The Process Fragment Metamodel Definition and Documentation

Valeria Seidita[1], Massimo Cossentino[2], Antonio Chella[1] and Salvatore Gaglio[1]

[1] Dipartimento di Ingegneria Informatica
Universitá degli Studi di Palermo, Palermo, Italy
{seidita, chella}@dinfo.unipa.it
[2] Istituto di Calcolo e Reti ad Alte Prestazioni, Consiglio Nazionale delle Ricerche
Palermo, Italy
cossentino@pa.icar.cnr.it

Abstract. This paper focuses in the field of Situational Method Engineering for the construction of agent based design processes. Whatever SME approach a method designer wants to use he has to manage two main elements: the fragment and the repository where it is stored. Specific fragment definition and documentation are useful during SME and system design activities. This paper aims at illustrating the fragment definition and documentation helpful for being principally used within PRoDe.

1 Introduction

The work presented in this paper starts and is based on the work done during the latest years towards the definition of the best way to create ad-hoc agent oriented design processes. The development of a multi agent system always requires great efforts in learning and using an existing design process. It has been said and heard several times that it does not exist one design process (or also a methodology or a method) for developing every kind of systems able to solve every kind of problems and there is the need of creating techniques and tools for a designer to develop an ad-hoc design process prior to use it on the base of his own needs. In order to solve this problem and to give means for one to develop an agent system using the "right" design process we adopted the Situational Method Engineering (SME) approach and started from pointing out what we intend for design process. In [3] the main elements of an agent-based design process have been identified, they fundamentally ground on three of the main elements a designer would always meet during design, they refer to the *stakeholders* that perform *activities* in order to produce *design results* (also labelled *work products or artefacts*); this important triad has been augmented by the consideration, also following the MDE[13] approach, that producing design results, is nothing else but instantiating elements from a (meta-)model. To be more precise the multi agent system solving a specific problem is that running on a platform and it is called the instance model, it represents the elements that exist as the system runs on the real-world platform (level M0); the instance model is created by compiling the code generated from what is called the user model (level M1) and finally at level M2 there is the metamodel that defines the language in which the user model is captured. Therefore the design process deals with all the elements (level M2, the system metamodel) that the designer instantiates in the user model and then in the running system when s/he is using a specific design process. SME provides tools and techniques for creating design processes by reusing portion of existing ones, called *method fragment*, stored in a *method base.*

The system metamodel is the fundamental element to be defined when following the Situational Method Engineering approach, PRoDe, we recently created [16], here we assimilated the process of constructing a new design process to the process for developing software [10]; softwares requirements have to be adequately analysed and used for defining the design process requirements that in turn are used to define the system metamodel that will allow the realization of a software architecture and the related set of components that will satisfy the system requirements. It can be said that the creation of a design process can be done by following specific phases from analysis to implementation; a lot of existing Situational Method Engineering approaches [6][12][1][11][9][7] are developed around three main phases: the *process requirements analysis*, the *process fragments selection* and the *process fragment assembly.*

The first activity in the PRoDe approach entails a set of steps that, starting from the process requirements, are able to produce the system metamodel or in any case a first draft of it. PRoDe, in fact, is iterative and after a first enactment of the new design process this might be modified/enhanced due to test results and new requirements identification. As regard selection and assembly the PRoDe approach provides a well defined set of activities for identifying and retrieving fragments from repository basing on some considerations made on the system metamodel resulting in the prioritization algorithm.

The prioritization algorithm produces a priority list defining in which order the elements of the metamodel have to be instantiated during the process workflow. Since each metamodel element may be instantiated by the activities to be performed within a process fragment, this priority list helps less experienced method engineers in retrieving fragments from repository in the correct way. After fragments selection, the PRoDe approach provides guidelines, resulting in the construction of what is called *component diagram*, for assembling the fragments in the new design process. The PRoDe activities, as well as other SME approach activities, are also highly grounded on the SME fundamental element, the *Process Fragment* (or method fragment or chunk or simply fragment - however it is named by different researchers), and obviously on the repository aimed at storing it. In order to apply a SME approach in the most fruitful way, a well done definition and documentation of process fragment is useful for properly storing, selecting and assembly new design processes whatever SME approach one wants to follow.

In this paper we focus on the process fragment definition and documentation by identifying its main elements and following a twofold aim: reuse, in the PRoDe point of view, hence providing all the information supporting the selection and assembly phases, and reuse in a general design point of view, hence providing the information for a designer to follow the guidelines for carrying out the portion of work described in the fragment and how to produce the related artefacts.

In the following section the definition of process fragment and its underpinning metamodel will be shown together with an example of documentation. Our aim is to give a process fragment definition aiming at well documenting it in order to be used both at system design time and for being reused in storage and assembly.

2 The Process Fragment Definition

The process fragment is a portion of design process and as such it stands, as said in the previous section, at the M2 level of the design process level categorization; the process fragment definition proposed in the following contains all the M2 elements that have to be instantiated in a model when the fragment is used inside a complete process and then in the running system.

2.1 The Process Fragment Metamodel

A *process fragment* is a portion of design process adequately created and structured for being reused during the composition of new design processes both in the field of agent oriented software engineering and in other ones (model driven engineering-based approaches are preferred fields of application for the proposed definition). The process fragment is, generally (this is the most common case), extracted from an existing design process and it is stored in a repository. Two things to be noted at this point: the process fragment definition together with the specific SME process (see for instance [16]) used for retrieving and composing fragments notably influence how the repository is conceived and constructed. Conversely, constructing from scratch a process fragment can be done in the same way a design process is composed using the SME approach.

Figure 1 shows the metamodel of the process fragment proposed in this paper, it contains all the elements useful for representing and documenting the fragment under the process, product and reuse point of view; the proposed fragment documentation template slavishly follows the proposed metamodel, its elements and their definition.

The root element, the *Fragment*, has been generally extracted from an existing design process, therefore an important information to be stored in the repository is the *Design Process* the fragment refers to, this serves for the designer to set the application context and the particular features the fragment would exhibit.

The process fragment is composed of activities, each of them is a portion of work that has to be performed by one or more stakeholders (*Process Roles*) and can be decomposable in other activities or can be atomic in the sense that it is

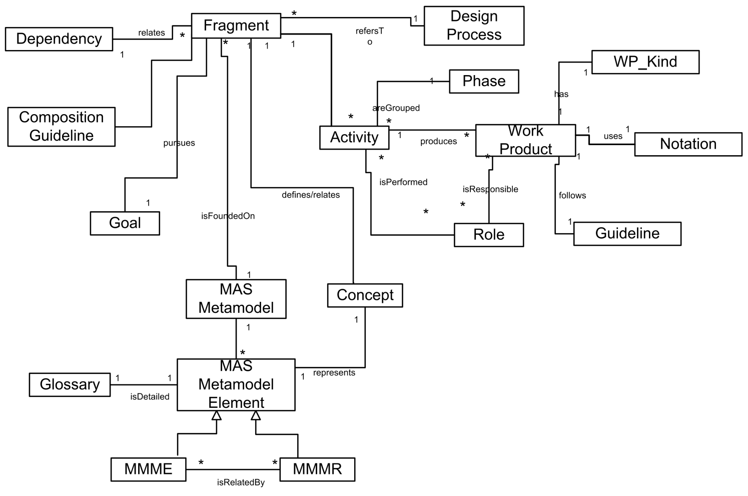


Fig. 1. The Process Fragment Metamodel

a single design action performed by only one process role. It is also important to indicate if the process role, while performing the activity, is the main performer, the main responsible, or he is assisting another role.

Design process usually is decomposed into phases, let us think for instance to the Unified Process (UP)[8], it presents different iterations during which Analysis, Design etc. phases are repeated with different levels of details and each phase is characterized by the fact that the activities here performed have a common scope, specific deadlines and constraints: for instance, Design phase can not start if Analysis phase is not finished. For that reason we need to identify the *Phase* the fragment refers to. In order to make easy storing and reusing fragments we identified in the past a taxonomy [14] for classifying phases, process roles and work products.

Activity delivers *Work Products*, where the results of design activities are drawn by using a specific *Notation* and each work product is developed under the responsibility of one process role. The notation to be used greatly influences the flow of work to be done for producing a work product and for this reason a fragment has to be supplied with a set of *Guidelines.* It is not mandatory to follow a specific notation, the same kind of diagram (for instance a structural one) may be expressed by using different notations without significant differences in the resulting expressiveness. Moreover, different kinds (*WP_Kind*) of work products can be delivered, we identified two main work product kinds: graphical and textual, the former when an activity results in a diagram the second when designers produce textual documents. Finally a work product can be of composite kind if it is a composition of the previous said kinds, for instance a document with a diagram and the text explaining it [15].

THE FRAGMENT METAMODEL	
Name	***Description***
Activity	A portion of work assignable to a performer (role). An activity may be atomic (sometimes addressed as Action) or composed by other activities.
Composition Guideline	A set of guidelines for assembling/composing the fragments with others. This may include notational specifications, and constraints (also addressing issues like platform to be used for system implementation and application area)
Work Product	The resulting product of the work done in the fragment; it can be realized in different ways (diagram, text,..) also depending on the specific adopted notation.
WP_Kind	Represents the specific kind one work product can be; it strictly depends on the means the adopted notation provides. One work product can be
Dependency	The description of specific dependencies of this fragment from other ones; it is useful for composition.
Design Process	It is the design process from which the fragment has been extracted.
Glossary	A list of definitions for the MAS metamodel elements.
Goal	The process-oriented objective of the fragment.
Phase	A specification of the fragment position in the design workflow. Usually referring to a taxonomy (i.e. Requirements, Analysis, Design, etc.)
Guideline	The description on how to realize the fragment's deliverable hence specifications about the activities for composing a deliverable following a specific notation.
MAS Metamodel	The structure of concepts underpinning the fragment.
MAS Metamodel Element Type	The main elements the fragments deals with, both in terms of outputs and inputs, for instance a fragment aiming at defining the system requirements has to define and to identify the concept of "requirements". It stands for Multi agent system metamodel element and relationship and represents the items the fragment's metamodel is composed of. Each metamodel component has to be defined during, at least, one portion of process work and has to appear in at least one work product.
MMME/MMMR	A MAS metamodel element can be an element or a relationship among elements; both the two have to be designed in a portion of work and represented in a work product.
Notation	Each deliverable can be drawn by using a specific notation. Concepts dealt by the fragment have to find a mapping in the notation. Notation usually includes a metamodel and a set of pictorial prescriptions used to represent the instantiation of metamodel elements.
Process Role	The stakeholder performing the work in the process and responsible of producing a deliverable (or a part of it) usually referring to a taxonomy (i.e. System Analyst, Test Designer, etc.)

Table 1. The Fragment Metamodel Definition

As well as in the design process definition, one of the most important elements in the fragment definition is the *MAS Metamodel* (Multi-Agent System Metamodel); each fragment underpins a metamodel that is obviously part of the metamodel of the design process it comes from. The metamodel contains the set of elements representing the system to be designed using a specific process fragment. In the case of fragment definition we have to consider that MAS metamodel are composed of elements (*MMME* - the concepts to be designed) or relatioships among them (*MMMR*).

The main aim of process fragment is to instantiate one or more MAS metamodel elements, one fragment should at least instantiate one MMME/MMMR and in so doing it may be requested to define relationships with other elements or to quote other elements and/or relationships; besides the result of defining an element or a relationship might be the refinement of existing elements or relationships. This fact led to the definition of the kinds of action to be done on a MAS metamodel element (see the following section for details). Finally the MAS metamodel element has a definition to be listed in a glossary; the definition is mainly useful during selection when the method designer must know which kind of MAS metamodel element better fits with the MAS metamodel element s/he is dealing with.

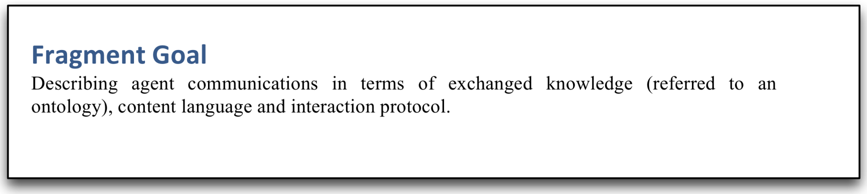
Until now the process and product part of the fragment metamodel has been explored though a set of elements that has to be necessarily present in the fragment documentation, now let us focus on the elements that principally deal with the reuse aspect of the fragment: *Goal*, *Dependency* and *Composition Guideline*.

The fragment goal is the objective the process part of the fragment wants to pursue and it is to be used during fragment selection from the repository. For this reason it is related to the new design process requirements, in other words, a goal describes the contribution a fragment may give to the accomplishment of some design process requirements. The dependency aims at describing specific constraints, if they exist, for the fragment to be composed with other ones, for instance, there can be fragments dealing with MAS metamodel elements that are very specific to particular application domains, in this case it should be possible that such fragments can be composed with fragments coming from the same classes of design processes.

Table 1 resumes the process fragment metamodel elements and their definitions.

2.2 Some Details about Process Fragment Definition

It is important noting that the way the work has to be performed inside one fragment may slightly change depending on the notation of the work product produced; if the result has to be a graphical work product the activity and the related guidelines are different if we want to use two different notations. But since the fragment aims at designing a specific MAS metamodel element type, we can consider the fragment itself independent from the specific notation, the same result can be obtained by producing different work products in different notations. Such a feature is one of the strengths of the proposed fragment definition that is highly reusable and composable being mainly oriented to the metamodel element it is aimed to define; for instance a fragment that delivers UML based work products can be easily composed to another fragment delivering free textual work product, it is only important that the two have a matching set of input/output metamodel elements. This fact overcomes the problem, until now present, of having all fragments producing work products with the same notation; at worst we could create design processes where different parts have different notation but also this problem can be overcame, even if it were a problem, by using a CAPE tool able to instantiate the right CASE tool for managing the enactment of the newly created design process. An example of such CAPE tool is Metameth, a prototype that we developed in the past in our laboratory [4] .


Fragment Goal

Describing agent communications in terms of exchanged knowledge (referred to an ontology), content language and interaction protocol.

Fig. 2. A Portion of the COD Fragment Document - The Fragment Goal

3 An Example of Process Fragment Description

All the elements of the fragment metamodel are reported and described in the fragment document structured basing on the following outline:

– Fragment Goal
– Fragment Origin
– Fragment Description - The Portion of Process workflow
– Fragment MAS metamodel - and the related Glossary
– Deliverable relationships with the MMM
– Deliverables
– Example of notation
– Preconditions and concepts to be defined
– Guideline
– Composition Guideline
– Dependency Relationships with other fragments

In the following an example of fragment documentation is given through a set of figures that have been captured from the document related to the *Communication Ontological Descritpion - COD* process fragment from PASSI [2]. Each figure represents a relevant portion of the document, the complete version of this fragment can be found in the FIPA DPDF working group website[3].

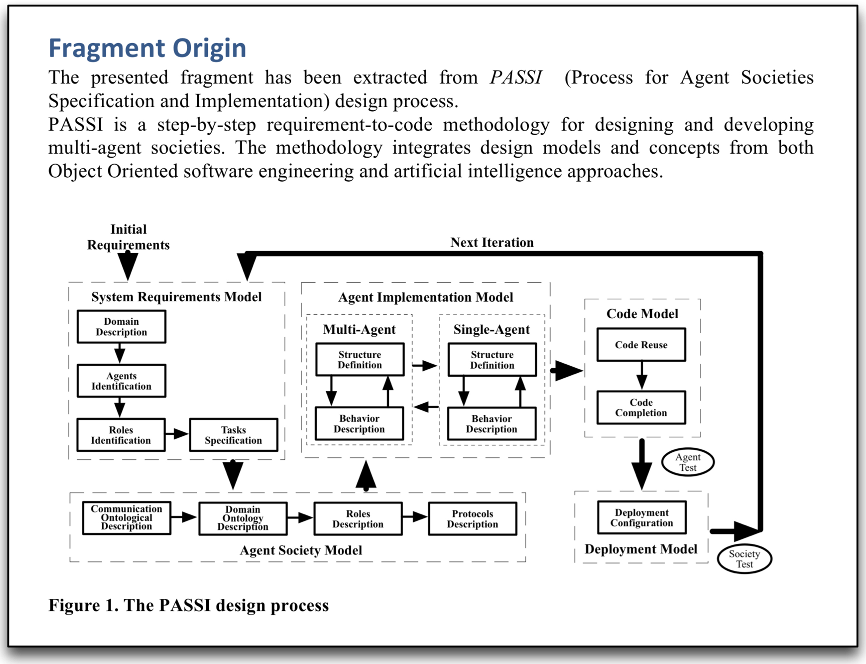
Looking at the fragment outline it can be seen that first of all we focus on the fragment presentation through its goal and its origin, in so doing we reach a twofold objective, letting the designer have a quick idea on the focus and the domain in which the fragment might work and allowing a sort of automatic or semiautomatic selection of the fragment.

Figure 2 shows the fragment goal. This is described in a very concise textual form that puts in evidence the main elements the fragment will deal with, for instance it can be noticed the words *agent communication*, *knowledge* and *protocol*. It is to be hoped that this part of the document were compiled using words focussing on the fragment scope.

Figure 3 shows a portion of the section dedicated to the design process the fragment has been extracted from, the importance of this early discussion has

[3] http://www.pa.icar.cnr.it/cossentino/fipa-dpdf-wg/docs.htm

been already said. The description can be augmented through diagrams and whatever kind of information is useful for illustrating the design process, it is recommended not to be excessively long in order not to loose the focus on the fragment process oriented part of the document. As regard the diagrams, it is highly recommended, but it is not mandatory, to use SPEM 2.0 notation as it has been decide within IEEE-FIPA-DFDP Working Group [5] and for SPEM presents a process metamodel grounded on the three elements we indicated at the beginning of this paper as the most important for representing a process: activity, role and artefact.



Fragment Origin

The presented fragment has been extracted from *PASSI* (Process for Agent Societies Specification and Implementation) design process.
PASSI is a step-by-step requirement-to-code methodology for designing and developing multi-agent societies. The methodology integrates design models and concepts from both Object Oriented software engineering and artificial intelligence approaches.

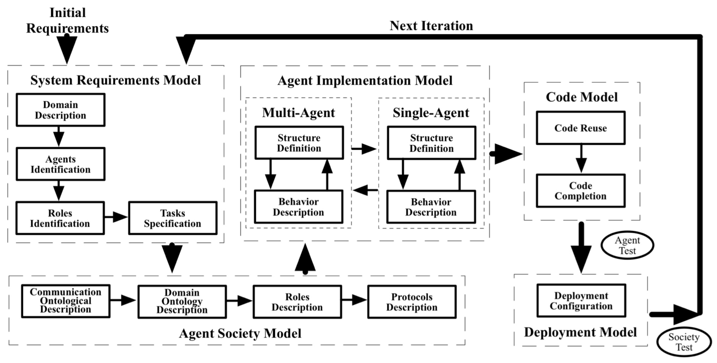


Figure 1. The PASSI design process

Fig. 3. A Portion of the COD Fragment Document - The Origin Design Process

Figure 4 refers to the fragment description section and it is aimed at detailing the fragment scope, its main inputs and outputs, the portion of process from which it has been extracted with all its input and output (in the case of this example the Agent Society phase) and then the fragment process workflow through a SPEM 2.0 activity diagram. The use of SPEM 2.0 together with the extensions proposed in [15] allow to represent the result of each part of the process in terms of the right work product kinds and to outline the process role

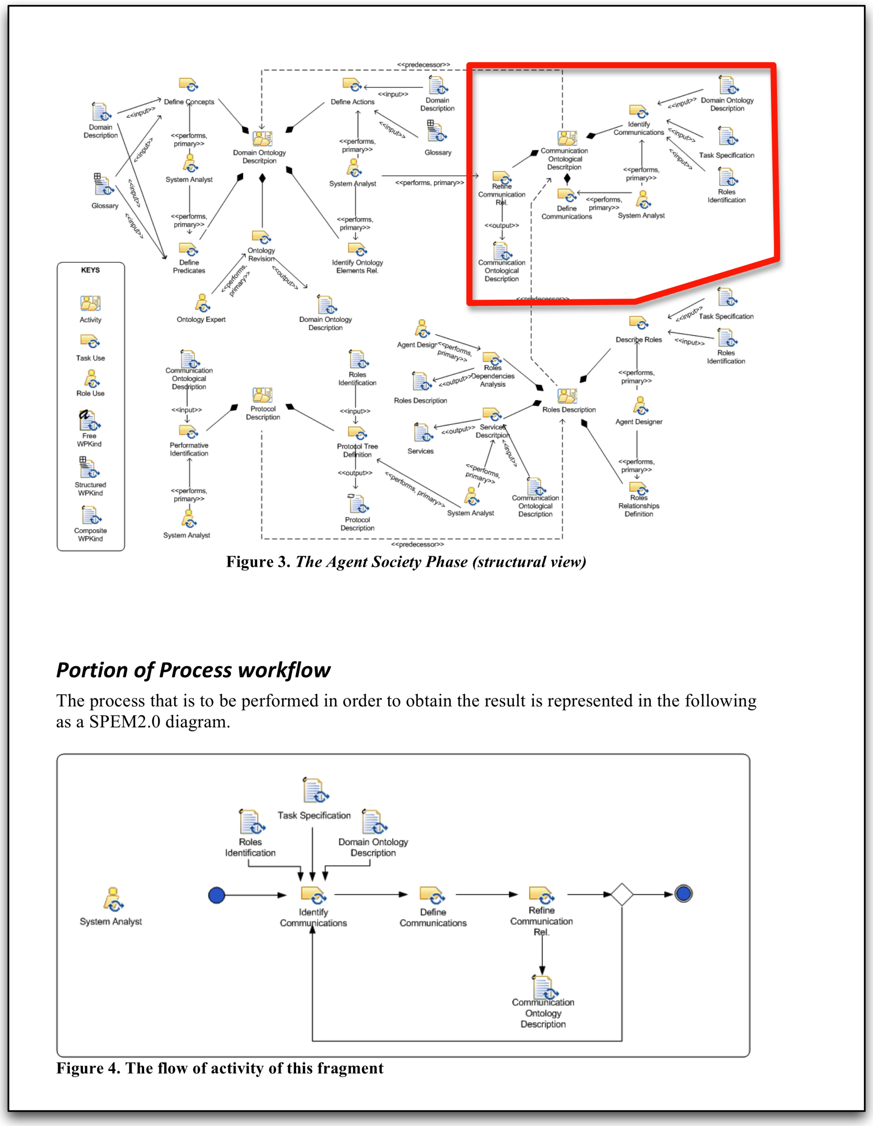


Figure 3. *The Agent Society Phase (structural view)*

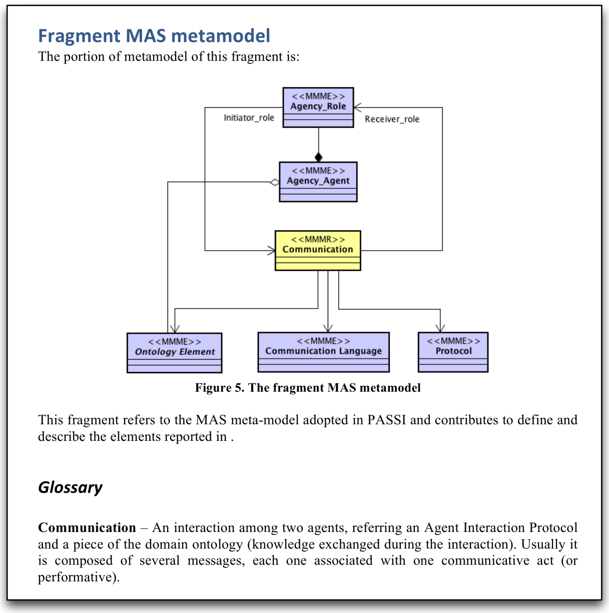
Portion of Process workflow

The process that is to be performed in order to obtain the result is represented in the following as a SPEM2.0 diagram.

Figure 4. The flow of activity of this fragment

Fig. 4. A Portion of the COD Fragment Document - The Fragment Description

performing or assisting during a specific task, for instance it can be seen that the “Refine Communication Rel.” produces the “Communication Ontology Description diagram” that is a composite work product; this latter is composed of a class diagram (see the first part of Figure 7) and the text and tables explaining it. As well as a design process each process fragment underpins a MAS metamodel composed of elements and relationships; the fragment document has to explore this issue and to show all the elements type to be defined/quoted/related in the fragment. Figure 5 shows the COD portion of metamodel; it can be seen that the two types of elements present different colours and stereotypes, besides each element has to be supplied with the proper definition to be used for design concerns but principally during the SME selection phase for the method designer to be able to identify the elements s/he needs and for deciding if the fragment can be used for her/his purposes. As already said the process fragment



Fragment MAS metamodel

The portion of metamodel of this fragment is:

<<MMME>> Agency_Role
Initiator_role
Receiver_role
<<MMME>> Agency_Agent
<<MMMR>> Communication
<<MMME>> Ontology Element
<<MMME>> Communication Language
<<MMME>> Protocol

Figure 5. The fragment MAS metamodel

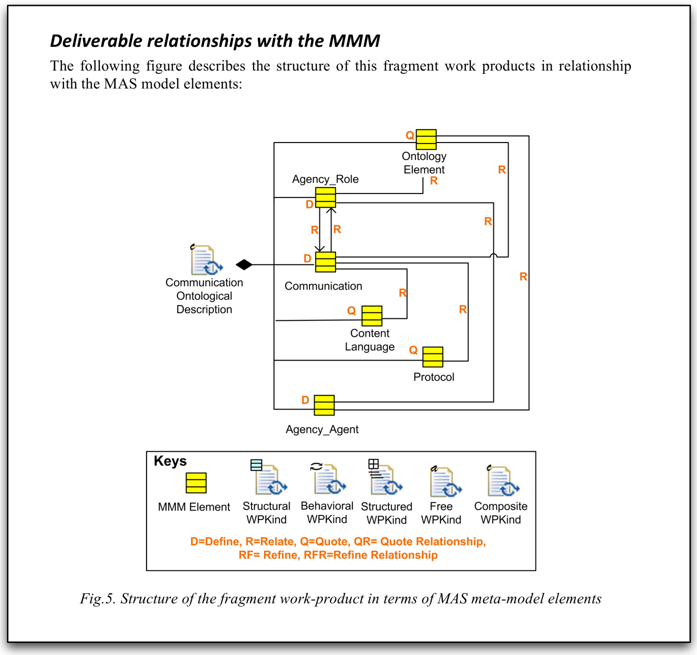
This fragment refers to the MAS meta-model adopted in PASSI and contributes to define and describe the elements reported in .

Glossary

Communication – An interaction among two agents, referring an Agent Interaction Protocol and a piece of the domain ontology (knowledge exchanged during the interaction). Usually it is composed of several messages, each one associated with one communicative act (or performative).

Fig. 5. A Portion of the COD Fragment Document - The Fragment MAS Metamodel

description, and documentation, is principally aimed at showing the process and product part of the fragment for easily identifying the way it can be reused. A section of the document is particularly important to this aim: the one devoted to deeply show the relationships among the product result (the work product) and all the elements the designer has to consider for producing it. In Figure 6 the COD fragment result, the *Communication Ontological Description* work product is reported with its relations to the previous shown metamodel elements, each of them tagged with a red letter indicating the kind of action the designer makes on them. **D** stands for *Define* and indicates the element is defined (instantiated) in the fragment workflow; during the design it might be requested to *Quote* - **Q** other elements, to define relationships among some of them (**R** - *Relate*) or to *Refine* (**RF**) some others. For instance in order to define the *Agency_Role* the designer has to quote *ontology elements* that is to be reported from another portion of the process, hence another fragment, and has to relate that. It is worth noting that quoting one element establishes a dependence of this fragment with another one and all the quote relationships constitutes paths for assembling.



Deliverable relationships with the MMM

The following figure describes the structure of this fragment work products in relationship with the MAS model elements:

Fig.5. Structure of the fragment work-product in terms of MAS meta-model elements

Fig. 6. A Portion of the COD Fragment Document - The Relationships among MAS Metamodel Elements and WP

The fragment document continues with an example and the explanation on how to produce the work product (see Figure 7) and with a set of composition guidelines and dependency relationships. Another important part of the document regards the definition of the process fragment preconditions and concepts to be defined both in form of the MAS metamodel elements and the workproducts depicting them (see Figure 8). These latter represent a pretty weak constraint since the input data (MAS metamodel elements) in the new assembled process may be made available by different documents thus obtaining the same result.

4 Discussions and Conclusions

In this paper we presented a proposal of fragment process definition. The fragment is based on the three classical elements accepted by most SME approaches:

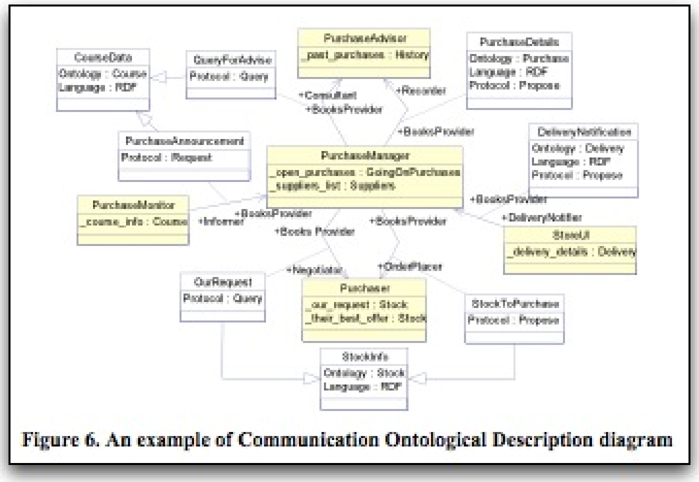
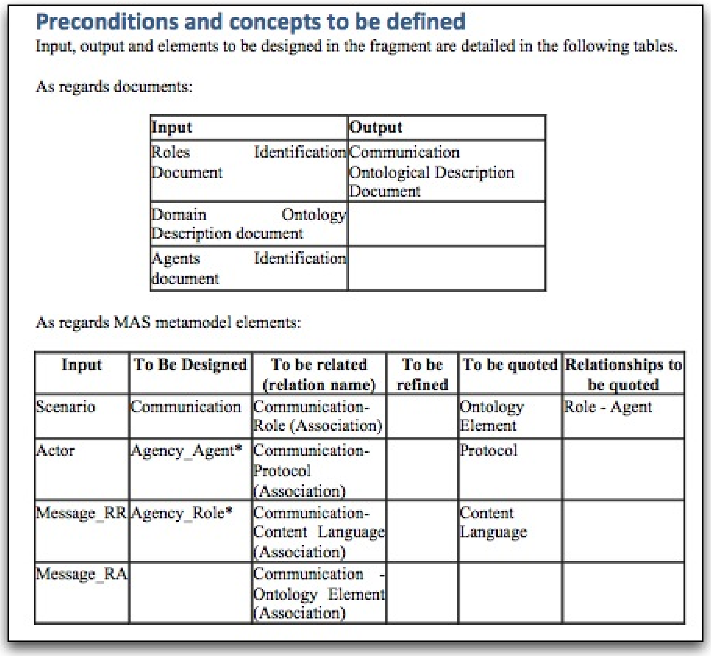


Figure 6. An example of Communication Ontological Description diagram

Fig. 7. A Portion of the COD Fragment Document - An Example of the Produced WP



Preconditions and concepts to be defined

Input, output and elements to be designed in the fragment are detailed in the following tables.

As regards documents:

Input	Output
Roles Identification Document	Communication Ontological Description Document
Domain Ontology Description document	
Agents Identification document	

As regards MAS metamodel elements:

Input	To Be Designed	To be related (relation name)	To be refined	To be quoted	Relationships to be quoted
Scenario	Communication	Communication-Role (Association)		Ontology Element	Role - Agent
Actor	Agency_Agent*	Communication-Protocol (Association)		Protocol	
Message_RR	Agency_Role*	Communication-Content Language (Association)		Content Language	
Message_RA		Communication - Ontology Element (Association)			

Fig. 8. A Portion of the COD Fragment Document The Fragment Preconditions and Concepts to be Defined

activity, deliverables and stakeholders. Moreover, it gives a special importance to the system metamodel seen as an essential choice towards the system requirements satisfaction. Afterwards, a documentation template for the fragment is proposed. We already adopted it in documenting large part of the PASSI fragments with good results and we are confident it can be general enough to support fragments from many other sources. The template is discussed with the support of an example taken from the PASSI process and describing the PASSI COD design activity.

Acknowledgment

This research has been partially supported by the EU project FP7-Humanobs and by the Italian Ministry for Education University and Research (MIUR) FRASI project.

References

1. Brinkkemper, S., Welke, R., Lyytinen, K.: Method Engineering: Principles of Method Construction and Tool Support. Springer (1996)
2. Cossentino, M.: From requirements to code with the PASSI methodology. In: Agent Oriented Methodologies, chap. IV, pp. 79–106. Idea Group Publishing, Hershey, PA, USA (Jun 2005), http://www.idea-group.com/books/details.asp?id=4931
3. Cossentino, M., Gaglio, S., Garro, A., Seidita, V.: Method fragments for agent design methodologies: from standardisation to research. International Journal of Agent-Oriented Software Engineering (IJAOSE) 1(1), 91–121 (2007)
4. Cossentino, M., Sabatucci, L., Seidita, V., Gaglio, S.: A collaborative tool for designing and enacting design processes. In: SAC '09: Proceedings of the 2009 ACM symposium on Applied Computing. pp. 715–721. ACM, New York, NY, USA (2009)
5. DPDF-WG: http://www.fipa.org/subgroups/dpdf-wg.html
6. Gupta, D., Prakash, N.: Engineering Methods from Method Requirements Specifications. Requirements Engineering 6(3), 135–160 (2001)
7. Henderson-Sellers, B.: Method engineering: Theory and practice. In: Karagiannis, D., Mayr, H. C., e. (eds.) Information Systems Technology and its Applications. pp. 13–23 (2006)
8. Jacobson, I., Booch, G., Rumbaugh, J.: The unified software development process. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA (1999)
9. Mirbel, I., Ralyté, J.: Situational method engineering: combining assembly-based and roadmap-driven approaches. Requirements Engineering 11(1), 58–78 (2006)
10. Osterweil, L.: Software processes are software too. In: Proceedings of the 9th international conference on Software Engineering. pp. 2–13 (1987)
11. Ralyté, J.: 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 pp. 271–282 (2004)
12. Saeki, M.: Software specification and design methods and method engineering. International Journal of Software Engineering and Knowledge Engineering (1994)
13. Schmidt, D.C.: Model-driven engineering. Computer 39(2), 25–31 (Feb 2006)
14. Seidita, V., Cossentino, M., Gaglio, S.: A repository of fragments for agent systems design. Proc. Of the Workshop on Objects and Agents (WOA06) (2006)

15. Seidita, V., Cossentino, M., Gaglio, S.: Using and Extending the SPEM Specifications to Represent Agent Oriented Methodologies. In: Agent-Oriented Software Engineering IX. pp. 46–59. Springer-Verlag (2009)
16. Seidita, V., Cossentino, M., Hilaire, V., Gaud, N., Galland, S., Koukam, A., Gaglio, S.: The metamodel: a starting point for design processes construction. International Journal of Software Engineering and Knowledge Engineering. (in printing). (2009)