From Internet of Things to Internet of Agents

Han Yu, Zhiqi Shen
School of Computer Engineering
Nanyang Technological University
50 Nanyang Avenue, Singapore
Email: {han.yu, zqshen}@ntu.edu.sg

Cyril Leung
Department of Electrical and Computer Engineering
The University of British Columbia
Vancouver, BC, V6T1Z4, Canada
Email: cleung@ece.ubc.ca

Abstract—**From sophisticated single agent in complex environments to multi-agent system (MAS) organizations, intelligent software agent research has come a long way in just under two decades. Many new branches of research in this field have emerged over the years which have enabled today's agents to perform a wide variety of human-like tasks such as learning, reasoning, negotiating, self-organizing and trusting each other, etc. Unfortunately, very few practical MASs have been deployed after such a long period of intensive research and development. For MASs to achieve higher popularity among end-users, we believe that agent oriented software engineering (AOSE) should adopt a new paradigm as has been done in Web 2.0 - to allow end-users to actively participate in developing or modifying features in agents at various stages of the agent's lifecycle. In this paper, we propose a vision for democratizing AOSE. We discuss what potential new researches need to be carried out in the areas of AOSE and agent learning in order to realize such a vision of moving from MASs to mass end-user agent development, and discuss potential challenges facing various aspects of this vision.**

I. Introduction

As Internet of Things (IoT) become strives towards the fusion of the physical, social and cyber spaces, the nodes are increasingly required to exhibit intelligent behaviors to varying degrees depending on the domain of application. It is natural to view and implement them as software agents in an IoT through Agent Oriented Software Engineering (AOSE). The AOSE paradigm has come a long way since its inception. Over the years, intelligent agents have evolved from software which responds to stimuli (*reactivity*) to more sophisticated pieces of program which take initiative (*autonomy*) to fulfill their goals (*proactiveness*) and interact with each other (*social ability*) [1]. The weak notion of agency [1] proposed by Wooldridge and Jennings has attracted much attention from the research community in the beginning of the 21st Century [1]. As agent applications grow in complexity from single agent systems to multi-agent systems (MAS), more research efforts have been focused on bringing the stronger notion of agency [1] into multi-agent systems. Research in the agents' ability to move around an electronic network (*mobility*) to save data communication overhead [2] and to display more human like traits such as emotion [3] has produced fruitful results in recent years. Nowadays, mobile agents and emotional agents are well recognized notions of agencies.

Unfortunately, there are not many multi-agent systems (MASs) deployed in practical applications nowadays. Although there has been some long running international forums, such as the International Conference on Practical Applications for Agents and Multi-agent Systems (PAAMS) and the Special Track in Innovative Applications in the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS), most of the agent applications published so far are targeted for usages that not often actively involve the general public as direct stakeholders who actively interact with the technology [4], [5]. This is somehow not entirely the case for applications in the domain of education where personal interactions between the agents and learners frequently occur [6], [7]. However, in these cases, the interactions are still limited by predefined processes which have been incorporated into these agents.

Compared to the field of MAS, Web 2.0 [8] has rapidly transformed the Internet within a short span of less than 10 years. Today, millions of users interact and create content and applications for Web 2.0 based platforms such as Wikipedia, Youtube, Facebook, Second Life and Twitter etc. Together with the paradigm of end-user development [9], [10], these Web 2.0 platforms have grown enormously successful by involving end-users as an auxiliary developer force into their digital ecosystems. It was estimated that in 2005, there were 55 million end-user developers in the U.S. alone as compared to 2.75 million professional developers. It is an active environment which can be viewed as crowdsourcing on steroid. By opening up the development of contents and applications for end-users, the Web 2.0 concept effectively democratized the Internet and created an ideal environment for explosive growth, mass collaboration and innovation.

While contemplating a way forward for MASs to become as successful as Web 2.0, we started realizing that the existing practice of institution-based approach to the research and development of agents and MASs are limiting its popularity with general public end-users. While researchers may be well positioned to know the value of certain technologies to MASs, it is the end-users, who are going to be direct stakeholders in deployed MASs, who are more likely to have insights into how to tweak the MASs to better serve their needs and achieve their goals. What they are lacking is the knowhow and tools to carry out these modifications or developments.

The current situation MASs find themselves in is reminiscent of that in the early days of the Internet. Imagine if the early Internet systems such as Usenet [11] did not take the stride towards opening up to the general public and becoming the World Wide Web, would Web 2.0 based platforms and applications exist today? While history cannot be assumed, we believe the answer would be a resounding ”no”. It is our vision that, in order for MAS research to move into the life of millions of end-users, we need to empower them with the tools and frameworks to help steer the development of future MASs. With the advancement of MAS research, we believe such a vision is close to being ready today.

In the following parts of this paper, we will give an overview of recent attempts in enabling end-users who are not technically savvy to help modify and even develop part of the features in existing agents. We will discuss the limitations in existing approaches and further illustrate our vision of a MASs open to end-user development by presenting what needs to be done from the perspective of the end-users and AOSE. At the end, we will complement the vision with some potential challenges that can be foreseen and possible ways around them.

II. State of the Art

For many researchers working in the field of MAS, the Java Agent DEvelopment Framework (JADE) [12] is the platform of choice when it comes to implementing their agents. It provides a set of sophisticated application programming interfaces (APIs) in the popular JAVA programming language to facilitate the implementation of various basic functional aspects involved in a typical software agent. Together with the Agent Communication Language (ACL) [13] defined by the Foundation for Intelligence Physical Agents (FIPA), multi-agent coordination in MASs can also be implemented.

JADE is often used in conjunction with the Belief-Desire-Intention (BDI) software model [14] to implement intelligent agents. The BDI model provides designers with a mechanism for separating the act of deliberating which action plan to select from the act of executing the currently selected action plan. Designers are responsible for creating these action plans in before they implement intelligent agents based on these action plans using the BDI model.

Based on the BDI model and JADE, an integrated design tool - the Prometheus Design Tool (PDT) [15] - was proposed by RMIT University, Australia. It provides a wide range features supporting the use of the Prometheus methodology which is a process following which designers can specify in detail the requirements and design for various aspects of a MAS. The PDT has been integrated with the Eclipse platform to provide a more closely coupled implementation process to MAS researchers.

Another agent design methodology, known as Tropos [16], has also been developed. It covers the agent oriented software design process from very early requirement analysis which identifies how humans and agents should interact in the target system all the way down to implementation. Agent mentalistic notions are used in all these different phases. They even provided a design language called the Tropos language for designers to conduct conceptual modeling.

These methodologies, tools and languages are designed primarily for researchers who are well versed in intelligent agent related technical concepts and the AOSE process. In order to efficiently use these techniques, the designers need to be familiar with specific areas of agent research and spend a lot of effort learning how to use the artifacts in these techniques. In addition, these methodologies are often designed for a team developing complex MASs to use. The design objectives of these approaches are not well aligned with our vision of next generation AOSE for mass end-user agent development.

Nevertheless, there are other existing attempts that are closer to our vision than the abovementioned approaches. One approach - SketchiXML [17] - proposed to use hand sketches on paper to automate the design of MASs. Designers can draw their MAS designs on paper and feed it to a computer with software capable of making sense of it. In most cases, the software is platform and language dependent. However, it has some inherent drawbacks. Firstly, users must draw out the design following closely the predefined shapes of the recognized entities involved. Secondly, as the design is being modified, the user must also follow predefined patterns to delete unwanted items or simply start all over again. The most limiting drawback is that the approach provides no way for the users to specify internal logic of each design item. Therefore, implementing a MAS based on these design sketches still requires a lot of programming effort.

Another approach - the POSH tools [18] - has been specifically developed for non-expert agent developers in the game industry to use for building intelligent agents. The tools include a simple agent management system; a development environment for programming the agents' properties and the communications among them; a graphical editor called ABODE for behavior and dynamic plans modification; and a set of classes implementing some predefined basic behaviors for agents in games. While it is a valuable step towards the realization of our vision, the POSH tools still lack generality to be applied domains other than game development. Nevertheless, some individual components in the POSH tools, such as ABODE, have great potential to become general tools for end-users to use.

The approach with design objectives most similar to our vision is the Goal Net Methodology [19]. Its main design artifacts are simplified into three categories based on human intuitions to facilitate easy understanding by non-technical people. They are goals, transitions and arcs. A goal represents the immediate milestones an agent need to achieve during its reasoning process; a transition defines a set of actions based on what is supported in the target operating environment that enables an agent to transit from one goal to the next; an arc is a design artifact that indicates the direction of the flow of logic of an agent's mental state. A design toolkit called Multi-Agent Development Environment (MADE) [20] has been developed based on the Goal Net Methodology to allow users to draw out their designs in the form of hierarchical goal nets. The design data is automatically converted into an agent by the MADE toolkit. The logic of the agent is interpreted and executed by the MADE Runtime system. Although commonly used agent behavior functions are implemented in the MADE toolkit, functions in each transition specified by the user still need to be implemented separately. Currently, they are required to be implemented as dynamic linked libraries (DLLs) so that they can be executed by the agents at runtime.

III. Opportunities for Further Research

From the summary of existing AOSE tools in Figure 1, it can be clearly seen that, currently, the main focus of the AOSE methodologies and tools are still to serve expert agent researchers.

Most of them are difficult to learn to use well, but once learnt, they can be used to design agents for a wide range

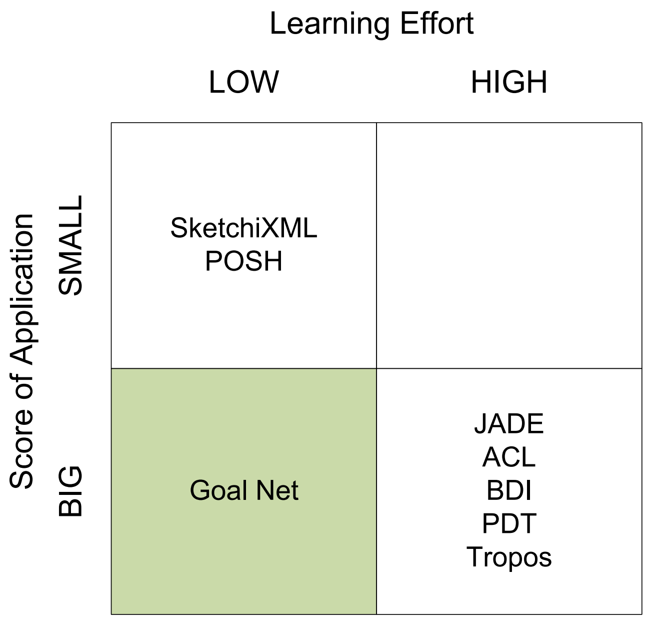

Fig. 1. Summary of Existing AOSE Tools.

of applications. Recent efforts made to enable end-users to develop their own agents are mainly focused on incorporating simplicity into agent development tools. The trade-off is that these tools typically can only be used to design agents for a specific domain of application. Although the Goal Net agent design tools have achieved a better balance between openness, simplicity and the inclusion of reusable building blocks, there are still limitations in its ability to infuse sophisticated learning capabilities into agents. The approaches are still aimed at people with considerable programming skills although they may not be required to have a deep understanding in agent research. While this step is necessary and still need to be pursued further, improvements in making agent development tools open to end-users, simple to learn and use, and equipped with building blocks for constructing intelligent agents should be explored.

Ideally, to facilitate non-technical end-users to participate in the development of agent features, several methods of input gathering should be allowed. Firstly, users should be able to modify or add only part of an agent's behavior without having to completely programming it from scratch. This requires agents to be designed in a modular fashion with a clearly defined boundary between core functionalities that should not be tampered with under any circumstances and functionalities that are open to user modification. The critical nature of the roles played by various agents in a given application must be systematically assessed in order to determine such boundaries.

Secondly, in order to involve more end-users and make agent development popular with the mass, the technological tools for them to modify or create agent features should be easy to understand and use. This objective requires a fine grained approach in implementing agent features so that modifying any individual feature does not require significant efforts. An example of a similar approach used in online task management is the IFTTT (if-this-then-that) system [21] where users can design processes for management multiple online accounts (e.g., emails, Facebook accounts, etc.) to automate certain tasks by simply specifying *if-then-else* rules through the IFTTT web portal. While this approach may be overly simple for modifying agent features in most cases, it is appealing to the end-users.

Thirdly, to exponentially expand the end-user agent developer population, future agents should be incorporated with the ability to modify its behavior based on the influence of ambient knowledge. Such knowledge may be from overhearing conversations among end-users or through observing the demonstrations of how to carry out an activity by a user. The key challenge here is to design a user interface for the end-user to show the agent what is intended without involving programming, and then enable the agent to abstract the idea behind the demonstration out so that it can automatically reprogram its behavior to include this newly learnt action. For the generalization part, the technique of transfer learning [22] may be a useful starting point for further research.

IV. Discussions

Apart from the technical challenges, the vision for involving end-users in the development of agents or parts of the agents also faces other non-technical concerns.

In real world, not all end-users are well-intentioned. Their modifications can be malicious and undermine the security in the MAS. In addition, there will always be mistakes made by people at various point when modifying an agent. A framework for managing risks due to end-user development in MAS, creating maintainable software, and eliminating inaccuracy and conflicts should be in place before end-users are invited to contribute.

On the other hand, end-users may not be motivated to contribute simply because tools facilitating them to do so exist. Incentives may be necessary to nudge them into the agent development ecosystem to kick start the process. Such incentives could be linked to popular platforms such as online social networks or virtual worlds which the end-users may be involved. A careful balance between quality assurance and design freedom will be a key concern for this vision.

Even if new agent features contributed by end-users pass the software quality assessment and is determined to be free of malicious content, there is still another higher level challenge remaining. Since interactions among agents in an MAS depends on environment changes and are often carried out in a non-linear way, the implications of changes on the wellbeing of the whole system can be unpredictable. Changes may be rational from an individual agent's perspective. However, when such changes are made, they may conflict with existing logics and cause unintended consequences. This can be very difficult to detect in advance. Therefore, research should be carried out for the design of appropriate methods and metrics to estimate the impact of new changes to the entire MAS. Proper accountability models should also be explored to pinpoint which changes need to be rolled back in case unintended consequences happen.

From the above discussion, it can be seen that both technical tools to lower the hurdle for end-users to contribute to agent designs as well as frameworks for managing such contributions are needed in order to realize the vision of a democratic AOSE process where end-users can help enhance the popularity of MASs. Although there have been significant progress made in the research of technical agent development tools for non-expert end-users, there is still a visible lack of interest in the area of end-user contribution management frameworks. Both of these areas need more attention from agent researchers in order to enable future AOSE approaches to help end-users create their own agents and bring sophisticated, dynamic and sustainable human-agent collectives into reality.

Acknowledgment

This research is supported in part by Interactive and Digital Media Programme Office (IDMPO), National Research Foundation (NRF) hosted at Media Development Authority (MDA) of Singapore under Grant No.: MDA/IDM/2012/8/8-2 VOL 01.

References

[1] M. Wooldridge and N. R. Jennings, "Intelligent agents: Theory and practice," *The Knowledge Engineering Review*, vol. 10, no. 2, p. 37, 1995.

[2] T. Schlegel, P. Braun, and R. Kowalczyk, "Towards autonomous mobile agents with emergent migration behaviour," in *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems (AAMAS'06)*, 2006, pp. 585–592.

[3] L. Morgado and G. Gaspar, "Towards background emotion modeling for embodied virtual agents," in *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems (AAMAS'08)*, 2008, pp. 175–182.

[4] V. Bevar, S. Costantini, A. Tocchio, and G. D. Gasperis, "A multi-agent system for industrial fault detection and repair," in *Proceedings of the 10th International Conference on Practical Applications for Agents and Multi-agent Systems (PAAMS)*, 2012, pp. 47–55.

[5] E. Shieh, B. An, R. Yang, M. Tambe, C. Baldwin, J. DiRenzo, B. Maule, and G. Meyer, "Protect: a deployed game theoretic system to protect the ports of the united states," in *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS'12)*, 2012, pp. 13–20.

[6] "Alice 3d authoring system. alice v2.0. carnegie mellon university," Available: http://www.alice.org/, Accessed in 2013.

[7] H. Yu, Y. Cai, Z. Shen, X. Tao, and C. Miao, "Agents as intelligent user interfaces for the net generation," in *Proceedings of the 15th International Conference on Intelligent User Interfaces (IUI'10)*, 2010, pp. 429–430.

[8] T. Oreilly, "What is web 2.0: Design patterns and business models for the next generation of software," *Communications and Strategies*, vol. 1, no. 65, pp. 17–37, 2007.

[9] A. J. Ko, R. Abraham, L. Beckwith, A. Blackwell, M. Burnett, M. Erwig, C. Scaffidi, J. Lawrance, H. Lieberman, B. Myers, M. B. Rosson, G. Rothermel, M. Shaw, and S. Wiedenbeck, "The state of the art in end-user software engineering," *ACM Computing Survey*, vol. 43, no. 3, pp. 1–44, 2011.

[10] H. Lieberman, F. Paternò, M. Klann, and V. Wulf, *End-User Development: An Emerging Paradigm*, ser. Human-Computer Interaction Series, 2006, vol. 9, pp. 1–8.

[11] M. Hauben, *Netizens: On the History and Impact of UseNet and the Internet*. Peer-to-peer Communications, 1996.

[12] "Jade - java agent development framework," Available: http://jade.tilab.com/, Accessed in 2013.

[13] "Fipa agent communication language specifications," Available: http://www.fipa.org/repository/aclspecs.html, Accessed in 2013.

[14] A. S. Rao and M. P. Georgeff, "Modeling rational agents within a bdi-architecture," in *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning*, 1991, pp. 473–484.

[15] L. Padgham, J. Thangarajah, and M. Winikoff, "Tool support for agent development using the prometheus methodology," in *Proceedings of the Fifth International Conference on Quality Software (QSIC'05)*, 2005, pp. 383–388.

[16] P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos, "Tropos: An agent-oriented software development methodology," *Journal of Autonomous Agents and Multi-Agent Systems (JAAMAS)*, vol. 8, no. 3, pp. 203–236, 2004.

[17] A. Coyette, S. Faulkner, M. Kolp, Q. Limbourg, and J. Vanderdonckt, "Sketchixml: Towards a multi-agent design tool for sketching user interfaces based on usixml," in *Proceedings of the 3rd Annual Conference on Task Models and Diagrams*, 2004, pp. 75–82.

[18] C. Brom, J. Gemrot, M. Bida, O. Burkert, S. J. Partington, and J. J. Bryson, "Posh tools for game agent development by students and non-programmers," in *Proceedings of the 9th International Computer Games Conference: AI, Mobile, Educational and Serious Games*, 2006.

[19] Z. Shen, C. Miao, and R. Gay, "Goal-oriented methodology for agent-oriented software engineering," *IEICE Transactions on Information and Systems*, vol. E89-D, no. 8, pp. 1413–1420, 2006.

[20] H. Yu, Z. Shen, and C. Miao, "A goal oriented development tool to automate the incorporation of intelligent agents into interactive digital media applications," *ACM Computers in Entertainment Magazine (CiE)*, vol. 6, no. 2, 2008.

[21] "Ifttt: If-this-then-that," Available: https://ifttt.com/, Accessed in 2013.

[22] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol. 22, no. 10, pp. 1345–1359, 2010.