Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Activity of the FIPA Methodology Technical Committee

M. Cossentino, Alfredo Garro

Rapporto Tecnico N.: 15
RT-ICAR-PA-05-15 **dicembre 2005**

Consiglio Nazionale delle Ricerche, Istituto di Calcolo e Reti ad Alte Prestazioni (ICAR)
– Sede di Cosenza, Via P. Bucci 41C, 87036 Rende, Italy, URL: *www.icar.cnr.it*
– Sede di Napoli, Via P. Castellino 111, 80131 Napoli, URL: *www.na.icar.cnr.it*
– Sede di Palermo, Viale delle Scienze, 90128 Palermo, URL: *www.pa.icar.cnr.it*

Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Activity of the FIPA Methodology Technical Committee

M. Cossentino [1], A. Garro [2]

Rapporto Tecnico N.: 15 ***Data:***
RT-ICAR-PA-05-15 **dicembre 2005**

---

[1] Istituto di Calcolo e Reti ad Alte Prestazioni, ICAR-CNR, Sede di Palermo Viale delle Scienze edificio 11 90128 Palermo
[2] Università della Calabria, Dipartimento di Elettronica, Informatica e Sistemistica Via P. Bucci, 41C - Arcavacata di Rende (CS)

Abstract

The method engineering paradigm enables designers to use phases, models and elements coming from different design processes in order to build up a new process expressly tailored for carrying out the task of developing a problem/domain specific system. This paradigm, widely studied in the object-oriented world, can be profitably applied to multi-agent systems design as witnessed by the recent activity of the FIPA Methodology Technical Committee (TC) which is presented in the paper.

I. INTRODUCTION

In the last years, the agent-oriented approach [30] has been recognized as very suitable for the development of complex software systems since it fully exploits the well known techniques of *Decomposition*, *Abstraction* and *Organization* [4] for helping to manage complexity. In particular in the context of complex software systems:

- the agent-oriented decomposition is an effective way of partitioning the problem space;
- the key abstractions of the agent-oriented mindset (agents, interactions, and organizations) are a natural means of modelling;
- the agent-oriented philosophy for modelling and managing organizational relationships is appropriate for dealing with existing dependencies and interactions [30].

The development of complex software systems by using the agent-oriented approach requires suitable agent-oriented modelling techniques and methodologies (in the paper the term methodology will be used as synonymous of software engineering process) which provide explicit support for the key abstractions of the agent paradigm.

Several methodologies supporting analysis, design and implementation of Multi-Agent Systems (MASs) have been to date proposed in the context of Agent Oriented Software Engineering (AOSE) [33]. Some of the emerging methodologies are Adelfe [3], Gaia [47], MaSE [16], Message [9], Passi [13], Prometheus [40], and Tropos [5] . Although such methodologies have different advantages when applied to specific problems it's a matter of fact that an unique methodology cannot be general enough to be useful to everyone without some level of customization. In fact, agent designers, for solving specific problems in a specific application context, often prefer to define their own methodology specifically tailored for their needs instead of reusing an existing one. Thus an approach that combines the designer's need of defining his own methodology with

the advantages and the experiences coming from the existing and documented methodologies is highly required.

A possible solution to this problem is to adopt the method engineering paradigm so enabling designers of MAS to (re-)use parts coming from different methodologies in order to build up a customized approach for their own problems [28]. In particular, the "development methodology" is constructed by assembling pieces of methodology (**method fragments**) from a repository of methods (**method base**). The method base is built up by taking method fragments coming from existing methodologies or ad hoc defined methods. Currently this approach has been adopted by the FIPA (Foundation for Intelligent Physical Agents) Methodology Technical Committee (TC) [19] in which the authors are active members. FIPA is currently moving to the IEEE Computer Society under the name of *IEEE FIPA Standards Committee*. Many of its TCs will become working groups in the new situation and the proposal of a Methodology working group is ongoing.

The FIPA Methodology Technical Committee (TC) has been constituted in 2003 with the aim of capitalizing the efforts of many researchers in the area of MASs design and identifying a design methodology for MASs that could fit the greatest number of needs.
More in details, the main goals of the TC were:

- **Definition of the method fragments meta-model**. It is necessary to formally represent method fragments in order to facilitate their identification, representation, integration and storing in the method base;
- **Identification of the method base architecture**. The method base needs of a technological infrastructure for the instantiation of the method meta-model previously defined;
- **Collection of method fragments**. They can origin from the most diffused methodologies and other specific contributions. After the formalization they can be introduced in the method base;
- **Description of techniques for methods integration**. It is necessary to define guidelines for methods integration in order to both construct the methodology (retrieving the method fragments from the method base and integrating them) and apply it in the real design work.

A more ambitious goal is enabling the use of:

- process aware $CAME$ (Computer Aided Method Engineering) tools that offer specific

support for the composition of a methodology from existing fragments; these tools should be able to support the definition of the process model as well as the reuse of fragments from the method base. They should enable the adoption of a specific process model (waterfall, iterative/incremental, spiral, ...) and the placing of different fragments in it. The CAME tool should "instantiate" a proper CASE tool (see below) that is specifically customized to support the designer in working with the composed methodology.

- $CASE$ (Computer Aided Software Engineering) tools that assist the designer in performing the development process based on the composed methodology. These tools should be the evolution of existing CASE instruments since they should be able to follow the order of phases defined at methodology composition time (accordingly to the adopted process model[10]) and guide the designer in profitably applying it.

The work already done by the FIPA Methodology TC can be summarized as follows: definition of a method fragment meta-model (including an XML-based method fragment representation, see section III); definition of a method base general architecture; representation of some methodologies using a process description language, the TC adopted OMG SPEM (Software Process Engineering Metamodel) notation [46], the described methodologies are: ADELFE, Gaia, and PASSI (see the TC documents [27], [25], [14]); collection of method fragments, this has been done by extracting method fragments from the previously listed methodologies according to the defined method fragment meta-model (see the TC documents [26], [25], [12]), a new fragment that is specific to deal with complex systems has been listed too [41]; and finally, identification of some approaches and guidelines for methods integration.

In the next sections these points will be discussed in details.

II. Agent-Oriented Method Engineering

The approach proposed by the FIPA Methodology TC is an extension of the method engineering paradigm [6][32][45] that incorporates the specific needs of designing a MAS instead of an object-oriented system. As already said, according to this approach, a methodology or better a *SEP* (Software Engineering Process) is built up by assembling pieces of the process (method fragments) [42][7][8] taken from a repository of methods that has been built by extracting pieces from existing design processes. The main advantages of this approach consist in the fact that the method engineer (who is responsible for the composition of the SEP) could obtain the

best process for the specific needs he is facing. From another point of view (that has been decisive in the FIPA context), this approach allows the contribution of a large community (several design processes for MAS design already exist in literature [34]) without imposing any kind of discrimination on what is “compliant” and what is not. The future FIPA compliant SEPs will be simply composed by reusing parts from the repository accordingly to the proposed guidelines, nothing more than that. Most of all, the different contributions to the repository are valuable because they are the consequence of some specific need, development context, application environment or theoretical background and as such they can be profitably reused when something similar is found in facing a new problem (just like a bridge pattern [21] can be reused whenever it is necessary to decouple an interface from an implementation, regardless of the original context that allowed the identification of this pattern). Some authors already started to work in this direction in the agent community [29], [31], [23], [15] thus confirming that the method engineering paradigm that has been already widely studied in the object-oriented world can be profitably applied to multi-agent systems design too.

Figure 1 describes the approach to the whole method engineering process proposed by the FIPA Methodology TC. It includes three main phases: the fragments repository construction, the SEP definition and the SEP use. The fragments repository is built by conveniently modularizing the existing design processes and converting them to the method fragment structure defined by the TC (see section III). The identification of guidelines for fragments extractions from existing processes is currently out of the scope of the FIPA Methodology TC activity although it has some effect on the fragment structure (for instance on its granularity). This problem is still an open research issue and some contributions from object-oriented method engineering can be found in [42] [8] while a more specific agent-oriented approach is presented in [15]. The method base can also list fragments not coming from an entire SEP but conceived as stand alone contributions to the repository. This is the case of the MaCMAS fragment that allows to deal with the analysis phase of complex systems [41]. The TC members think that this is a relevant contribution to the research on AOSE: it is no more necessary to prepare an entire design process in order to study one single aspect of agency. A researcher can focus his attention on the preferred topics and then complement the resulting method fragment(s) with others coming from the method base thus quickly completing a process that he can use to test the new parts resulting from his work.

During the SEP construction, the method engineer, has to consider several different factors

that effect his work:

- the SEP should be fitted for the specific family of problems it will be applied. This means that if the problem is typically effected by some constraint (e.g. real-time or security issues) it should include proper methods to explicitly deal with them.
- the SEP is to be used by persons. This means that the method engineering should compose a SEP that is coherent with their skills (or at least not too far); a group of designer already skilled with some design practice should not be forced to change this use if it is possible to adopt (eventually part of) the old approach to solve the new problem.
- designing agents is different from designing objects. Several papers deal with this issue (this is out of the scope of this paper, see [36], [47] for further details), by now it's worth noting that designing an agent society is characterized by fundamental choices like the social structure (peer-agents, hierarchical organizations and so on) or single-agent architecture (reactive, BDI, state-based, ...) that are a kind of requirement for the SEP. They often descend from the development context or the specific problem to be solved: a society that has a consolidated tradition in adopting BDI agents organized in groups of peer agents will more likely choose a similar general architecture for solving future problems rather than change it if not really necessary (or remarkably profitable); as a consequence, the new SEP should encompass these aspects.
- agent is not a well defined concept; several different definitions can be found in literature [44], [39], [20] and this also includes most of the concepts used when defining a MAS (role, task, behavior, goal, ...), as a consequence while basis of object-oriented programming are well defined and widely accepted, it is not so for agent-oriented programming. The MAS is usually designed and implemented by considering abstractions and components that could be significantly different. Several studies have been carried on during these last years about MAS meta-models [37], [2], [9], [17] and a final result has not been achieved while in the OO context it exists a consolidated meta-model of the system that has been clearly defined (it is sufficient to think about the M2 level of the UML 2.0 Infrastructure [38]). The same absence of a real, pure agent-oriented coding language is the consequence of this situation (most diffused solutions are Java-based). The FIPA Methodology TC approach encourages studies in this direction and allows an easier application of their results since

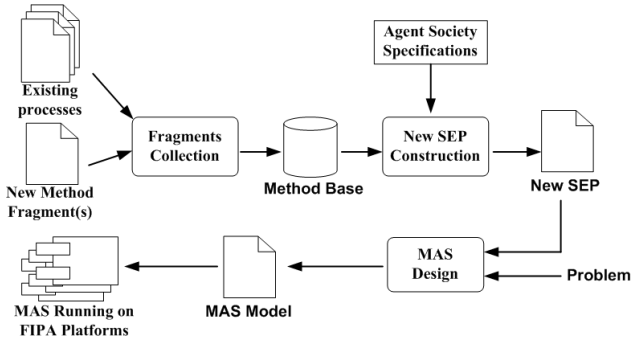

Fig. 1. The adopted Agent-Oriented Method Engineering process

the *method fragment definition* (see section III) given by the TC explicitly considers the MAS meta-model elements involved in its workflow and artifacts.

The resulting SEP will be essentially composed of a flow of activities to be done, the descriptions of a set of artifacts, the related guidelines and the suggested notation for the required artifacts.

During the last phase, the system designer, adopts the new SEP (and related CASE tool) to design the MAS solution to the problem he is facing and in so doing, assisted by the cited CASE tool, produces the required artifacts by following the guidelines of the process.

In the following section the *method fragment definition* will be introduced.

III. METHOD FRAGMENT DEFINITION

A SEP can be see as mainly composed of activities to be performed (requirements elicitation, analysis, ...), the artifacts produced during these activities and the related guidelines. Indeed, several other aspects should be considered as well:

- the stakeholders involved in the process and their roles (some of them can be responsible of a certain activity, some others can assist and so on);
- the specification of the agent concept: it is still an open question in the research field;
- the meta-level structure of the system to be built;
- the modelling language: a standardization effort is carried on in parallel to the one of the FIPA Methodology TC by the FIPA Modeling TC that starting from some initial proposals for an extension of the UML is now defining the structure for a future agent modelling language.

Copying with this situation the FIPA methodology TC started his work by defining the method fragment and complementing it with some studies on MAS meta-models and a glossary of terms. As regards the method fragment [35], it is considered as a portion of the development process, composed of the following parts:

1) A specification of the portion of process which defines what is to be done by the involved stakeholder(s) and in what order. The fragment specification prescribes the use of the OMG SPEM language [46] for describing its procedural aspect of the work flow; despite the use of it has pointed out that a more versatile and complete version of the language would be desirable, it still proved sufficient for the FIPA Methodology TC purposes and several members reported convincing experiences about its use. According to the SPEM syntax, the FIPA fragment can be regarded as a *process component*.
2) One or more deliverables such as AUML/UML diagrams [1] and text documents; these should be part of the fragment specification in form of a description of their structure (in order to clarify what is the expected output of the presented activities) also including a reference to the suggested (or adopted, in the original methodology from which this fragment has been extracted) modelling syntax.
3) Some preconditions which represent a kind of constraint specifying when it is possible to fire the activities specified in the fragment. They are usually related to the required input data; these preconditions can be thought as the similar preconditions in a contract between two classes. In particular, the preceding fragment (or the *n* preceding fragments) is/are responsible for establishing the conditions that will enable the successful execution of the following fragment. The formalization of this preconditions would allow the introduction of some kind of automatic assistance in the composition of the fragments but a formal language has not been specified nor adopted yet and the only considerations that can be easily automated by now, regards the required input set in terms of already defined MAS meta-model components (see below).
4) A list of components of the MAS meta-model (which are a part of the MAS meta-model subsumed by the methodology from which the fragment was extracted) to be defined or refined through the specified process; while this list could be theoretically speaking void (this is for instance the case of a fragment whose purpose consists in selecting between

two different paths in the design process accordingly to the evaluation of some aspects of the actual design), all the fragments that have been up to now identified, are concerned with some components to be defined/refined, thus showing that the community is, by now, still more concerned about a product-oriented identification of fragments rather than a process-oriented one.

5) Application guidelines that illustrate how to apply the fragment and the related best practices; the same formalization of these aspects in the existing agent-oriented methodologies has its own specific importance since otherwise, except for a few well documented approaches, guidelines often remain bounded to the personal knowledge of some skilled designers or the methodology creators.
6) A glossary of terms used in the fragment; this avoids misunderstandings if the fragment is reused in a context that is different from the original one; in order to facilitate this part of the fragment documentation the FIPA Methodology TC has discussed a list of definitions for many commonly used terms and these are made available from the TC website.
7) Composition guidelines which describe the context/problem addressed by the specific fragment and that is behind the methodology from which is has been extracted.
8) Aspects of fragment; they are textual descriptions of specific issues such as platform to be used for system implementation, application area, etc; this helps in delimiting the proper application field for the fragment.
9) Dependency relationships useful to assemble fragments. When fragments granularity is fine grained (and the FIPA repository is conceived to allow the introduction of different sized fragments) it is frequent to reuse more fragments from a specific methodology since their adoption probably correspond to the choice of some philosophy for the solution of a specific portion of the design process composition problem.

It should be noted that although a complete description of the fragment in all of the above listed field is advisable if possible, not all of these elements are always mandatory; some of them (for instance deliverable notation or guidelines) could be not applicable or not necessary for some specific fragment.

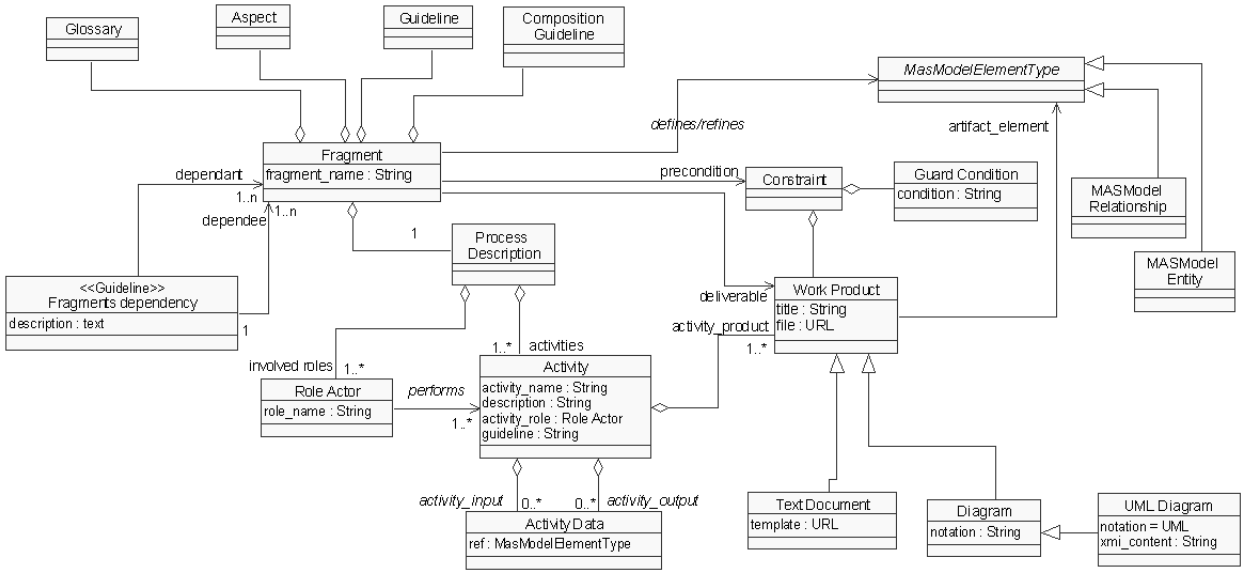

Fig. 2. The method base model as currently defined by the FIPA methodology TC

IV. METHOD BASE ARCHITECTURE

According to the reported definition of method fragment the **method base** structure proposed by the FIPA Methodology TC is an XML based repository storing a collection of XML document, each one representing a method fragment, validated by a DTD or an XML Schema. It is possible to formally represent the proposed method base structure by using an UML Class Diagram as showed in Figure 2.

The repository basically includes two different portions of information: on the left part of Figure 2 are represented the elements of the method fragment already described in the previous section, the elements in the right part are helpful in order to introduce a support for the fragments assembling that also includes the possibility of verifying the compliance with data-related composition constraints. The repository is oriented toward a MAS meta-model based classification of fragments; each one of them is in fact labeled with the MAS meta-model components that are defined or refined during its activities. These components are described in the fragment work products (text documents or diagrams with a link to the adopted notation) that are related to the activities performed by the role actors in the fragment. Each activity has some inputs and produces some outputs in terms of defined/refined elements of the MAS meta-model. The fragment preconditions are represented in terms of required work products or guard

conditions (these can for instance detail the required refinement level of some elements of the MAS meta-model).

The showed repository model can be easily translated in a XML *Document Type Definition* (DTD) or in a *XML Schema* that can be used for validating an XML document representing a particular method fragment (all elements not directly related to fragment specification are not necessarily required but their use enable the realization of assembling features in the methodology construction supporting tools that will be built in the future). The validation process ensures that the particular method fragment was extracted and defined according to the FIPA Methodology TC method fragment meta-model. It is worth to point out that the XML document representing a fragment is not self-contained but could contain some URI that point to resources that can not be coded in XML but that constitute portions of the fragment.

Further details about the repository implementation or querying approaches are considered out of the scope of the work of the TC and are left to tool implementers. A prototypical implementation of the method base is already available in the committee web pages and is being filled in with the fragments of the available methodologies.

V. Method fragments integration

Method fragments integration is the process of composition of the new methodology and usually consists of two different and complementary phases: the selection of the reused fragments from the method base and their assembling. Several approaches exist in literature to deal with these crucial phases, among the others the work of Ralyté where fragments are composed by association and integration [43], and the Brinkkemper's paper [48] where the composition process is based on three orthogonal dimensions: perspective, abstraction and granularity.

At the writing of this paper, the FIPA Methodology TC members are still discussing about this topic and, specifically, they are mainly studying two basic approaches for the construction of the agent-oriented SEP by using methods integration [22]: (i) *meta-model driven*, which starts from the definition of some agent society aspects (mainly the MAS meta-model) preferred by the designer (because of preceding experiences, developing environment or customer specific requirements) for the development of a MAS that are suitable for a specific problem in a specific application domain; (ii) *development process driven*, which is based on the instantiation of a software development process in which each phase is carried out using appropriate method

fragments selected on the basis of the supported activities and of the resulting work products.

Both of the proposed approaches have been experimented by the authors in various applications and case studies. In particular, the *meta-model driven* approach to method fragments integration was exploited for the construction of a SEP suitable for developing a MAS for the prediction of the three-dimensional structures of proteins [24] and a MAS for E-Learning and Skill Management context [23]. The *development process driven* approach to method fragments integration was exploited for the construction of a SEP for the modelling and the validation through simulation of MAS [18]. These two approaches will be further explored and then compared in the following sub-sections.

A. The MAS meta-model driven approach for method fragments integration

To build a SEP by exploiting the *meta-model driven* approach, the designer must:

- choose or define some key aspects of the agent society, mainly a MAS meta-model suitable for the specific problem and/or the specific application domain;
- identify the elements that compose the defined meta-model of the MAS under development; the list of these elements could descend from the developing environment (specific agent platform often impose a relevant number of them), agent reasoning and architecture paradigm (the choice of a BDI architecture naturally involves the adoption of concepts like belief, desire, intention), involved stakeholders expertise (the availability of analysts used to some kind of requirement analysis approach induces to include resulting elements); and so on.
- choose the method fragments that are able to produce the identified meta-model elements; the main criterion here is related to the complete coverage of the meta-model instantiation procedure;
- defining a development process characterized by a *method fragments execution order* on the basis of the relationship existing among the meta-model elements produced by each fragment; in this phase a kind of dependency matrix among the artifacts produced by the fragments could help together with opportunity considerations (for instance, in an agile approach, test planning is done as soon as it is possible).

Hence, the obtained SEP is able to completely ensure the MAS meta-model instantiation for the given problem in a specific application domain.

B. The development process driven approach for method fragments integration

The development process driven approach focuses on the instantiation of a software development process that completely covers the development of MAS and complies some specific needs related with it (like the creation of an extensive documentation or the flexibility in managing new requirements).

To build a SEP by exploiting the *development process driven* approach, the designer must:

- choose or define a software development process model (waterfall, evolutionary or incremental, transformation, spiral, . . .) most suitable for the specific problem and for the specific application domain; this means for instance the adoption of a waterfall process model if the customer explicitly requires it (as it happens in some government contracts) or an iterative/incremental one to cope with evolving requirements and development risks management;
- instantiate the development process by selecting, for each phase of the process model, some suitable method fragments, chosen from the method base or even ad-hoc defined.

It's worth noting that if two subsequent phases (P_1 and P_2) are carried out by using method fragments coming from different methodologies, it is required to elaborate the work products of phase P_1 to obtain the information needed to drive the construction of the work products of phase P_2. In other words, the work products produced in a given phase might constitute the input for the subsequent phase provided that they contain all the information required for initializing it.

C. Comparison of the approaches

As it is usual in software engineering, each of the proposed approaches has advantages and drawbacks, beginning from the first, in particular, the *meta-model driven* provides flexibility for the definition of many aspects of the MAS to be developed; this is probably the most suited one if social rules coming from a specific domain play a relevant role in the problem to be solved. Conversely, it is characterized by a difficulty of integration of different fragments due to the different semantics of the concepts they represent in the meta-models subsumed by the methodologies from which they have been extracted; further more, the a-priori selection and/or definition of the meta-model to adopt for the specific problem and/or application domain is a difficult and at the same time crucial task.

The *development process driven* approach is characterized by the following advantages: flexibility for the construction of a SEP by means of the instantiation of each stage of the development process according to a selected process model. On the other hand, the disadvantages are the following: (i) low flexibility of the system meta-model since the meta-model results from the sum of the element defined by the selected method fragments; (ii) adaptation among the work products which is sometimes difficult to achieve; (iii) choice and definition of the process to instantiate for the specific problem and/or application context; (iv) low level of help in selecting the fragments that descends from the lone process model choice (several degrees of freedom still exist and other guidance are needed to select the proper method fragments).

Each one of above listed points represents an open problem and a challenge for the agent community; although most of these issues are common to both agents and objects research contexts, the first one has still to explore some peculiarities that are related to the agent paradigm, the most important probably being the role that the agent social organization plays in the composition of the new process.

The proposed approaches to the integration of methods fragments (*meta-model driven* and *development process driven*) are not mutually exclusive; rather, hybrid approaches containing features of both of them might be defined as well. An example of process composition that mixed both of the proposed approaches has been used to create one of the first agile processes for MAS design, PASSI Agile [11]; it started from the general model of an agile process and adopted a MAS meta-model that has been obtained by conveniently reducing the conventional PASSI one. Different approaches can be considered as well (for instance some based on the attributes of the resulting process [31]) and their use is not in contrast with the presented ways.

VI. Conclusions

This paper presents the activity of the FIPA Methodology Technical Committee which aims at adopting the method engineering approach for the design of MASs. This approach once moved to the agent-oriented context presents new research challenges that have been faced; conventional object-oriented studies although important for the new application context are not sufficient: the concept of agent and agent societies are to be introduced and specifically managed in the whole process with the consequence of significant changes to the existing state-of-the-art. As regards the actual results of these studies, they are: a specification of the method fragment structure

(that includes agent-related aspects and formalizes the reusable part of a design process), a description of the method base that could allow an easier interchange of fragments produced in different contexts, some guidelines about the assembling of customized design process. Some of these issues have not still found a definitive solution (and they are still a work in progress) but interesting papers have been presented that evaluated/adopted the FIPA Methodology TC results [11], [18], [23], [24] or follow similar approaches [29], [31]. Future works include the attempt of enabling the interoperability between the TC specifications and other existing frameworks like the OPEN [28] one.

Acknowledgments

The authors wish to thank all the other members of the FIPA Methodology Technical Committee for their precious work and priceless support which made this work possible.

The authors are also grateful to all the other members of the AgentLink AOSE Technical Forum Group for their valuable hints and suggestions.

REFERENCES

[1] B. Bauer, J.P. Muller, and J. Odell. Agent UML: A Formalism for Specifying Multiagent Interaction. In Paolo Ciancarini and Michael Wooldridge, editors, *Agent-Oriented Software Engineering*, pages 91–103. Springer-Verlag, Berlin, 2001.

[2] C. Bernon, M. Cossentino, M.P. Gleizes, P. Turci, and F. Zambonelli. A Study of some Multi-agent Meta-Models. In *Proc. of the Fifth International Workshop on Agen-Oriented Software Engineering (AOSE-2004) at The Third International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2004)*, New York, USA.

[3] C. Bernon., M.P. Gleizes, G. Picard, and P. Glize. The Adelfe Methodology For an Intranet System Design. In *Proc. of the Fourth International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS)*, Toronto, Canada, 2002.

[4] G. Booch. Object-Oriented Analysis and Design with Applications. Addison Wesley, 1994.

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

[6] S. Brinkkemper. Method engineering: engineering the information systems development methods and tools. In *Information and Software Technology*, 37(11), 1995.

[7] S. Brinkkemper, K. Lyytinen, and R. Welke. Method engineering: Principles of method construction and tool support. *International Federational for Information Processing*, 1996.

[8] S. Brinkkemper, M. Saeki, and F. Harmsen. Meta-modelling based assembly techniques for situational method engineering. In *Information Systems*, 24(3), 1999.

[9] G. Caire, F. Leal, P. Chainho, R. Evans, F. Garijo, J. Gomez, J. Pavon, P. Kearney, J. Stark, and P. Massonet. Agent Oriented Analysis using MESSAGE/UML. In *Proc. of the 2nd In-ternational Workshop on Agent-Oriented Software Engineering (AOSE)*, LNCS 2222. Springer-Verlag, Berlin, 2002.

[10] L. Cernuzzi, M. Cossentino, and F. Zambonelli. Process Models for Agent-based Development, *Journal of Engineering Applications of Artificial Intelligence (EAAI)*, 18(2), 2005, Elsevier.

[11] A. Chella, M. Cossentino, L. Sabatucci, and V. Seidita. From PASSI to Agile PASSI: Tailoring a Design Process to Meet New Needs. In *Proc. of IEEE/WIC/ACM International Joint Conference on Intelligent Agent Technology*, Beijing, China, September 2004.

[12] M. Cossentino. PASSI fragments: All fragments, draft. rel 0.1, Feb. 2004. *[http://www.pa.icar.cnr.it/ cossentino/FIPAmeth /docs/passi_fragments_0_1.zip]*

[13] M. Cossentino. From Requirements to Code with the PASSI Methodology. *Agent-Oriented Methodologies*, B. Henderson-Sellers and P. Giorgini (Editors). Idea Group Inc., Hershey, PA, USA. 2005.

[14] M. Cossentino, L. Sabatucci, and V. Seidita. Expressing PASSI Methodology using Spem. *FIPA Methodology TC, working draft v. 1.0/04-03-15, [http://fipa.org/activities/methodology.html]*.

[15] M. Cossentino and V. Seidita. Composition of a New Process to Meet Agile Needs Using Method Engineering. In *Software Engineering for Large Multi-Agent Systems vol. III*, LNCS Series, Elsivier Editor, 2004.

[16] S. A. DeLoach, M. Wood, and C. Sparkman. Multiagent system engineering. *International Journal of Software Engineering and Knowledge Engineering*, 11(3):231–258, April 2001.

[17] J. Ferber, and O. Gutknecht. A Meta-model for the Analysis and Design of Organizations in Multi-agent Systems. In *Proc. of the 3rd International Conference on Multi-Agent Systems (ICMAS'98)*, pages 128–135, 1998.

[18] G. Fortino, A. Garro, and W. Russo. From Modeling to Simulation of Multi-Agent Systems: an integrated approach and a case study. In Gabriela Lindemann, Jorg Denzinger, Ingo J. Timm, Rainer Unland, editors, *Multiagent System Technologies*, Lecture Notes in Artificial Intelligence (LNAI) volume 3187, pages 213–227, Springer-Verlag, Berlin Heidelberg, Germany, 2004.

[19] Foundation for Intelligent Physical Agents (FIPA). *[http://www.fipa.org]*.

[20] S. Franklin, and A. Graesser. Is it an Agent, or Just a Program?: A Taxonomy for Autonomous Agents. In *Proc. of the 3rd International Workshop on Agent Theories, Architectures, and Languages*, LNAI Series, volume 1193, pages 21–35, Springer Verlag, 1996.

[21] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns Elements of Reusable Object Oriented Software*, Addison-Wesley, 1994.

[22] A. Garro, G. Fortino, W. Russo. Using Method Engineering for the Construction of Agent-Oriented Methodologies. In *Proc. of WOA 04 - Dagli Oggetti agli Agenti, Sistemi Complessi e Agenti razionali*, pages 51–54, Torino, Italy, December 2004.

[23] A. Garro and L. Palopoli. An XML Multi-Agent System for E-Learning and Skill Management. In Ryszard Kowalczyk, Jorg P. Muller, Huaglory Tianfield, Rainer Unland, editors, *Agent Technologies, Infrastructures, Tools, and Applications for E-Services* , Lecture Notes in Artificial Intelligence (LNAI) volume 2592, pages 283–294, Springer-Verlag, Berlin Heidelberg, Germany, 2003.

[24] A. Garro, G. Terracina, and D. Ursino. A Multi-Agent System for supporting the prediction of protein structures. *Integrated Computer-Aided Engineering (ICAE)*, 11(3):259–280, 2004. IOS Press, Amsterdam, The Netherlands.

[25] A. Garro, P. Turci, and M.P. Huget. Expressing Gaia Methodology using Spem. *FIPA Methodology TC, working draft v. 1.0/04-03-15, [http://fipa.org/activities/methodology.html]*.

[26] M. P. Gleizes et al. Adelfe fragments, rel.0, March 2004. *[http://www.pa.icar.cnr.it/ cossentino /FIPAmeth/docs/adelfe_fragments_v0.pdf]*

[27] M.P. Gleizes, T. Millan, and G. Picard. ADELFE:Using SPEM Notation to Unify Agent Engineering Processes and Methodology. *IRIT/2003-10-R, [http://fipa.org/activities/methodology.html]*.

[28] B. Henderson-Sellers. Method Engineering for OO Systems Development. *Communications of the ACM*, 46(10), 2003.

[29] B. Henderson-Sellers. Creating a comprehensive agent-oriented methodology - using method engineering and the OPEN metamodel. In B. Henderson-Sellers and P. Giorgini, editors, *Agent-Oriented Methodologies* Idea Group, 2005.

[30] N. R. Jennings. An Agent-Based Approach for Building Complex Software Systems. *Communications of the ACM*, 44(4), 2001.

[31] T. Juan, and L. Sterling, and M. Winikoff. Assembling Agent Oriented Software Engineering Methodologies from Features. In *Proc. of the Third International Workshop on Agent-Oriented Software Engineering, at AAMAS'02*.

[32] K. Kumar and R. Welke. Methodology engineering: a proposal for situation-specific methodology construction. In *Challenges and Strategies for Research in Systems Development*, pages 257–269, 1992.

[33] J. Lind. Issues in Agent-Oriented Software Engineering. In *Proc. of the First International Workshop on Agent-Oriented Software Engineering (AOSE)*, LNCS 1957, pages 45–58. Springer-Verlag, Berlin, 2001.

[34] M. Luck, R. Ashri, and M. D'Inverno. Agent-Based Software Development, 2004, Artech House Publishers.

[35] Method Fragment Definition. *FIPA Methodology TC, working draft, Nov. 2003, [http://www.fipa.org/activities/methodology.html]*.

[36] J. Odell. Objects and Agents Compared. In *Journal of Object Technology*, 1(1):41–53, 2002.

[37] J. Odell, M. Nodine, and R. Levy. A Metamodel for Agents, Roles, and Groups. In *Agent-Oriented Software Engineering (AOSE) V*, 2005.

[38] OMG. UML 2.0 Superstructure Specification, August, 2002.

[39] OMG Agent Platform Special Interest Group. Agent Technology - Green paper, version 1.0, September, 2000.

[40] L. Padgham and M. Winikoff. Prometheus: A methodology for developing intelligent agents. In *Proc. of the Third International Workshop on Agent-Oriented Software Engineering (AOSE)*, LNCS 2585, Springer-Verlag, Berlin, 2003.

[41] J. Pena, and R. Corchuelo. MaCMAS/UML: A Methodology Fragment for the Analysis Stage of Large Complex/Complicated Multi-Agent Systems. The Distributed Group, University of Seville, 2004.

[42] J. Ralyté and C. Rolland. An approach for method reengineering. *Lecture Notes in Computer Science*, pages 27–30, 2001.

[43] J. Ralyté, and C. Rolland. An assembly process model for method engineering. In *Proceedings of the 13th Conference on Advanced Information Systems Engineering, CAISE01*, Interlaken, (Switzerland), (2001)

[44] S. Russell, and P. Norvig Artificial Intelligence; A Modern Approach, 1995, Prentice Hall.

[45] M. Saeki. Software specification & design methods and method engineering. In *International Journal of Software Engineering and Knowledge Engineering*, 1994.

[46] Software Process Engineering Metamodel Specification, Version 1.0, formal/02-11-14. Object Management Group Inc. , November 2002.

[47] F. Zambonelli, N. R. Jennings, and M. Wooldridge. Developing Multiagent Systems: The Gaia Methodology. *ACM Transactions on Software Engineering and Methodology*, 12(3):317–370, 2003.

[48] S. Brinkkemper, M. Saeki, and F. Harmsen. Meta-modelling based assembly techniques for situational method engineering. *Information Systems*, 24, (1999)