

Intelligent Agent Software for Medicine

Henry Lieberman

Media Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA

Cindy Mason

University of California, Berkeley, CA, USA

Abstract

An important trend for the future of health technology will be the increasing use of *intelligent agent* software for medical applications. As the complexity of situations faced by both patients and health care providers grows, conventional interfaces that rely on users to manually transfer data and manually perform each problem-solving step, won't be able to keep up. This article describes how software agents that incorporate learning, personalization, proactivity, context-sensitivity and collaboration will lead to a new generation of medical applications that will streamline user interfaces and enable more sophisticated communication and problem-solving.

Keywords: Intelligent agents, interface agents, machine learning, collaborative agents, distributed artificial intelligence, multi-agent systems, medical imaging, medical records, medical Web sites

Why Agents?

Medicine is fast becoming a science of information. Information about patient history, symptoms, functions and lifestyle; information about diseases, diagnostics, drugs, and treatment methods play an ever-increasing role. But there can be too much of a good thing.

The dilemma faced by doctors, patients and medical administrators is to find and utilize the relevant information at the right time. Computers, of course, help manage information, but the current generation of computer inter-

faces can't keep growing at the same rate medical information is growing. We can't keep piling on new applications, adding new icons to our screens and new choices to our menus. Sooner or later, our software will have to work smarter, not just add more features.

Intelligent Agent software [2] is a new paradigm of computer software that aims to bring human-like capabilities to software -- such as learning, proactivity, inference, heuristic problem solving, goal-based action and communication. This doesn't necessarily mean that we can make the computer as smart as a person -- that goal is not yet achievable -- but it is a profound change from today's software, which mostly acts like a passive tool, performing fixed data-manipulation functions only on demand.

Interface agents are agents that act directly as assistants to the user of an interactive graphical interface. Like a secretary, travel agent, stockbroker, or medical assistant, they can provide suggestions and help lessen burdens on those that employ them. The best ones provide personalized assistance, remembering past interactions, and using them to infer the user's preferences. They automate commonly performed tasks, constantly improving their behavior and being always on the lookout for opportunities.

Collaborative agents can also work behind the scenes as well as directly in the user interface. Collaborative agents take information from many different sources and locations. They collaborate when multiple agents act as a team in analyzing data and solving problems, bringing multi-perspective and multi-expertise capabilities to bear in solving problems.

Collaborative agents communicate using *agent communication language*. These languages, unlike simple protocols that simply transfer data, contain complex social interactions that reflect a community of agents -- providing help or asking for it, providing and requesting opinions and perspectives, organizing, monitoring, and so forth. Collaborative agents' interactions and dynamic groupings often model the flow and function among systems and people they are intended to help, providing a natural example of form following function and coincidentally, a model of information flow among the organization(s) in which they operate.

The premise of the collaborative agents paradigm is that legacy systems are leveraged as much as possible. Collaborative agents are thus an important enabling technology for rapidly integrating and prioritizing diverse data sources in large, complex distributed information systems. The compositional stance of the paradigm also provides an operational design for growth, allowing hospitals to evolve newly deployed support systems alongside pre-existing information structures, thus ensuring scalability for future information system requirements.

Connecting the personal with the public

One of the most intriguing areas for the use of agents in medicine is in helping to make connections between personal information and public information. Example of personal information in medicine would be information specific to a particular patient, health-care provider, or facility, such as patient history and treatment records, test results, a physician's personal notes, and hospital records. These records are usually stored in private databases. Examples of general information would be medical reference books, medical Web sites (either for the physician or the patient), the medical research literature, and public health statistics. These are stored in databases which are public, or intended for a large audience.

Right now, these sources of information are completely separate, and the task is left to people to explicitly establish these connections. Which medical references pertain to a specific person's condition? Which Web sites should a particular patient look at? Which journal articles help a hospital decide which piece of equipment to buy for their needs?

Integrating private and public information is difficult, because nobody has the time or energy to monitor and consider all sources of potentially relevant information. What people really want to know is "What in this is relevant to me and to my current situation?". We can delegate to the agent the job of at least trying to guess what might be relevant and bring it to our attention, thus potentially saving us time and trouble.

One of the strengths of agents is that they can synthesize partial information that comes from different sources to present an integrated view to the user. Tools like search engines help find information, of course, but everyone

who types the same phrase into a search engine receives the same hits, so a search engine cannot deliver personalized results. The software has got to know something about you.

We'll present some examples of how agents can perform the function of connecting personal and public information. The first will be oriented toward the patient, addressing the question of providing relevant assistance to the ever-increasing number of patients who educate themselves on personal medical topics via the Web. The second scenario, more oriented towards physicians, will show how the physician's immediate context can be used to facilitate semi-automatic annotation and retrieval of images from a database of medical images. The capabilities demonstrated in the patient scenario could, of course, be equally valuable in some circumstances to the physician, and vice versa. Both scenarios are rough, of course, but we hope this will illustrate the potential and power of agent software.

Interface agents can discover serendipitous connections

Search engines are great when you know exactly what you want, but people never quite know exactly what they want, and what they want to know keeps changing with changing conditions, interests and what they discover. Sometimes the most interesting information pops up via *serendipitous connections* -- when two seemingly unrelated pieces of information turn out accidentally to have a strong link.

Medicine is full of cases where finding unexpected relationships between diverse pieces of information is important. Drug interactions is an obvious example -- patients may be taking two different drugs for totally unrelated conditions, but if the two drugs have a bad interaction, knowing that can mean the difference between life and death. Some rare diseases are characterized by an unusual combination of symptoms that most harried physicians might not think to check for.

It is hard to imagine a traditional search engine that could find serendipitous connections automatically, but we have developed a browsing assistant agent, *Letizia* [4], [5], [7] that does just that.

Letizia records the user's choices in a standard Web browser, then reads and analyzes each page that the user sees. It compiles a user profile, a weighted list of keywords, that represents the user's interests. Letizia "follows you around" on the Web, updating this profile dynamically,

Then, when the user is viewing a particular page, Letizia initiates an incremental, breadth-first search starting from each link on the page. That is, it looks first at every page one link away from the user's choice, then two links away, and so on. Every so often, it filters the pages searched through the user's profile and delivers the best one so far as a recommendation. When the user jumps to a new page, so does Letizia's search. This process is illustrated below.

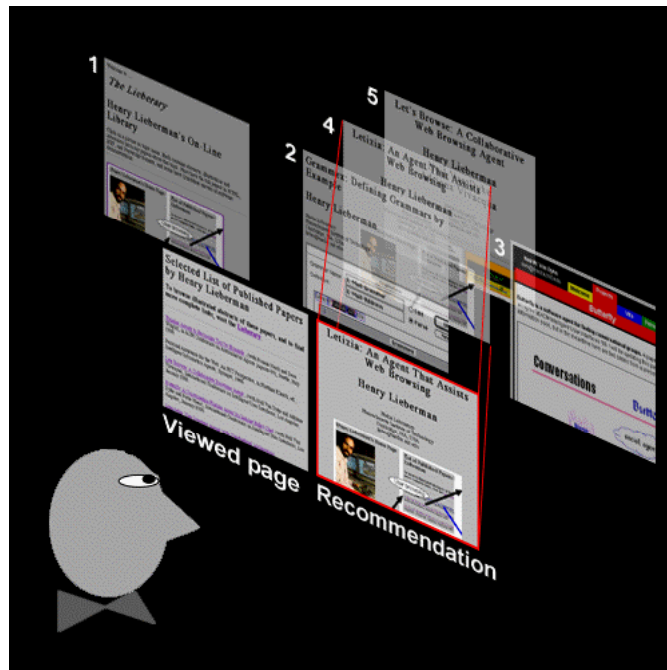


Figure 1. Letizia's search. The user is looking at the page on the left marked "Viewed Page". Linked to it directly are pages 1, 2, and 3. 4 and 5 are two and three links away, respectively. Letizia chooses 4 as best fitting the user's interest, and recommends it.

How does this process discover serendipitous connections? User browsing tends to be "chunky" in the sense that users look at several pages covering related topics for a certain interval of time, punctuated by topic shifts. Just because they change topics doesn't mean that they cease to be interested in the old topic, and so Letizia has the job of maintaining the user's "persistence of interest". When a topic happens to satisfy several of the user's interests simultaneously, even if they are seemingly unrelated, Letizia will award it a high score. That will bring up pages of high interest even if they are a few links away. Let's look at an example.

Suppose we have a female user who likes to browse about health topics on the many Web sites devoted to health. She is a smoker, and would like to quit. She browses several sites that give advice about quitting smoking. From that browsing pattern, Letizia would infer keywords such as the following:

Smoking, tobacco, nicotine, patch, Nicorette, emphysema, cancer, counseling, group, addiction...

Much later, perhaps in a different session, she would examine an unrelated topic, birth control methods. Looking at Planned Parenthood, and other sites, she happens upon articles concerning the pill. The profile would now also contain:

Birth control, condom, pill, diaphragm, estrogen, prostaglandin,...



Figure 2. A patient browses Web sites concerning different topics. Letizia learns an interest profile that includes keywords from both these topics.

Now, again much later, and perhaps during a different session, the user visits a general site containing Medical news, here Healthlink, maintained by the Medical College of Wisconsin. It contains news items on a wide variety of medical topics.

At first glance, none of the headline articles might seem of interest to our user. However, while the user is scanning the headlines, Letizia is busy searching the subsidiary links from the Healthlink home page, without any explicit action on the part of the user. After a short while, Letizia brings up an article it found several links away in the Q & A section, one that warns about the risks of birth control pills to smokers, which our user might not have known.



Figure 3. When the user arrives at the news page, Letizia automatically retrieves an article about birth control and smoking, two topics of high personal interest to the user.

Of course, had she typed "smoking birth control pollen allergies ACL injuries...", and all her other health interests into a search engine, she might have found the same article, but she probably wouldn't have thought to do that. Because smoking and birth control were in her Letizia interest profile, we can bring up the "news she can use". Some sites let you select topics of personal interest via questionnaires, but the problem is people often forget to update them as their interests change. Should our user succeed in quitting smoking or switch birth control methods, static questionnaires wouldn't reflect such changes.

In contrast to a search engine, which might retrieve pages from anywhere on the Web, Letizia's search is localized, starting with the current page and widening over time. In our example, though there might be many pages about the smoking-and-birth-control-pill issue on the Web, Letizia is much more likely to retrieve the Medical College of Wisconsin's article rather than one from an arbitrary source. Because Letizia tends to keep a user "in the neighborhood", sources are most likely to be those with which the patient is already familiar and trusting.

Letizia is an example of how agents that learn personalized information from watching the user's actions, and make suggestions proactively, can help users connect personally relevant information with information that is aimed at a general audience, such as on the Web.

Interface Agents can use context to associate metadata with data

Another function that an interface agent can assume is to use the *context* of the user's work to assist in problem solving. Most of today's applications do not make any significant use of context; your text editor and your spreadsheet don't communicate, even if you are using them both on a common task where they might profitably share information. This is because the current paradigm of software is based on the notion of "applications" -- that each piece of software is a world unto itself and transfers of information occur only via files and cut and paste. This is silly.

A full discussion of the importance of context in user interaction can be found in [6]. In this article, we'll show how agents that help with context can assist in a task of annotation and retrieval of medical images. As the fidelity and accessibility of computer graphics and digital cameras increases, imaging has become more and more important in medicine. Pictures can communicate in a way that words alone cannot. Doctors and hospitals maintain enormous collections of patient images, and soon patients will manage their own collections of their own medical images.

Like any image library, medical images are useful only if you know what they mean, and can retrieve them as needed. Annotating images with metadata information about data, in this case information about the images such as patient identification, time, place, equipment, diagnostic information, etc. is tedious and time-consuming. Similarly tedious and time-consuming is using query applications to retrieve or browse collections. Thus many opportunities to use images are lost.

Our idea was to integrate the annotation and retrieval of images into a single, integrated application, and entrust a software agent, Aria [8] with the job of detecting opportunities for annotation and retrieval of images in the context of the user's everyday work. Aria stands for Annotation and Retrieval Integration Agent.

Aria is composed of a text editor that can be used to compose e-mail messages or Web pages and a photo library, the Shoebox. When the user types in the text editor, Aria is continuously analyzing the text typed and extracting keywords, which are automatically used to retrieve images from the image database. The query is recomputed at every keystroke.

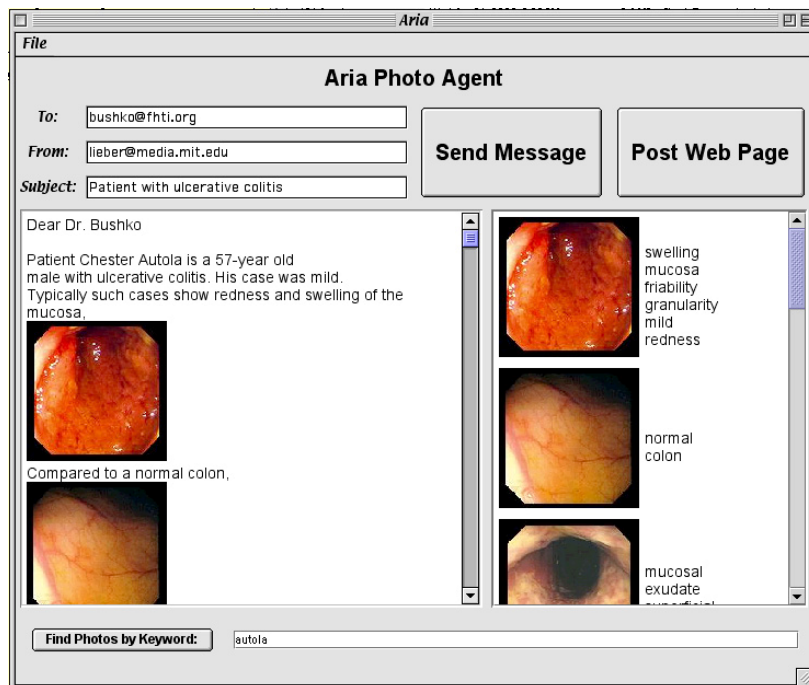


Figure 4. As the physician discusses a case, Aria automatically retrieves relevant images, and uses keywords from the discussion to annotate the image in the database.

Thus as soon as the user types a word, he or she sees the picture column recomputed, ordering the pictures from most relevant top to least bottom. The user can simply drag an image into the text to include it.

Dragging the image into the text also directs the system to scan the surrounding text, and use those keywords as annotations that automatically go back into the database, without any user action necessary.

In the example above, as soon as Dr. Lieberman types "57-year old male with ulcerative colitis" into the e-mail message, pictures from a relevant database were retrieved by Aria. The words "ulcerative colitis" were typed not as a query to a search engine, but to communicate these words to the e-mail correspondent. This "recycling" of user input reduces the total number of user interactions necessary. Why should we retype a query for "ulcerative colitis" when we already told the computer we're interested in it?

Similarly, typing "redness" brings up an image that displays that characteristic, which the doctor inserts to illustrate his point. But he also types "swelling of the mucosa" to describe that picture, so the keywords "swelling" and "mucosa" are automatically associated with the image. He can, of course, edit out annotation that the system gets wrong. The next time he or a colleague types a description of ulcerative colitis with swelling of the mucosa, that image will automatically appear. To make his point, he would like to contrast the patient's colon with a normal colon, and as soon as the words "normal colon" are typed, a picture of a normal colon appears.

Contrast this interaction with unwieldy conventional annotation and retrieval applications. We were able to complete the interaction with:

- No dialog boxes
- No filenames
- No saving and loading files
- No cut and paste
- No queries typed to a retrieval application
- No annotation application

Agent software holds great potential to streamline the user interface by allowing agents to pick up information from its natural context, avoiding the explicit user interactions that would otherwise have to take place.

The previous scenarios all concerned a single agent assisting a single user. We'll now move to consider a broader picture where multiple computer agents, distributed over time and space, assist multiple people working together in an organization. This scenario is supported by the technology of collaborative software agents.

A Night in the Life of a Hospital Agent

The emergence of collaborative agent teams will enable new ways to manage diverse distributed data sources, saving valuable patient and hospital time and reducing potential for errors in tedious procedures. In the following hypothetical scenario, we project how a team of collaborative agents might work through the night in a Los Angeles hospital. Although we have not implemented this particular scenario, the collaborative agent technology behind it is real, and we predict that it will become the future of integrating software in hospitals.

Agents assigned to computers in diagnostics, x-ray, and other testing labs are working around the clock with other agents in the hospital to ensure data integrity, consistency and efficient use of hospital and staff resources, improving patient care and reducing anxiety and conflicts among staff and family members.

What are these agents doing? As afternoon rounds wind to a close, Dr. Brown stops in on his last patient, Betty, who is scheduled for transplant surgery in the morning. Dr. Brown holds Betty's hand and describes what will happen next. Dr. Brown communicates a series of tests to be performed to Hal, the agent in Betty's wristband, that listens while Dr. Brown visits with Betty.

Hal's job is to ensure all procedures and interactions related to Betty go as smooth as possible while she is here. Betty received the wrist band when she was admitted and will wear it until she is released. Hal knows a lot about Betty because he's already talked to the other agents that are down in the pharmacy and in Dr. Brown's office. Hal also knows other agents that might be of use if anything goes wrong with Betty during the night. Hal will know because the agents assigned to monitor her wearable computing devices are working for him. As Hal begins the task of creating the orders, reports, requests and schedules to fill Dr. Brown's order, he consults his knowledge base regarding the surgery and Betty's history. He reasons that before issuing a request for a blood draw at the 2pm rounds, it would be a good idea in this situation to check with the agents in lab records to determine if these tests has been done before on Betty and retrieves any notes or other relevant information regarding the tests.

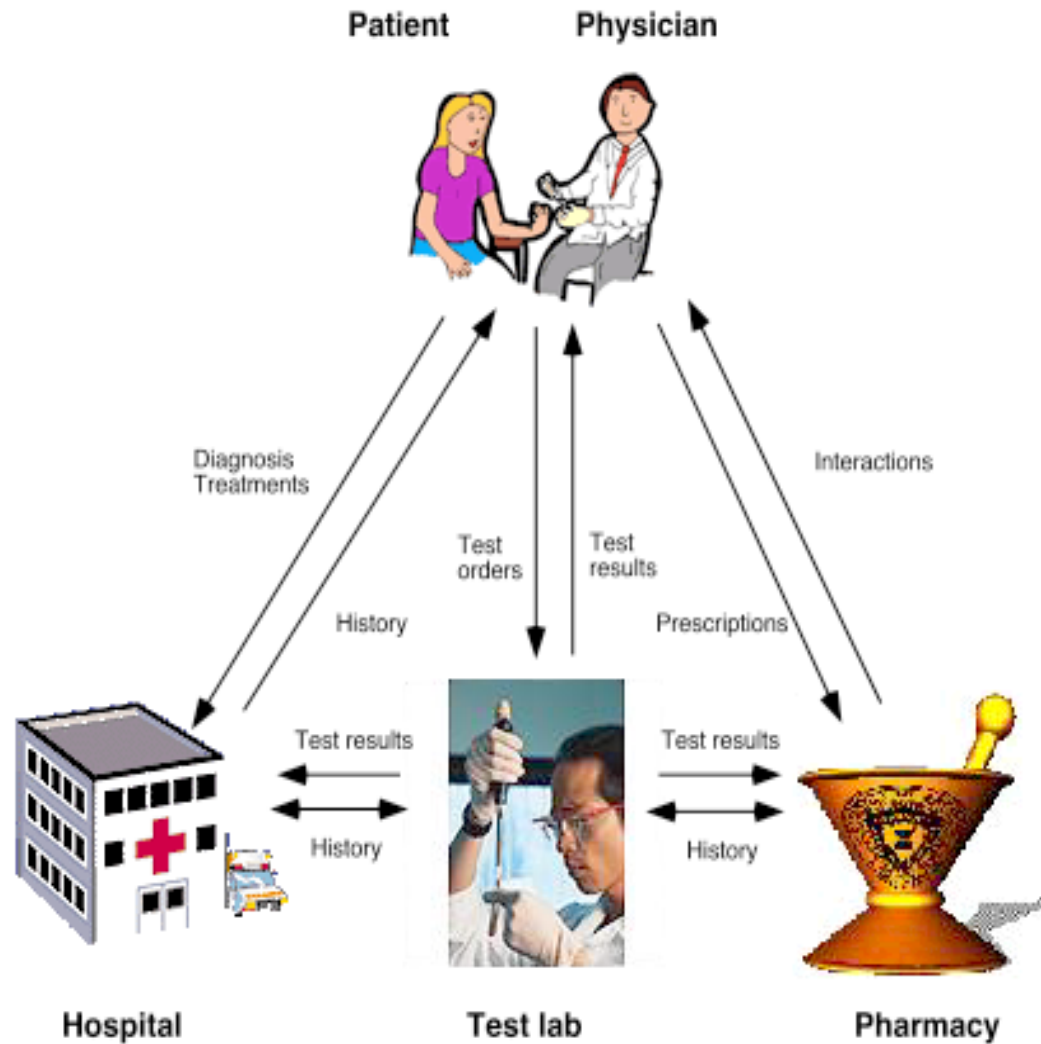


Figure 5. Partial diagram of information flow between software agents

While Hal is waiting for the reply from records to arrive, he also requests information from the pharmacy agent regarding any medications that could interfere with the tests he is about to order and checks these against the ones he knows Betty is on. Hal notices the returned information from the lab, and places a note for the scheduling agent that blood should be taken from Betty's right foot, as her veins have a history of collapsing and there has been difficulty drawing Betty's blood lately. All of the preliminary checks appear in order, so Hal begins tending to other tasks.

Around 3:20am, as results from automated data analysis complete and arrive in Hal's data repository, Hal begins to examine the incoming results, looking to see if there is anything unusual to raise an alarm about.

He notices the CBC sed rate is rather unusual, and requests the test be repeated. The agent down in the lab has received over a dozen requests to repeat this test, so it does a self-monitoring test and realizes its own meter is not functioning. It raises an alarm with the lab attendant who manages to correct the situation and re-start the lab computer. Hal runs a few other routine checks and looks at the trends in monitoring data that were collected during the night before disseminating results and patient status. Hal shares the results with agent managing the patient chart, the agent collating and prioritizing the PCP's morning reports, and the surgery room. Hal's agent protocol ensures the chart agent, the badge agent, and PCP agent pass an identifier consistency check as the patient enters the surgery room at 5:30 am as morning rounds begin.

Collaborative Agents

Notice how different the above scenario is from the way today's medical information systems are organized. Typically, patient records, lab tests, hospital administration records, pharmacy systems, etc. would all be implemented in separate computer systems and the job is left to people to get information in and out of the systems when needed, and fill out forms, translating between different formats when necessary. It has been estimated that over 30% of a physician's time is consumed with routine information entry and information seeking tasks. In our scenario above, these tasks are largely accomplished autonomously by software agents themselves. But to do that, the software agents need to reason about goals, methods, resources, and communication strategies.

The fields of Multi-Agent Systems and Distributed Artificial Intelligence [1, 3] study the principles of communication and processing for collaborative agent problem solving. A focus of research in this area is *agent communication languages*, which express higher level concepts than today's data transfer protocols. These languages allow agents to communicate not only simple transfers of data, but also concepts such as goals and methods, or asking particular agents about their capabilities. Agent toolkits [10] provide general facilities for things like how to find out which agents are capable of solving a given problem, keeping track of which agents are working on which problems, where and when goals are satisfied, sharing partial results, etc. Analysis of agent communication patterns can yield valuable strategic information such as how to manage the tradeoff between expending

effort in advance to determine which agent can solve a problem, or letting several agents try and integrating the results [9].

Conclusion

The new paradigm of intelligent agent software holds great promise in helping medical practitioners and patients deal with the problems of "information overload" that result from both the explosion of available medical information and the complexity of conventional computer software. By helping to connect personal information with public information, utilize information that is implicit in a user's context, and perform collaborative problem solving, software agents might be just what the doctor ordered for the next generation of medical software.

References

1. Bond, Alan H. and Les Gasser (Editors). Readings in Distributed Artificial Intelligence. Morgan Kaufman Publishers. 1988.
2. Bradshaw, Jeffrey, ed. Intelligent Agents, MIT Press, 1996.
3. Ferber, Jacques, Multi-Agent Systems : An Introduction to Distributed Artificial Intelligence, Addison-Wesley, 1999.
4. Lieberman, H. Letizia: An Agent That Assists Web Browsing, International Joint Conference on Artificial Intelligence IJCAI-95, Montréal, August 1995.
5. Lieberman, H., Autonomous Interface Agents, ACM Conference on Computers and Human Interface CHI-97, Atlanta, May 1997.
6. Lieberman, Henry and Ted Selker, Out of Context: Computer Systems that Learn From, and Adapt To, Context, IBM Systems Journal, Vol 39, Nos 3&4, pp.617-631, 2000.
7. Lieberman, Henry, Christopher Fry and Louis Weitzman, Why Surf Alone? Exploring the Web with Reconnaissance Agents, Communications of the ACM, to appear, 2001.
8. Lieberman, Henry, Elizabeth Rosenzweig and Push Singh, An Agent for Integrated Annotation and Retrieval of Images, forthcoming.

9. Mason, Cindy, Introspection as Control in Result-Sharing Assumption-Based Reasoning Agents, International Workshop on Distributed Artificial Intelligence, Lake Quinalt, Washington, 1994.
10. Mason, Cindy, ROO: A Distributed AI Toolkit for Belief-Based Reasoning Agents, International Conference on Cooperative Knowledge-Based Systems, Keele, England, 1994.

About the Authors

Henry Lieberman has been a Research Scientist at the MIT Media Laboratory since 1987. His interests are in using artificial intelligence and machine learning to improve the user interface. He works on Programming by Example, a technique for enabling non-expert users to teach new behavior to a machine by demonstrating example solutions. He is also building agents for the Web, for digital photography, information visualization, programming environments and other applications. From 1972-87, he was a researcher at the MIT Artificial Intelligence Laboratory, where he developed some of the early implementations of the Logo language and graphics systems for educational applications. He worked on the Actors model of object-oriented parallel computing, developing the notion of prototypes and the first real-time garbage collection algorithm. He recently published a book, *Watch What I Do: Programming by Example* (Morgan Kaufman, 2001). He holds a doctoral-equivalent degree (Habilitation) from the University of Paris VI and was a Visiting Professor there.

Cindy Mason is a Assistant Research Engineer at the University of California, Berkeley. Cindy's research interests include intelligent agents, distributed systems, and speech interfaces. She received her Ph. D in 1992 and was awarded Outstanding Student Contribution to the field of Distributed Artificial Intelligence for her work on collaborative belief revision. She was recipient of the National Research Council Research Associateship at NASA Ames Research Center from 1992 to 1995 and a Research Fellow in the Stanford School of Medicine in 1996. She joined the Berkeley Initiative on Soft Computing in 1995. She teaches courses on intelligent agents and is active in the area of multi-agent systems, and the development of human centered computing technologies, including speech interfaces and affective computing.