Fragmenting ADELFE using the Medee Framework

Thomas Liu de Almeida, Sara Casare, Anarosa A. F. Brandão

Laboratório de Técnicas Inteligentes
Escola Politécnica – Universidade de São Paulo (USP)

{thomas.almeida, anarosa.brandao}@usp.br, sjcasare@uol.com

Abstract: *This work describes the fragmentation of the agent-oriented method ADELFE following the approach defined by the Medee Framework. The fragmentation goal is to increase the population of the Medee Method Framework repository. The Medee method framework supports the development of multiagent systems based on the Situational Method Engineering approach. To achieve our goal, the paper briefly explains the Medee Method Repository structure and its associate process for fragmenting methods. Then, it's described how this process was applied to define ADELFE fragments.*

1. Introduction

The multiagent paradigm is based on the interaction of autonomous elements in order of achieving goals in a given system. It is a fitting solution for various situations of informatics, specially nowadays when huge amount of information and fast changes on the virtual environment request some kind adaptation and application of artificial intelligence techniques. However, the lack of tools and frameworks, combined with complexity of systems based on this paradigm makes its adoption scarce in the software industry.

The Medee Method Framework [Casare et al, 2013] aims to provide this kind of support for developing multiagent systems. It allows the building of customized methods on demand according to the project specifications. This is based on Situational Method Engineering concepts, which is a section of Method Engineering [Brinkkemper, 1996] that focuses on these tailored solutions to each given situation. They are called situational methods.

In order to achieve that, the framework stores pieces of agent-oriented methodologies, such as Tropos [Giorgini et al, 2004], Gaia [Zambonelli et al, 2003] and PASSI [Cossentino, 2005], as well as organizational models, like MOISE+ [Hubner et al, 2002]. Each one of these pieces is called a fragment. Fragments are standardized and represent a coherent part of a method that can be used and re-used to solve entire problems or parts of them. This paper consists in the fragmentation of the agent-oriented method ADELFE [ADELFE, 2003] utilizing the process proposed in the Medee Framework.

2. Medee Method Framework

The Medee Method Framework aims to provide methods for developing organization centered MAS. It allows you the combination of advantages from AOSE methods and agent organizational models by reusing portions of these two types of MAS development approaches.

The Medee Method Framework is basically composed by the Medee Conceptual Model, the Medee Method Repository and the Medee Delivery Process [Casare et al, 2013].

2.1 Medee Method Repository

The architecture of the Medee Method Repository consists in three pillars: the Medee Elements, the Medee Fragments and the Medee Methods. The first pillar provides SPEM method elements [SPEM, 2008], that are: tasks, work products, roles and guidance. They will serve for elaborating Medee MAS method fragments stored in the second pillar, which in their turn provide fragments to compose methods stored in the third pillar. SPEM is a standard for describing methods and processes.

Method fragments in the second pillar pertain to one of the following layers of granularity: activity, phase, iteration or process. They are also standardized according to Medee MAS Method fragment definition, in order to make the reuse and composition of methods with fragments from different sources possible.

The Medee Methods pillar stores Medee Situational Methods and Medee AOSE Methods. The former is composed according to a given project situation, and the latter is formed by a set of fragments usually captured from on single source.

2.2. Medee Delivery Process

The Medee Delivery Process specifies how to populate the three pillars of the Medee Repository. It is composed of three phases: Method Elements Capture, Method Fragment Elaboration and Medee Method Composition, each one describing one pillar of the Medee Repository, from the first to the third, respectively. During the description of ADELFE fragmentation the phases will be outlined.

3. Fragmenting ADELFE using the Medee Delivery Process

3.1 ADELFE in a nutshell

The name ADELFE is the French acronym for "toolkit to develop software with emergent functionality" (Atelier pour le DEveloppement de Logiciels à Fonctionnalité Emergente) [Bernon et al, 2005]. It is a method based on object-oriented methods, which follows the Rational Unified Process (RUP) and uses UML and AUML notations [Bauer et al, 2001] . ADELFE aims to guide the development of Adaptative Multiagent Systems (AMAS). These systems are embedded in a dynamic environment, where agents are continuously searching and adapting themselves to keep cooperating in collective tasks. It is composed by four phases: preliminary requirements, final requirements, analysis and design. Figure 1 shows the ADELFE workflow detailing its phases.

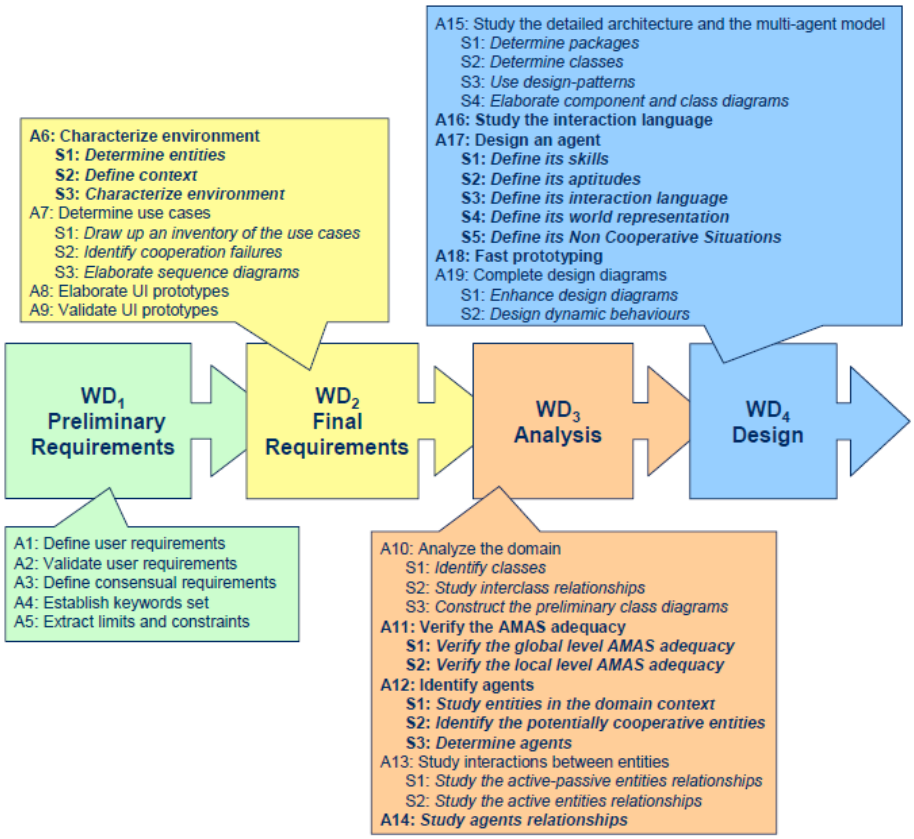

Figure 1: ADELFE workflow, sourced from [Bernon et al, 2005].

3.2 Fragmenting ADELFE

Although the Medee Delivery Process is composed of three phases, fragmentation itself is conducted by phases one (*Method Element Capture*) and two (*Method Fragment Elaboration*).

The *Method Element Capture* phase comprises the following tasks: Outline Method Content; Detail Method Content; Build AOSE Method As Is; Publish AOSE Method As Is, and its outcomes are SPEM elements such as work products, roles, tasks, guidelines captured from the target method, and the method as it is using those elements. It is all made using the tool EPF Composer [Haumer, 2007], a tool that implements SPEM.

Firstly, an understanding about the method was needed. It was made through [Bernon et al, 2005] and [ADELFE, 2003] . In addition, [Cossentino and Seidita, 2005] presented ADELFE fragmentation following another approach that uses an extended version of SPEM to describe fragments. Almost the same was presented in [ADELFE, 2003] . the difference between these them is that the first contain just one Phase for Requirements, while the second divides it in Preliminary Requirements and Final Requirements. Then, populating the first pillar of the repository was possible by mapping the SPEM elements already defined and describing them using the EPF Composer. To store the SPEM elements as tasks, roles, work products and guidance, a plugin named A: ADELFE Content Elements was created in EPF, containing the Main ADELFE Content Package, with divisions for each one of the elements, as can be seen at Figure 2.

Then, the method elements were organized in the form of process, to represent the method as the combination of SPEM elements, following [Bernon et al, 2005]. Therefore, the "method as it is" was stored in another EPF plugin, named A: ADELFE As Is. Figure 3 shows the work breakdown structure of the process related to the method.

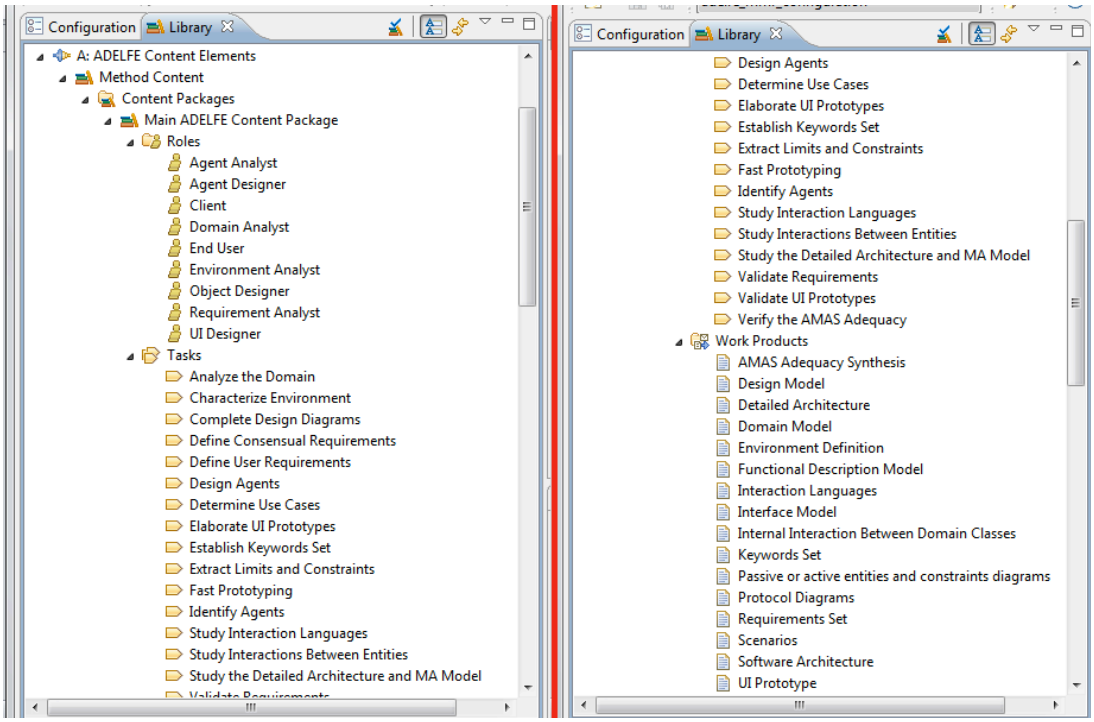

Figure 2: SPEM elements captured stored in the EPF Composer. The red line indicates a junction. Actually, it's a vertically continuous list.

Once that all method elements are defined and documented, the *Method Fragment Elaboration* phase begins, in order to populate the second pillar of the Medee Repository with fragments sourced from ADELFE. It consists in the following tasks: Build MAS Variability; Build Activity Method Fragment; Create Iteration Method Fragment; Create Phase Method Fragment; Create Process Method Fragment. Fragments must be defined according to granularity layers such as activity, phase or iteration and process. Also, they must be standardized as proposed at [Casare et al, 2013], in order to provide reuse and combination with other fragments sourced from different methods.

An activity fragment is a set of one or more tasks, considered as the smaller part of the method that can be fragmented and reused. For example, the activity showed on Figure 4 has four tasks. In our case, no iteration was created. A phase fragment is a set of activity fragments and here phase fragments were mapped to the phases proposed in "ADELFE as it is" process. A process fragment is a combination of phases. The phases are grouped in the ADELFE Base Method, which is basically the entire process that describes the method and can be used as a fragment. The fragments are stored in an EPF plugin named B: ADELFE Method Fragment. Figure 4 shows the Medee method fragments sourced from ADELFE, detailing at the right side the MMF Requirements Description with ADELFE one.

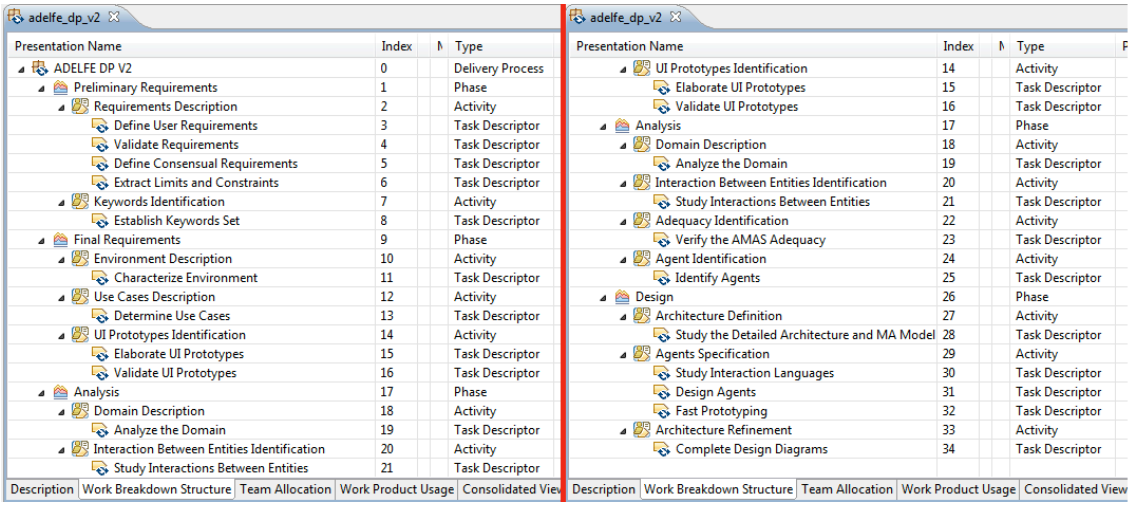

Figure 3: ADELFE method as it is. The red line indicates a junction; actually, it's a vertically continuous list.

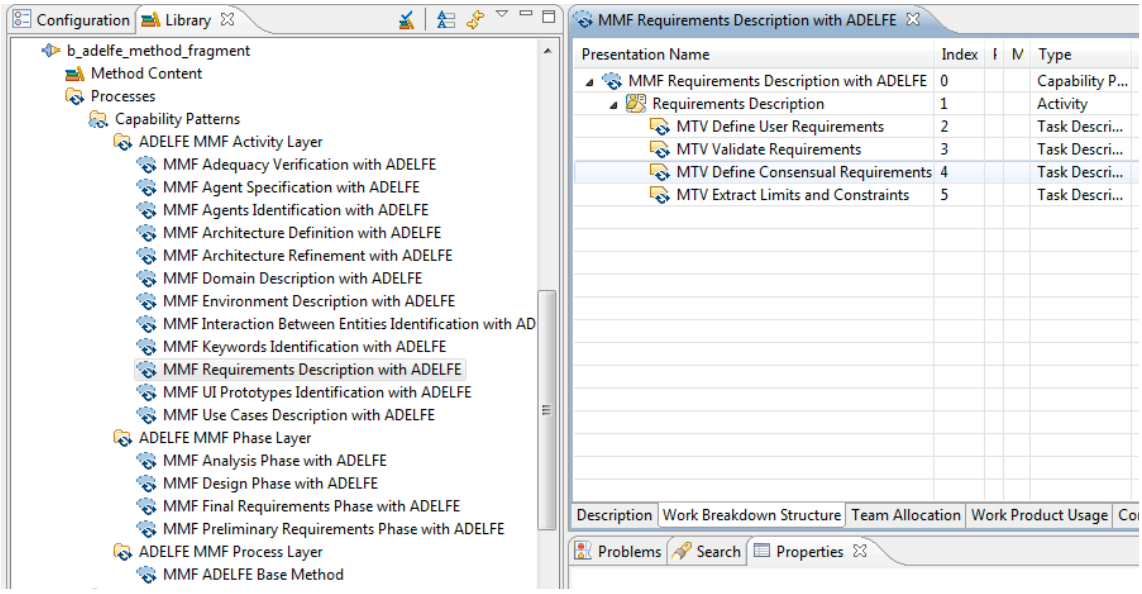

Figure 4: Overview of the fragments, organized in folders according to its granularity. On the right, details of an activity fragment, containing four task descriptors.

4. Conclusion

In this paper we describe the fragmentation of ADELFE following the Medee Delivery Process. The adoption of SPEM as the underlying technology to describe fragments facilitated the work while mapping existing description of ADELFE according to SPEM elements. Nevertheless, existing description didn't provide tool support for creating and storing fragments in CASE tools that implements SPEM, such as EPF Composer, in a sense that they adopts an extended version of SPEM that are not supported yet.

Seventeen new fragments were stored at the Medee Method Repository, increasing its population and diversity of sources in order to provide more flexible combination of fragments to create new situational methods to support MAS development.

Acknowledges

Thomas Liu de Almeida is partially supported by grant #2013-3326, Institucional CNPq. Anarosa A. F. Brandão is partially supported by grant #010/2640-5, São Paulo Research Foundation (FAPESP).

References

[ADELFE, 2003] ADELFE Process Description available at http://www.irit.fr/ADELFE/ENGLISH/AdelfeDescription.html. Accessed on 03/11/2014

[Bauer et al, 2001] Bauer, B.; Müller, J.; Odell, J.. Agent UML: A Formalism for Specifying Multiagent Software Systems. International Journal of Software Engineering and Knowledge Engineering, Vol. 11, No. 3, pp.1-24, 2001.

[Bernon et al, 2005] Bernon, C; Camps, V.; Gleizes, M.Picard, G.. Engineering Adaptive Multi-Agent Systems: The ADELFE Methodology. Published in B. Henderson-Sellers and P. Giorgini, Es, Agent Oriented Methodologies, pages 172-202, Idea Group Publishing, 2005.

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

[Casare et al, 2013] Casare, S. J. ; Brandão, Anarosa A.F. ; Guessoum, Z. ; Sichman, J.S. 2013. Medee Method Framework: a Situational Approach for Organization-Centered MAS. Autonomous Agents and Multi-Agent Systems (Dordrecht. Online), v. tbd, p. 1-44, 2013. (http://dx.doi.org/10.1007/s10458-013-9228-y).

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

[Cossentino and Seidita, 2005] Cossentino, M. ; Seidita, V. . SPEM Description of ADELFE Process. 2005 Rapporto Tecnico N: RT-ICAR-PA-05-07 available at: http://www.pa.icar.cnr.it/cossentino/paper/adelfe_spem_05-07.pdf

[Giorgini et al, 2004] Giorgini, P. et al. 2004. The Tropos Methodology. In: Bergenti, V; Gleizes, M. P.; Zambonelli, F.(Ed.), Methodologies and software engineering for agent systems, Kluwer Academic Publishers, pp. 89-106.

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

[Hubner et al, 2002] Hubner J., Sichman J., Boissier O. 2002. A model for the structural, functional, and deontic specification of organizations in multiagent systems. In: Bittencourt, G. & Ramalho, G. L. (Eds.), 16th Brazilian Symposium on AI, SBIA'02, LNAI 2507, Berlin: Springer, p. 118-128

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

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