Methodology vs. Development Process: A Case Study for AOSE

Alma Gómez-Rodríguez and Juan C. González-Moreno

Departamento de Informática, University of Vigo
Ed. Politécnico, Campus As Lagoas,
Ourense E-32004, Spain
{alma,jcmoreno}@uvigo.es
http://gwai.ei.uvigo.es/

Abstract. There is a general agreement in the fact that Agent Oriented Software Engineering (AOSE) needs development process definition for an accurate process management. The main trends in the field identify process and methodology in order to approach the process definition. This paper focusses in the idea that process and methodology must be considered independently. This means that not only the same process can be used for different methodologies but also that the same methodology can be used following different processes. The most suitable process can be selected by developers depending on several factors such as: human resources available, time restrictions, costs, etc. The previous approach is justified introducing a case study, which shows how different development processes can be applied while the team is following the same methodology (in particular, INGENIAS methodology).

Keywords: Multi-Agent Systems, Development Process, SPEM, Metamodel, AOSE Case Study.

1 Introduction

Agents and multiagent systems (MAS) have proved to be a powerful technology to face the complexity of a variety of Information Technology based systems. The construction of such systems within a Software Engineering (SE) perspective implies the use of methodologies which guide the developer along this process. To this end, a variety of methodologies to discipline and support the development process of a MAS have been defined in the past years [1,5,13,16]. All of them introduce the conceptual abstractions that must be taken into account in any MAS development.

Nowadays, in the field of quality assurance, one of the more relevant lines of work is the study and improvement of processes for software development and maintenance. The relevance of processes for quality assurance lies on the direct relation between process quality and final product quality. Moreover, in SE, the processes of development are fundamental when referring to cost and quality of products. Historically, the quality assurance has been tightly related with the definition of methodologies which guide the development team in the steps of development.

The agent-oriented software engineering processes proposed until now need a deeper formalization for an accurate process management. Moreover, FOSE-MAS [19] encourages the necessity of obtaining models of the processes for the development of MAS, that define its structural and behavioral issues. The anticipated definition of the processes is useful to obtain the right process for the development. Moreover, the definition of the process models is one of the basis for automating the development process, in the same way as it is done in other engineering fields, and opens the possibility of customizing CASE tools. This customization can be done at two levels: methodology and process. With methodology level, we mean the capacity of the tool to provide support for the concepts and models of a particular methodology. The process level refers to the ability of the tool for incorporating the development process and guide the user in what model to define and when it must be defined.

In this paper a new approach for process definition and usage is proposed. Instead of identifying methodology and process, this work is based on considering that the chosen methodology is independent from the development process. This means that the same methodology may follow different development processes.

Taking into account the previous hypothesis this paper shows with a practical example how a development team could benefit from using this approach, changing the underlying development process in a MAS project. In the example, the team must readapt the process due to a particular circumstance (arising time restrictions); but, there are other possibilities which can make the development team change the process of development.

The remainder of the paper is organized as follows. After this introduction, sections 2 and 3 address the theoretical background in the field of processes for AOSE methodologies. Next, these theoretical concepts are used for showing a simple example of application, where the development process of a system is changed dynamically and the system is constructed enhancing reutilization. Finally, in section 5 the conclusions and future work are introduced.

2 Processes versus Methodologies

There are many works in the field of AOSE which deal with methodology and process concepts. In the work [7], software development process is defined as *the coherent set of policies, organizational structures, technologies, procedures, and artifacts that are needed to conceive, develop, deploy, and maintain a software product.* Attending this definition in [3] a methodology is described as a collection of methods (way of preforming a particular activity) covering and connecting the different stages of a process.

The initial approaches in AOSE [6] try to compare methodologies applying a generic evaluation framework which considered the same aspects for all the methodologies under evaluation. The intended goal of these works was to identify methodologies' robustness and weakness and obtaining the "best" fragments.

Next evolution in the field considers the relationship between methodologies and processes. In AOSE, this issue has been addressed many times in literature,

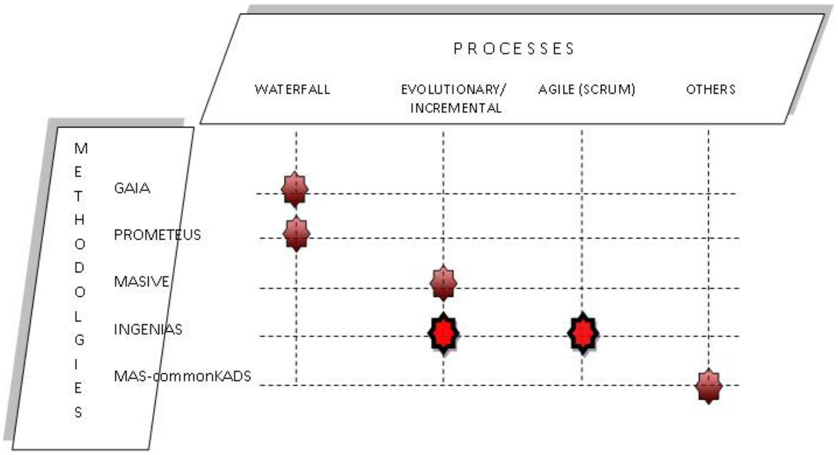

Fig. 1. Process and Methodology are considered independently in this work

but following different approaches. For instance, the work [12] tries to obtain a standard process for AOSE development. In this paper, the authors compare four process metamodels in order to obtain a global process of development with independence of the chosen methodology. Other works relate methodologies with a particular process model. This is the case of [3] which provides a survey of the most relevant AOSE methodologies and classifies them according to their process models. For this taxonomy, the paper relates directly each methodology with a type of development process. Another work that introduces a similar approach is [18]; in this case, processes and methodologies are described jointly.

In this paper, as it has been said before, the methodology and the process are considered independently. The underlying idea is that following the same methodology several development processes can be implemented. This idea is graphically addressed in Figure 1 where some methodologies are associated with their intended processes. The INGENIAS methodology is a particular case, because the process proposed for the methodology by its authors was Incremental; in particular OpenUp process; but, a previous work [8] has achieved the definition of a different processes for the INGENIAS methodology: the Scrum process. This latest definition has been done by identifying common tasks in the different development processes and reordering them to construct a new process. Moreover, it is thought that the process followed can be changed during development to readapt the development to new constraints. Next sections address how this change can be managed.

3 Process Definition and Formalization

Following the accepted concepts of process and methodology introduced in section 2, several process models considering the stages, tasks and artifacts that must be accomplished have been defined.

A Software Process Model (SPM) is defined as a description of structural and behavioral issues of a process, using as formalism for description a Process Modeling Language (PML) [2]. The definition of process models used in this paper is based on considering a software development process, at a high level of abstraction, as a simple dependency graph with three basic components: the process participants (*roles or workers*), the consumed and generated products (*work products*) and the *activities* and *tasks* achieved during the process, which constitute particular instances(*work definitions*) of the works that must be done.

A fundamental challenge in software process modeling is to find a standard PML for process definition. For MAS based process modeling, the *Foundation for Intelligent Physical Agents* (FIPA) through its *Methodology Technical Committee* [4] has suggested the use of Software Process Engineering Metamodel (SPEM) [14]. SPEM is a standard defined by the Object Management Group (OMG) to model software development processes. This standard is specified as a UML profile and currently has a stable version 2.0 that adapts the original proposal to the UML-2.1.2 standard [15].

There are other standards, such as ISO 12207, which has be described as the international standard for defining all the tasks required for developing and maintaining software. This standard is industry focused, so it has the main objective of supplying a common structure so that the people involved with the software development use a common language. This common language is established in the form of well defined processes. The structure of the standard was intended to be conceived in a flexible, modular and adaptable way and is based on two principles: modularity of processes (minimum coupling and maximum cohesion) and responsibility (that is, to establish a person responsible of each process). It distinguishes three types of processes: basic, for support and organizational. In addition, the set of processes, activities and tasks can be adapted to a particular software project.

Based in the concepts previously introduced, different processes used for an agent-oriented methodology have been formalized. In particular, two processes have been modeled for the INGENIAS methodology: the OpenUp Process and the Scrum process [9,8]). In these works, process modeling, like software modeling, is thought to present several orthogonal and complementary views. Each of these views provides a partial explanation of the whole process and allows its gradual definition. A detailed description of the steps proposed to map a well-established methodology/process into a new process can be found in these works; as well a description of how the OpenUp and Scrum processes have been defined following these steps. The standard selected for definition was SPEM, as it is the recommended by FIPA group and has proved its suitability for this kind of systems.

4 A Practical Example

There are several advantages in considering the process of development independently from the methodology. One of them is that the process can be selected for each development, although the same methodology is used. Other important advantage relies on the possibility of changing the process during the development.

This latest case is the one addressed in this paper. To understand properly what we mean, let's see the following example. A team of development must obtain a system for accessing through the Internet a corpus containing Spanish illustrated emblems. The main objective of the system is to provide global access to that kind of data, which, due to its scarcity and dispersion, is rarely available. In addition, the system will supply access to the studies related to a particular emblem, providing in this way a complete view of the information available about it.

For developing the system previously described, the team is following the INGENIAS methodology and its intended process of development, which is an adaptation of OpenUP. Following this process (formally modeled in [9]) and using the IAF framework as supporting tool, they achieve a certain degree of development. Although the system is simple the team spend too much time in defining system requirements, and obtaining the initial models of the system. Due to this delay, the project starts to diverge from the initial scheduling and the team has to find a solution.

In traditional approaches of development, the process of development will remain the same and the solution provided for this deviation would be to accept the deviation (what means accepting a delay in delivering the system) or to provide more resources to the team (more people, more time, in brief, assuming more costs). In Agile processes of development, the team may accept doing more iterations, what in the end will imply more time and more costs for obtaining the final system. The approach presented in this work tries to overcome that kind of problems changing dynamically the process of development.

At the moment, two different process have been defined for INGENIAS methodology: OpenUp and Scrum. So the team may change from the process initially selected to Scrum , trying to overcome the problems of delay it is facing. The agile characteristics of Scrum, jointly with the reuse of previous developments may result in a reduction of the development time.

In SCRUM, the production of a software version (release) is usually done in a couple of months. Each release is produced within a number of iterations from 2 to 4 weeks called Sprints. At the end of each Sprint, the team produces a product increment which is potentially releasable. All the work is done in two basic phases: the *Preparation Phase* and the *Sprint Phases.*

The development team will change the scheduling of the project according to Scrum and will accomplish the steps and the activities and tasks formally defined in [8]. Roughly speaking this would imply to:

– Define the Sprints.
– For each sprint: Plan Sprint,Update Product Backlog, Daily Works, Manage Problems, Conduct SCRUM Daily Meeting, Review Sprint, and Conduct Retrospective.
– Produce release, which is a functional software.

In addition, for the sake of quickness, the team must identify previous developments that can be reused as basis for each sprint; producing in this way a rapid

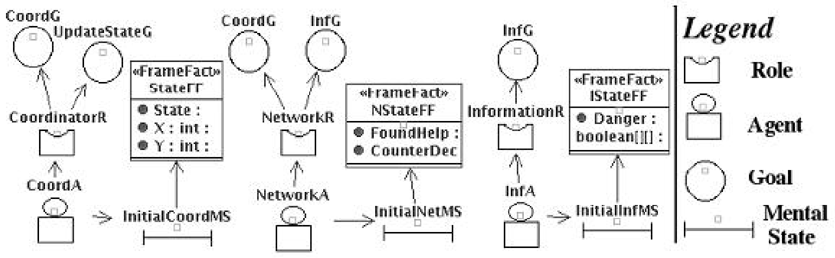

Fig. 2. Role and Agent Definitions of the crisis MAS (Diagram taken from [10])

release of software. There are several MAS that could be rapidly constructed with INGENIAS by reusing previous developments. Recently, the INGENIAS Agent Framework (IAF) [11] for JADE has been proposed and documented as a successful approach in this context.

In the example treated in this section, the team could reuse the crisis management system introduced in [10]. This system tries to manage situations of crisis involving groups of people and is one of the habitual case studies proposed in the Artificial Intelligence community. In order to explain how reutilization can be done, we will choose as example the Role and Agent Diagram taken from [10], that is shown in Figure 2. This diagram introduces the three roles that are assigned to three different types of agents: coordination-agents, network-agents and information-agents.

Although the diagram of Figure 2 reflects the Roles and Agents of a completely different system in a different domain, it can be modified and reused for the system of the example. In this particular diagram *Information Agent* can be reused for modeling the agent which will coordinate the access to the different databases of emblematic information. This agent will organize the results obtained from the different databases in order to provide results and show them.

The *Network Agent* will have exactly the same meaning and roles in both systems, providing communication between user and the Information Agent. This role isolates the potential communication problems that may occur in an environment such as the Internet.

Finally, the *Coordinator Agent* will provide access to the system to different users, attending their queries and showing the obtained results.

Once the different kinds of agent have been identified and reused, the rest of the diagrams containing the components that have been reused from the Role and Agent Model are easier to adapt. The interaction models, tasks models or even the implementation will have minor modifications, because many of the interactions or functionalities will be coincident with the ones of the original model.

5 Conclusions and Future Work

Most times a methodology proposes a particular development process in its description. However, this paper tries to show how a system can be constructed following different processes along its development, and adapting the process to the restrictions the team must face.

Methodologies have usually a supporting tool which provides facilities for creating the different models; for instance, INGENIAS provides de IAF [11]. Nevertheless, in general, the tool does not provide support to development process. At the moment, the group is working in the construction of a tool (for INGENIAS methodology) that makes easier the SMA development. This tool will simplify the construction of a MAS using the INGENIAS-IAF and following different processes of development (at the moment only Scrum and OpenUp). The tool will allow collaborative work among users using as invariant the INGENIAS models [17]. In addition, the developer will choose the process to follow and can assign different roles to the users involved in the development. This selection will fix the tasks each user must fulfill and to which entities and models he/she will have access.

Although all the concepts introduced in this paper are focussed on the AOSE field, we consider that the conclusions obtained can be generalized for Software Engineering with independence of the kind of system. This affirmation can be justified by the fact that no restrictions are considered in the paper about the system to construct. The AOSE approach is used only for illustrating the problem with an example.

In the future, the work done in this paper must be extended to other methodologies and processes. The results obtained from this study can assist the MAS designer in selecting the appropriate methodology and process model for a specific MAS, team of development or other particular circumstances.

Other important feature than can be addressed is to divide the model of the process in pieces, called fragments. Fragments from different processes can, after, be integrated to define a new process. This approach is one of the current lines of investigation in the field [12]. The definition of fragments can facilitate also the changes in process during the development, by providing pieces of process that must be accomplished completely.

Acknowledgements. This work has been supported by the project *Methods and tools for agent-based modeling* supported by Spanish Council for Science and Technology with grant TIN2005-08501-C03-03 co-financed with FEDER funds.

References

1. Bernon, C., Cossentino, M., Pavón, J.: Agent-oriented software engineering. Knowl. Eng. Rev. 20(2), 99–116 (2005)
2. Breton, E., Bezivin, J.: Model driven process engineering. In: Int. Computer Software and Applications Conf (COMPSAC 2001), Chicago, pp. 225–230 (2001)
3. Cernuzzi, L., Cossentino, M., Zambonelli, F.: Process models for agent-based development. Engineering Applications of Artificial Intelligence 18(2), 205–222 (2005)
4. Cossentino, M., Garro, A.: Activity of the FIPA Methodology Technical Committee. Technical report, Consiglio Nazionale delle Ricerche (2005)

5. Cuesta, P., Gómez, A., González, J., Rodríguez, F.J.: The MESMA methodology for agent-oriented software engineering. In: Proceedings of First Int.Workshop on Practical Applications of Agents and Multiagent Systems (IWPAAMS 2002), pp. 87–98 (2002)
6. Cuesta, P., Gómez, A., González, J.C., Rodríguez, F.J.: Evaluating agent oriented software engineering to propose MESMA. In: Proceedings of the 3rd Int. Workshop on Practical Applications of Agents and Multi-Agent Systems (IWPAAMS 2004), pp. 103–114 (2004)
7. Fuggetta, A.: Software process: a roadmap. In: ICSE 2000: Proceedings of the Conf. on The Future of Software Engineering, pp. 25–34. ACM, New York (2000)
8. García-Magariño, I., Gómez-Rodríguez, A., Gómez-Sanz, J., González-Moreno, J.C.: INGENIAS-SCRUM Development Process for Multi-Agent Development. In: International Symposium on Distributed Computing and Artificial Intelligence (DCAI 2008), Advances in Software Computing (2008)
9. García-Magariño, I., Gómez-Rodríguez, A., González, J.C.: Definition of Process Models for Agent-based Development. In: 9th International Workshop on AOSE, Lisbon, Portugal. Springer, Heidelberg (2008)
10. García-Magariño, I., Gutiérrez, C., Fuentes-Fernández, R.: Organizing multi-agent systems for crisis management. In: Antunes, L.M.L., Pavón, J. (eds.) 7th Ibero-American Workshop in Multi-Agent, Lisbon, Portugal, pp. 69–80. Springer, Heidelberg (2008)
11. Gómez-Sanz, J.: Ingenias Agent Framework. Development Guide V. 1.0. Technical report, Universidad Complutense de Madrid (2008)
12. Henderson-Sellers, B., Gonzalez-Perez, C.: A comparison of four process metamodels and the creation of a new generic standard. Information and Software Technology 47(1), 49–65 (2005)
13. Mas, A.: Agentes Software y Sistemas Multi-Agentes. Pearson Prentice Hall, London (2004)
14. O. M. G. OMG. Software Process Engineering Metamodel Specification. Version 1.1, formal/05-01-06 (2005), http://www.omg.org/
15. O. M. G. OMG. Unified Modeling Language (UML). Version 2.1.2, formal/2007-11-04 (2007) http://www.omg.org/
16. Pavón, J., Gómez-Sanz, J.: Agent Oriented Software Engineering with INGENIAS. In: Mařík, V., Müller, J.P., Pěchouček, M. (eds.) CEEMAS 2003. LNCS, vol. 2691, pp. 394–403. Springer, Heidelberg (2003)
17. Pavón, J., Gómez-Sanz, J.J., Fuentes-Fernández, R.: The INGENIAS Methodology and Tools, article IX, pp. 236–276. Idea Group Publishing (2005)
18. Penserini, L., Perini, A., Susi, A., Mylopoulos, J.: High variability design for software agents: Extending Tropos. ACM Transactions on Autonomous and Adaptative Systems 2(4), 16–27 (2007)
19. Weyns, D.: The Future of Software Engineering and Multiagent Systems. Held at AAMAS 2008 (2008)