What we talk about when we talk about software agents

There's so much talk these days about software agents—and close relatives with names such as softbots, knobots, and interface agents—that we are reminded of the early days of AI (see the “Early AI” sidebar). The ideas are creative, early-stage, and all over the map. Here at Stanford alone, you will find agents that sort your mail, adaptively recommend Web pages, assist with scheduling, find people with interests similar to your own, translate between different knowledge bases, and have individual electronic personality and graphical depiction. Elsewhere, you can also find agents that help manage your network, shop for you, migrate in the network, have a natural-language understanding capability, and much more. Many of these academic projects have been coupled with commercial efforts—some of them highly publicized. Indeed, while using the term agent in connection with software has been around for as long as anyone can remember, somehow in recent years it has developed particular cachet.

A state of hype

1995 offered the first confidential, high-priced industry report on the “software-agent revolution.” Journals, including this one, have since devoted special issues to software agents. In 1997, the first agents conference took place, and shortly thereafter the first academic journal devoted to agents appeared. Halfway between academia and industry, we encounter efforts to establish agent specifications in international standard committees. And, squarely on the industrial side, we see special panels on agents in industry symposia, agent technology groups in corporations, and, inevitably, a host of startups predicated on “agent technology” of one sort or another.

What happened to lend such sex appeal to software agents all of a sudden? Is it that somehow the technology matured all at once? Or is it that new needs have arisen that make the agent technology (whatever that means) suddenly relevant? Is it that software agents embody good science and will yield commercial success, or are they merely a flash in the intellectual-industrial pan?

The short answer is that technology has evolved at a respectable pace, but there have been no quantum breakthroughs in recent years. The main catalysts behind the agents hoopla have been two rapidly accelerating trends in computing—consumer-based computing and the spread of the Internet. Together, these trends have presented new opportunities for innovation, many of them in the realm of software agents.

Two elements, however, threaten to spoil the party—overhype and the confusing amalgamation of quite different ideas and motivations under the agents umbrella. The hype is unfortunate; alongside fairly creative ideas, quite shoddy ones are seeking legitimacy under the umbrella. The danger is that negative reaction to the lesser ideas will be directed indiscriminately towards all work in the area. However, my purpose here is not to pass judgement on this or that work. What I’d like to do instead is tease apart various strands of ideas in the area, thus placing some structure on the space of agents that will hopefully facilitate intelligent discussion on the topic.

How to think about software agents

Having an intelligent conversation about software agents isn't easy. No one can give you a clear and comprehensive definition of the notion. At best, what you'll get is a clear definition of one person's version of the concept, guaranteed to exclude various elements
that others will swear are the essence of software agents. At worst, you’ll get an answer so general and imprecise that it has little informational content. This state of affairs has caused more than one reasonable person to scoff at the very notion of software agents, which is unfortunate, because there are some exciting things going on in this space.

So if we can’t define agents, what can we do to shed light on them and provide a framework for evaluating work in the area? One approach is to identify dimensions or axes of software agency—that is, various properties that characterize this or that version of agents. The idea behind such an exercise, which we see attempted in various places, is to be able to group together kinds of software agents that embrace roughly the same properties.

Here are some properties that have been proposed (for reasons that will become apparent, I make no attempt to define them precisely, or to attribute them to specific authors):

- **Ongoing execution.** Unlike software routines that are invoked to achieve particular tasks and then disappear, agents function continuously for a lengthy period of time.
- **Autonomy.** Agents do not require constant human control or supervision.
- **Environmental awareness.** Agents model the environment in which they operate, track it, and react to changes in it.
- **Adaptiveness.** Over time, agents adapt their behavior to suit the preferences and behavior of individual users.
- **Intelligence.** Agents embody sophisticated techniques—for example, one based on probabilistic reasoning, machine learning, or automated planning.
- **Agent awareness.** Agents model other agents, reason about them, and interact with them (for example, communicate or negotiate) using specialized protocols.
- **Mobility.** Agents can migrate in a network.
- **Anthropomorphism.** Agents exhibit human-like mental qualities; they respond to queries about their “beliefs” or “obligations,” convey “understanding” or “confusion” through icons depicting facial expressions, or display an animation that connotes “fear” or “friendliness.”

Categorizing agents along these or similar dimensions is not without merit, but it has three flaws, in increasing order of severity. First, most of these properties are vague. Second, as you start categorizing kinds of agents in this fashion, things simply don’t aggregate nicely. You can point to some natural groupings. For example, various information-retrieval agents share a strong reliance on adaptive techniques. Similarly, other agent work is predicated on having code migrate in the network, which makes this group of work internally consistent but fairly unrelated to the first group. However, these are exceptions to the rule, and overall the various agent-related work simply doesn’t arrange itself tidily into clumps in this scheme. But the problem goes deeper. The scheme’s worst deficiency is that it reinforces the myth that there is at heart one pure agent functionality, and the agent-related work is grouped according to the way in which it departs from this ideal. The reality is that in the agent-related field, there is work that is so dissimilar from other work in goals, orientation, and technology, that it is misleading to even speak of such goals in the same breath.

It is therefore much more useful to first home in on the primary orientation of the work. To understand this orientation, we must keep in mind two trends that in recent years have shaped much of the agenda of computing—the explosion of the Internet and of consumer- (as opposed to business-) based computing. These have given rise to the following three strands of agent-based work:

- **Exotic distributed systems (EDS):** Building novel middleware to increase developers’ productivity, creating novel infrastructure to leverage the (local or wide-area) network, and, in general, contributing to making real the slogan about the network being the computer.
- **Anthropomorphic design (AD):** Rendering computers more accessible to the nontechnical user by endowing them with looks and behaviors that are at least superficially human-like.

Another use of the term agent is prevalent nowadays within the AI community but differs from the ones above. A growing number of researchers argue that it is a mistake to study particular mechanisms of intelligence in isolation. Rather, it is said that researchers should tackle real-world problems and build complete systems that solve these problems. For example, research on robot planning cannot be conducted independently from the realities of robot sensing and actuation hardware. The reasoning is that when we attempt to program a real robot, we are led to embrace and develop techniques that are quite different from traditional planning techniques. This methodological stance is having a strong impact on the field and has resulted, among other things, in increased emphasis on probabilistic methods and adaptive techniques. Often in AI the term ‘agent’ is used loosely to describe systems that are designed from this methodological standpoint. However, because this orientation is internal to the AI research community, I won’t discuss it further.

Given a particular brand of software agent, how do you categorize it within this ternary scheme? Here is a simple litmus test. Think of the intended direct beneficiary of this work, and place it on the continuum shown in Figure 1.

Broadly speaking, the EDS-motivated agent work attempts to make the lives of folks at the left end of the spectrum easier, and the AD-inspired work attempts to do the same for the right end of the spectrum. The NES-motivated work by nature runs the gamut, but it tends to concentrate on the middle, directed primarily towards computer-literate users but not developers.

Beyond mere categorization, these three orientations facilitate some qualitative assessments of the practical impact of agent-related work. Consider three questions:
• How mature is the technology?
• What is the area’s potential impact?
• How long will it take to realize this potential?

The three orientations line up the same on these three scales (see Figure 2), which I’ll discuss further below.

NES. The technology here is fairly mature—the fruit of 10 or more years of research. This is particularly true in work that relies on machine learning and probabilistic techniques, areas that have seen significant progress in the past decade. As far as the significance of the work in this area goes, there is no doubt that it is important, but I think it’s fair to say that it won’t change the face of computing. On the other hand, the technology is ready today. Indeed, many industry people, entrepreneurs as well as investors, display excitement about the near-term opportunities in this area.

EDS. Infrastructure for leveraging resources external to the code developed by the individual has come a long way since the introduction of the clipboard. Manual data sharing evolved into data sharing among applications (such as direct data exchange). Linked libraries allow sharing code, not just data. Remote procedure calling extended this functionality to the network. The advent of object-oriented programming has reinforced these trends, especially with the recent movement toward distributed object systems (OMG’s CORBA, Microsoft’s COM/DCOM, and Sun’s (Enterprise) JavaBeans).

Some very recent agent work attempts to go a step beyond this and introduce yet more powerful tools to leverage the network. It is useful to subdivide the EDS agent work into two subgroups. The first subgroup aims to enrich the infrastructure itself in a domain-independent fashion. The second subgroup is closer to the application level and is driven primarily by electronic commerce. For convenience, I’ll call these two subgroups EDS/INF and EDS/EC.

Two strands of work stand out in the EDS/INF work: facilities for remote programming featuring mobile code, and communication languages (such as KQML) featuring high-level semantic primitives. The technology in this area is not that mature, having been developed in the past five (or even fewer) years. Because of this, and the fact that this work entails some measure of standard setting, it will likely take a while before it significantly impacts industry—five years isn’t an overly conservative prediction. However, the potential impact of this work is very significant; it promises to change the way developers, if not users, think about computing. The vision of the platform being the network will eventually come true, and the area of EDS/INF-oriented agents will likely play an important role in realizing this vision.

The main focus of EDS/EC work consists of novel negotiation protocols over the Internet. It is widely recognized that the Internet is changing the way we do business. One particularly important change is how prices are set, with the Internet helping to convert fixed pricing to dynamic pricing. Consumer-based auctions are clearly all the rage, but conventional wisdom has it that the shift to dynamic pricing will have a profound impact on businesses. There is an identifiable interest among computer scientists to devise novel negotiation mechanisms that are network-aware, and this is where much of the EDS/EC work is focused. The technology is more mature than the EDS/INF work, because it builds on years of results from economics and game theory, and some of the results have already influenced the marketplace. The ultimate impact is fairly substantial, though perhaps more on the economy than on the world of computing.

AD. True breakthroughs in AD-oriented agents could bring about the most radical change to the computing world. They could change not only how professionals practice computing, but also how mass consumers conceive of and interact with the technology. However, here it is easy to lapse into wishful thinking. I think it’s fair to say that this area has the least novel technology to point to; it’s based primarily on a change in attitude.

This is a broad statement that naturally admits exceptions. One exception worth mentioning is natural-language (in particular, spoken-speech) understanding, where the technology has advanced to the level of commercial relevance and where companies are investing significant industrial resources. Other aspects of AD-type agents, in particular ones that attempt to capture humanlike behavior at a deeper level, are less mature. Generally speaking, the pressure to bring consumers into the computing world will undoubtedly accelerate innovations in this area, but I hesitate to predict a timeline, if for no other reason than the measure of progress in this area is less obvious. Also, the key driver behind AD work, namely the need to make the complex PC accessible to the masses, might be weakened by the nascent trend toward non-PC appliances.

The bottom line

So what is the conclusion from all this—are software agents a significant concept? If so, which brand of agents? Answering first from the intellectual point of view, I believe that some exciting work takes place today under the software agents umbrella (again, as well as some pretty silly work). All three orientations offer exciting intellectual playgrounds. I am most attracted to the NES-oriented arena. Here I see, on the one hand, solid technology that we can sink our teeth into, and, on the other hand, enough open-endedness to allow for novel paradigm setting and creative metaphor creation.

In contrast, the emphasis in the EN-oriented work will not be on breaking new ground, but rather on perfecting, experimenting with, and customizing existing techniques in the context of specific appli-
In a groundbreaking development, the IEEE Computer Society has joined forces with the American Institute of Physics to jointly produce Computing in Science & Engineering magazine. This new bimonthly publication continues the strong tradition of focused theme issues and topical articles in the spirit of IEEE CS&E, augmented by the comprehensive departments that have long been the hallmark of Computers in Physics.

As both magazines have demonstrated, scientists, engineers, and mathematicians can learn much from each other about advances in computing. The new magazine bridges these disciplines and brings a fuller understanding of both the scientific and the practical aspects of computing.

Also, in addition to the many timely departments and columns we've been presenting to IEEE CS&E subscribers, you now get new coverage on:
- Lab Applications
- Algorithms
- Scientific Programming
- Computer Simulations
- Web Mechanics
- Practical Visualizations

If you haven’t already done so, contact our Customer Service office to subscribe:
- 714-821-8380 (phone)
- 714-821-4641 (fax)
- membership@computer.org (e-mail)

The area of software agents also presents attractive commercial opportunities. As I’ve mentioned, the NES-oriented work already enjoys substantial interest among entrepreneurs and investors. The main interest appears to be related to information retrieval, filtering, and mining, fueled undoubtedly by the exploding number of information sources available online, as well as by the growing number of potential users of this information who have online access. The EDS-oriented work is a higher-risk and higher-reward proposition. The risk results not from technical difficulties but, ironically, from a lack thereof (which limits the possibilities of traditional protection in the form of patents or trade secrets), as well as from greater investment needed and some reliance on having the right industry standards. With a few exceptions, I would be more hesitant to predicate a new commercial venture on AD-oriented work; the uncertainties that make it such an important research area are also the deterring factors from the shorter-term commercial point of view.

Yuvi Shoham received his PhD in computer science from Yale University, and he has been a faculty member in the Computer Science Department at Stanford University since 1987. In 1989, he coined the term agent oriented programming, and he has been struggling to understand it ever since. His work in recent years has taken place at the intersection of AI and game theory. He wrote this article in part while on sabbatical with Interval Research Corporation. Contact him at shoham@cs.stanford.edu.