Letters

The Challenge of Finding Intelligent Agents

Agents have been intensively studied for more than 20 years. However, we don’t see any real application of agents in the Web. Why? The answer is simple. Most application software can be developed and deployed without using such an intelligent framework.

The same thing applies to Web services. You don’t need to understand UDDI, WSDL, SOAP, and all WS-* technologies to build a Web application. You can just write a code in JavaScript or any other toy language, simply use HTTP GET and POST, and you’re done! If RDF-based RSS is too complicated, go Atom and REST!

This is the technical essence of Web 2.0, which “mash-ups” existing contents in a simplistic manner. This primitive composition works fine in most situations, and that fact irritates me.

Looking back in history, object-oriented technology became widespread in industry because people wanted to develop GUI (actually, MS Windows) applications. OO was a natural way to model and program windows and widgets. (Using Xlib in C was a nightmare.) Technologies such as compilers, structured programming, and relational databases all started in academe, and industry eagerly adopted them. Real programmers in real industries strongly needed such new techniques to implement real applications.

Now, do the real programmers need technologies such as the Semantic Web or agents? In other words, what are the killer applications of the Semantic Web and agents? I have to be pessimistic. But this pessimism isn’t the fault of OWL or the Foundation for Intelligent Physical Agents’ Agent Communication Language by themselves. The real questions are, is there any need of further advancement of software technology? Are smaller hardware, faster processors, and wider network bandwidth just sufficient for future computation? At least, those are all I need. Everything else is just fine with me.

Hiroki Suguri
Comtec
suguri@comtec.co.jp

I’ve been working in an agent-related industry and would love to present my humble opinion, based on my own experience, in response to your challenge (see “A Letter from the Editor,” May/June).

Just in the first year and a half of my work, I’ve been a principal investigator for two ongoing Small Business Innovation Research projects, both of which heavily employ agent technologies. These technologies continue to attract a huge community and have significant commercialization in various fields and applications, mostly defense, aerospace, healthcare, and e-commerce.

The first project, funded by DARPA, explores human-agent collaboration (HAC)—that is, how human experts and software agents cooperate with each other as peers to solve distributed problems. Particularly, we’re evaluating how HAC, as a decision-making enhancement, could assist human-to-human activity collaboration (H2AC) that reflects real-life scenarios and problems. The key idea is for software agents to focus on what they’re good at—namely, computational tasks—while humans focus on tasks requiring manual operations.

The second project, funded by the US Air Force Research Lab, focuses on assessing and tracking team performance in collaborative environments. This is a fairly new field in the agent community; traditional technologies for performance assessment have focused on textual analysis and human-systems approaches. Agent concepts and technologies enable a suitable knowledge representation method, automated intelligent processing capability (compared with traditional human manual, subjective evaluation), and more.

I wasn’t surprised to learn about your challenge: “I see no evidence for the imminent widespread use of this technology such as we were promising a decade ago.” That’s because the IT industry and the general population haven’t been educated enough to recognize the importance of agents yet. I had the opportunity to talk to a few companies whose major product is agent technologies. The CEOs and managers of those companies all pointed out the difficulty in educating potential customers, not to mention the general public, about agent technologies.

The Internet’s success benefits from the fact that what the Internet has to offer is tangible—anyone with a suitably configured Web browser can access it. I believe that software agents, as standalone products (for example, ticket-purchasing agents and scheduling-assistance agents), aren’t yet widespread because they

• are embedded in or associated with containing hardware and software,
People can purchase an agent product as a helper for decision making (for example, reminding them about their daily schedule or choosing the best deal for a purchase), but not as middleware alone. Just as in the aforementioned DArPA project, such HAC agents assist H2AC (for example, gathering human expertise and matching it with user requests), but these agents alone merely serve a general—not necessarily independent—purpose.

In order for the next generation of agent technologies to have a good chance in those fields, the following should happen:

- Ontological approaches should pave the way for integrating the currently dispersed agent implementations into semantically related components.
- Software agents should assist humans in conventionally challenging tasks (for example, computation-intensive decision making and gathering human expertise and matching it with user requests) rather than replace human intelligence.
- Agent technologies should continue to merge with other areas (such as human systems and psychology, and biological and biomedical technologies) to provide novel approaches and solutions.

(The views in this letter are the author’s and don’t represent the official position or policy of the research sponsors mentioned in the letter.)

Wei Chen
Intelligent Automation Inc.
wechen@i-a-i.com

Clearly, James Hendler is more than qualified to state (May/June issue), “What we don’t seem to have are intelligent-agent-based systems in any serious way!” I also fail to detect signs that agent technologies are about to penetrate real-world applications with the exponential growth that’s typical for information technology.

Personally, I’m leading two projects aimed at deploying agent technology. Notably, these projects aren’t exactly applying mainstream agent technologies. The agent community has developed a consensus that all agents must have goals and are actively pursuing them. However, real-world applications comprise much more than goal-oriented components. Current agent technology is like a soccer team without a defense.

Some smaller research communities are looking at this problem. For instance, one community is looking into environments for multiagent systems. Their research doesn’t stop at providing an IT infrastructure. It provides a better connection to the real world. The February 2007 Autonomous Agents and Multi-Agent Systems special issue on this topic provides an introduction on the developments and reveals how applications benefit.

In the two projects I mentioned, the key elements aren’t intelligent agents but intelligent beings (see Paul Valkenaers et al., “From Intelligent Agents to Intelligent Beings,” to be published in Holonic and Multi-Agent Systems for Manufacturing, V. Marik, V. Vyatkin, and A.W. Colombo, eds., LNAI 4659, Springer, 2007). Intelligent beings aren’t goal oriented. Instead, they reflect some part of reality. In many ways, these intelligent beings are the descendants of geographical maps, which have developed into powerful service providers.

In mainstream agent technology, the services of intelligent beings are provided by ontologies and the internal world models of the agents. Ontologies fail to contain information about the state of the world: they might contain information about the concept of a car but don’t reflect that your car is in the parking lot of your university or company. Internal models fail to share information: information is private, and its representation serves its owner’s needs first.

In summary, agent technology today is like route-planning software without maps. Some vital components are missing. Whether these are intelligent beings, environments as a first-class abstraction, or something else remains an open issue. It probably will take a combination. A serious challenge in this matter is the need for truly multidisciplinary research teams. Indeed, those missing items must crystallize invariant facts about both the natural and artificial world.

To conclude, two remarks about the concept of an intelligent being. First, experience in other technologies supports separating the intelligent agent (goal-oriented decision making) from the intelligent being (reality reflecting). In nature, humans and animals comprise both. In nature, birds have integrated lift and propulsion devices. However, in the world of manmade artifacts, airplanes have separate components for lift (wings) and propulsion (jet engine).

Second, a short story illustrates how the intelligent being remains unaffected when integration conflicts need resolving. It indicates how large collections of useful software can be developed through intelligent beings without crumbling under their complexity: reality comes to the rescue. The story recaps a conversation over the maritime radio waves between an intelligent agent and an intelligent being:

- Intelligent being: “Ship ahoy. This is CL273. You are on a collision course with us. Please change your heading immediately.”
- Intelligent agent: “This is USS129. This is the United States Navy. You change your course.”
- Intelligent being: “This is Canadian Lighthouse 273 …”

Hoping the thoughts presented in this letter might help agent technology become a major socioeconomic contributor.

Paul Valkenaers
Katholieke Universiteit Leuven
paul.valkenaers@mech.kuleuven.be

At the first NASA Workshop on Radical Agent Concepts in 2002, Tim Finin posed essentially the same challenge. To paraphrase (loosely and with apologies for any unintended inaccuracy), he told the workshop participants that “both agent technology and the Internet were created in the same year—guess which one is more important now?” I’m sure you can answer his question.
today with exactly the same response as was appropriate then—agent technology hasn’t achieved the potential many of us believed it could, would, and should have reached.

Tim Finin made his comment in the context of the “need for a universal ontology” debate, which loomed large at that time. Other noted experts at the conference expressed concern that without an “Esperanto” agent ontology, the widespread adoption of agent technology would be dramatically constrained. I could well believe the shift in focus to agent architecture interoperability you described at DARPA was inspired by similar concerns. My answer to “Where are all the agents?” is that many of the ontology initiatives motivated by the concerns expressed in the past, as well as the interoperability approach you undertook at DARPA, were bad ideas with regard to the successful propagation of practical and powerful agent technologies into real-world applications.

In all honesty, that conference was a very dispiriting experience. First, some participants seemed overly concerned about expressing and enforcing agent theory “political correctness” (dare I challenge the notion that “a rock can be an agent”?). In addition, I felt that many of the leading evangelists of agent technology were losing sight of agents’ true power and potential, which was foreseen and described by many of the founding fathers of agent theory (for example, “early” Hector Levesque). Specifically, the ontology and architecture interoperability approaches constrain the power of massively distributed agent populations to solve real-world problems using only their combined “agent sized” actions—all in the context of real-world failures, uncertainty levels, and incomplete data, and all the narrowing circumstances that the real world imposes on people’s actions and plans. (Some important agent characteristics and capabilities include populations of genuinely autonomous, autonomic [with regard to the OS] agents, and agents that are capable of platform and system migration, ad hoc task delegation, the absolute right of task refusal, “volunteerism” driven by “principle,” a high level of role self-determination [that is, dynamic specialization], the ability to “learn” anything, the ability to destroy other agents [such as for virus control], and the ability to clone themselves at will.)

In short, agents have been chained to rigid ontology- and architecture-constrained solutions in a misplaced effort to ensure set-piece reliability and transaction success, rather than letting agent technologies and populations dynamically evolve (as human populations have) to achieve “good enough” reliability and (fuzzy) system success in getting many jobs done. Apologies to the agent community from a true believer, but agents seem to have become over the last five years “just another overly complex and impractical technology.” There can be few alternative explanations for why more agents and agent solutions aren’t in use when agents would be the superior solution in so many situations.

With all due respect to the agent ideologues, the only agents worth discussing here are software objects, because object theory wedded to distributed-computing techniques provides the great initial framework for practical and powerful agent computing solutions. We must emancipate computer object agents from their current constraints if agent technology is to ever achieve its great potential.

We must emancipate computer object agents from their current constraints if agent technology is to ever achieve its great potential. Unless agents can become emancipated objects, I have to wonder if there will be an agent community at all in five more years.

—Mark Bobick
CTO, Correlation Concepts
m.bobick@correlationconcepts.com

James Hendler isn’t the only one wondering where the agents have gone. Observing the fact that agent technology hasn’t moved into mainstream computing isn’t difficult. However, finding out the cause of this is more demanding.

Let me start with some general remarks on AI research and computing. My impression is that many AI researchers are trying to develop cutting-edge technology—technology that’s intelligent, adaptive, learning, natural, and so on. The main concern in this type of research is how close we can get to human performance rather than how close we can get to mainstream computing. The connection between AI techniques and the rest of computing science is of little interest to these researchers. This isn’t to say that it’s not important! However, we need other researchers with interests in software-engineering issues to bridge this gap.

With agent technology a similar division of interests is visible. The people interested in the theoretical issues don’t care about the efficiency of the systems and standardization of results. On the other hand, the people doing agent-oriented software engineering care little about being faithful to BDI (belief-desire-intention) theory. They want to engineer products that nonagent people can use. Although the basic ideas about software agents are quite intuitive and appealing, they’re far from fully developed. Therefore, the theory doesn’t give software engineers enough ground to base their implementation decisions on. Lacking any backup from theory, these people made their own decisions based on pragmatic considerations. This led to many different agent-programming platforms and languages. Because people emphasize different aspects of the agent systems, the platforms and languages also differ in many aspects and are hard to use together—let alone that a standard could be developed.

Two efforts to alleviate this problem were coming from DARPA through the CoABS (Control of Agent-Based Systems) project and from the Foundation for Intelligent Physical Agents through its efforts to standardize agent interfaces. Unfortunately, the CoABS software never made its way outside the US. The fact that it’s now commercialized doesn’t make it easier to use as a standard infrastructure. The FIPA Agent Communication Language has become a de facto standard for agent communication. However, its relation to other Web services communication standards coming from the World Wide Web Consortium is unclear. Therefore, it’s difficult for industry to start using this standard because it doesn’t link to existing industry software standards.

What would help is a standard agent platform or language that can be used in conjunction with other existing software. This...
in, say, 1982. Yet, that was a full two decades after the invention of the OO concept in 1962, which is approximately the length of time that agent computing concepts have now been in circulation. In 1982, of course, Java was still 13 years away from public release. We’re not claiming here that agent technologies will have as great an impact on computing as have object technologies, and there are many reasons why they are unlikely to do so (see the AgentLink Roadmap, www.agentlink.org/roadmap/index.html, for a more detailed analysis). Nevertheless, we see agent technologies currently being used, both experimentally and commercially. This is the case despite the lack of a foundational “prediscipline” that, for instance, the Semantic Web enjoyed (in extensive prior research on knowledge representation and logic) before it coalesced into its current form.

Once dynamic and open systems become the norm, they’ll need to adopt agent technologies as fundamental. The conceptualization of agents (and possibly the technologies) might not then be distinct domains of study but could become a part of the accepted view of what constitutes computing. Indeed, as with object orientation, which we usually don’t consider to be a specific technology, it’s likely that agents will simply be the normal way of conceiving and engineering computer systems. Such a development will lead to increased opportunities for agent technology application and deployment because in a world of pervasive distributed computing, we’ll require methods and systems for negotiation, auctions, trust and reputation, coalitions, and so on. As computer scientist Christos Papadimitriou noted at his presentation at the 2001 Annual Symposium on the Theory of Computing about software engineering design, “All design problems are now mechanism design problems” (see http://doi.acm.org/10.1145/380752.380883). Will it be an indication of failure or of success when no one talks of agents anymore because agent approaches are so pervasive in the next generation of computing?

Related Resources

**Agentis Software**: www.agentissoftware.com
**Agent Oriented Software Group**: www.agent-software.com
**Eurobios**: www.eurobios.com
**Lost Wax**: www.lostwax.com
**Magenta Technology**: www.magenta-technology.com
**Whitestein (and Living Systems)**: www.whitestein.com


Letters

continued from p. 5

sounds quite simple. However, I bet most people would at this moment disagree about which components should be part of such a standard platform. Why is that? Mainly because agent theory didn’t solve a number of central issues, and people are avoiding tackling them. Such issues include these:

- How do you balance proactive and reactive behavior in a principled way?
- How can agents behave socially? That is, which social concepts should the theory include?
- Where does intelligence go—into the reasoning, into the interaction, or both?
- How is that determined, designed, and implemented?
- How can we combine learning and adaptivity with reasoning?

But probably most important is that the agent community doesn’t seem to have a way to discuss these questions and give some general direction to the research.

This doesn’t mean that nothing good happens in agent research. There are efforts to integrate agents and Web services. It would be good if the results of these efforts were available for the whole community.

Also, some progress is being made in using ontologies. Recent work has been published on negotiating ontologies on an as-needed basis. Relatively simple protocols are used to exchange concepts so as to enable all the communication that’s needed to perform some joint task.

Research on game-theoretic issues could be used to design the interaction patterns in multiagent systems in a way that leads to optimal behavior.

So, my message is mixed. Some good things are happening and should be made clearer to the outside world. However, a number of important issues are left unsolved, and no concerted effort seems to be underway to address them.

Unless we face these issues we will lose momentum, and people (read: industry) will start losing trust in agent technology.

Frank Dignum
Utrecht University
dignum@cs.uu.nl

Peter McBurney is a faculty member of the University of Liverpool’s Agent Applications, Research and Technology Group. Contact him at p.j.mcburney@csc.liv.ac.uk.

Michael Luck is a professor of computer science at King’s College London. Contact him at michael.luck@kcl.ac.uk.