Enhancement of Development Technologies for Agent-Based Software Engineering

Andre Karpištšenko

Tallinn Technical University, Ehitajate tee 5
19086 Tallinn, Estonia
andre@lap.ee

Abstract. Current trends in software development show a move towards supporting autonomous components (agents). The accurate timing of interactions between such components is growing in importance. However, the modeling tools provide insufficient support for effective specification and implementation of such systems. This paper gives an overview of a doctoral work, where an effort is made to integrate the existing and emerging tools for providing a development platform for time-aware multi-agent systems. As the work is in its early stages, this paper presents an overview of the preliminary and expected results.

1 Background

Modern systems are software-intensive and must meet increasingly demanding requirements in order to cope with significant degrees of uncertainty, dynamic environments, and to provide greater flexibility. Agent-based development metaphors and technologies provide well suited structures and techniques to handle such complexity [1]. However, when developing complex open systems, the tools provide insufficient support for effective specification and implementation of such systems. At the same time, software projects involve many partners, where greater attention to specification and collaboration is critical to success [2].

Elaborating the agent concept, the inherent complexity of a large system means that it is impossible to know *a priori* about all the potential links between the multiple agents: interactions will occur at unpredictable times, for unpredictable reasons, between unpredictable components [1]. Components need to have the ability to initiate (and respond to) interactions in a flexible and timely manner.

The paradigmatic model of computation (i.e., Turing Machine model) is too weak to describe areas like multi-agent interactive systems, web-distributed programming and distributed databases [3]. The evolution of computer science has gradually reached the understanding that networks are better explained by means of interactive computations [4], and by the extended notion of algorithm [5]. Intuitively it is believed that a suitable underlying model should be the interaction-centered model of computation [6]. Unfortunately, the appropriate and widely accepted formalism for such a model is not available; there are some experimental models and a set of concepts.

With the official release of UML 2.0, the fundamental definition of behavior will offer means for modeling interactions and for specifying systems independently of where the system/environment boundary is drawn [7]. This would suggest that the utilization of Model Driven Architecture (MDA) offers a content platform for modeling multi-agent systems. However, as critics of MDA suggest, it is distinctly nontrivial – many would argue impossible – to support and evolve semantically correct platform-specific models (PSM) for complex platforms such as J2EE or .NET [8]. Drawing the parallel with inherently complex multi-agent systems, additional means of modeling have to be considered.

Ongoing research, carried out by a group of enthusiasts at Tallinn University of Technology (TTU) and University of Tartu (UT), is focused on developing time-aware interaction-based models of computing that are closely related to the long-term goal of building multi-agent systems and formal analysis methods usable at early stages of software development (e.g. model processor for analyzing subtle behavioral properties) [6].

2 Purpose and Possible Impacts

The purpose of this doctoral work is to study the software development process in order to modify relevant steps in that process so as to foster support for software implementation of time-aware multi-agent systems (TA-MAS). The focus is on the effective modeling practices in the early stages of software development.

2.1 Problem Solving Approach

This doctoral work is aiming at uniting the existing and emerging tools in the domain of modeling TA-MAS.

In the context of Model Driven Development (MDD), framework called MDA is considered as a starting point. Significantly Real-Time UML profile (RT-UML) provides the conceptual foundations needed for *quantitative* modeling of the time-related characteristics of systems. The emerging Agent UML (AUML) is considered as a reasonable starting point for modeling agents [9]. As an alternative for completely UML-based modeling, the study of a meta-programming based environment [10] is at its early stages. The work will also benefit from analysis of existing agent-oriented methodologies like TROPOS [11].

In addition to the MDD approach (that focuses on modeling the product), the Object-oriented Process, Environment, and Notation [12] (OPEN) approach (that focuses on the development process) will be studied at later stages, for possible improvement ideas. OPEN provides a process meta-model, which enables focus on agile software development methodologies that are suitable for small and medium enterprises.

In general, the problem is addressed in three basic steps:

1. Studying the domain of agent-oriented development, UML profiles (Real-time; Agent) and MDD in order to identify requirements and possibilities for interaction-based models. The study is based on literature and experimentation with KRATT [6], the agent generator developed by joint research between TTU and UT, and JADE [13] combined with AUML prototype Ingenias [14].
2. Generating TA-MAS implementations of complex systems specified in UML or alternative languages like Meta Programming System (MPS) [10], in order to study the constraints and concrete possibilities by using samples.
3. Develop mechanisms for verifying and controlling the agent interactions in system's implementation against the model and specified constraints. The results will be evaluated by comparing the results of the models to the real behavior of systems implemented in ad hoc sensor network environment.

2.2 Possible Impacts

In terms of software engineering, the results could provide means for modeling efficient collaboration tools for intra-organizational software projects. From the perspective of agent-orientation, the work could create possibilities to automatically control and verify agent-oriented implementations against their model specifications. This would facilitate the engineering of complex systems, as the problems associated with the coupling and interaction of components are significantly reduced and the problem of managing control relationships among the software components is reduced.

The expected results of this research would improve the understanding of contemporary software process that should match the requirements of new computing systems (e.g. ubiquitous computing, proactive and autonomic computing, mobile and pervasive computing). In particular, the impact is expected on software modeling and formal analysis methods resulting from integration of RT-UML and AUML. This thesis is expected to contribute to the behavioral modeling aspects – elaborate the list of behavioral features that can be analyzed in early stages of the software process, and methods applicable for this analysis [15].

3 Preliminary Results

Due to the early stage of the doctoral work, the majority of results are "work in progress". For this reason this paragraph is organized as an overview of the relevant topics with a glimpse of the ongoing work.

3.1 Modeling Time-Aware Multi-Agent Systems with UML

The existing trend in practical applications has led to (partial) merging of two domains – multi-agents and real-time embedded systems [16]. For modeling real-time systems there is a widely acknowledged RT-UML. However when it comes to modeling agents, the formalizations and standards are still evolving. A well known modeling language is AUML which heavily relies on OMG's UML [9].

As RT-UML puts a lot of attention to concurrency and communication issues, which are inherent characteristics of multi-agent systems, the doctoral work is on merging the relevant parts with the existing AUML. As the authors of AUML (FIPA modeling TC) have stated, that UML is not relied upon, but reused [9], open-source AUML tools are preferred for ease of modification.

When it comes to modeling agents, special attention must be paid to behavioral aspects. For example, the sequence diagram allows one to make explicit the messages exchanged, in addition temporal dimension can be introduced intuitively for modeling temporal interaction. Integration with RT-UML is required, as current timing constraints and the underlying time model, defined in AUML do not satisfy the needs of real-time systems.

The first group of prototype models will include a model of an ant colony simulation [17] and in the near future models will be developed for agents collecting information (e.g. for digital maps) in heterogeneous computing network with dynamically changing topology (e.g. ad hoc sensor networks, currently such an experimental environment is being built in TTU).

3.2 Meta-programming of Domain-Specific Agents

Tools developed for MDA provide means for coping with complexity by means of abstraction – allowing transformation from Business Model (CIM) to Platform Independent Model (PIM) and then to PSM. However, as discussed in the background section of this work, UML based models of TA-MAS might prove inefficient in practice. Since attention to methods, tools, and models that directly support domain-oriented programming is crucial [18], a study of MPS is planned in parallel with the work on UML profiles.

MPS is a research project developed by Jetbrains, offering programming environment for any domain-specific language with full support from the IDE [10]. Similarly to MDA, MPS allows to raise the level of abstraction by supporting definition of domain-specific languages which in turn have mappings to the target format (e.g. Java). The variety of languages and formalizations required for designing multi-agent systems [19] suggest that seamless interoperability on source code level (through a common base language) is a must for successful code generation and the maintenance of TA-MAS.

As the MPS project is currently in a research phase and working prototype of the system was made available in late May 2005, no tangible results can be presented here. However, further research within this doctoral work is expected to result in a prototype language for TA-MAS with possible candidates for target format (code generation) being KRATT or JADE. Close integration with work on UML profiles could result in MPS being MetaObject Facility (MOF) compliant allowing interoperability with MDA tools and thus with the enhancements in the profiles.

4 Conclusion

This thesis is to fill the gap between the existing techniques in software engineering and the actually required characteristics of agent-based software developing, with a focus on time-sensitive systems. The explicit contributions of the doctoral work are expected to be: the integration of RT-UML and AUML profiles for TA-MAS modeling; prototype language based on MPS for code generation of TA-MAS.

References

1. Jennings, N.R., Bussmann, S.: Agent-Based Control Systems. Why are they suited to engineering complex systems? IEEE Control Systems Magazine, vol23, (2003) 61-73
2. Light, M.: Web Sources and Quality vs. Time in the Age of SODA. (2003) Gartner Research
3. Wegner, P.,Eberbach, E.: New Models of Computation. The Computer Journal, vol47 (2004)
4. Goldin, D.Q., Smolka, S.A., Wegner, P.: Turing Machines, Transition Systems, and Interactions. Electronic Notes in Theoretical Computer Science vol. 52(1), www.elsevier.nl/locate/entcs
5. Blass, A., Gurevich, Y.: Algorithms: A quest for absolute definitions. Bulletin of European Association for Theoretical Computer Science, no. 81, (Oct 2003) 195-225
6. Motus, L., Meriste, M., Kelder, T, Helekivi, J.: Agent-based Templates for Implementing Proactive Real-time Systems. Proc. International Conference on Computing, Communications and Control Technologies, Austin, Texas, ISBN: 980-6560-17-5, (2004) 199-204
7. Bock, C.: UML 2 Activity and Action Models. Journal of Object Technology, vol. 2. no. 4. (Jul-Aug 2003) 43-53 http://www.jot.fm/issues/issue_2003_07/column3
8. Thomas, D.: MDA: Revenge of the Modelers or UML Utopia? IEEE Software, May/June. (2004) 22-24
9. FIPA Modeling TC: Agent UML Web Site. www.auml.org
10. JetBrains: Meta Programming System Project. www.jetbrains.com/mps
11. Tropos: Requirements-Driven Development for Agent Software. www.troposproject.org/
12. OPEN, Object-oriented Process, Environment and Notation. www.open.org.au/
13. Telecom Italia Lab: Jade – Java Agent DEvelopment Framework. jade.tilab.com/
14. Grasia: Ingenias IDE. ingenias.sourceforge.net/
15. Selic, B., Motus, L.: Using models in real-time software design, IEEE Control Systems Magazine, vol.23, no.3, (June 2003) 31-42
16. Motus, L., Meriste, M., Dosch, W.: Time-awareness and Proactivity in Models of Interactive Computation. ETAPS – Workshop on the Foundations of Interactive Computation. Edinburgh, Scotland, (April 2005). www.elsevier.nl/locate/entcs
17. Kimlaychuk, V.: Creating intelligent agents in JADE (an example of ant colony simulation). In Proc. of Int. Conf. on Education and Information Systems: Technologies and Applications, EISTA 2004 (Orlando, FL, July 2004), v. 1, IIIS (2004), 55-60
18. Thomas, D., Barry, B.M.: Model Driven Development: The Case for Domain Oriented Programming. Companion of the 18th Annual ACM SIGPLAN Conf. Object-Oriented Programming, Systems, Languages, and Applications, ACM Press (2003) 2–7
19. Ferber, J.: Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence. Addison-Wesley (1999) 22-24