or rank characteristics by their order of importance.

B. *Contextualization methodology*

In order to define the context for a given method and its components, we propose an approach based on a process (Fig. 5) modeled with the MAP formalism (See Appendix 1).

There are two possible ways to define the context: top-down or bottom-up. By the top-down approach, the engineer defines the method context and then instantiate it for each method component. By the bottom-up approach, the engineer specifies the contexts of all method components and assemblies them into the method context.

Both method and method component contexts can be defined following two strategies: *By deduction* and *By generation*. It depends on the characteristic type. The generic characteristics are *deduced* from generic context typology and the specific ones are *generated* from method description. These strategies could be applied as many times as possible characteristics exist.

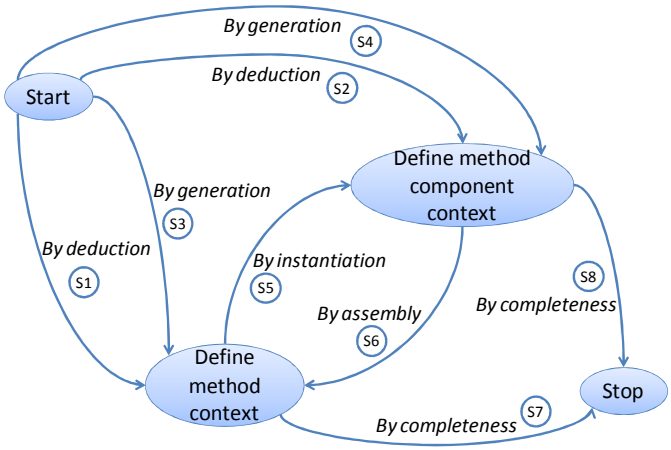

Figure 5. Contextualization MAP.

This MAP has two main intentions: *Define method context* and *Define method component context*. The achievement of these intentions implies the definition of the context characteristics set for method or for method components respectively. The definition of method components contexts includes also the attribution of values to the defined characteristics.

The contextualization Map includes eight sections, as shown in Table VII.

TABLE VII. CONTEXTUALIZATION MAP SECTIONS

Section	<Source intention, Strategy, Target intention >
S_1	<Start, By deduction, Define method context>
S_2	<Start, By deduction, Define method component context>
S_3	<Start, By generation, Define method context>
S_4	<Start, By generation, Define method component context>
S_5	<Define method context, By instantiation, Define method component context>
S_6	<Define method component context, By assembly, Define method context>
S_7	<Define method context, By completeness, Stop>
S_8	<Define method component context, By completeness, Stop>

All these sections are explained bellow. Operators are defined for each section in order to indicate how to proceed for carrying out its execution.

<Start, By deduction, Define method context>. The generic characteristics deduction is based on the context

typology. This section gives a selection of characteristics carried out by the IS method engineer. The result of this strategy is a sub-set of generic characteristics available for a given project.

The corresponding operator is:

Select Context Characteristic ()

<Start, By deduction, Define method component context>. This section includes the selection of characteristics form generic typology like the previous one and includes furthermore the attribution of values to these characteristics. The result of this strategy is a sub-set of generic characteristics available for a given project with corresponding values.

Two following operators are applied consecutively:

Select Context Characteristic ()

Attribute a Value to Context Characteristic ()

<Start, By generation, Define method context>. The specific characteristics generation is based on the method description. The method engineer defines them by analyzing different aspects which are organized into four facets: intentional, satisfaction, decisional and internal.

This section includes four operators. Each of the following operators is applied depending on the corresponding characteristic's facet:

Analyze Method Goal () [for intentional facet]

Measure Method Satisfaction () [for satisfaction facet]

Analyze Method Argumentation () [for decisional facet]

Measure Method Characteristics () [for internal facet]

<Start, By generation, Define method component context>. The definition of specific characteristics for method components context is the same as for method context (the previous section) but also requires the attribution of characteristics values.

This section uses the same four operators and adds another one that deals with the attribution of values to the characteristics. This last one is applied after each of the first four operators for defining concrete values of the identified specific characteristics.

Analyze Method Goal () [for intentional facet]

Measure Method Satisfaction () [for satisfaction facet]

Analyze Method Argumentation () [for decisional facet]

Measure Method Characteristics () [for internal facet]

Attribute a Value to Context Characteristic () [for all facets]

<Define method context, By instantiation, Define method component context>. The context characteristics instantiation is common for both characteristics types and is applied in the top-down approach. This section allows defining a sub-set of generic and specific method characteristics with an associated value for each method component separately.

This section contains two operators applied consecutively:

Retain Context Characteristic ()

Attribute a Value to Context Characteristic ()

<Define method component context, By assembly, Define method context>. In the case of the bottom-up approach, the strategy *By assembly* follows the definition of the method component context *By deduction* or *By generation*. The method engineer groups method components characteristics together. As a result, the method context includes all characteristics of its components contexts.

This section is carried out by the following operator:

Group Characteristics ()

<Define method context, By completeness, Stop> and **<Define method component context, By completeness, Stop>.** These sections are the same in both top-down and bottom-up approaches and include verification of completeness and coherence of the described context.

The associated operator is:

Verify Context Completeness ()

V. APPLICATION WITHIN IS PROJECT PLANNING

The process of our case study is expressed with the intentional model MAP (cf. Appendix 1). Before introducing our example, the relationships between the intentional model MAP and the context must be made. The MAP model describes the ways of processing requirements by introducing the stakeholders' intentions and the strategies they used in order to reach these intentions. Moreover, the formulation of an intention is related to the considered domain. Thus, the MAP model allows the description of the *semantic* and the *cognitive* aspect of a context. For instance, the MAP model has been already used to describe the alignment links between business requirements and IT/Business operational components [32].

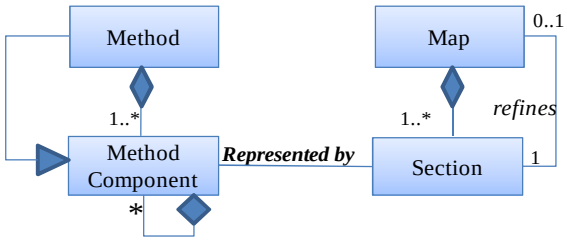

Figure 6. Section and Method component correspondance

The key concept of a Map is the notion of Section (cf Appendix 1). Each Map section is linked to a particular method component, as shown in Fig. 7. As shown above, a method component may be an entire method composed of other method components (themselves composed of other less complex components). The method components are formalized following the meta-model shown in Fig. 1. Each section of the IT project portfolio map is then linked to a specific method component. The engineer is guided in the usage of method components through the map by selecting intentions and strategies. We describe below a possible scenario based on the current map.

At a given *situation*, the designer must select a method component in order to improve the product compliance with requirements. Method components are saved into a repository. Based on the description of their context, the designer makes a request on the repository in order to extract the appropriate component to use. In our proposal (Fig. 6), there are ten sections, each one linked to a component.

The MC selection is based on the description of context in order to extract appropriate components for the design situation. The component ensures the transformation of a *product part* into a desired product: the *source* is a product part (for instance a UML diagram) which will be transformed by the design process (*process part*) into a *target* product part.

A. Case study overview

This section addresses the formalization of *method components* related to IT project portfolio management and proposes the application of the previous typology of context.

Project Portfolio Management (PPM) is a term used to describe methods for analyzing and collectively managing a group of current or proposed projects based on numerous *key characteristics*. Its main purpose is to direct the financial distribution between projects - PPM is applied to IT Project Portfolio Management (IT-PPM) for IT purposes. IT portfolio management is an enabling technique for IT Governance requirements. It is related to both IT Service Management and Enterprise Architecture, and might even be seen as a bridge between them. We propose here to sustain the IT-PPM activities which consist in *identifying*, *evaluating* and *prioritizing* IT projects for their implementation. The related components are saved into a method base which includes their description and methodological guidelines for their application.

We take into account the *intentional* paradigm previously presented to formalize the methodological *process* which aims at making the *product* used to support IT-PPM activities (information system dedicated to IT-PPM). We formalize this product with UML class diagrams.

We base our approach on the IT governance requirements by using a MAP model (cf. Appendix 1) which shows the intentions of IT decision makers in steering the IT processes and resources [30].

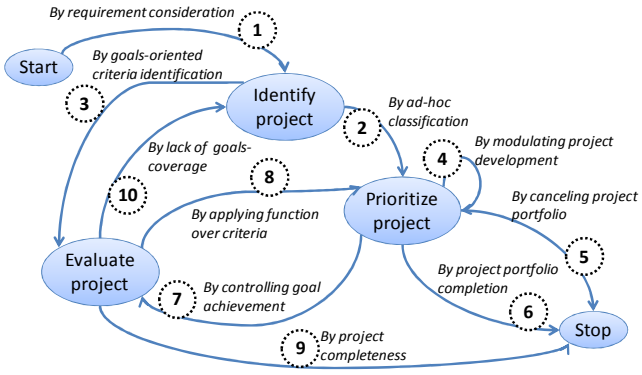

Figure 7. IT-PPM Map.

The Fig. 7 shows the project portfolio intentional map which describes the ways to manage a project within a portfolio. This Map is a refinement of the section < Define Risks, By Project Planning, Align IT and Business Process > of the map dealing with IT governance presented in [33]. The MAP formalism helps to show the intentional way to consider the project portfolio process. It describes the intentional links between the components.

In the following, we define the context of method components shown in Fig. 6 (sub-section B) and we illustrate the usage of the identified characteristics in order to guide the engineer through the IT-PPM Map within a case study (sub-section C).

B. Contextualization process application

In our case study, we consider the design activities dedicated to a project of an organization. The time dedicated to this activity must be minimized. The main goal of this project is then *Minimize design delay*. Moreover, within this organization a new designer has been hired as back-office project manager. This situation leads to the contextualization of the required course of action to reach the main goal. Therefore, the experience level of the designer is a variable point from which to choose the navigation path through the IT-PPM Map. The designer use a particular set of characteristics, representative of his context, which guide him through the selection of method components. We describe the characteristics identification process below.

The engineer has selected the top-down approach of the contextualization process. It guides the engineer through the definition of the method characteristics before the definition of MC characteristics. Fig. 8 shows the path used in the navigation through the contextualization Map (Fig. 5) in this particular case study.

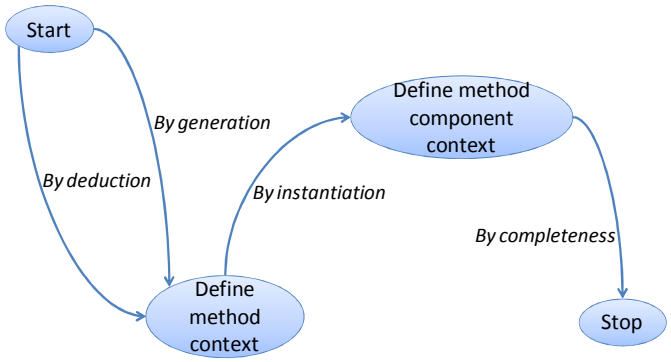

Figure 8. Path used in the Contextualization Map

This path contains four sections of the Contextualization Map (cf. Tab. VII) that we represent within the three following steps: (1) Definition of the method Context (S_1 and S_3), (2) Definition of the method components contexts (S_5), and (3) Verification of the process completeness (S_8).

1) Definition of Method Context

This step contains the execution of the two sections: S_1 and S_3.

Definition of the generic characteristics (S_1). The engineer uses the characteristics presented in Table III and

Table IV as generic characteristics to specify the context of the method.

The engineer has applied the operator *Select Context Characteristic ()* in order to define a sub-set of generic context characteristics according to the given project. He has selected only three generic characteristics for this example: *expertise degree*, *expert role* and *application type* (Tab. VIII).

TABLE VIII. GENERIC FACET VALUES

Characteristic	Value domain
Human facet values	
Expertise degree	{low, normal, high}
Expert role	{tester, developer, designer, analyst}
Application domain facet values	
Application type	{intra-organization application, inter-organization application, organization-customer application}

Definition of the specific characteristics *(S_3).* The engineer also uses *specific* context characteristics (cf. Table IX) leading the designer to choose a method component. This specific context is depicted by the constraints of the business environment (the design *situation*), the *intention* of the designer and the *strategy* for reaching the intention. So, the three operators were applied to identify the specific characteristics. *Analyze Method Goal ()* is used to identify the *Intention* which is related to the intentional type of specific characteristic. *Measure Method Satisfaction ()* allows defining the *Situation* in the Satisfactional facet (as it describes the satisfaction degree of the previous intention). Finally, *Analyze Method Argumentation ()* defines the *Strategy* in the decisional facet of the specific characteristic.

TABLE IX. SPECIFIC FACET VALUES

Characteristic	Value domain
Intentional facet values	
Intention	text
Satisfactional facet values	
Situation	text
Decisional facet values	
Strategy	text

2) Definition of Method Components Context (S_5)

The method context defined at the previous step is now instantiated for each component. A value is affected to each characteristic in order to help the case process execution guidance. The following operators are applied to each method characteristic: *Retain Context Characteristic ()* and *Attribute a Value to Context Characteristic ().* The results are presented in Table X.

3) Verification of the process completeness (S_8)

The engineer has decided that the identified context characteristics are sufficient to allow a satisfying guidance through the portfolio project management by the operator *Verify Context Completeness ()* application.

C. Guidance application within the Case Study Map

The characteristics values are used in the navigation process to help the component selection. In this particular case study, the engineer has ranked these characteristics following his preferences which are:

$$\text{Intention} \succ \text{Expert role} \succ \text{Degree of expertise}$$

First step. At the beginning of the design process, there is only one candidate section. Each section holds a method component. The identification of a project is supported by the section ① depicted in Fig. 6. The corresponding method component is described in Fig. 9: the analyst has to refine his requirements into goals and to organize them by project. The requirements are inputs (source product part) of the project portfolio process. By applying the "By requirement consideration" strategy, requirements are analyzed in order to identify a project and define its related goals. This leads to the transformation of the source product part (for instance, *Requirement* class) by extending it with the *goal* and *project* classes (cf. target product part).

TABLE X. SPECIFIC CHARACTERISTICS INSTANTIATION

Characteristic	IT-PPM Method Components									
	①	②	③	④	⑤	⑥	⑦	⑧	⑨	⑩
Generic										
Degree of expertise	Normal	Low	High	High	Low	Low	Low	Normal	Low	Normal
Expert role	Analyst designer	Analyst designer	Analyst designer	Designer	Designer	Designer	Designer	Analyst	Designer	Analyst
Application type	Intra-Org.	Intra-Org.	Intra-Org.	Intra-Org.	Intra-Org.	Intra-Org.	Intra-Org.	Intra-Org.	Intra-Org.	Intra-Org.
Specific										
Situation	Problem statement	Project identified	Project identified	Project prioritized	Project prioritized	Project prioritized	Project prioritized	Project evaluated	Project evaluated	Project evaluated
Intention	Identify project	Prioritize project	Evaluate project	Prioritize project	Stop	Stop	Evaluate project	Prioritize project	Stop	Identify project
Strategy	By requirement consideration	By ad-hoc classification	By goals-oriented criteria identification	By modulating project development	By canceling project portfolio	By project portfolio completion	By controling goal achievement	By applying function over criteria	By project completeness	By lack of goal coverage

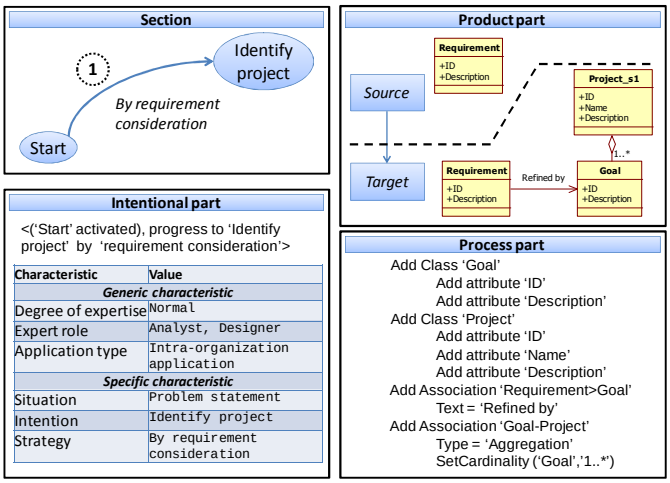

Figure 9. MC for project identification

Second step. According to the new *situation*, the intention *Identify Project* is reached. The possible path is twofold: the designer can consider the intention (i) *Evaluate Project* or (ii) *Prioritize Project*. The first one is more complex than the second. In our case the designer is a beginner and the *degree of expertise* characteristic leads to the selection of the MC dedicated to project scheduling: section ② is chosen. The project class is extended with the attribute *number* for the project classification and the set of projects are organized within a *portfolio*. Fig. 10 summarizes the method component used for this situation.

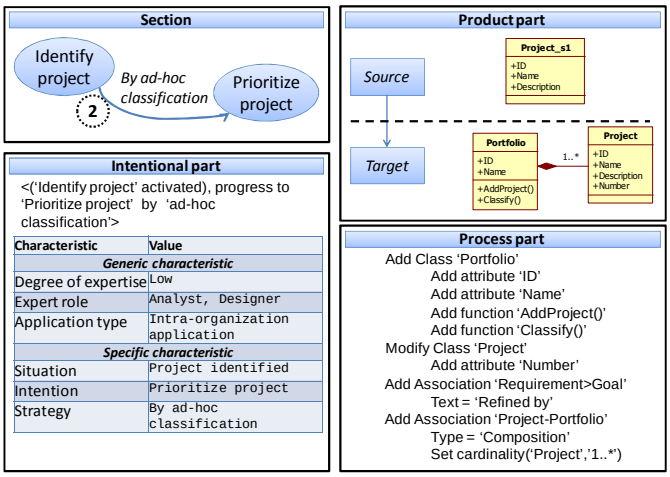

Figure 10. MC for project prioritization

Third step. The intention *Prioritize Project* is now reached. Considering this situation, the designer can select between three envisioned intentions:

- (i) to complete the intention of prioritizing project by *modulating* project. Modulation consists in allocating more or less resources to a project;
- (ii) to evaluate the achievement of a project. In this case, control processes and audits are performed in order to evaluate the completion of a project;
- (iii) to stop the process.

In our case study, the designer aims at formalizing the project evaluation. This is a specific constraint (intention) which leads to select the component related to the section ⑦.

The resulting component is depicted in Fig. 11: the project is a set of goals which must be evaluated by a control process over criteria. The instantiated product is a m by n matrix G including m criteria and n goals in which $x_{m,n}$ is the evaluation of the m^{th} criteria over the n^{th} goal.

$$G = \begin{bmatrix} x_{1,1} & \cdots & x_{1,n} \\ \vdots & \ddots & \vdots \\ x_{m,1} & \cdots & x_{m,n} \end{bmatrix}$$

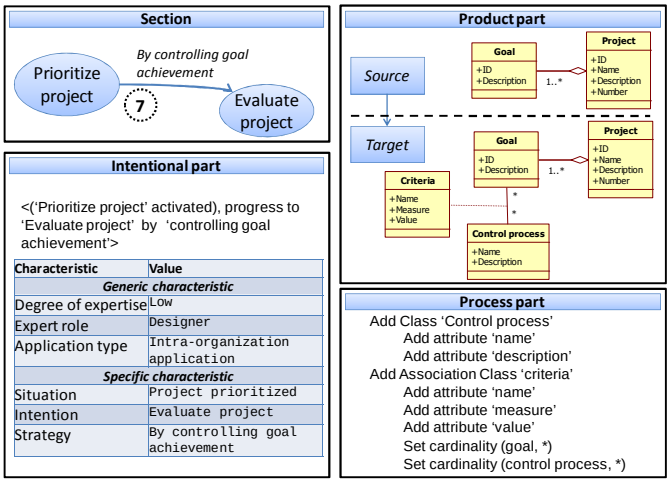

Figure 11. MC for project evaluation

Fourth step. The intention *Evaluate Project* is now satisfied: this evaluation can lead (i) to identify intermediate goals and intermediate projects; (ii) to prioritize project based on the evaluation of criteria; (iii) and to stop the IT-PPM process when projects are completed. At this stage, the designer does not need to identify or prioritize project. This leads to enact section ⑨, and the UML specification which must be improved in order to integrate the completion measure of a project: this is done by extending the project class with the *Completion* attribute and the operation EvaluateCompletion(). Fig. 12 represents the method component used at this step.

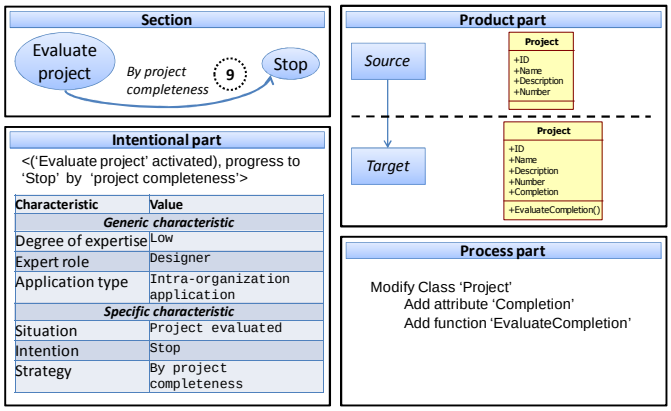

Figure 12. MC for project completion

Discussion. The presented scenario shows how the engineer was guided through the IT-PPM Map and led to execute four method components. The Map used in this case study may lead to various executable processes depending on the different situations. This point highlights the intrinsic variability of the MAP model. We notice that a sub-set of the characteristics (six in this example) may be sufficient to steer the design process.

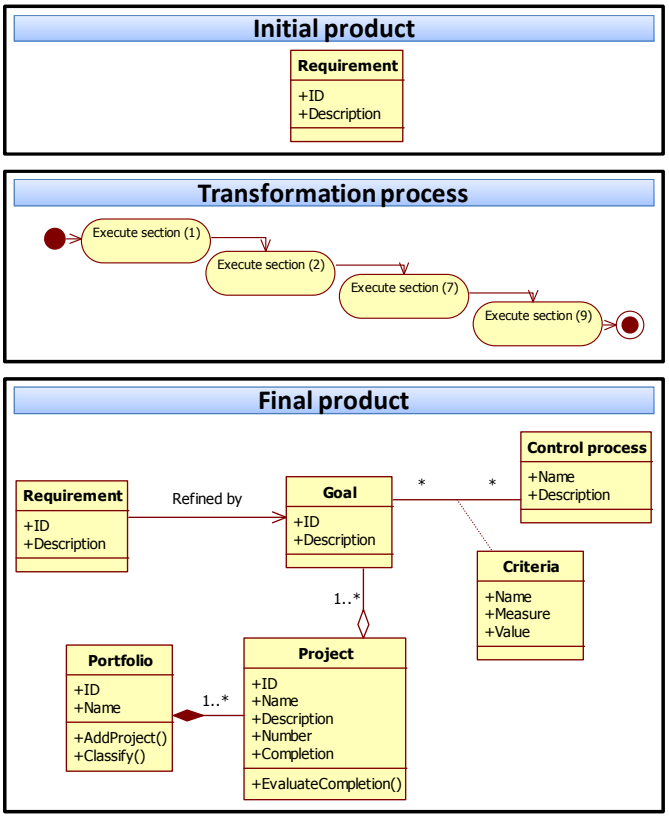

Figure 13. Method overview

As the application of a method component depends on the context characteristics, more than one method component can be a candidate for the execution of a MAP section: the selection of a MC candidate is operated as the MAP execution is performed by context characteristics analysis.

At last, Fig. 13 shows the final result of the design process. This application is based on the previous meta-model shown in section IV.

VI. Conclusion

This paper provides a contextual description of method components and a way to specify it. This kind of description allows a better retrieval of method components according to project specificities.

This proposal can be applied in different IS engineering situations such as the selection of a component for enhancing the existing IS engineering method (for instance, extension-based approaches) or a selection of several components for constructing a new one (for instance, assembly-based approaches). We have applied the proposed model for project portfolio management within ISDM. It contributes to the study of the relatively unexplored domain of IT governance from the SME point of view.

Our future work aims at:

- Developing an approach to define the method context by aggregating method components' characteristics.
- Ensuring the adaptability of methods with regards to the context specificities.
- Proposing a method for a formalized selection of method components following their characteristics values.
- Validating and experiencing the current proposal.
- Developing other method components using the current assumptions.

VII. References

[1] F. Harmsen, S. Brinkkemper and J. L. Han Oei, "Situational method engineering for information system project approaches", *Methods and Associated Tools for the Information Systems Life Cycle conference*, 1994, pp 169-194.

[2] C. Rolland, "L'ingénierie des méthodes : une visite guidée", *E-revue en Technologies de l'Information (e-TI)*, Invited Talk, 2005.

[3] R. Deneckère, A. Iacovelli, E. Kornyshova, and C. Souveyet, "From method fragments to method services", *Evaluation of Modeling Methods in Systems Analysis and Design (EMMSAD)*, Montpellier, France, June 2008.

[4] J. Ralyte, and C. Rolland, "An assembly process model for method engineering", *International Conference on Advanced information Systems Engineering (CAISE)*, Interlaken, Switzerland, June 2001.

[5] G. Guzélian and C. Cauvet, "SO2M : Towards a service-oriented approach for method engineering", *Proceedings of the international conference IKE'07*, Las Vegas, Nevada, USA, 2007.

[6] C., Gonzales-Perez, "Supporting situational method engineering with ISO/IEC 24744 and the work product tool approach", *Proceedings of the International IFIP WG8.1 Conference ME 07*, Springer, Geneva, Switzerland, 2007.

[7] E. Kornyshova, R. Deneckère, and C. Salinesi. "Method chunks selection by multicriteria techniques: an extension of the assembly-based approach", *Proceedings of the International IFIP WG8.1 Conference ME 07*, Springer, Geneva, Switzerland, 2007.

[8] C. Rolland, N. Prakash, A. Benjamen, "A multi-model view of process modelling", *Requirements Engineering*, Springer-Verlag London Ltd, 4:4, 1999.

[9] S. Brinkkemper, "Method engineering: engineering of information systems development method and tools", *Information and Software Technology*, 38:7, 1996.

[10] C. Rolland and V. Plihon and J. Ralyté, "Specifying the reuse context of scenario method chunks", *Proceedings of the international conference CAiSE'98*, Pise, 1998.

[11] K. Wistrand and F. Karlsson, “Method components: rationale revealed”, *Proceedings of CAISE 04*, Springer-Verlag. Riga, Latvia, 2004.

[12] B. Henderson-Sellers, “Process meta-modelling and process construction: examples using the OPF”, *Ann. Software Engineering*, 14(1-4), 2002.

[13] A. Iacovelli, C. Souveyet, and C. Rolland., "Method as a service (MaaS)", *International Conference on Research Challenges in Information Science (RCIS)*, Marrakech, Morocco, 2008, pp. 371 - 380.

[14] Y. R. Nehan, and R. Deneckère., "Component-based situational methods - A framework for understanding SME", *Proceedings of the International IFIP WG8.1 Conference ME 07*, Springer, Geneva, Switzerland, 2007.

[15] Ralyté, J., Deneckere and R., Rolland, C.: Towards a generic model for situational method engineering, in proceedings of the conference CAISE’03, Springer Verlag, Velden, Austria, 2003

[16] N. A. Bradley and M. D. Dunlop, “Toward a multidisciplinary model of context to support context-aware computing”, *Human-Computer interaction*, Lawrence Erlbaum Associates, 2005.

[17] H. Bunt, “Context and dialogue control”, *Proceedings of CONTEXT’97*, 1997.

[18] J. Coutaz and G. Rey, “Recovering foundations for a theory of contextors”, 4th ICCADUI, Valenciennes, France, 2002.

[19] B. Schilit, N. Adams, and R. Want. "Context-aware computing applications" (PDF). *IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'94)*, Santa Cruz, CA, US, 1994, 89-101.

[20] M. Rosemann and J. Recker, "Context-aware process design: exploring the extrinsic drivers for process flexibility", 18th international conference on advanced information systems engineering. *Proceedings of workshops and doctoral consortium*, Luxembourg: Namur University Press., 2006, pp 149-158.

[21] K. Bessai, B. Claudepierre, O. Saidani and S. Nurcan, "Context-aware business process evaluation and redesign", *Proceedings of BPMDS'08*, 2008.

[22] M. Kirsch Pinheiro, Y. Vanrompay, and Y. Berbers, "Context-aware service selection using graph matching", *Proceedings of ECOWS 2008*, vol. 411, 2008.

[23] M. A. Rosen, S. M. Fiore, E. Salas, M. Letsky and N. Warner, “Tightly coupling cognition: understanding how communication and awareness drive coordination in teams”, *International C2 journal*, 2:1, 2008.

[24] J. L. Drury and S. D. Scott, “Awareness in unmanned aerial vehicle operations”, *International C2 journal*, Geoffrey N. Hone, 2:1, 2008.

[25] I. Mirbel, “Contributions à la modélisation, la réutilisation et la flexibilité des systèmes d’information », HDR thesis, Nice University, 2008.

[26] J. Ralyté and C. Rolland, “An approach for method reengineering”, Proceedings of the 20th International Conference on Conceptual Modeling (ER2001), Yokohama, Japan, November 2001. H. Kunii, S. Jajodia, A. Solvberg (Eds.), LNCS 2224, Springer-Verlag, 2001, pp.471-484.

[27] I. Mirbel. and J. Ralyté. “Situational method engineering: combining assembly-based and roadmap-driven approaches”, *Requirements Engineering*, 11(1), 2006, pp. 58–78.

[28] K. Van Slooten and B. Hodes, “Characterising IS development projects”, IFIP WG8.1 Conference on Method Engineering, 1996.

[29] F. Harmsen, “Situational method engineering”. Moret Ernst & Young, 1997.

[30] A.F. Harmsen, J.N. Brinkkemper and J.L.H. Oei, "Situational method engineering for information systems project approaches", Int. IFIP WG8. 1 Conference in CRIS series : "Methods and associated Tools for the Information Systems Life Cycle" (A-55), North Holland (Pub.), 1994.

[31] F. Karlsson and P.J. Agerfalk, Method configuration: adapting to situational characteristics while creating reusable assets, Information and Software Technology 45, 2004, pp 619-633.

[32] L-.H. Thevenet, C. Rolland and C. Salinesi, “Alignement de la stratégie et du système d’information – Présentation de la method INSTAL”, in RSTI-ISI, Evolution des systèmes d’information, Paris, France, n°14, 2009.

[33] B. Claudepierre and S. Nurcan, “ITGIM: An intention driven approach for analyzing the IT governance requirements”, *Workshop on Requirements, Intentions and Goals in Conceptual Modeling*, 2009.

[34] C. Rolland, "Method engineering: “Towards methods as services”, *International Conference on Software Process (ICSE-ICSP)*, Springer-Verlag, Leipzig, Germany, 2008.

VIII. Appendix 1

A MAP illustrates a given process of IS engineering. The *MAP* model [8] is a representation of process models expressed in intentional terms. It allows specifying process models in a flexible way by focusing on the process intentions, and on the various ways to achieve each of these intentions.

A map is presented as a diagram where nodes are *intentions* and edges are *strategies*.

The directed nature of this diagram shows the precedence links between intentions. An edge enters a node if its associated strategy can be used to achieve the target intention (the given node). Since there can be multiple edges entering a node, a map is able to represent the many ways for achieving an intention. The following figure shows the structure of a map with the UML formalism (see Fig. 14).

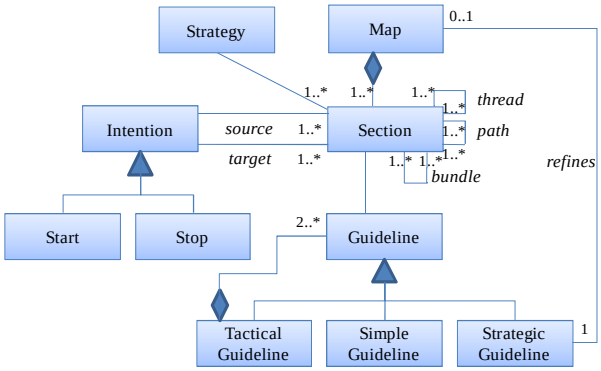

Figure 14. MAP model.

The key concept of a map is the section. A section is an aggregation of two specific intentions, the *source intention* and the *target intention*, linked together with a *strategy*. It embeds the knowledge corresponding to a particular process step to achieve an intention (the target intention) from a specific situation (the source intention) following a particular technique (the strategy).

An intention is a goal that can be achieved. There are two predefined intentions contained in any map, namely “Start” and “Stop”, which mean accordingly the beginning and the end of the process. A specific process to achieve an intention is captured in a section. All sections having the same source and target intentions represent all the different strategies that may be used to achieve this target intention. In the same way, there may be several sections with the same source intention but different target ones, which show all the intentions that may be reached after the realisation of the source intention.

There are three kinds of relationships between sections. A *thread* relationship shows that a target intention can be achieved (from the same source intention) in several ways. Each of these ways is expressed as a section in the map. A *path* relationship establishes a precedence relationship between sections. A *bundle* relationship shows that several sections having the same source and target intentions can be mutually exclusive.

There are three types of guidelines: simple, tactical and strategic. A *simple* guideline may give informal content advice on how to proceed in handling the situation in a narrative form. A *tactical* guideline is a complex guideline, which uses a tree structure to link its sub-guidelines. A *strategic* guideline is a complex guideline which shows that a section of a map can be refined by another map. This relationship implies that each map may be represented as a hierarchy of maps.

The MAP model defines the process through the combination of observable situations in which a certain number of specific intentions can be achieved. The work to be made is described in the process as depending on both situation and intention. In other words, it depends on the context in which a method engineer must act at a given point in time. By modelling *intentions* and the ways (*strategies*) to reach them, the process has the ability to represent the *cognitive* context as defined by Bunt. Moreover, by relating method service [34] (or method component [15]) to a *section*, Rolland extends the context expressiveness of the MAP to the *semantic* context of Bunt. This approach allows identifying several context aspects. More precisely, this model includes a set of guidelines which help an engineer navigate through the process model. The navigation is carried out by *arguments* that allow the engineer to choose the adapted variant within the process model. These arguments express the context of a given process model.

IX. Appendix 2

In order to access to *up to date* information about authors, please scan the following codes.

Rébecca Deneckère

Elena Kornyshova

Bruno Claudepierre

For scanning abilities, you may install the following barcode scanner software on your mobile:
http://www.lynkware.com/support_devices.php