
The Future of Software Engineering and Multi-Agent Systems

**Danny Weyns¹, H. Van Dyke Parunak²,
Onn Shehory³**

¹DistriNet Labs, Department of Computer Science,
Katholieke Universiteit Leuven, Belgium
E-mail: danny.weyns@cs.kuleuven.be

²NewVectors, a Division of TechTeam Government Solutions, Inc.
Ann Arbor, USA
E-mail: van.parunak@newvectors.net

³IBM Haifa Research Lab
Haifa, Israel
E-mail: onn@il.ibm.com

Abstract:

In spite of the massive research efforts by the multi-agent system community, the state of the art in multi-agent systems is insufficiently reflected in state of the practice of complex distributed systems. Triggered by this observation, leading researchers and engineers in the field came together at the AAMAS 2008 conference to discuss the future of software engineering and multi-agent systems. Technical and organizational obstacles were identified that hamper industrial adoption of multi-agent systems. Prominent obstacles include poor awareness of industrial needs, disconnection from conventional software engineering, immaturity of technology, and a research bias towards scientific challenges. To address these obstacles several opportunities were identified. Organizational opportunities include, among others, improving the communication with conventional software engineers and industrial clients. On the technical side, several topics were proposed to facilitate industrial adoption of multi-agent systems. Topics such as goal-oriented design, architectural patterns, and validation and verification, can be developed as research areas in the field of multi-agent systems.

Keywords: multi-agent systems, software engineering

Reference to this paper should be made as follows: D. Weyns, H. V. D. Parunak, and O. Shehory (2008) 'The future of software engineering and multi-agent systems', Special issue on Future of Software Engineering and Multi-Agent Systems, International Journal of Agent-Oriented Software Engineering (IJAOSE).

Biographical Notes: Danny Weyns is a post-doctoral researcher at the Katholieke Universiteit of Leuven. He obtained a PhD in computer science in 2006 for research on the connection between multi-agent systems and software architecture. Danny's main research interests are in software architecture and middleware for self-managing systems and decentralized systems.



H. Van Dyke Parunak is Chief Scientist at NewVectors, where he leads basic and applied research projects in applications of stigmergic coordination to a wide array of practical problems, including robotic control, prediction, and information exploitation.

Onn Shehory is a researcher at the IBM Haifa Research Lab, where he leads an international research project on software engineering of autonomic systems. His research span across theory and practice in multiple disciplines, including multi-agent systems, software engineering, system management, game theory and AI.


1 INTRODUCTION

Since the early 1990s, multi-agent system researchers have developed a large body of knowledge on the foundations and engineering principles for designing and developing complex distributed systems. In spite of the enormous research efforts and a number of successful industrial applications, the state of the art in multi-agent system research is insufficiently reflected in state of the practice of complex distributed systems. This observation was the incentive to bring together the leading researchers and engineers in software engineering of multi-agent systems in a session dedicated to the future of software engineering and multi-agent systems (FOSE-MAS). The session was organized at the Seventh International Conference on Autonomous Agents and multi-agent Systems 2008 [1]. More than 100 researchers and engineers participated in a lively discussion. This article summarizes the outcome of the discussion.

This article is structured as follows. In the next section, we give some additional explanation of the scope of the FOSE-MAS discussion. Section 3 elaborates on a number of obstacles that hamper industrial penetration of multi-agent systems. In section 4, we present opportunities that were identified for the future of software engineering and multi-agent system. Finally, section 5 wraps up and draws conclusions.

2 SCOPE

The general objective of FOSE-MAS was to reflect on the state of the art in software engineering and multi-agent systems and explore key directions for future research in the field. To steer the discussion, the following set of guiding questions was proposed:

- What are the main aspects that hamper progress in software engineering and multi-agent systems?
 - Why is state-of-the-art in multi-agent systems research and engineering insufficiently reflected in state-of-the-practice in complex distributed systems?
 - What is the future for agent-oriented methodologies?
 - What are the strong and weak points of state-of-the-art agent programming languages?
- 

- What makes software engineering of multi-agent systems different from mainstream software engineering?
- What are the key research challenges for software engineering and multi-agent systems?
- What is it that we have to do to promote industrial adoption of agent-oriented software engineering? Can we do that, and how?
- What actions are required to advance research in software engineering and multi-agent systems?

Starting from these questions four authors were invited to produce position statements. These statements, together with a selection of submitted position statements from the community served as input for the discussion. A selection of revised position statements is included in this special issue.

3 OBSTACLES

During the discussion, several obstacles were identified that hamper industrial adoption of agent-based technology. We make a distinction between organizational and technical obstacles. Although technical aspects may seem more essential to allow technology adoption, it is clear the organizational aspects are very important too. Further, in the case of an emerging technology such as agent technology, the organizational obstacles may by themselves generate, or affect, the technical obstacles. Therefore, we present them first.

3.1 Organizational Obstacles

Aspects such as community objectives, technology misperception, limited information exchange between communities, and other, affect the adoption of agent technology by the industry. We refer to some of the prominent obstacles below.

Different Objectives. The academic community in general has different objectives from those of industry. The agent community is not an exception. Specifically, scientists are interested in studying challenging, unsolved problems and then publishing the solutions arrived at, advancing the state of the art. The commercial value of such solutions is usually not a factor. In many cases, robustness of solutions and ease of implementation and use are not considered either. Industry has different objectives: new, innovative solutions of challenging problems are important, but only if, at the end of the day, they generate profits. Further, ease of use and implementation are of great importance and have a major effect on the commercial value. As a result of objective dissonance (at least in part), academic researcher are, rather often, not aware of industrial needs and limitations. Consequently, their research results do not address well industrial needs.

Lack of Focus and Misperception. The multi-agent research community is rather diverse. Members of the agent community bring in multiple backgrounds and expertise. One consequence of this diversity is an apparent difficulty to focus the community on a small set of goals. This in turn hampers the delivery of practical, industrially applicable results, particularly software deliverables. The latter are essential for technology adoption.



Agent technology, being associated with AI, is perceived by many practitioners as complex, difficult to understand, and inefficient for practical needs. Whereas for some agent technologies this is indeed the case (as discussed in this issue), this perception is not always based on facts. Although the value proposition for agent technology is sometimes complex compared to alternatives (e.g., services), the multi-agent community has a myriad of technology offerings, many of which could be deployed at an acceptable level of effort. However, industrial (mis)perception regarding "heavy" AI techniques interferes with technology adoption.

Industrial solutions are aimed at providing the customer with some function, at some level of quality. As long as the functional and non-functional goals meet the specifications, it does not matter much what the underlying technology is. This calls for offering agent technology based on functional advantages it provides. However, as it stands now, agent technology is usually promoted focusing not on function but on the conceptual change it brings about. This is of course counterproductive to technology adoption. Example applications and use cases, which could provide for functional demonstration, are usually not present.

Poor Connection with Mainstream Software Engineering. The connection and information exchange of the agent research community with the conventional software engineering community is very limited. Research in multi-agent systems classifies itself as an isolated community. This may in turn form artificial boundaries, adding difficulty in convincing mainstream software developers of the agent approach merits. A clear evidence of the poor connection with mainstream software engineering is the lack of citations of agent research in the general software engineering literature.

Lack of Industrial Involvement. Given the above obstacles, and the resulting industry view of agent technology, commercial support of agent technology declines over time. Having difficulty to identify potential profits from the use of agent technology, industrial organizations avoid investments in this technology. Further, without commercial interest, even some research funders reduce their support of the field. This of course further influences the adoption of agent technologies.

Following the limited enthusiasm of industry with agent technologies, there is a very limited industrial involvement in agent-related activities. For instance, participation in agent technology training is scarce. Definition of cases for developing agent-based applications is also not common. Weak industry involvement and feedback prevent development of agent technologies to fit industrial needs. Without the involvement of domain experts from industry, researchers end up solving the wrong, or irrelevant, problems.

3.2 *Technical Obstacles*

It is evident that, although agent research, and in particular agent oriented software engineering, have advanced significantly in recent years, several technological barriers still exist.

Immaturity of Technology. The first and most prominent obstacle is immaturity of agent technology. The agents research community has produced multiple languages, methods and tools for engineering agent-based systems. Although for research purposes diversity is bliss, it hampers industrial adoption. In particular, lack of a common understanding of key multi-agent concepts, a common set of notations and models, and flexible, industrial

strength methods and techniques for developing multi-agent systems, has a major negative effect on adoption. Additionally, many agent-oriented methodologies are proposed without providing a solid underlying technical layer. In such cases, even if the methodology is of excellent quality, practical adoption is unlikely. Another facet of diversity is the use of non-standard agent-specific terms. These may be very useful for describing agent-based systems, but their translation to mainstream software engineering terms is non-obvious.

Lack of Integration with General-Purpose Technologies. Another obstacle to industrial adoption is the lack of (support for) integration of multi-agent systems with other, general-purpose technologies. Industrial systems are complex, diverse, and commonly integrated from multiple components. Thus, when introducing agent-based solutions into such systems, it is necessary to integrate with existing software environments such as legacy systems, frameworks, services, etc. It is however difficult to perform such integration and fully maintain agents' autonomy behavior encapsulation. Hence, additional research is needed to infer best practices for performing this integration.

Research Bias towards Scientific Challenges. Yet another issue hindering industrial adoption is the bias of scientific research towards problems of scientific challenge. Solutions of these are not necessarily of simple or direct practical use. Additionally, multi-agent research tends to adopt scientific models and languages which are not always known to, or understood by, software developers (e.g., the use of Formal Logic). Thus, the state-of-the-art in multi-agent research and engineering is not always directly applicable, and thus not well reflected, in state-of-the-practice in complex distributed systems. Clearly, the state-of-the-practice in software engineering consists of useful and easily comprehensible concepts and tools. The multi-agent research community commonly delivers rather complex concepts and tools, not always accessible to the typical software designer and developer. Thus, although useful, the agent concepts end up being severely under-represented.

Limited Scope of Programming Languages. In addition to concepts and tools, software developers are in need of adequate programming languages, without which technology adoption is practically infeasible. Although agent-oriented programming is widely studied, practical work on the software engineering aspects of agent programming languages is limited in scope. That is, scientific research on agent-oriented programming usually does not address programming language design issues. It further does not address the specification, realization and assessment of actual prototype programming languages.

4 OPPORTUNITIES

Our discussion identified several opportunities for addressing these obstacles. Again, we summarize them under the main categories of organizational vs. technical.

4.1 Organizational Opportunities

Just as many of challenges are organizational in nature, progress will depend on organizational actions. Our discussion identified several of these.

Better Communication to Industrial Clients. Rather than trying to sell "agent-oriented software" as a monolithic package, the community needs to focus on the key ideas that agents enable (including modularity, local modifiability and quality control, and "bottom-



up” optimization of operations). A basic pedagogical principle is that “you can’t teach someone something that they don’t almost know already.” In this spirit, the agent community should focus on incremental introduction of its ideas to industry. We should emphasize our commonalities with current software practice, rather than only the distinctions. It is essential to build on conventional methods and terminologies, and to translate concepts and techniques from multi-agent systems into the framework used in conventional software engineering and systems development environments. Successful deployment of a single agent (perhaps an enhancement of an existing service) to handle some dirty, dull, or dangerous job will open the door for others. Above all, the agents community must better formulate and communicate its value proposition. First, it must appreciate the shortcomings in current software engineering methods from the customer’s perspective. Then it must articulate how agents address these challenges. One key area to emphasize is the increased flexibility and agility that agents can offer in managing complexity and dealing with changing requirements and demands. Ultimately, industry cares about ease of use, training, and lower development costs, such as those resulting from increasingly mobile computing. Demonstration projects (such as those referenced in a number of position papers, elsewhere in this issue) can increase the confidence of potential adopters that the technology can deliver improvements, and that the improvements are really due to the technology.

Better Communication to Conventional Software Engineering Community. In addition to communicating more effectively with industrial users, the agent community must do a better job of communicating with the conventional software-engineering community on which industry does rely. Agent researchers could reach out to the broader software community by presenting their work in software-engineering venues (and paying attention to the peer reviews as a valuable source of insight into how to integrate agents into the mainstream software engineering perspective). There are numerous promising points of contact, including the natural alignment of services in a service-oriented architecture with agents. Promoting agent workshops at software conferences and establishing liaisons with software engineering groups will be crucial steps, adumbrated by Jim Odell’s success in introducing agents to the Object Management Group.

Personnel-based Technology Transfer. The research community often labors under the misconception that the basis of successful technology transfer is publication of a landmark paper. Industry is often far more interested in hiring a landmark Ph.D. student, who has internalized the technology and can introduce it to a company while learning first-hand the company’s needs and priorities. Current Ph.D. programs are overwhelmingly slanted toward academic careers, often producing a surfeit of potential professors. Recognition by the academy that many of its students will in fact pursue industrial careers, and preparation of them for such work by exposure to elements of business education in the course of their technical training, could greatly facilitate this promising mode of technology transfer.

Standards Promotion. The agents community has launched numerous efforts at standards development, motivated by idealistic visions of system integration. Historically, standardization rooted in the research community (e.g., KIFF with DARPA support) has had much less influence than standards that originate from industry (e.g., UML). Researchers are naturally driven by creativity and diversity, but to the degree that they need standards to get their work done, they could enhance the chances for transition by relying as much as possible on industrially-based standards.



4.2 Technical Opportunities

On the technical side, adoption of agent technology will be facilitated if the research community will broaden its agenda to increase questions of interest to the industrial community. Several topics could be developed as research areas within the agent community.

Goal-oriented Design. A major business need is adaptability. Goals are what make agents adaptable, by capturing the objective the agent is to pursue rather than the specific process it is to follow in pursuing it. Focusing on specifying behavior in terms of goals and developing ways to prosecute them persistently and flexibly can yield results of direct interest to industrial users.

Software Architecture and Patterns. Multi-agent system architectures are known for addressing quality attributes such as adaptability, robustness, openness and scalability. In conventional software engineering, common architectural knowledge is documented by means of architectural patterns or styles. Architectural patterns exhibit particular quality attributes providing recurring solutions to architectural problems. Research on architectural patterns and styles for multi-agent systems is crucial to document and mature knowledge and practices with multi-agent systems. Pinpointing the quality attributes the patterns embody will delineate a convincing motivation for an architect to adopt a particular multi-agent system approach.

Theoretical Refinement. There is scope for theoretical refinement in distinguishing among passive representational elements (such as objects and components), non-autonomous reactive components (such as services), and autonomous reactive components (agents). An overarching theory that includes these and related concepts will provide a roadmap for transitioning from simpler constructs to more complex ones.

Validation and Verification. Industry will not rely on technology unless it has a methodical way to ensure both that it meets the requirements and is robust and dependable. Validation and verification is recognized as a legitimate pursuit in many engineering disciplines, and offers a rich field of research for those concerned with the broader deployment of agent-based technologies.

Tangible Deliverables. Industrial users need CASE tools, reusable design guides, and methodologies that support agents before they will make use of them. In keeping with the incremental approach recommended earlier, these resources might first be developed as extensions to existing software engineering suites, and tools that focus on agents should in turn support simpler technologies so that they can be integrated into a mixed environment.

5 CONCLUSIONS

The awareness in the multi-agent system community grows that the results of their research are insufficiently transferred into practice. During the session on the future of software engineering and multi-agent systems, leading researchers and engineers reflected on the underlying problems and identified opportunities to bring a change. In order to amplify industrial adoption of multi-agent systems, the following main opportunities were identified:



- Better communicate with industry. Focus on incremental introduction of agent concepts, and in particular, build on conventional methods en terminologies, and translate concepts and techniques from multi-agent systems into conventional software engineering practice.
- Improve the connection with the conventional software engineering community. Agent researcher could present their work in conventional software engineering venues, and participate actively in related research initiatives.
- Broaden the research agenda. In particular:
 - Focus on goal-oriented design. Goals are what make agents adaptable, and adaptability is a major concern of today software systems.
 - Enhance research on architectural patterns and styles. Patterns embody know-how in a established form and allow architects to adopt a particular multi-agent system approach.
 - Extend research on validation and verification. Guarantees about the stakeholder requirements is a prerequisite for industrial adoption of multi-agent systems.

Multi-agent systems are characterized by local autonomy, social interaction, adaptability, robustness, and scalability. These are key properties of complex distributed systems. The extent to which the knowledge and experience developed by the multi-agent system community will be adopted in practice will largely depend on the openness of the multi-agent system community to obey to industrial needs, integrate with conventional software engineering, and adapt its research agenda accordingly.

Acknowledgement

We are grateful to the IJAOSE editors-in-chief for supporting the dissemination of the results of FOSE-MAS. We thank the AAMAS 2008 chairs, Simon Parsons, Joerg Mueller, Lin Padgham, and David Parkes for supporting the organization of FOSE-MAS. Thanks to the members of the advisory board for their collaboration: Michael Luck, Onn Shehory, Michael Winikoff, Alessandro Ricci, John Thangarajah, and Mehdi Dastani. We thank the panelists Scott DeLoach, Monique Calisti, Danny Weyns, and Michael Georgeff. Last but not least, we thank all the participants of the FOSE-MAS session for the inspiring discussion.

References

- [1] AAMAS. Seventh International Conference on Autonomous Agents and Multiagent Systems. *http://gaips.inesc-id.pt/aamas2008/*, 2008.

