

# Intelligent Agents on the Internet: **Fact, Fiction, and Forecast**

Oren Etzioni and Daniel S. Weld  
University of Washington

"THE MOST PROFOUND TECHNOLOGIES  
ARE THOSE THAT DISAPPEAR."

—MARK WEISER<sup>1</sup>

**E**LECTRIC MOTORS HAVE EXERTED a dramatic influence on our lifestyle. There are, for example, more than twenty such machines in the average American home. But most people don't notice electric motors in daily life; instead, they focus on the activities the devices enable: washing clothes or listening to a CD. Similarly, computer technology has dramatically enhanced our ability to generate, deliver, and store information. Unfortunately, our tools for locating, filtering, and analyzing that information have not kept pace. Consequently, instead of "disappearing" like electric motors, the "information superhighway" looms larger and more prominent in the lives of its users every day. Intelligent software agents can reverse this trend.

Such agents have two ways of making the information superhighway disappear:

- **Abstraction:** Details of the technology underlying the agent and the resources the agent accesses are "abstracted" away—that is, they are user-transparent. The agent enables a person to state what information he or she requires; the agent determines where to find the information and how to retrieve it.
- **Distraction:** The agent's character or persona helps distract the person from what might be a tedious or complex task.

*IN THE FUTURE, INTELLIGENT SOFTWARE AGENTS WILL  
HELP US NAVIGATE THE INFORMATION SUPERHIGHWAY BY  
SERVING AS BACKSEAT DRIVERS OR TAXI DRIVERS. BETTER  
YET, THEY HAVE THE POTENTIAL TO ACT AS SOPHISTICATED  
CONCIERGES WHO MAKE IT UNNECESSARY FOR US TO  
APPROACH THE HIGHWAY AT ALL.*

**Forecast:** While cute agent personalities will have tremendous impact in the entertainment market, people will not appreciate them in operating systems, utilities, or business applications. Microsoft's "Bob" will become extinct for the same reason that consumers rejected talking cars.

For intelligent software agents, we believe that abstraction will win out over distraction.

The pervasive interaction style for today's computers is direct manipulation: the user clicks on, drags, and drops icons. Although direct manipulation is handy for performing simple tasks such as printing files or invoking applications on a personal computer, it does not scale to searching massive networks for information. Many visionaries recognize

this point. Alan Kay writes:

A retrieval "tool" won't do because no one wants to spend hours looking through hundreds of networks with trillions of potentially useful items. This is a job for intelligent background processes that can successfully clone their users' goals and then carry them out. Here we want to *indirectly manage agents*, not directly manipulate objects.<sup>2</sup> (p. 203)

Nicholas Negroponte uses the task of making one's bed as an illustration:

Today, notwithstanding the skill, I cherish the opportunity of delegating the task and have little interest in the *direct manipulation* of my bed-sheets.... Likewise, I feel no imperatives to

manage my computer files, route my telecommunications, or filter the onslaught of mail messages, news, and the like. I am fully prepared to delegate these tasks to agents I trust as I tend to other matters....<sup>3</sup> (p. 347)

**Facts:** The Information and Interactive Services Report states that more than 10,000 people are signing up for on-line information services each day (New York Times, April 9, 1995). The Internet Society estimates that more than 30 million people worldwide have access to the Internet on any given day (New York Times, May 11, 1995).

Both Negroponte and Kay suggest that direct manipulation is appropriate for enjoyable tasks. Although some people enjoy browsing Internet content, most people would gladly delegate tedious tasks such as searching for information on the Internet to a competent agent.

**Fact:** There are at least 1,500 general-purpose information-finding services (staffed by human information brokers) currently doing business nationwide (Harvard Magazine, May-June, 1995, p. 23).

## What is an agent?

Researchers interpret the term *intelligent software agent* in a number of ways.

By *software agent*, we mean a computer program that behaves in a manner analogous to a human agent, such as a travel agent or an insurance agent. In essence, the term refers to software that supports a social interface metaphor—a dialogue between a person and the agent. Researchers<sup>4-7</sup> have proposed the following characteristics as desirable qualities of software agents:

- **Autonomy:** An agent takes initiative and exercises control over its own actions in the following ways:
  - Goal-oriented:* accepts high-level requests indicating what a human wants and is responsible for deciding how and where to satisfy the requests.
  - Collaborative:* does not blindly obey commands but can modify requests, ask clarification questions, or even refuse to satisfy certain requests.
  - Flexible:* actions are not scripted;

able to dynamically choose which actions to invoke, and in what sequence, in response to the state of its external environment.

—*Self-starting:* unlike standard programs directly invoked by the user, an agent can sense changes of its environment and decide when to act.

- **Temporal continuity:** An agent is a continuously running process, not a one-shot computation that maps a single input to a single output and then terminates.
- **Personality:** An agent has a well-defined, believable personality that facilitates interaction with human users.
- **Communication ability:** An agent can engage in complex communication with

## ALTHOUGH SOME PEOPLE ENJOY BROWSING INTERNET CONTENT, MOST PEOPLE WOULD GLADLY DELEGATE TEDIOUS TASKS, SUCH AS SEARCHING FOR INFORMATION ON THE INTERNET, TO A COMPETENT AGENT.

other agents, including people, to obtain information or enlist help in accomplishing its goals.

- **Adaptability:** An agent automatically customizes itself to the preferences of its user on the basis of previous experience. It also automatically adapts to changes in its environment.
- **Mobility:** An agent can transport itself from one machine to another and across different system architectures and platforms.

Although no single agent has all these characteristics, several prototype agents embody a substantial fraction of them. Currently, there is little agreement about the relative importance of the different properties, but most researchers agree that these are the characteristics that differentiate agents from simple programs.

**Fiction:** General Magic has developed mobile, intelligent agents actively roaming the Internet.

**Fact:** General Magic has developed Telescript, a language for specifying mobile, scripted agents. Actual intelligent agents are still under development. Furthermore, for mobility, each site receiving an agent must be running a Telescript interpreter. Few such sites exist today.

## Types of agents

Here we describe a selective sample of software agents currently under development, to give the reader an idea of the agents that will emerge in the next few years. More comprehensive collections are available.<sup>8,9</sup> We organize our discussion by the agents' sophistication and the degree to which they make the information superhighway disappear.

Following the information superhighway metaphor, an intelligent agent may be a backseat driver who makes suggestions at every turn, a taxi driver who takes you to your destination, or even a concierge whose knowledge and skills make it unnecessary for you to approach the superhighway at all.

**Tour guides.** People often feel lost or disoriented when navigating through the World Wide Web. An agent that may help alleviate this feeling is a tour guide.<sup>10</sup> A tour guide helps the user answer the question "Where do I go next?" Various tour guides have been implemented for relatively small hypertext systems, and prototype systems are being developed for the World Wide Web. For example, WebWatcher interactively advises Web users about which hyperlink to follow next; it learns by observing the user's reaction to its advice.<sup>11</sup> WebWatcher researchers are focusing on developing the mechanisms by which the agent learns good hyperlink recommendations.

Far from making the Web disappear, tour guides draw a person into the Web while attempting to provide a friendlier or even a "customized" experience. Of course, badly designed tour guides act much more like backseat drivers, constantly making annoying suggestions.

**Indexing agents.** The most popular agents on the World Wide Web are indexing agents such as Lycos, WebCrawler, and InfoSeek. Indexing agents carry out a massive, autonomous search of the Web (scanning over a million documents) and store an index of words from document titles and document texts. The user

can then query the agent by asking for documents containing certain key words.

Indexing agents can deliver quick responses, but they have a number of technological limitations. Key word queries can be awkward. For example, it would be difficult to find a reliable Chicago florist or the Boston weather forecast or the White House fax number using key word queries. The indexes are not personalized in any way. As a result, most queries get many false hits. The number of false hits appears to grow quickly with the size of the Web, so indexing agents are unsatisfactory in the face of the Web's expected exponential growth in content. Furthermore, indexing agents fail to consider information stored in databases, servers, and information services that are accessible from the Web but aren't actually Web documents.

Current indexing agents are not selective in their search of the World Wide Web. Quite the contrary, they attempt to be as exhaustive as possible, given their resource limitations. To further extend the information superhighway metaphor, indexing agents are a bit like taxi drivers. They take you to your destination (a particular location on the Web), but there they drop you off and leave you on your own. Furthermore, you may find that you are not at the destination you wanted to reach.

**Forecast:** In the next two years, a new breed of indexing agents will emerge. These agents, which we might call automated information brokers, will choose among existing indexes on the basis of factors such as cost of access, coverage, and speed. Information brokers will prune the results returned by indexing agents, using clues about the contents of each page. For example, it is easy to tell what organization is behind a particular Web document, so we might ask our information broker to show us only product catalogs that originate with AT&T or US West. An automated information broker is currently under development at the University of Washington. (See <http://www.cs.washington.edu/research/metacrawler/>.)

**FAQ-finders.** A far more selective agent emerging on the Internet is one that guides people to answers to frequently asked questions (FAQs).<sup>12,13</sup> We call these agents FAQ-finders, after a project at the University of Chicago. People tend to ask the same questions over and over again. In response, news-

groups, support staffs, and other organizations have developed files of frequently asked questions and their answers. For example, the Gopher FAQ file contains answers to questions such as "What is Gopher?" and "Where can I get Gopher software?" FAQ files have two distinctive features. First, they address an isolated question—for example, "What is the appropriate key sequence for booting my Mac without operating system extensions?" Second, the FAQ archives are highly structured, facilitating agent-based access to the information.

A person approaching the Internet with a question may be convinced that the answer is out there somewhere in a FAQ file but may be unable to locate the appropriate file. FAQ-

*WE THINK OF SOFTBOT AS A  
CONCIERGE BECAUSE IT  
ELIMINATES A PERSON'S NEED  
TO DRIVE THE INFORMATION  
SUPERHIGHWAY AT ALL; THE  
PERSON DELEGATES THAT JOB  
TO THE SOFTBOT.*

finders address this problem by indexing large numbers of FAQ files and providing an interface by which people can pose their questions in natural language. The agent uses the text of the question to locate the appropriate answer. In contrast to indexing agents, FAQ-finders retrieve answers only to questions that appear in the FAQ files they index. Because of the semistructured nature of the files, and because the number of files is much smaller than the number of documents on the World Wide Web, FAQ-finders are potentially much more reliable than general-purpose indexing agents.

**Fiction:** Eventually all information will be available on the Internet.

**Expertise finders.** All the systems described thus far presuppose that the information being sought is directly available on the Internet. But as Kautz, Milewski, and Selman<sup>14</sup> argue, much of the information people are looking for is not available on-line but is only "in people's heads." Therefore, another task

that intelligent software agents can perform is expertise location—putting the questioner in touch with appropriate experts.

Tour guides and indexing agents operate by suggesting locations on the Web to the user. The suggestions are based on a relatively weak model of what the user wants and what information is available at the suggested location. In contrast, the University of Washington's Internet Softbot<sup>5</sup> represents a more ambitious attempt to determine what the user wants and understand the contents of information services. The agents we have described thus far access unstructured documents or semistructured information such as FAQ files. The Internet Softbot tackles a different component of information on the Internet: structured information services such as weather map servers, stock quote servers, Federal Express's package tracking service, the NetFind service, and others.

Because the information is structured, the Softbot need not rely on natural language or information retrieval techniques to "understand" the information provided by a service. Instead, the Softbot relies on a model of the service for the precise semantics associated with information provided by the service. As a result, the Softbot can answer focused queries with relatively high reliability. We do not argue that the Softbot is the ideal software agent, but it illustrates several important themes.

## Case study: Internet Softbot

The key idea behind the Softbot is reflected in its name, which is derived from *software robot*. Instead of a mobile robot's arms and wheels, the Softbot has software commands such as *ftp*, *print*, and *mail* for effectors; instead of a sonar array or television camera, the Softbot uses commands such as *list files* and Internet services such as *Finger* and *NetFind* to gather information. Internally, a least-commitment planner provides behavioral control of the Softbot. Several technical innovations were necessary to make this approach successful in the complex world of the Internet.

**Objective: a concierge.** The Internet Softbot is a prototype implementation of a high-level assistant, analogous to a hotel concierge. In contrast to systems for assisted browsing or information retrieval, the Softbot accepts high-level user goals and dy-

natically synthesizes the appropriate sequence of Internet commands to satisfy those goals. The Softbot executes the sequence, gathering information to aid future decisions, recovering from errors, and retrying commands if necessary. We think of Softbot as a concierge because it eliminates a person's need to drive the information superhighway at all; the person delegates that job to the Softbot. The following are examples of the types of requests that the Softbot handles:

- *Notification requests:* The Softbot can monitor a variety of events—disk utilization, user activity, bulletin boards, remote FTP (file transfer protocol) sites—and communicate its findings to the user in a beep, an on-screen message, an e-mail message, or printer output. The Softbot determines autonomously how to monitor events (that is, which commands to execute and how often to execute them) as well as how to notify the user of its findings.
- *Enforcing constraints:* In addition to filling information-gathering requests, the Softbot can act on the world around it. For example, a user can ask it to ensure that all files in a shared directory are group-readable or that the contents of an archival directory are compressed.
- *Locating and manipulating objects:* The Softbot can compile source code, convert documents from one format to another, and access remote databases. Since users find it convenient to communicate partial descriptions of people, objects, and information sources, the Softbot automatically executes commands until it has disambiguated the request. For example, a request to print a file quickly causes the Softbot to determine the queue size and status (for example, "jammed") of nearby printers. When given a person's name (but not his or her e-mail address), the Softbot uses a combination of *whois*, *netfind*, *staffdir*, *finger*, and other utilities to determine an electronic address.

This list is not exhaustive nor are the tasks mutually exclusive, but they illustrate our main point: The Softbot allows a user to specify *what* to accomplish, while it handles the decisions of *how* and *where* to accomplish it. Thus, the Internet Softbot is an excellent example of a goal-oriented agent.

## For further information

For details about several intelligent-agent projects, browse the following Web sites:

- WebWatcher, a tour guide being developed at Carnegie Mellon University: <http://www.cs.cmu.edu:8001/afs/cs.cmu.edu/project/theo-6/web-agent/www/project-home.html>
- The FAQFinder project at the University of Chicago: <http://cs-www.uchicago.edu/burke/faqfinder.html>
- SIMS, an information mediator being developed at the Information Sciences Institute: <http://www.isi.edu/sims/sims-homepage.html>
- The Autonomous Agent Group at MIT's Media Lab: <http://agents.www.media.mit.edu/groups/agents/>
- The Robotics Research Group at Stanford University: <http://robotics.stanford.edu/groups/nobotics/home.html>
- The Internet Softbot project at the University of Washington: <http://www.cs.washington.edu/research/softbots>

**Specifying goals.** Goal-orientation is useful only if users find specifying requests easier than carrying out activities themselves. The agent must satisfy three criteria to make goal specification convenient for users:

- *Expressive goal language:* If common goals are impossible to specify, a goal-oriented software agent is of limited use. To avoid this pitfall, the Softbot accepts goals containing complex combinations of conjunction, disjunction, negation, and nested universal and existential quantification. This allows specification of tasks such as "Get all of Kambhampati's technical reports that aren't already stored locally." Note that the Softbot can handle this goal even though the underlying service (for example, FTP) does not handle this combination of universal quantification and negation. The Softbot determines which of Kambhampati's reports are new and issues FTP commands to obtain them.
- *Convenient syntax:* Despite the expressive power of mathematical logic, many users are unable (or unwilling) to type long, complex, quantifier-laden sentences. For this reason, the Softbot supplies a forms-based graphical user interface and automatically translates forms into the logical goal language. Users can quickly construct complex quantified goals with a few mouse clicks and a minimum of typing. (Natural language input, an alternative approach pursued by many researchers, is not yet incorporated in the Softbot.)
- *Mixed-initiative refinement dialogue:* Even with a well-engineered interface, it is difficult to specify orders precisely. For this reason, human assistants often engage their superiors in dialogue: "Do you *really* need it delivered by 8 a.m.? It

would cost much less to arrange for delivery by 9 a.m. instead." The current Softbot interface has only limited support for iterative goal refinement, but we are designing a new interface that will allow the Softbot to pose questions (while it continues to work) and allow the user to add information and constraints.

**Softbot architecture.** The Softbot architecture consists of four major modules—task manager (approximately 10% of the code), XII planner (25%), model manager (25%), and Internet domain models (30%)—in addition to miscellaneous support code (10%).

*Task manager.* The task manager resembles an operating system scheduler. It controls all important Softbot activities, including cognitive tasks such as planning and active tasks such as connecting to a Gopher server. As a result, the Softbot can schedule activities for future execution.

*XII planner.* The XII planner is an extension of UCPOP,<sup>15,16</sup> a partial-order planner that handles planning with incomplete information.<sup>17</sup> XII searches the space of plans—partially ordered sets of partially specified actions with both observational and causal effects—until it finds a plan that will achieve the current goal. Since incomplete information requires that the Softbot gather information to determine a plausible course of action, XII alternates planning and execution. For example, when requested to "Get all of Kambhampati's technical reports that aren't stored locally," XII executes actions to locate Kambhampati's host, connects to see what reports are available, executes actions to see which are stored locally, and only then plans the detailed FTP

actions that will achieve the goal. See Golden, Etzioni, and Weld<sup>18</sup> for a description of the innovations necessary for efficient planning in the face of incomplete information.


**Model manager.** The model manager is a specialized database that stores everything the Softbot has observed about the world. To support the XII planner, the model manager implements pattern-directed queries via unification. The most novel aspect of the model manager is its capacity for local closed-world reasoning.<sup>19</sup> Closed-world reasoning—the ability to draw conclusions based on the assumption that one knows about the existence of all relevant objects—is essential for goal-directed behavior.<sup>20</sup> For example, when directed to find the cheapest direct flight, travel agents assume that after accessing their collection of databases, they have information about every relevant flight.

Underlying the Softbot model manager is the insight that closed-world reasoning is both essential and dangerous. Clearly, the Internet is so vast that the Softbot can't assume it knows the contents of every database on every host. However, after accessing the SABRE reservation system, the Softbot must be able to conclude that it has *local* closed-world information—that it knows the price of every flight between the cities in question. The Softbot model manager performs fast inference on local closed-world information; if the user later specifies that the carrier must be United Airlines, the Softbot need not access SABRE again. But if the Softbot is informed of the creation of a new flight, or a change in the desired destination, it retracts its conclusion of local closed-world information and gathers more information.

**Internet domain models.** The Internet domain models provide the Softbot with background information about the Internet. The most important part of these models is declarative encodings of the actions available to the Softbot—that is, descriptions of Unix commands and Internet databases and services. The models also include search control heuristics and background information about people (job title, phone number, e-mail number) and machines (location, performance characteristics).

**Agent properties.** The Softbot has many, but not all, of the desirable properties of agents we discussed earlier. The Softbot is highly autonomous; it is goal-directed, flexible, and

self-starting. In principle the Softbot has temporal continuity, but in practice it rarely survives more than a few days before needing rebooting. We are currently striving to augment the Softbot's rudimentary collaborative, communicative, adaptive, and personality characteristics, but the Softbot is not mobile and we see no reason for it to be so. Given the reach of its effectors, there appears to be no need for the Softbot to transport itself to another host machine; indeed, mobility might compromise the Softbot's ability to keep user goals and preferences confidential.



## JUST AS HUMANS AND ANIMALS ARE AWARE OF CHANGES IN THEIR ENVIRONMENT, SOFTWARE AGENTS MUST BE ABLE TO SENSE THEIR SURROUNDINGS.

### Research challenges

Although working prototypes of several significant intelligent software agents are now in existence, considerable research is necessary before such agents fulfill their potential capabilities. The following research challenges seem most crucial to achieving agents with goal orientation, collaborativity, and adaptivity:

- **Algorithms for planning with incomplete information:** The Internet's dramatic dynamism suggests that these planners must gather information as part of the planning process. The Internet's vast size implies that planners must be extremely efficient, have excellent search control heuristics, or exploit past experiences using case-based techniques. We have used planning technology successfully to guide the Internet Softbot, but the problem of planning with incomplete information is certainly not solved. The Softbot requires extensive, hand-coded search control rules to handle the example tasks listed in our case study section. If it could learn

effective search control rules automatically, the Softbot would achieve many more goals.

- **Multiagent communication techniques:** Software agents must communicate with each other and with their human masters. In particular, they should be able to refine requests and queries through evolving dialogue. It is likely that all effective communication and collaboration techniques will rely on shared encodings of large amounts of common-sense information about computers, networks, useful services, and human activities. But the communication algorithms should allow one agent to convey new vocabulary (or the capabilities of a third agent) to another.
- **Learning algorithms:** Machine-learning techniques will allow an agent to adapt to its user's desires as well as to a changing environment.
- **Cyberperception:** Just as humans and animals are aware of changes in their environment, software agents must be able to sense their surroundings. Machine load, network traffic, and human activities greatly affect which actions a software agent should execute. Agents must also be able to extract facts from unstructured and semistructured documents (perhaps using image-processing and information retrieval algorithms). Until full natural-language processing is a reality, semantic mark-up formats will be crucial. We must provide information service authors a formal description language that makes the contents of their information sources accessible to software agents.
- **Safety systems:** Without a guarantee that it holds its master's goals and interests inviolate, a powerful software agent is more dangerous than desirable. We need techniques of ensuring that an agent will not inadvertently harm its owner while carrying out a request.<sup>21</sup>

**Forecast:** The Internet and the World Wide Web will eventually disappear. They will become invisible layers, much as registers and machine instructions have been supplanted by higher programming languages in today's computers.

**T**HE KEY TO AN AGENT'S SUCCESS is the degree to which it understands the information it is manipulating. One reason the Internet Softbot succeeds is its focus on structured information of the sort found in databases and information services. It is relatively straightforward to represent the semantics of that information to the Softbot.

In the future, naive users, busy executives, and people requesting information over low bandwidth channels (for example, from hand-held computers via cellular links) will refuse to be thrust directly onto the information superhighway. Instead, they will delegate responsibility to reliable, intelligent agents that handle simple tasks without supervision and provide concise answers to complex queries.

## Acknowledgments

This research was funded in part by Office of Naval Research grants N00014-94-1-0060 and 92-J-1946, and by National Science Foundation grants IRI-9357772 and IRI-9303461. We acknowledge the members of the Internet Softbot Group at the University of Washington: Tony Barrett, David Christianson, Terrance Goan, Keith Golden, Cody Kwok, Neal Lesh, Sujay Parekh, Mike Perkowitz, Richard Segal, Erik Selberg, and Ying Sun.

## References

1. M. Weiser, "The Computer for the 21st Century," *Scientific American* (Special Issue: The Computer in the 21st Century), Vol. 6, No. 1, 1995, pp. 78-89.
2. A. Kay, "User Interface: A Personal View," in *The Art of Human-Computer Interface Design*, B. Laurel, ed., Addison-Wesley, Reading, Mass., 1990, pp. 191-207.
3. N. Negroponte, "Hospital Corners," in *The Art of Human-Computer Interface Design*, B. Laurel, ed., Addison-Wesley, Reading, Mass., 1990, pp. 347-353.
4. P.R. Cohen et al., "An Open Agent Architecture," *Working Notes of the AAAI Spring Symp.: Software Agents*, AAAI Press, Cambridge, Mass., 1994, pp. 1-8. To order, contact sss@aaai.org.
5. O. Etzioni and D. Weld, "A Softbot-Based Interface to the Internet," *Comm. ACM*, Vol. 37, No. 7, July 1994, pp. 72-76.
6. O. Etzioni, N. Lesh, and R. Segal, "Building Softbots for Unix" (preliminary report), Tech. Report 93-09-01, Univ. of Washington, Seattle, 1993. Available via anonymous FTP from pub/etzioni/softbots/ at cs.washington.edu.
7. C.A. Knoblock and Y. Arens, "An Architecture for Information-Retrieval Agents," *Working Notes of the AAAI Spring Symp.: Software Agents*, AAAI Press, 1994, pp. 49-56. To order, contact sss@aaai.org.
8. *Working Notes of the AAAI Spring Symp.: Software Agents*, O. Etzioni et al., eds., AAAI Press, 1994. To order, contact sss@aaai.org.
9. *Working Notes of the AAAI Spring Symp.: Information Gathering from Heterogeneous, Distributed Environments*, C. Knoblock et al., eds., AAAI Press, 1995. To order, contact sss@aaai.org.
10. T. Oren et al., "Guides: Characterizing the Interface," in *The Art of Human-Computer Interface Design*, B. Laurel, ed., Addison-Wesley, Reading, Mass., 1990, pp. 367-381.
11. R. Armstrong et al., "WebWatcher: A Learning Apprentice for the World Wide Web," *Working Notes of the AAAI Spring Symp.: Information Gathering from Heterogeneous, Distributed Environments*, AAAI Press, 1995, pp. 6-12. To order, contact sss@aaai.org.
12. K. Hammond, et al., "Agent-Amplified Communication," *Working Notes of the AAAI Spring Symp.: Information Gathering from Heterogeneous, Distributed Environments*, AAAI Press, 1995, pp. 69-73. To order, contact sss@aaai.org.
13. S.D. Whitehead, "Auto-faq: An Experiment in Cyberspace Leveraging," *Proc. Second Int'l WWW Conf.*, Vol. 1, 1994, pp. 25-38. See also <http://www.ncsa.uiuc.edu/SDG/IT94/Proceedings/Agents/whitehead/whitehead.html>.
14. H.A. Kautz, A. Milewski, and B. Selman, "Agent-Amplified Communication," *Working Notes of the AAAI Spring Symp.: Information Gathering from Heterogeneous, Distributed Environments*, AAAI Press, 1995, pp. 78-84. To order, contact sss@aaai.org.
15. J.S. Penberthy and D. Weld, "UCPOP: A Sound, Complete, Partial Order Planner for ADL," *Proc. Third Int'l Conf. Principles of Knowledge Representation and Reasoning*, 1992, pp. 103-114. Available via FTP from pub/ai/ at ftp.cs.washington.edu.
16. D. Weld, "An Introduction to Least-Commitment Planning," *AI Magazine*, winter 1994, pp. 27-61. Available via FTP from pub/ai/ at ftp.cs.washington.edu.
17. O. Etzioni et al., "An Approach to Planning with Incomplete Information," *Proc. Third Int'l Conf. Principles of Knowledge Representation and Reasoning*, Morgan Kaufmann, San Francisco, Calif., 1992. Available via FTP from pub/ai/ at ftp.cs.washington.edu.
18. K. Golden, O. Etzioni, and D. Weld, "Omnipotence Without Omniscience: Sensor Management in Planning," *Proc. 12th Nat'l Conf. AI*, AAAI Press, 1994, pp. 1048-1054.
19. O. Etzioni, K. Golden, and D. Weld, "Tractable Closed-World Reasoning with Updates," *Proc. Fourth Int'l Conf. Principles of Knowledge Representation and Reasoning*, 1994, pp. 178-189.
20. R. Reiter, "On Closed-World Databases," in *Logic and Databases*, H. Guillaire and J. Minker, eds., Plenum Press, New York, 1978.
21. D. Weld and O. Etzioni, "The First Law of Robotics (A Call to Arms)," *Proc. 12th Nat'l Conf. AI*, 1994. Available via FTP from pub/ai/ at ftp.cs.washington.edu.

**Oren Etzioni** is an assistant professor of computer science and engineering at the University of Washington, where he launched the Internet Softbot project in 1991. The Softbot was one of five finalists in the 1995 Discover Awards for Technological Innovation in Computer Software. His research interests include software agents, machine learning, and human-computer interaction. Etzioni received an NSF Young Investigator award in 1993. He received a BS in computer science from Harvard University in 1986 and a PhD from Carnegie Mellon University in 1991. Readers can contact Etzioni at the Dept. of Computer Science and Engineering, University of Washington, Seattle, WA 98195-2350; etzioni@cs.washington.edu.

**Daniel S. Weld** is an associate professor in the Department of Computer Science and Engineering at the University of Washington. He received a Presidential Young Investigator award in 1989 and an ONR (Office of Naval Research) Young Investigator award in 1990. He is an associate editor of the *Journal of AI Research*, was a guest editor of *Computational Intelligence*, and is program chair of the 1996 National Conference on Artificial Intelligence. Weld has published 50 technical papers and two books. He received bachelor's degrees in computer science and in biochemistry from Yale University in 1982 and a PhD from the MIT Artificial Intelligence Lab in 1988. Weld can be reached at Dept. of Computer Science and Engineering, University of Washington, Seattle, WA 98195-2350; weld@cs.washington.edu.