

Using and Extending the SPEM Specifications to Represent Agent Oriented Methodologies

Valeria Seidita¹, Massimo Cossentino², and Salvatore Gaglio^{1,2}

¹ Dipartimento di Ingegneria Informatica, University of Palermo,
Palermo, Italy
{seidita,gaglio}@dinfo.unipa.it

² Istituto di Calcolo e Reti ad Alte Prestazioni, Consiglio Nazionale delle Ricerche
Palermo, Italy
cossentino@pa.icar.cnr.it

Abstract. Situational Method Engineering for constructing ad-hoc agent oriented design processes is grounded on a well defined set of phases that are principally based on reuse of components coming from existing agent design processes; these components have to be stored in a repository. The identification and extraction of these components could take large advantages from the existence of a standardized representation of the design processes they come from. In this paper we illustrate our solution based on SPEM 2.0 specifications for modelling agent design processes and extending them when necessary to meet the specific needs we faced in our experiments.

1 Introduction

Our research is focussed on the field of Situational Method Engineering (SME) [9][2][8][10] for the construction of ad-hoc multi agent systems design processes. Applying Situational Method Engineering requires executing a well defined set of phases [7][6][17]: Process Requirements Specification, Process Fragments Selection and Process Fragments Assembly.

SME is based on the reuse of components coming from existing design processes, the so called method fragments or simply fragments; the request for reusable fragments leads to the need for a repository containing standardized fragments that could be easily selected and assembled in new design processes (i.e. methodology; this term is commonly used in the agent community and in the SME one, for avoiding confusion in this work we consider it as a synonymous of design process or process).

Since the repository is composed of fragments coming from existing design processes, its construction cannot be done without considering: the knowledge of a set of existing processes, their standard description through a standard notation and a precise definition of the fragment notion and of the process itself.

We decided to use SPEM (Software Process Engineering Metamodel) 2.0 [11], both for design process and fragments representation; it is an OMG standard and it is based on a metamodel containing three main elements: activity, work

product and process role. We found the use of SPEM very promising and suitable for our purposes; in our previous works [5][16] we identified and defined the main elements a design process and a fragment are composed of, these elements can be easily represented using the SPEM metamodel main elements.

Besides, in the agent oriented context, according to our view, a process is devoted to design a MAS model whose elements are represented in the work products resulting from the enactment of a specific activity. A MAS model element is an instance of a MAS metamodel element; the MAS metamodel represents the structure of the system that is being built with the specific design process.

The key point of our approach consists in using an ontological metamodel providing the right semantics to the element of the domain we are dealing with [15][3] (this metamodel will for instance make use of relationships that are typical of the ones used in ontologies). The MAS metamodel is one of the most important factors of our approach, as it is not present in the SPEM specifications we decided to extend these specification.

In this paper we present how we use SPEM, its elements and diagrams, for representing a design process and how we extended it by adding elements and diagrams to SPEM specification, in order to meet our needs.

The paper is structured as follows: in section 2 an overview on the Situational Method Engineering approach for creating new agent oriented design processes is given; in section 3 the main SPEM elements we use are illustrated, the needed extensions are justified and motivated and an example on applying SPEM is provided; finally in section 6 some conclusions are drawn.

2 The Formal Description of a Design Process

The construction of a new design process following Situational Method Engineering principles is based on three main phases [6][13]: Method Requirements Engineering, Method Design and Method Construction.

Our approach for applying SME in the agent oriented context is based on these three phases too but it is specialized for the agent context, as shown in Figure 1; here we sketch the whole process a method designer has to carry out in order to construct a new agent oriented design process. The first step a method designer must undertake is to create a repository of fragments starting from those extracted from a set of existing design processes and/or constructing new ones from scratch. For this aim he needs the set of existing design processed to be well defined, in a standardized fashion, in order to easily and quickly identify the portions of process devoted to become fragments.

During the Process Requirements Analysis activity, the method designer considers inputs coming from the development context (tools, languages, available skills, etc.) and the type of problem he has to solve. These inputs are used to define the process life cycle (that establishes the structure the designer has to follow during process fragments assembly activity), the MAS metamodel concepts and the other process elements (available stakeholders, required activities or work products) that are used for selecting fragments from the repository.

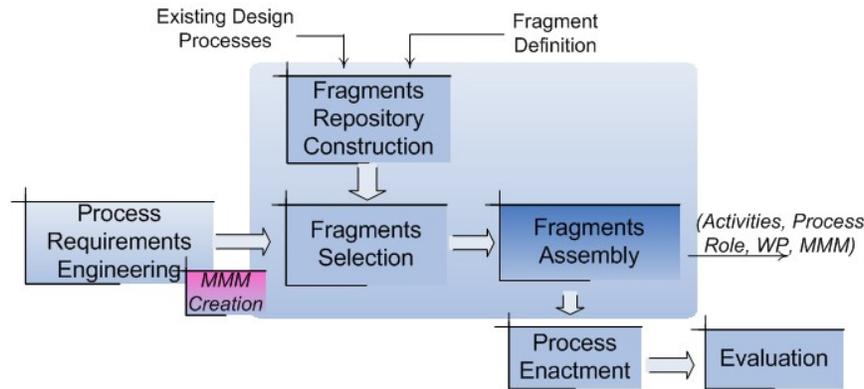


Fig. 1. Our Approach for Creating an Agent Design Process

It is worth noting the fundamental difference between the MAS metamodel that will be instantiated in the actual agent system during design and the process (or method fragment) metamodel that will be used to define the design process.

The output of the process requirements analysis contributes to the selection of process fragments and to their assembly; once the process is created, it can be enacted and then evaluated for eventually iterating the whole construction process.

In this work we want to point out our attention to the fragments repository construction, for which we need two elements: a set of existing design processes used for extracting fragments to be stored and a specific definition of design process. A specific definition of fragment is also needed for enabling the method engineer to correctly describe the existing design processes.

As regards the definition of design process and fragment we consider a design process as the set of activities to be performed in order to produce an output, the way of performing some activities (guidelines or techniques), and the resources and constraints this requires. In [5] we gave a definition of multi-agent system design process and of fragment (we call it *process fragment*). In Figure 2 we show the main elements we use for describing the agent design processes; these elements are the base for the design process fragmentation.

A design process is composed of activities, each activity is performed by a process role that is responsible for one or more work products that are structured by a work product kind representing a specific category, for instance, text document, code and so on.

A design process is devoted to design a MAS Model that is composed of MAS model elements each of which can be represented in one or more work products; a MAS model element is an instance of a MAS metamodel element so in each work product there is a correspondence with one or more MAS metamodel elements.

A process can be decomposed into (process) fragments that are self-contained pieces of the whole process, with all the elements characterizing the process it-

self (activity, process role, work product and MAS metamodel element) and that instantiate one or more MAS metamodel elements described in the work product(s) resulting for the fragment itself.

In our approach we assume that each fragment has to deliver at least one work product, thus, basing on this hypothesis and on the proposed fragment definition, we can say that our fragment extraction activity is work product driven.

Starting from the aforementioned definition of design process we decided to follow a top-down approach for a clear description and an easy retrieval of the main elements of a process: starting from highest level activities, we decomposed them (and the corresponding fragment) down to the atomic steps that compose the work to be done; at each level of detail, we report the produced work products (with their work product kind) and the specific process role that perform/assist the work and above all the description of the MMM element that is defined/refined/quoted in each work product.

Because of our formalization needs we decided to adopt SPEM 2.0 (Software Process Engineering Metamodel) Specification; it is based on the idea that *"a software development process is a collaboration between abstract entity called process role that performs operations called activities on tangible entities called work products"* [11]. This well defined conceptual model allows to represent every kind of process lifecycle (iterative, incremental, waterfall and so on), SPEM in fact is composed of a breakdown structure allowing to represent all design processes.

In the next section we will describe which elements and diagrams we use of this specification and for each of them we will illustrate the related definition (from [11]).

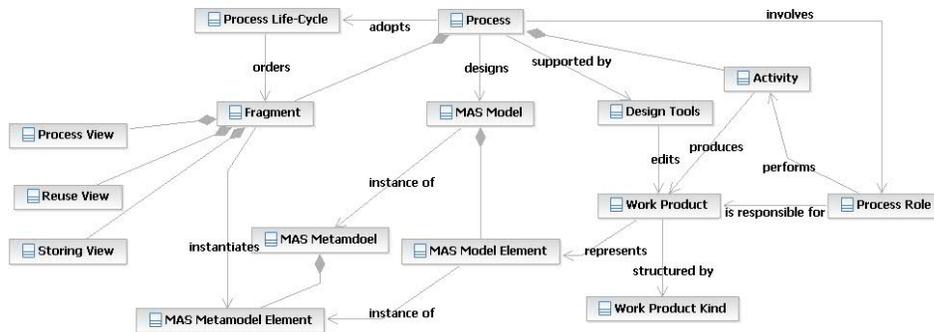


Fig. 2. The Agent Oriented Design Process Definition

3 Using SPEM for Representing Agent Oriented Design Process

Software Process Engineering Metamodel (SPEM) [11] is a meta-modelling language used for the description of development design processes and their components.

The SPEM 2.0 presents a metamodel structured in seven main packages; only three of them will be illustrated in this paper in order to justify their specific use for our purposes, they are *Process Structure*, *Method Content* and *Process With Method* Packages:

- The *Method Content* package contains all the elements for creating a set of reusable methods, its aim is to illustrate which are the goals that a method has to reach, which resources are used and which roles are involved.
- The *Process Package* is composed of the main elements for modelling a process: Activities, nested in a breakdown structure where the performing Role classes and the input and output Work Product classes for each activity are listed.

These elements are used to represent a high-level process that when instantiated on a specific project takes the method content elements and relates them into partially-ordered sequences that are customized to specific types of projects.

For our purposes and at a first level of abstraction, a process, through the Process Package elements, can be represented in its general structure without any reference to a specific project and without detailing the inner content description of each activity.

It is worth to note that in SPEM 2.0 the concepts of "Method" and "Process" have a specific meaning that allows their use respectively for representing and modelling the fragment and the design process.

SPEM 2.0 presents two levels for process representation: Method is considered at an higher abstraction level, where there is no reference to a specific project (and above all it is considered as an auto consistent portion of process) whereas a Process is the concrete representation of a specific development situation.

For all these reasons we found SPEM 2.0 suitable both for our top-down decomposition/representation of a design process and for the representation of each fragment. We can use the *Method Content* Package elements to represent a (process) fragment and the *Process With Method* Package elements to represent an existing design process; note that a development process, in our approach, is composed of process fragments (see section 2).

Finally in the *Process with Method* Package the central element is the BreakdownElement; in SPEM 2.0 processes are represented with a breakdown structure of Activities that nest BreakDown Elements; they are generalization of any type of Process Element such as other Activity instances, Task Uses and Role Uses. Therefore Activity represents a basic unit of work within a Process as well as a Process itself.

Role Uses represent the performer of a Task Uses and defines a set of related skills, competencies and responsibilities of a set of individuals and *Task Uses*

define the unit of work that is performed by Roles; a Task Use has a clear purpose in which the performing roles achieve a well defined goal. *Work Product Uses* are the artifacts, produced, consumed or modified by Task Uses³; Roles use Work Products to perform Tasks and produce Work Products in the course of performing Tasks.

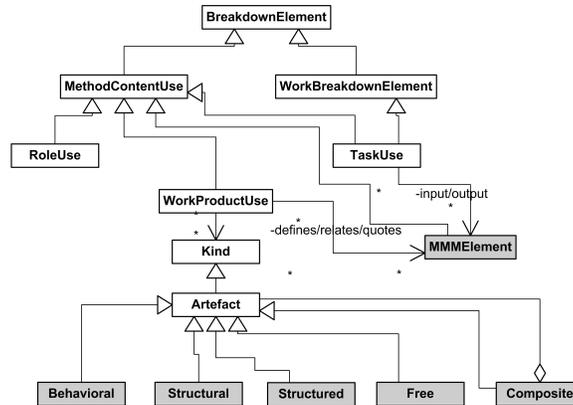


Fig. 3. The Proposed Extension to Process With Method Package Metamodel

The *Process With Method* Package contains the same elements of the process metamodel we presented in section 2, the Activity can be related/mapped to Task Use, The Process Role to Role Use and obviously the Work Product to Work Product Use. These elements together with the rationale of the work breakdown structure are sufficient enough for describing a complete design process for our purposes following a top-down approach, from the higher level definition of the work to be done until the details of each task with the roles performing it and the artifacts produced.

SPEM provides other two useful elements for grouping a set of activities under a common application theme, they are the *Process Component* containing one Process represented by an Activity and a set of Work Product Ports that define the inputs and outputs for the Process Component, and the *Phase* that represents a significant period in a project, with milestone or set of Deliverables.

With all the discussed elements a whole process can be divided into process components that groups the activities under a common theme, set of activities are also grouped into phases that impose specific milestones to the work to be done. Techniques, methods and guidelines for each activity are given for tasks that are performed by roles and consumes/produces (has input/output) work products.

³ From now on, for the sake of brevity, Task and Task Use, Role and Role Use, Work Product and Work Product Use will be indifferently used, with the same meaning.

4 Extending SPEM specifications

In the previous section we saw which elements of SPEM metamodel packages we use for our purposes of providing a standard representation for agent oriented design processes. In this section we illustrate the motivation that led us to extend SPEM.

We had to deal with three specific factors that had a direct consequence in the extension of SPEM specifications: the MAS Metamodel element and the Work Product Kind that are elements of our process metamodel not provided by SPEM Packages and the existing dependencies among work products.

Regarding the MAS metamodel element, its definition is the core of the work done in each portion of process resulting in the delivery of a work product; as section 2 illustrates, using a specific design process for developing an agent system means to instantiate its metamodel, this instantiation results in a precise set of actions that we identified to be done on each MMM element; a designer can in fact

- **define** an element (i.e. instantiate it) thus establishing its properties and/or attributes; the resulting instantiated element is, of course, reported in the work product produced by the fragment the designer is executing
- similarly, the designer can **relate** a MMM element to other elements or
- simply **quote** it for introducing relationships among work products.

SPEM 2.0 does not provide means for representing actions to be made, in each activity, on the MMM elements.

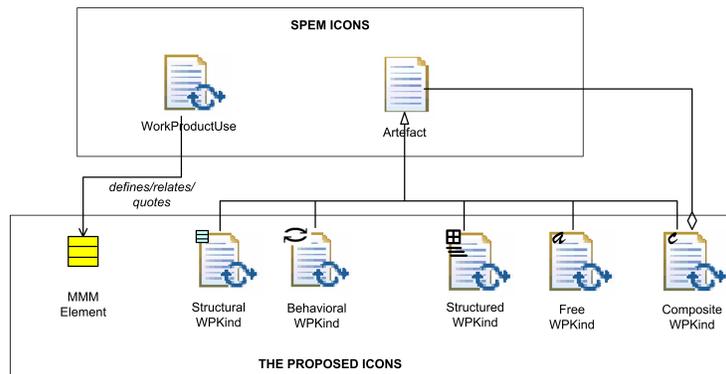


Fig. 4. The Proposed Icons

An important factor in the process representation is also modelling work products dependencies; SPEM specifications provide a way for modelling dependencies through the so called Work Product Dependency Diagram.

The Work Product Dependency diagram supplies way for representing three kinds of dependency relationships among work products: Composition, Aggregation and Dependency. The first expresses that a work product is a part of another one, the second indicates that a work product uses another work product and finally the third indicates that a work product depends on another one.

In our case a well specific dependency among a work product and another exists, it is due to the relationships among elements in the MAS metamodel; for instance let us consider an hypothetical requirement elicitation phase of a given process, we could think that this portion of process is represented by a metamodel where the concept of actor is related to that of scenario. Now, let us further suppose that the designer performing this phase begins his work by defining the concept of actor and produces a document listing all the actors he has identified. When the designer has to define the concept of scenario, because it is related to the concept of actor, he has to look at the last document he created, he cannot proceed without knowing all the defined actors; in so doing all the relationships among metamodel elements are reflected upon a precise dependency among the work products that act upon them.

In order to represent such situations, we need a diagram reporting both the dependencies among work products (and this is provided by SPEM) and the correspondence between each work product and the MMM elements, this second point led us to the creation of a specific diagram.

The SPEM Work Product Kind allows to represent a work product when it: *i*) is an intangible one or it is not formally defined (in this case it is of the kind: Outcome), *ii*) it aggregates other work products (the kind is Deliverable) and *iii*) it defines a tangible work product consumed, produced or modified by a task (the kind is Artifact).

In our work we defined five kind of work products [16] as a result of our need for adequately storing process fragments in our repository; the defined kinds on work products are:

1. Behavioural, it is a graphical kind of work product and is used to represent the dynamic aspect of the system (for instance a sequence diagram representing the flow of messages among agents along time);
2. Structural, it is also a graphical kind of work product and is used for representing the static aspect of the system, for instance a UML class diagram;
3. Structured, it is a text document ruled by a particular template or grammar, for instance a table or a code document;
4. Free, it a document freely written in natural language.
5. Composite, this work product can be made by composing the previous work product kinds, for instance a diagram with a portion of text used for its description.

These definitions together with that proposed by SPEM allow us to consider the work product kind we use as a specialization of the SPEM Artifact Work Product Kind (Figure 3).

Therefore for applying our design process definition we made some specific extensions to the Process With Method Package by adding some elements; in

Figure 3 we show the portion of Process With Method metamodel that we extended; the white elements are all the pre-existing SPEM ones whereas the gray elements are newly introduced one.

To sum up, we added five elements and for each of them the related icon was created (see Figure 4); the relationship between Work Product Use and MM-Element found its realization in a new diagram that represents the correspondence between each work product and the MMM elements it defines/relates/quotes.

Figure 5 shows the kind of diagram we created in order to represent dependencies among Work Product through MMM elements and shows the artefact produced during the PASSI Agent Society phase; it is composed of work products of three kinds (structural, behavioural, structured) and for each of them all the MMM elements they work on are shown, the letter indicates the specific action that is made on the MMM element: **D** stands for Define, **R** for Relate and **Q** for Quote.

Referring to the Agent Society phase that one of the Process Component elements of the PASSI process, as it will be seen in the next section, using the presented kind of diagram we can see that it results in the Agent Society model, modelled through a Work Product Use composed by two structural work products, two behavioral ones and one structured.

One of the greatest advantages of using the diagram shown in Figure 5 is to have a complete vision on all the metamodel elements that are defined/related/quoted in each work product delivered from each process activity. Mind that the proposed SPEM representation aims at identifying reusable fragments from existing design processes and that in our approach each process fragment is devoted to instantiate (hence define) at least one MMM element. Such a view on the existing dependencies among MMM elements and work products allows us to quickly identify the possible fragments (how the fragments extraction is carried out is not the focus and is not detailed in this paper).

5 Representing PASSI with SPEM 2.0

Our representation of a design process follows a well specific method, it is carried out in a top-down fashion in order to reach the right level of granularity allowing the extraction of process fragments.

Therefore a process is represented, in a first time, as a package of components through the *Process Component Diagram* that allows to represent all the portions of a design process with the needed input and output, this diagram models the design process at a high level of detail.

In Figure 6 it can be seen, for instance, the Process Component Diagram related to PASSI [4], it is composed of five components, each of them representing a phase, a portion of work for which a specific outcome and milestones can be identified and represented in this diagram.

The second step is to detail the work done in each component through a SPEM Activity Diagram, Figure 7 shows all the activities nested in the Agent

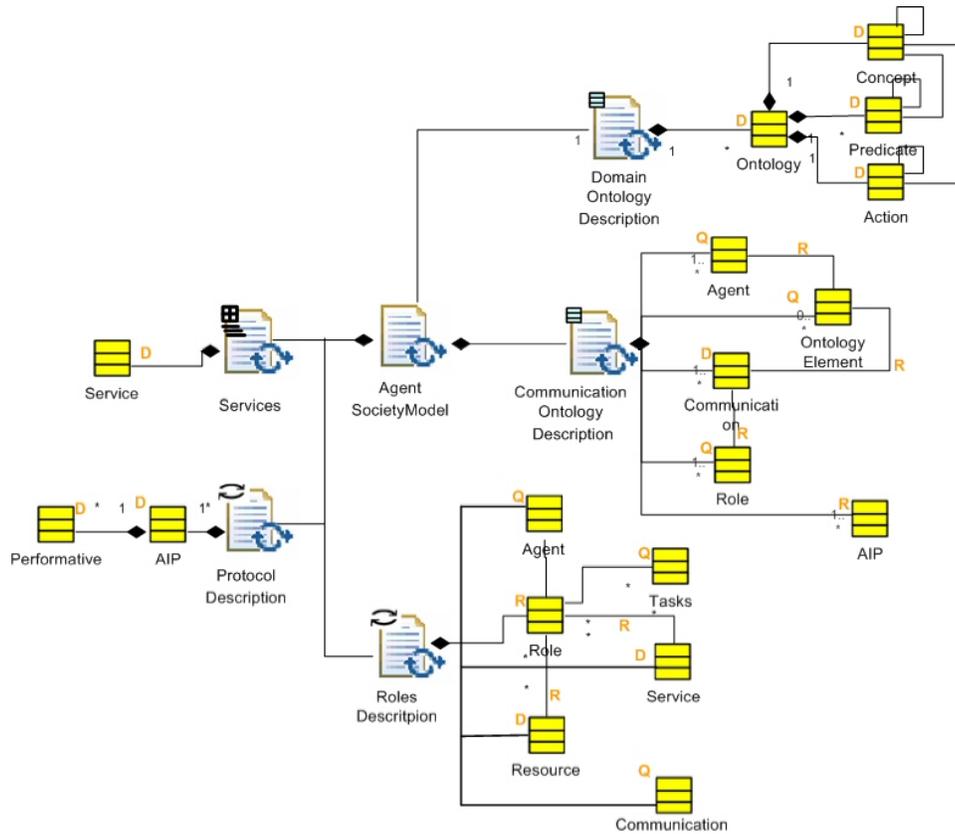


Fig. 5. The Correspondence among Work Product and MMM elements

Society component, the activities sequence is indicated through the «predecessor»stereotype (the pointed activity is the predecessor of the other one). Each activity is composed of tasks, roles performing the task and input/output work products, no tasks sequencing is necessary at this step, it is only useful to give the idea of the set of elements an activity is composed of (in the figure only the Role Description activity is shown for space reasons).

When complete, this diagram is full of information that could be accompanied by definition, explanation and so on, but it can result too huge, in alternative another diagram can be produced where only activity and input/output work products are shown.

This level of detail is sufficient in order to identify all the process fragments that can be extracted from a design process.

For a detailed documentation of the whole design process other diagrams have to be produced; these diagrams allow to model and document all the techniques, the methods and the guidelines involved in each task, i.e. the dynamic part of

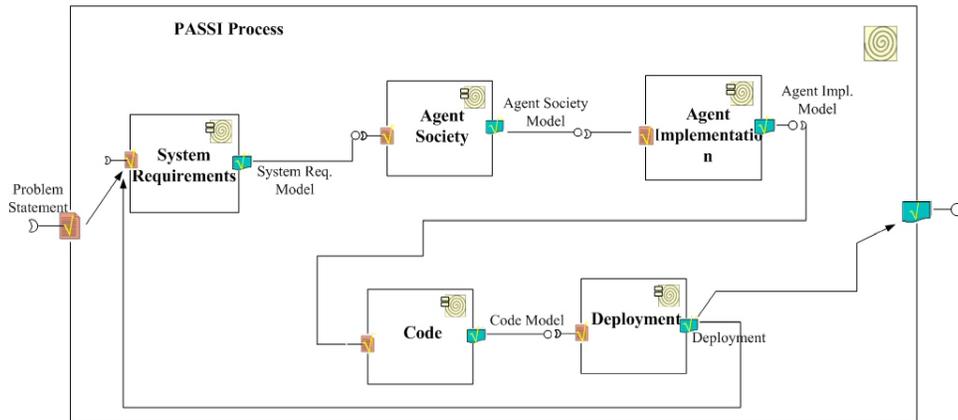


Fig. 6. The PASSI Process Component Diagram

each portion of process, for space reasons we do not specify this part in this paper.

As already said, our process fragment extraction is work product-oriented [5], in the sense that we look at a work product and extract from the whole process only the portion of work delivering the selected work product.

For instance looking at the Role Description we can see that there are two outputs, Services and Role Description diagram; then from Figure 5 (this is the final diagram we draw when we model a design process) we can identify the MMM elements these work products act on and so we could decide to extract a fragment dealing with the concept of role and delivering the Role Description diagram and/or one dealing with the concept of service; of course it is also possible to identify a fragment delivering both the work products.

The novelty of the presented work is principally the possibility of easily retrieving a set of reusable fragments starting from the representation in a standard way of available design processes and then, once the fragments are identified and extracted, they can be documented in a similar standard way.

In our approach, in order to represent fragments, we also need elements such as activities, roles, work product and MMM elements and the way we indicate to use SPEM 2.0 specification and our extension will serve this scope.

Besides it is important to note that the standard representation of the fragments allows, during the application of the whole SME process for creating agent design processes, an easy and quick selection from the repository for the assembly phase [16].

In [5] we illustrated how to document a fragment, here we give a brief hint on it, let us suppose to start from the representation of PASSI (see the previous subsection) and suppose that from the diagrams reported in Figures 5 and 7 we had identified the MMM element we want to be designed in the fragment, the related work product and then the portion of work to be done to have this

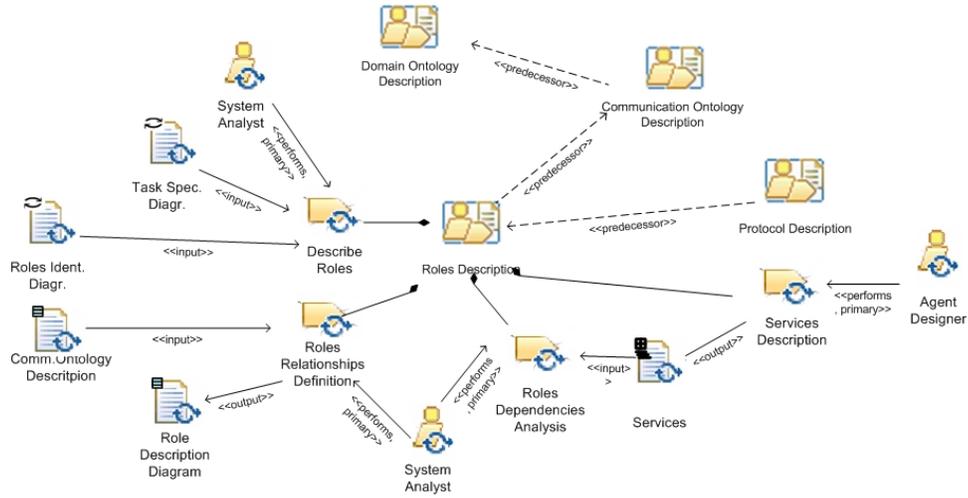


Fig. 7. The PASSI Process Activity Diagram

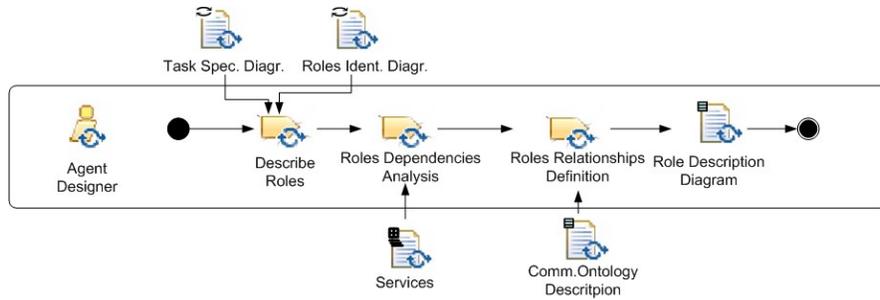


Fig. 8. The "Roles Description" Fragment Description

result. What we have to do now is to model/represent this portion of work with its resulting product; in Figure 8 a SPEM workflow diagram is shown, here we can see the flow of work the specific role has to perform in order to produce the "Role Description" diagram and all the inputs (obviously coming from other fragments) needed.

This latter diagram introduces a deeper level of detail with respect to the diagram shown in Figure 7, in fact it details the sequence of activity; it is helpful for the fragment documentation.

6 Discussions and Conclusions

The approach we use for adapting Situational Method Engineering to agent oriented design processes construction is composed of three phases, as it is in other approaches in literature [6][12]; in this paper we mainly consider the part

regarding the (process) fragments documentation in a form that encourages the fragment extraction from an existing process and the repository construction.

We started this work from a set of existing design processes that we represented and fragmented in a standard fashion in order to extract a set of fragments coherent with the definition we gave.

The first problem we faced was to find a way for representing a design process coherently with the process metamodel we consider in our approach (activity, process role, work product and MAS metamodel element); we decided to follow a top-down approach for the design process representation using and extending, when necessary, the SPEM 2.0 specifications.

We decided on SPEM .0 and we think it is well suited to our purposes because it is an OMG standard, in our work we always had great attention for the standards, and above all because of the conceptual metamodel it is based on. The main elements we find in SPEM metamodel are: activity, process role and work product; due to their definition these elements well fit our definition of agent design process elements we gave in [5].

The breakdown structure, SPEM is composed of, let us to represent PASSI only using three kinds of diagrams; one of them is the extension we made on the work product dependency diagram provided by SPEM. This extension together with the other two ones we made was due to our concerns about MAS metamodel, whose elements cannot be represented used SPEM packages elements, and for representing the kind of work product we use in our work for categorize the fragments in the repository [16].

All the approaches present in literature draw on a more or less huge or formalized and structured repository of fragments and each researcher in the field of SME takes care of populating it by extracting fragments from existing design processes [1][14][7][13].

However guidelines on how to do that are still lacking; only in [12] a complete work on that can be found. There Ralyté proposes an approach based on the decomposition of existing processes into smaller components aiming at satisfying specific goals, but it seems that this approach is to bound to the rigidity of existing design processes (not originally created to be modular, as the same author says) and to the quality of the model provided for the process and the product parts of the design process.

Our approach is also based on the concept of decomposition but it tries to overcome these problems by firstly adopting a way for representing in the proper fashion a design process and then extracting from them fragments.

In the past we used SPEM 1.1 in our work and we modelled several agent design processes, we found several difficulties that are now over with SPEM 2.0; one of the greatest advantages we found in using SPEM with the proposed extension is that it is possible to represent a whole design process only through three diagrams from which all the essential information for fragmentation can be easily and quickly gathered.

Our research is principally focussed, and this was the starting point, on the application of Situational Method Engineering for the construction of ad-hoc

agent oriented design processes but while working we realized that, due to the use of metamodel, to the elements we identified for a design process definition and to the use of SPEM for representing them, our approach results general enough to be applied to every kind of design process construction.

In the future we are going to experiment and to test SPEM 2.0 for other processes and to formalize a set of guidelines for correctly model an agent design process for the scopes of Situational Method Engineering applied to every application contexts and research fields.

References

1. Henderson-Sellers B., Serour M., McBride T., Gonzalez-Perez C., and Dagher L. Process construction and customization. *j-jucs*, 10(4):326–358, apr 2004.
2. S. Brinkkemper. Method engineering: engineering the information systems development methods and tools. *Information and Software Technology*, 37(11), 1996.
3. Atkinson C. and Kuhne T. Model-driven development: A metamodeling foundation. *IEEE Software*, 20(5):36–41, September/October 2003.
4. M. Cossentino. From requirements to code with the PASSI methodology. In *Agent Oriented Methodologies*, chapter IV, pages 79–106. Idea Group Publishing, Hershey, PA, USA, June 2005.
5. M. Cossentino, S. Gaglio, A. Garro, and V. Seidita. Method fragments for agent design methodologies: from standardisation to research. *International Journal of Agent-Oriented Software Engineering (IJAOSE)*, 1(1):91–121, 2007.
6. D. Gupta and N. Prakash. Engineering Methods from Method Requirements Specifications. *Requirements Engineering*, 6(3):135–160, 2001.
7. A.F. Harmsen, M. Ernst, and U. Twente. *Situational Method Engineering*. Moret Ernst & Young Management Consultants, 1997.
8. Brian Henderson-Sellers. Method engineering: Theory and practice. In *ISTA*, pages 13–23, 2006.
9. K. Kumar and R.J. Welke. Methodology engineering: a proposal for situation-specific methodology construction. *Challenges and Strategies for Research in Systems Development*, pages 257–269, 1992.
10. I. Mirbel and J. Ralyté. Situational method engineering: combining assembly-based and roadmap-driven approaches. *Requirements Engineering*, 11(1):58–78, 2006.
11. Software process engineering metamodel. Version 2.0. Final Adopted Specification ptc/07 03-03. Object management group (omg), march 2007.
12. J. Ralyté. Towards situational methods for information systems development: engineering reusable method chunks. *Procs. 13th Int. Conf. on Information Systems Development. Advances in Theory, Practice and Education*, pages 271–282, 2004.
13. J. Ralyté and C. Rolland. An approach for method reengineering. *Lecture Notes in Computer Science*, pages 27–30, 2001.
14. Brinkkemper S., Saeki M., and Harmsen F. *Assembly Techniques for Method Engineering*, volume 1413/1998 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 1998.
15. Shavrin S. Ontological multilevel modeling language. *International Journal Information Theories & Applications*, 14, 2007.
16. V. Seidita, M. Cossentino, and S. Gaglio. A repository of fragments for agent systems design. *Proc. Of the Workshop on Objects and Agents (WOA06)*, 2006.

17. V. Seidita, J. Ralyté, B. Henderson-Sellers, M. Cossentino, and N. Arni-Bloch. A comparison of deontic matrices, maps and activity diagrams for the construction of situational methods. In *CAiSE'07 Forum, Proceedings of the CAiSE'07 Forum at the 19th International Conference on Advanced Information Systems Engineering.*, pages 85–88, Trondheim, Norway, 11-15 June 2007.