Towards a New Approach for MAS Situational Method Engineering: a Fragment Definition

Sara Casare[1], Zahia Guessoum [2], Anarosa A. F. Brandão[1], Jaime Sichman[1]
[1] Intelligent Techniques Laboratory – University of São Paulo - Brazil
{sara.casare, anarosa.brandao, jaime.sichman}@poli.usp.br
[2] Laboratoire d'Informatique de Paris 6 - LIP6 – University Pierre et Marie Curie - France
zahia.guessoum@lip6.fr

Abstract. This paper introduces a new definition of method fragment intended to represent MAS development approaches in a more standardized and coherent way, thus facilitating the configuration of situational methods. In order to do that, we take into account three complementary notions: (i) a method fragment *description* based on SPEM 2.0 elements; (ii) two method fragment *perspectives*, the internal and the external view, and (iii) four method fragment granularity layers. Moreover, this definition establishes some mechanisms for method fragments' encapsulation and identification. The proposed method fragment definition is illustrated through an example using Tropos.

Keywords: multiagent oriented software engineering, situational method engineering, method fragment, SPEM

1 Introduction

In order to structure the development and to manage the complexity associated with Multiagent Systems (MAS), several development methods have been proposed during the last decade, e.g. Gaia [18], Tropos [2], PASSI [6], and Adelfe [1]. The variety of Agent Oriented Software Engineering (AOSE) methods is due to the specific needs raised on MAS development and to the different approaches adopted by MAS developers. It shows that a method cannot be general enough in order to be applied to every MAS development project without some level of customization [11]. Moreover this customization requires deep knowledge on both the method and the MAS research field. Nevertheless, it seems that reinventing a new method for each new project situation wouldn't be a best practice, given that there are a great number of available methods for MAS development. This scenario suggests that Method Engineering techniques and, particularly, Situational Method Engineering [3] seems to be promising approaches to be considered for MAS development.

Situational Method Engineering is the sub-area of Method Engineering that addresses the controlled, formal and computer-assisted construction of situational methods out of method fragments. Roughly speaking, building a situational method consists of reusing parts of existing methods taking into account a given project situation that encompasses, for example, notions related to the class of the desired application (like traditional and pervasive computing) and project perspectives.

Several approaches concerning the notions of a *part of a method* and *situational method building* have been proposed in the Situational Method Engineering field. For instance, Brinkkemper and colleagues [3][4] introduce a Method Fragment notion, and Karlsson [14] introduces a Method Component notion. Method Fragments [3][4] are standardized building blocks based on a coherent part of a method that can reside on one of five layers of granularity: method, stage, model, diagram or concept. The notion of coherence should be interpreted while considering a method as a connected graph of products or processes. For instance, an entire process can be considered as a method fragment. A situational method can be built by combining a number of method fragments in a bottom-up fashion. Such a combination must follow certain assembly rules in order to adhere to the construction principles into the process perspective and the product perspective.

A Method Component [14] consists of an exchangeable and reusable part of method composed of descriptions for actions, notations, artifacts and concepts that can be viewed into two perspectives: an internal view and an external view. While the internal view presents all method component elements (as action, artifacts, and roles), the external view aims to describe method component output in order to identify how it contributes to a chain of goal achievements. On the one hand, this approach emphasizes principles as method modularization and method reusability. On the other hand, it proposes a way for using these principles in order to define a procedure for method configuration involving the notion of Base Method: a method chosen as starting point for the configuration process, allowing a top-down fashion to create situational methods, eliminating, adding or exchanging additional fragments captured from another method.

This paper proposes a definition of method fragment that combines these two notions of part of a method. This definition allows representing MAS development approaches in a more standardized and coherent way. Moreover, it establishes mechanisms for method fragments encapsulation and identification in order to provide a solid base for developing / building MAS situational methods. The paper is organized in five sections. Section 2 presents the proposed definition for MAS method fragment, while Section 3 shows an application of such definition to Tropos. Section 4 presents an overview of the current research concerning situational method engineering applied to MAS field. Finally, Section 5 presents a discussion about the proposed approach for MAS method fragment definition.

2 A New Definition for MAS Method Fragment

The MAS Method Fragment definition proposed in this paper has been mainly inspired on the Method Fragment notion proposed by Brinkkemper and colleagues [3][4] as well as on the notions of method component view and Base Method proposed by Karlsson [14]. From Brinkkemper and colleagues we adopted the simple and intuitive idea of *part of a method* and from Karlsson we adopted the black box perspective of *part of a method* offered by the Method Component view, as well as the Base Method notion to provide a solid foundation for top-down situational method configuration. Additionally, we have adopted some concepts of Software Engineering proposed by Jacobson and colleagues [13] and have used SPEM 2.0 (Software and

Systems Process Engineering Metamodel) [15] as a common meta-model for describing method fragment. The former is among the most popular software development processes and the latter is the standard "de facto" to model development process.

Our proposed definition is:

"*A MAS Method Fragment is a **standardized building block** based on a **coherent part** of a **MAS development approach**".*

The standardization of building blocks considers the notions of (i) identification of method fragments using well established naming rules in order to convey their desired semantics; (ii) encapsulation of original work products; (iii) utilization of common roles for MAS developers to be used as task performers; and (iv) classification of method fragment based on a semiotic criteria [5]. The coherence of method fragments is assured by the notions of (i) a *method fragment description* based on the SPEM 2.0 elements (task, work product, role, activity and so on) and their associations; (ii) the proposition of *two method fragment views* (internal and external views); and (iii) the use of four *method fragment granularity layers* (activity, phase, iteration, process).

In the following subsections, we will describe the main characteristics of the proposed definition for coherence.

2.1 Standardizing Building Blocks

In order to have a standardized and common semantics for specifying method fragment objectives and work products, we have defined a MAS Work Product Framework mainly based on the MAS components proposed in the Vowel approach [9] - **A**gent, **E**nvironment, **I**nteraction, **O**rganization. This approach offers a natural and coherent way for describing MAS components and has been adopted in several MAS research [16] with successful results. Nevertheless, it does not deal with the notion of users requirements that should be gathered before specifying MAS components. Then, the proposed MAS Work Product Framework involves also an element related to MAS **U**ser Requirement, in order to encapsulate work products used to describe the system-to-be requirements.

Such a work product framework is used to encapsulate original MAS development work products, explicitly stating their involvement with user requirements or to the main MAS components. This approach allows enhancing work product flow into a situational method and making clear the main goal of each work product independently of their name in the context of the original MAS development approach. Finally, the method fragment characteristics are specified through the MAS Semiotic Taxonomy [5] that provides a set of semiotic criteria to categorize MAS Method Fragments taking into account their meaning, usage, structure and so on.

2.2 MAS Method Fragment Main Elements

The proposed MAS Method Fragment description is based on SPEM 2.0 elements and related associations. To improve readability we use Arial font to concepts proposed in

this work and Comic Sans font to describe SPEM elements. Therefore, the main elements used to compose a MAS method fragment description are: Process Pattern, Activity, Phase, Milestone, Iteration, Task Definition, Task Use, Step, Role Definition, Role Use, Work Product Definition, Work Product Use, Category, and Guidance. As proposed by SPEM 2.0, these elements are separated into method content elements (Task Definition, Step, Role Definition, Work Product Definition) and their application in the development process (Process Pattern, Activity, Phase, Milestone, Iteration, Task Use, Role Use, Work Product Use).

A Process Pattern represents building blocks for assembling processes. It describes a reusable cluster of Activities that provides a consistent development approach to common problems. An Activity represents a general unit of work assignable to specific roles, relying on input work products and producing output work products. We have chosen this as the main element of MAS Method Fragment in the Activity Layer. A Phase consists of a significant period in a project, ending with major management checkpoint, as a Milestone that represents a significant event for a development project. We have used Phase and Milestone as main elements of MAS Method Fragment in the Phase Layer. Moreover, Milestone is used to define fragments in the Process Layer. An Iteration is a set of nested Activities that are repeated more than once, allowing the organization of work in repetitive cycles. It has been used to define MAS Method Fragments in the Iteration Layer.

A Task Definition represents an assignable unit of work involving generally a few hours to a few days and usually affecting one or only a small number of work products. A Step describes a meaningful and consistent part of the overall work described for a Task Definition. A Task Use represents a proxy for a Task Definition in the context of one specific Activity. Tasks and Steps constitute the main element of the Activities used to build MAS Method Fragments.

A Role Definition describes a set of related skills, competencies, and responsibilities of an individual or a set of individuals. A Role Use represents a Role Definition in the context of one specific Activity. Roles represent both the development roles originally specified by the AOSE methods and the common MAS role set involved in the MAS Method Fragments definition. Work Product Definition represents pieces of work that are used, modified, and produced by Task Definitions, while a Work Product Use represents a Work Product Definition in the context of a specific Activity. Work Product Definitions represent the artifacts proposed by the MAS development approaches, as models, specifications and diagrams.

A Category represents the classification structure used to group SPEM elements based on the user's criteria. It allows defining tree-structures of nested categories used for browsing MAS Method Fragments based on a semiotic criteria [5]. A Guidance represents a specific description related to other SPEM elements. It can be a formal description, such as concepts description, or informal description such as guidelines, white papers, checklists, examples or roadmaps. Guidance represents some elements proposed by the MAS development approaches, such as work product examples, main references, concepts and tool mentor.

The aforementioned elements will be depicted in conjunction of the proposed method fragment views and some standardization notions in **Fig. 1**. Finally, we have applied an important notion proposed by SPEM - called Variability Elements - to improve tasks and work products defined according to the original MAS development approaches. It allows reaching the completeness and standardization involved in a MAS Method Fragment definition without modifying the original method elements. For example, we can use a Task Variability to complete a task definition introducing performing role and a Work Product Variability to define composite work products, as Agent Model or Requirement Model.

2.3 Method Fragment Views

The *Internal* and *External* views of MAS Method Fragments depict, respectively, the whole set of elements that compose them and the main elements that constitute their interface. They have been inspired on the method component views concept proposed by Karlsson [14]. The Internal View offers a detailed and deep representation of a method fragment that allows analyzing all elements involved in its composition, such as activities, roles, tasks, guidance, categories, work products and milestone, as well as their relationships (see **Fig. 1** for details).

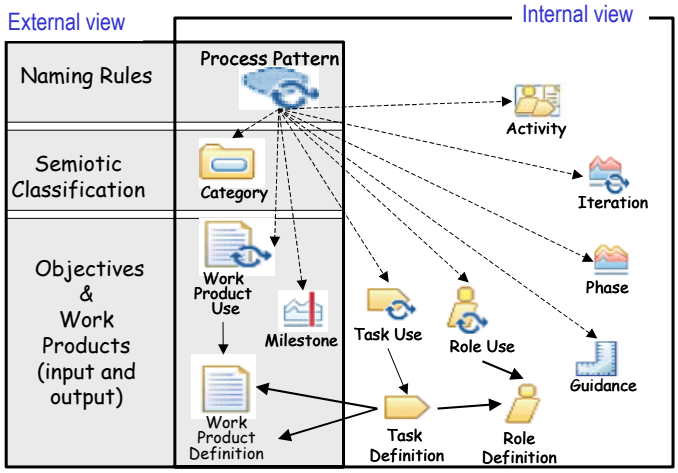

Fig. 1 Main elements of MAS Method Fragment

The External View goals are twofold. First, it describes, in a standard way how a method fragment can be used in a situational method configuration by: (i) identifying the method fragment through naming rules, (ii) specifying method fragment objectives in terms of milestones and/or work products, and (iii) describing fragment characteristics as meaning, usage, structure and so on. Second, it describes how a method fragment can contribute to achieve a situational method objective, specifying fragment output work products through a common semantic. The naming

rules adopted to identify method fragments are based on simple concepts and will be explained through the method fragments examples in the next section.

These two method fragment perspectives improve method fragment coherence because it allows analyzing method fragments as standard black boxes without loosing the details of the description. **Fig. 1** shows the views and associated elements in a diagrammatic perspective. For instance, it depicts Process Pattern as a kind of "logical container" for building MAS Method Fragments, Category to define part of the External View of MAS Method Fragment, called Method Fragment Semiotic Classification and the use of Milestone to represent MAS Method Fragment expected objectives.

2.4 Method Fragment Layers

The four layers of MAS Method Fragment – activity, iteration, phase and process - have been defined according to Jacobson et al [13] and SPEM homonym concepts.

The definition of a MAS method fragment in the Activity Layer, for short an Activity Method Fragment, is based on the notion of Activity proposed by Jacobson et al [13]. An Activity Method Fragment consists of a tangible unit of work performed by a worker that yields a well-defined result based on a input set of artifacts. The unit of work has defined boundaries that are likely to be referred in a project plan when tasks are assigned to individuals.

Fig. 2 depicts the components of an Activity Method Fragment. It is worthy to notice that this figure represents only the main relationships between SPEM elements used to define an Activity Method Fragment. For instance, relationships between Category and Roles, Tasks and Work Products are not depicted. Moreover, we have labeled the relationship arrows with cardinalities that constitute a constraint over SPEM elements definition. For example, in general a Process Pattern can be associate to zero or many Activities, while the Process Pattern used in the context of an Activity Method Fragment must be associated with exactly one Activity.

An Activity Method Fragment is composed of one Process Pattern associated with one Activity and one or more Categories. Such Activity must be associated with at least one Task Use that must produce one or more Work Products as output. However, in the context of an Activity Method Fragment an Activity must not be associated with an Iteration nor to a Phase or Milestone elements, given that these elements are used to define other layers of method fragment. As we can see in Fig. 2, Category and Work Product elements constitute the external view of the Activity Method Fragment. In summary, an Activity Method Fragment must contain one Activity that is composed by at least one Task, one Role and one output Work Product. Moreover, it must be classified by Categories.

The definition of a MAS method fragment in the Phase layer, for short a Phase Method Fragment, is based on the notion of phase proposed by Jacobson et al [13]. These authors consider that a phase should be concluded with a major milestone. Moreover, they state that any software process needs to have a sequence of clearly articulated milestone in order to be effective. Therefore, a Phase Method Fragment represents a significant period in a project and consists of a Process Pattern

classified by Categories, associated with exactly one Phase, one (major) Milestone and several Activity Method Fragments and/or Iteration Method Fragments.

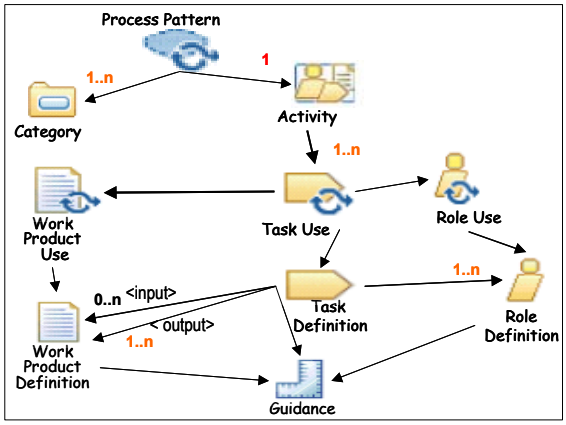

Fig. 2: Activity Method Fragment representation as UML Class Diagram

The definition of a MAS method fragment in the Iteration layer, for short an Iteration Method Fragment, is based in the SPEM homonym element. An Iteration Method Fragment consists of a Process Pattern that involves a set of Activity Method Fragments and/or a set of Phase Method Fragments that are repeated more than once during the process development lifecycle, offering a structuring fragment to organize work in repetitive cycles. Moreover, it must be classified by Categories.

Finally, a MAS method fragment in the Process layer, for short a Process Method Fragment, represents a whole MAS development cycle. It is composed of a Process Pattern classified by Categories that contains several Phase Method Fragments and/or Iteration Method Fragments and ends with a (major) Milestone. The goal of a Process Method Fragment is twofold. First, it depicts the notion of Base Method presented in Section 2, allowing a top-down fashion to configure a MAS situational Method. Second, it allows describing MAS original methods as a MAS Method Fragment ready to be used when an existing MAS method totally matches a given project situation.

The main advantages of having these four method fragment layers are: (i) the reuse of original AOSE methods in several level of granularity; (ii) the representation of MAS development approaches that do not provide a full development method (such as method fragments related to Agent Organizations models), and (iii) the utilization of bottom-up or top-down mechanisms for situational method configuration.

3 Applying the Proposed Definition to Tropos

In this section, we use the proposed definition to describe MAS method fragments sourced from Tropos. Tropos proposes a process for building MAS involving the following phases: *Early Requirements, Late Requirements, Architectural Design*, *Detailed Design* and *Implementation*. The goal of the first two phases is to provide a set of functional requirements as well as non-functional requirements for the system

to be built, while *Architectural Design* and *Detailed Design* phases focus on the system specification. Finally, the *Implementation* phase transforms the results of the preceding phases using an agent development platform in order to code the MAS.

We have used the Eclipse Process Framework Composer (EPF Composer) [12] to represent the MAS Method Fragments extracted from Tropos. EPF Composer is a tool developed by Eclipse Foundation that fully implements SPEM 2.0. After using the adequate SPEM element to represent each activity, step, role, diagram and model proposed by Tropos we have defined MAS Method Fragments into the Activity Layer, Phase Layer and Process Layer. We have not defined fragments into the Iteration Layer because Tropos does not propose iteration development cycles. Due to space constraints, we will describe only one example of fragment for each layer.

Fig. 3 depicts the External View of the Process Method Fragment called MMF Tropos Base Method, represented as a Process Pattern (called Capability Pattern in EPF Composer). On the left frame we can see that this fragment is classified in several categories of the MAS Semiotic Taxonomy, e.g. in the social level and iteration degree, it is classified as part of the Low Iteration Fragment Category.

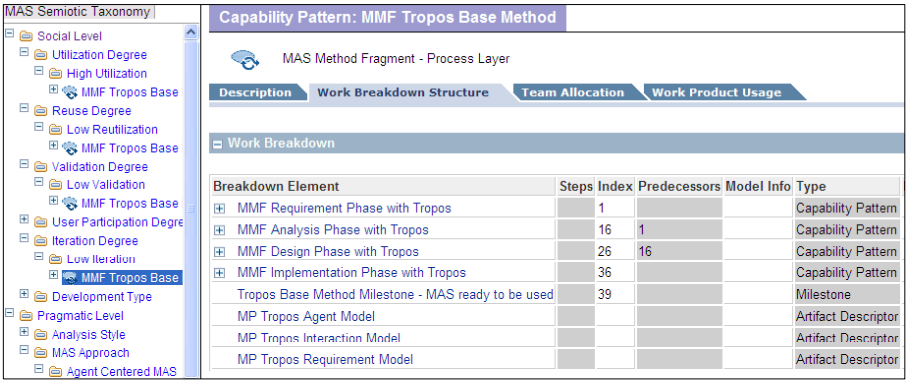

Fig. 3 External view of the MAS Method Fragment Tropos Base Method

On the right frame we can see that this fragment is composed of four Phase Method Fragments - MMF Requirement Phase with Tropos, MMF Analysis Phase with Tropos, MMF Design Phase with Tropos and MMF Implementation Phase with Tropos. As said before, we propose applying standard naming rules for method fragment identification. For instance, phase names should convey either the software development discipline covered by the phase - as *Requirement, Analysis, Design, Implementation, Test* - or the phase main development goal - as *Inception, Elaboration, Construction, Transition* – as currently used in iterative development approaches [13].

Moreover, the MMF Tropos Base Method fragment is composed of a Milestone called Tropos Base Method Milestone - MAS ready to be used that involves three work products: MP Tropos Agent Model, MP Tropos Interaction Model and MP Tropos Requirement Model, where MP stands for **M**AS work **P**roduct. Such work products encapsulate Tropos original work products and provide a standardized and common semantic for describing MAS work products based on the Vowels approach, as

proposed in Section 2. For instance, the work product MP Tropos Requirement Model encapsulates the *Tropos Actor Diagram* and *Tropos Goal Diagram,* both related to Tropos requirement phase.

As we have seen in Section 2, given that MMF Tropos Base Method is a Process Layer Fragment it can be used into two distinct ways: as standard representation of Tropos or as a Base Method in a top-down configuration mechanism in order to build a MAS situational Method.

Fig. 4 depicts the External View of the method fragment called MMF Requirement Phase with Tropos as well as some elements of its Internal View (highlighted rectangle of the right frame). On the left we can see the fragment classification using the MAS Semiotic Taxonomy. On the right we can see that this fragment is composed of three Activity Method Fragments: MMF Identify Initial Requirement with Tropos, MMF Detail Requirement with Tropos, and MMF Identify Additional Requirement with Tropos. Moreover, it contains the MAS Objectives Described Milestone that involves the work product MP Tropos Requirement Model.

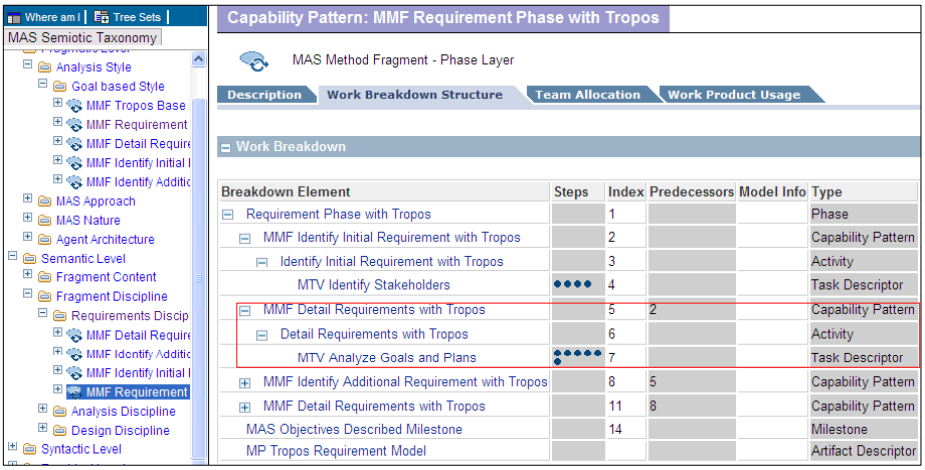

Fig. 4 Phase MAS Method Fragment - Requirement Phase with Tropos

The Internal View of the MMF Detail Requirement with Tropos involves an homonym *Activity* that contains a *Task Use* (Task Description in EPF Composer) called MTV Analyze Goals and Plans. The acronym MTV used as task name prefix stands for **M**AS **T**ask **V**ariability and indicates that we are dealing with a task that extends another one. Such task variability has been defined over the Tropos original task in charge of analyzing goals and plans in order to provide basic MAS roles. In addition, it also changes Tropos original work products by the related MAS work products that forms the External Fragment View (see **Fig. 5** for more details).

It is worthy to notice that the fragment MMF Detail Requirement with Tropos is executed twice in the MMF Requirement Phase with Tropos: first after identifying initial requirements (concerning stakeholders identification) and latter after identifying additional requirements (defining system actors).

Finally, **Fig. 5** complements the Internal View of the fragment MMF Detail Requirement with Tropos showing its main elements.

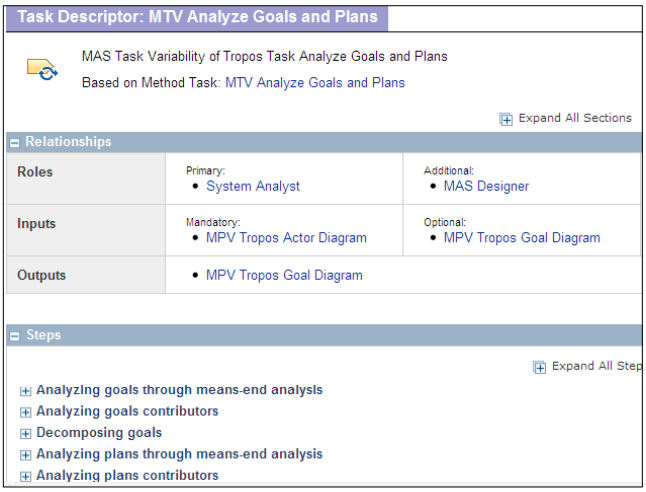

Fig. 5 Internal View of the fragment MMF Detail Requirement with Tropos

As the fragment MMF Detail Requirement with Tropos consists of an Activity Method Fragment, its elements are defined through the task MTV Analyze Goals and Plans. These are the following: (i) the roles System Analyst and MAS Designer, (ii) the mandatory input work product MPV Tropos Actor Diagram, (iii) the output work product MPV Tropos Goal Diagram, and (iv) several steps, as Analyzing goals through means-end analysis and decomposing goals.

These examples of usage show that the proposed definition of MAS Method Fragment offers a more standardized way for representing each phase and activity of Tropos, as well as Tropos own method as a whole. Moreover, it establishes mechanisms for the encapsulation and identification of original Tropos tasks, roles and work products. Finally, it allows reusing method fragments even in the context of its original method.

4. Related Work

In AOSE field there are several researches concerning method fragment notion. Among them we can cite the FIPA method fragment definition [10] and its refinement proposed by Cossentino et al. [7], as well as the fragment description for adaptive methodology proposed by Rougemaille et al. [17] and the three method fragment levels of granularity introduced by [8]. In general, such researches propose the use of SPEM as a common meta-model for representing MAS method fragment. For instance, Rougemaille et al. [17] highlight how SPEM can participate to design adaptive methodology process and claim that being compliant with SPEM is important to broaden the use of agent-oriented methodologies and principles. Nevertheless, the method fragment levels of granularity (atomic, composed, and phase level) proposed in [8] do not involve any metamodel. Instead, these method fragment levels are only briefly outlined in natural language.

The preliminary version of the FIPA method fragment definition [10], published in 2003 by the FIPA Methodology Technical Committee (TC)[1], states that a method fragment is a portion of the development process that involves the following elements: (i) a definition of a portion of process using SPEM describing what should be done; (ii) one or more deliverables; (iii) a required input data representing the preconditions to start the process specified in the fragment; (iv) a list of concepts related to the MAS meta-model to be defined / refined for the fragment; (v) guidelines that illustrate how to apply the fragment as well as best practices related to that; (vi) a glossary of terms used in the fragment; (vii) composition guidelines describing the context/ problem treated by the fragment; (viii) aspects of fragment such as the platform to be used, and finally (ix) the dependency relationships useful to assemble fragments. It is worthy noting that some of these elements are not mandatory, as input data and guidelines.

Our MAS method fragment definition is based on SPEM, as shown in Section 2, and it is fully compliant with this FIPA definition since the portion of process is represented by a `Task` element while `Work Product` elements describe fragment deliverables as well as fragment inputs. Moreover, we use `Guidance` elements to describe MAS concepts, to provide fragment guidelines and glossary of terms. Finally, our method fragment definition encompasses an external view that aims to describe the aspects that should be taken into account during method fragment selection and assembling, applying `Category` elements to classify the fragments based on their context use, their deliverable work products and their development platforms, among other criteria covered by the MAS Semiotic Taxonomy.

The refinement of FIPA definition proposed by Cossentino et al. [7] involves four different point of views for describing method fragments: (i) the process fragment view that deals with the process related aspects of a fragment, including workflows, activities and work products; (ii) the reuse fragment view for representing fragment elements such as the MAS meta-model, glossary of terms, guidelines and fragment dependency; (iii) the fragment storing view that deal with retrieving method fragments from the method base and, finally, (iv) the implementation view that concerns the implementation aspects of the process fragment view elements.

This refinement is distinct of our MAS method fragment definition in two main points: the SPEM compliance and the representation of MAS components. First, our approach is fully defined over SPEM (since we do not introduce new elements neither new associations in order to define method fragment elements) while Cossentino et al. [7] introduce new elements such as the Workflow and the MAS Model elements. In our opinion the SPEM compliance offer important benefits: on the one hand, SPEM is the "de facto" standard for method metamodel and, on the other hand, such compliance allows using SPEM based tools, as EPF Composer, for building the Method Base and configuring the MAS Situational Method. Second, this refinement requires dealing with MAS metamodel elements in a fine grained granularity and involves MAS metamodel integration while we propose representing MAS components in a coarse grained granularity based on the Vowel approach (Agent, Environment, Interaction, Organization). Therefore, our definition of method

[1] The activity of this TC stopped during the transition towards the new FIPA structure as part of IEEE Computer Society Standards Committee since 2005

fragment does not depend on a previous MAS metamodel integration in order to configure MAS situational methods. Such independence represents an important benefit since it allows taking advantage of Situational Method Engineering techniques for creating MAS situational methods without waiting for MAS community reaching a consensus about MAS main concepts.

Summing up, at the best of our knowledge, in the AOSE field there is no explicit method fragment definition tailored to facilitate the use of situational configuration mechanisms and to offer a standardized description of a method fragment elements and objectives. The IEEE-FIPA Design Process Documentation and Fragmentation Working Group[2] (DPDF WP), recently formed to deal with a definition of method fragment for situational method engineering process, corroborates that these topics constitute open issues in AOSE field.

5 Conclusions

In this paper, we presented a new definition for MAS method fragment based on Situational Method Engineering and Software Engineering techniques that can be used to represent MAS development approaches in a more standardized and coherent way. It offers two method fragment perspectives - internal and external fragment view - and four method fragment granularity layers: activity, iteration, phase and process. According to the proposed definition, a method fragment is considered coherent when its internal view is described using SPEM 2.0 and their elements and associations follow one of these four method fragment layers definition.

Moreover, we show how this definition can be used to represent method fragments extracted from Tropos, that is a well known MAS method. However, we think that it is not possible to identify a univocal criteria to set the best level of granularity for extracting method fragments, since fragment creation depends more on the experience of the method engineers than on a rigid extraction criteria. We claim that the absence of such a univocal criteria could be mitigated in two ways. First, using a common model, like SPEM, to represent method fragment. Second, establishing a procedure for extracting and storing fragments in a method repository. Such procedure is part of our research future work.

It is worthy to notice that part of the proposed method fragment definition is general enough to be applied to classical software development methods and not only to MAS methods. In fact, the MAS methods distinctive aspects are taken into account mainly through the following notions of our definition: the MAS Work Product Framework based on Vowel approach and some semiotic criteria, as MAS approach (agent /organization centered) and MAS nature (open/closed MAS). Therefore, in general lines, the proposed method fragment definition could be used to represent fragments sourced from classical software development methods after replacing these notions for other more suitable to a giving development paradigm.

We claim that this MAS Method Fragment definition can provide the backbone for defining a method repository for MAS situational method configuration, given that it provides mechanisms for (i) identifying method fragments using a well established

[2] http://www.fipa.org/subgroups/DPDF-WG.html

naming rules, (ii) conveying method fragment objectives and encapsulating original work products using a common work product framework, and (iii) categorizing method fragment based on a semiotic criteria. Finally, given that we propose distinct method fragment granularity layers, we can build a method repository involving fragments extracted from AOSE methods as well as from other MAS development approaches, such as Agent Organization models. In a top-down situational method configuration fashion, the former could provide a Base Method for the configuration while the latter could contribute with specific Activity Method Fragments in order to improve and complete a original MAS method.

Acknowledgments. This work is a product of the MEDEIA project, supported by FAPESP (Brazil) and CNRS (France). The first and the last two authors are partially supported by CNPq and CAPES (Brazil).

References

[1] Bernon, C., Camps, V., Gleizes, M.-P., Picard, G.: Engineering Adaptive Multi-Agent Systems: The ADELFE Methodology. In: Henderson-Sellers, B., Giorgini, P. (eds) Agent-Oriented Methodologies, Idea Group Pub. USA, 171—202 (2005)

[2] Bresciani, P.; Giorgini, P.; Giunchiglia, F.; Mylopoulos, J.; Perini, A.: Tropos: An Agent-Oriented Software Development Methodology. Journal of Autonomous Agents and Multi-Agent Systems, vol 8(3), 203--236 (2004)

[3] Brinkkemper, S.: Method Engineering: Engineering of Information Systems Development Methods and Tools. Information and Software Technology, vol. 38 (4), 275--280 (1996)

[4] Brinkkemper S.; Saeki, M.; Harmsen, F. Meta-Modelling Based Assembly Techniques for Situational Method Engineering, *Information Systems,* 24(3), 209--228 (1999)

[5] Casare, S.; Brandão, A. A. F.; Sichman, J. S. A Semiotic Perspective for Multiagent Systems Development (Extended Abstract), Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010), van der Hoek, Kaminka, Lespérance, Luck and Sen (eds.), May, 10–14, 2010, Toronto, Canada, 1373-1374 (2010)

[6] Cossentino, M.: From Requirements to Code with the PASSI Methodology. In: Henderson-Sellers, B., Giorgini, P. (Eds.), Agent-Oriented Methodologies, Idea Group Publishing, 79--106 (2005)

[7] Cossentino, M.; Gaglio, S.; Garro, A. ; Seidita, V. Method Fragments for agent design methodologies: from standardization to research. In: International Journal on Agent Oriented Software Engineering (IJAOSE). 1(1) (2007)

[8] Cossentino,M.; Fortino, G.; Garro, A.; Mascillaro, S. ; Russo. W..PASSIM: a simulation-based process for the development of multi-agent systems. Int. Journal of Agent-Oriented Software Engineering, 2(2):132:170, Inderscience Enterprises Ltd., UK (2008)

[9] Demazeau, Y. From interactions to collective behavior in agent-based systems. Proc. of the 1st. European Conference on Cognitive Science. Saint-Malo, 117—132 (1995)

[10] FIPA. Foundation for Intelligent Physical Agents, Methodology TC. Method Fragment Definition, Preliminar version 2003/11/21 (2003) Available on <http://www.pa.icar.cnr.it/cossentino/FIPAmeth)>.

[11] Guessoum, Z; Cossentino, M.; Pavón, J.: Roadmap of Agent-Oriented Software Engineering – The AgentLink Perspective. In: F. Bergenti, M. P. Gleizes, & F. Zambonelli

(Eds.), Methodologies and software engineering for agent systems, Kluwer Academic Publishers, 431--450 (2004)

[12] Haumer, P. Eclipse Process Framework Composer – Part 1 – Key Concepts. (2007) Available on: <http://www.eclipse.org/epf>.

[13] Jacobson, I.; Booch, G.; Rumbaugh, J. The unified software development process. Addison-Wesley (1999)

[14] Karlsson, F.: Method Configuration - Method and Computerized Tool support. Doctoral Dissertation Dept. of Computer and Information Science, Linkoping University (2005)

[15] OMG. Object Management Group. Software & Systems Process Engineering Meta-Model Specification, version 2.0. OMG document number: formal/2008-04-01 (2008) Available on http://www.omg.org/spec/SPEM/2.0/PDF.

[16] Pavon, J. Ingenias: Développement Dirigé par Modèles des Systèmes Multi-Agents. Dossier d'Habilitation à Diriger des Recherches de l'Université Pierre et Marie Curie. Paris, France (2006)

[17] Rougemaille S.; Migeon, F.; Millan, T.; Gleizes, M.-P.: Methodology Fragments Definition in SPEM for Designing Adaptive Methodology: A First Step. In: Luck, M.; Gomez-Sanz, J.J. (Eds**.):** AOSE 2008**,** LNCS, vol. 5386, Springer, 74–-85 (2009)

[18] Zambonelli, F., Jenningns, N. R.; Wooldridge, M.: Developing multiagent systems: The Gaia methodology. ACM Transaction on Software Engineering and Methodology, vol 12(3), 417--470 (2003)