Attentive Agents

In the anticipated symbiotic partnership, men will set the goals, formulate the hypotheses, determine the criteria, and perform the evaluations. Computing machines will do the routinizable work that must be done to prepare the way for insights and decisions in technical and scientific thinking.

J.C.R. Licklider, 1960

The current broadband infrastructure promises to bring huge volumes of information to us faster than ever before. Even now, reports, data, music, and movies flood our electronic and paper worlds. Once only executives and researchers were required to handle large amounts of information. Today, home users are swamped with information from their broadband connections. It seems most of us live in a kind of semi-organized information soup. How long will the simple tool metaphor of direct manipulation graphical user interfaces (GUIs) support our expanding information needs?

By Paul P. Maglio and Christopher S. Campbell

Executives and other busy people employ someone—an assistant—to relieve them of the effort required to manage daily activities and information, freeing them to focus on and perform more important tasks more efficiently. In fact, in today's information-packed world everyone could use an assistant. But current information systems provide only passive tools for organization and searching. By contrast, assistants actively filter incoming information, communicate in an appropriate manner, and are aware of the supervisor's needs and goals. Moreover, good assistants pay attention, are polite, and are easy to talk to.

Our goal in building attentive agents is to create good assistants. This goal is not new (see [5]). Licklider was perhaps the first to imagine that computers could behave more like assistants than like calculators [4]. People establish goals and computers take action in support of these goals. On this view, computers are active participants in getting work done, and engage with their users in ongoing activity. The nature of this user-computer relationship is very important to understand, as effective communication depends on it.

Consider the relationship users have with most GUI-based applications and tools. Such systems might often seem rude, interrupting the user in the middle of an ongoing task, popping up modal dialogue boxes while the user is typing into a different text field, or using precious computational cycles searching for the latest network driver just as the user urgently searches for the calendar. By all accounts, such systems are bad assistants that no reasonable person would tolerate.

IT SEEMS MOST OF US LIVE IN A KIND OF SEMI-ORGANIZED INFORMATION SOUP. HOW LONG WILL THE SIMPLE TOOL METAPHOR OF DIRECT MANIPULATION GUIS SUPPORT OUR EXPANDING INFORMATION NEEDS?

Attentive Agents

By contrast, attentive agents are computational systems that attend to what users do so they can attend to what users need, just as good assistants do. By closely watching users work with information and modeling the user's state, attentive agents can communicate with users more effectively than non-attentive agents, provide timely and relevant information, and support rapidly changing user interests and goals. For instance, attentive agents might filter information to help manage the user's attention (see the article by Shell, Selker, and Vertegaal in this section), or might scout out information ahead of the user to suggest promising links to follow [3].

To be attentive, agents must collect information about the user as well as information about the world, including what data the user works with, and what physical objects are in the user's environment. Knowledge of the user's activities and environment provides common ground between user and attentive agent, enabling effective and natural communication [2, 9]. Knowledge of user activities allows attentive agents to inform or notify the user of potentially helpful information at opportune moments. For example, a robotic head called PONG senses where the user is looking—and what the user is saying—and responds by appropriately shifting eye gaze and facial expression (see the sidebar by Koons and Flickner). By collecting information about the user's state, an attentive agent such as PONG can communicate engagement and emotion to convey system states naturally.

To learn how to create effective attentive agents that act like good assistants, our work has centered on two main issues. First, we have explored ways to build attentive agents with our Simple User Interest Tracker (Suitor) system, which collects information about users and the world—such as the weather, stock prices, who is in the office today—and uses this information to provide additional information on topics of current interest. Second, we have used the Suitor framework to evaluate methods of displaying such additional information to users, showing significant performance benefits for a specific type of scrolling display over other types of displays.

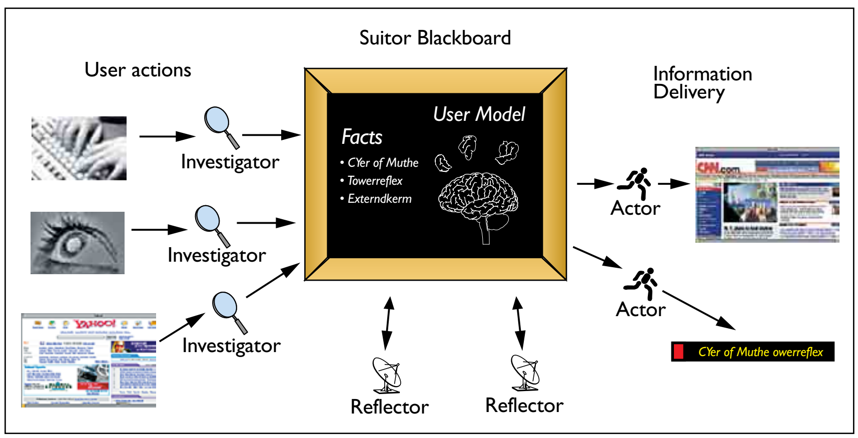


Figure 1. Suitor's architecture includes investigator agents that monitor the user and the state of the world, and add information to the blackboard; reflector agents that make inferences based on what is on the blackboard, possibly developing and maintaining a model of the user; and actors that present information from the blackboard to the user.

Suitor: A Framework for Attentive Agents

We developed Suitor as an extensible framework for building attentive agents [8]. Suitor can be used to create customized agents that monitor user actions, search the world for information (investigators), process user actions or world events (reflectors), and act on the data received (actors). We use Suitor to create individual attentive devices or computers, and to distribute agents across devices in the environment to create attentive spaces. Suitor has also been used to develop applications that perform specific attentive functions, such as task-specific help or Web navigation assistance, as well as large-scale attentive systems that monitor multiple modalities

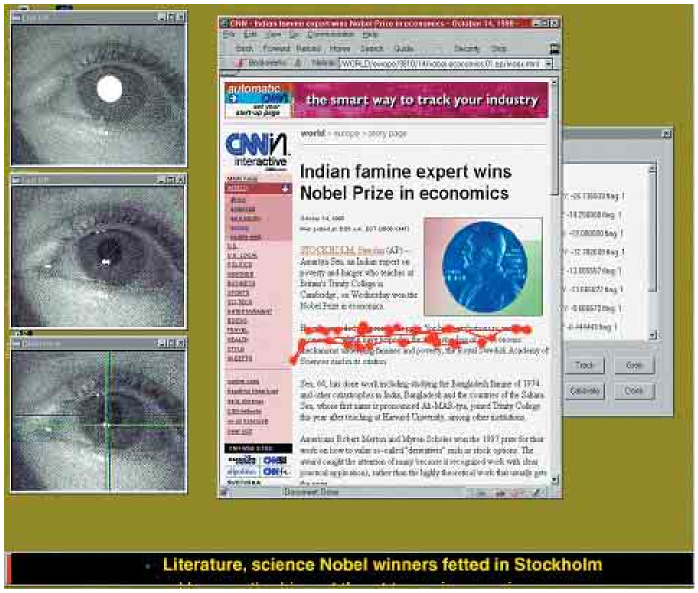


Figure 2. Suitor uses eye tracking to monitor the user's reading activity on a Web page. It then displays potentially interesting headlines of related stories found in the ticker display at the bottom of the screen. The display of eye images on the left is for illustrative purposes only; they allow an experimenter to verify that online gaze tracking is active. At the moment this screenshot was taken, the user was reading the second paragraph on the Web page. The gaze points on the Web page are marked with red dots and are connected by red lines according to the sequential pattern of eye movements. Suitor detects this pattern indicates the user is reading, collects the read text and title of the article, infers that the user is interested in the topic of the article, and automatically displays a related story headline in the ticker window.

and perform complex inference.

Figure 1 illustrates Suitor's architecture. All processing in Suitor revolves around the blackboard—a shared memory and scheme for dispatching information to interested agents. Investigators collect data and post that information on the blackboard. Reflectors receive posted information, process it, and post new information on the blackboard. Actors receive posted information and take action, such as notifying the user. When information is posted to the blackboard, Suitor notifies all reflectors and actors registered to receive that type of information.

Investigator agents gather information from the world outside of Suitor. They monitor user actions, watch Web sites for changes, or scan for database updates. Investigators automatically register themselves with Suitor and post information on the blackboard about the user or about the world. Investigator agents can be created to gather any type of information, including user interactions with the operating system, user identity, and information from network databases. However, investigator agents cannot gather information posted on the blackboard.

In attempting to build effective attentive agents for individual users, we created a variety of investigators that monitor running applications, applications the user is currently working in, keyboard input, mouse movements, Web browsing, Web searching, news information on the Web, stock quotes, and user eye gaze. We paid extra attention to eye tracking as eye gaze is a powerful source of evidence of user information interests [12]. We developed an investigator agent that monitors the user's eye gaze and calculates the coordinates of gaze direction. Data provided by this gaze investigator allowed us to determine whether a user is reading [1] or searching.

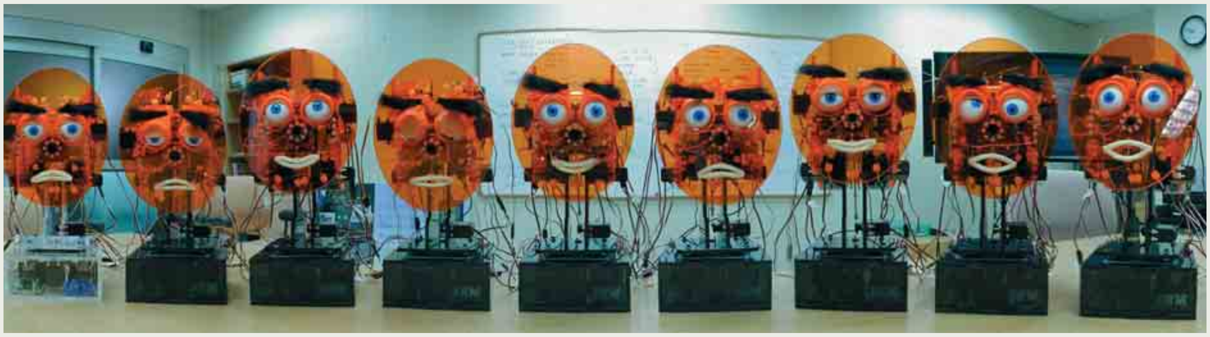
Reflector agents post information to the blackboard and consider what has been posted there by other agents. Reflectors decide what to do about information discovered by investigators and other reflectors. They can construct a model of the user's interests and they can gather information based on the user's interests. For example, text gathered by investigators monitoring interactions with the computer—keyboard input, email received, Web pages read, files opened—can be combined and analyzed to produce key words derived from words that occur more frequently in the pooled text than would be expected given their overall frequency in the language. In this case, the user's current interest can be represented in a user model as a list of words that distinguish the sorts of text being written and read. Investigators constantly post information about what the user is typing and what the user is viewing to Suitor's blackboard. As this information arrives, reflector agents determine word frequencies and update the current list of keywords. As user interests change—as the user shifts attention from one task to another—the key words that represent interests change.

Actor agents are the inverse of investigators: They act on information posted to the blackboard but cannot post information themselves. Actors perform some action on the outside world, such as displaying information to the user. For instance, a scrolling ticker can display headlines to the user based on the list of currently relevant key words in the user model and on additional information gathered from the outside world. Reflector agents prioritize news and other facts that investigators have gathered by comparing them with the user model, and only information that has some overlap with the user's interest is selected for display, ordered by how much overlap exists.

Single-User Scenario

Putting these pieces together, we implemented an application that monitors the user's Web-browsing

PONG: The Attentive Robot



Dave Koons and Myron Flickner

PONG is an attentive agent that watches the user, reacts to user actions, and conveys attention and emotion. PONG expresses happiness on seeing the user and sadness when the user leaves. PONG engages by looking directly at the user and maintaining eye contact during conversation. PONG communicates confusion and surprise to inform the user that it does not understand an action or statement. PONG provides a compelling demonstration of how attentive agents naturally engage and communicate with people.

PONG's namesake ping-pong ball eyes and surgical tubing lips are simple components that create an entertaining demonstration. PONG uses joint audio and video processing [2] to interact with a person. A microphone array is used to orient the head to the sound source. A camera system finds the user's eyes so PONG can establish eye contact. Automatic speech recognition enables PONG to have a conversation. PONG knows its name and age, is very good at arithmetic, and can spell. PONG demonstrations in elementary classrooms have resulted in excellent feedback from future inventors. It is possible build your own PONG (see [1]). C

PONG senses and responds to a user's gaze, voice, and facial expression.

References

1. Koons, D. PONG: The attentive robot. How to build a Pong robot. *IBM Research Report* RJ10213 (2001); www.research.ibm.com/resources/paper_search.shtml.
2. Haritaoglu, I., Cozzi, A., Koons, D., Flickner, M., Yacoob, Y., Zotkin, D., and Duriswami, R. Attentive toys. *International Conference on Multimedia and Expo* (2001).

Dave Koons was a researcher at the IBM Almaden Research Center. **Myron Flickner** is the manager of the Attentive Environments group at IBM Almaden Research Center, San Jose, CA.

activity [6], monitors the user's eye gaze to determine what text the user is reading [1], finds additional relevant information on the Web [8], and displays the additional information in a ticker window. In this scenario (shown in Figure 2), news headlines scroll by in the ticker display at the bottom of the screen, and the user clicks on one of them to show the associated story in the browser window. If the user starts reading a story in the browser (indicating interest), Suitor collects this information and stores it in the user's model. Some time later, Suitor may find a new story related to what was read and show the headline of that new story in the ticker window. If the user wants to continue reading about the topic, the user can click on the related headline with the mouse. This functionality gives the Suitor application many properties of a good assistant. Suitor pays attention to user actions, and uses its observations to provide relevant information. In this case, Suitor can effectively integrate new information into the user's information environment by following the shifting focus of user attention.

Another pivotal attribute of a good assistant is to provide additional information in an unobtrusive manner, for instance, keeping users from being distracted from their primary tasks by information they no longer have interest or in which they have only a passing interest (see the article by Shell, Selker, and Vertegaal in this section). A scrolling ticker display located at the margin of the user's main screen is often intended to be both informative and unobtrusive, suited to display peripheral information—information not central to the current task, but that might be helpful to it or be informative in other ways [7].

The information provided by the attentive Suitor application discussed previously (and shown in Figure 2) may be informative and thus helpful to the user, but it will typically not be central to the user's current task. We found a scrolling ticker display located at the margin of the user's main screen to be unobtrusive yet accessible to the user, and thus suited for displaying peripheral information.

To verify our ticker design, we experimentally evaluated this interface component with respect to its relative informativeness and distraction in comparison with a variety of alternative scrolling ticker displays [7]. Our results show that scrolling tickers in which text scrolls in very rapidly and stops for a time—discretely scrolling tickers—are as informative and less distracting than continuously scrolling tickers. Our results also show that discretely scrolling tickers are as informative and less distracting than non-scrolling or instantly updating tickers. In the case of a continuously scrolling ticker, too much motion in the user's visual field tends to make the display distracting. Some motion, however, turns out to be helpful, as it updates the user about when to schedule glances at the ticker display.

By placing information in the margin of the screen, Suitor can present peripheral information without being distracting. By controlling the display of peripheral information, Suitor can effectively integrate this type of information delivery with the user's shifting focus of attention. Unlike modal dialogue boxes, for example, the ticker display does not reprioritize user activities, and thus minimizes interruption.

Suitor is implemented in Java. It is a framework for developing attentive applications. Developers can create their own agents by extending the appropriate agent type and adding the desired functionality. For example, adding a new sensor to the system means creating a new type of investigator agent with the appropriate code to extract data from the sensor. An instance of this new agent is registered with Suitor automatically when its constructor is called, and all data collected by the agent is posted on the blackboard. Developers can create reflector and actor agents in the same manner by extending these types as appropriate.

Conclusion

Like good human assistants, paying attention to user actions helps attentive agents anticipate user needs, thus providing appropriate help with information management. In developing Suitor, we created a programmable framework that enables easy combination of evidence from sensors—camera, keyboard, application use—to make inferences about user interests and states. By studying how attentive agents notify or inform, we found that certain display characteristics (such as motion) cause distraction and affect user performance. Throughout, we have tried to uncover necessary attributes of attentive systems. Although we have discussed attentive agents that react to users, more proactive agents are also possible.

Attentive agents are assistants that aim to effectively communicate with users, provide timely and relevant information in a nondistracting way, and support rapidly changing user interests and goals. By exploring means for agents to collect data about users and the world, as well as methods to display results to the user using peripheral displays, we have begun to realize this potential to create systems that are helpful and easy to use—systems that meet people's expectations and provide natural modes of communication. C

References

1. Campbell, C.S. and Maglio, P.P. A robust algorithm for reading detection. In *Proceedings of the ACM Workshop on Perceptual User Interfaces* (2001).
2. Clark, H.H. *Using Language.* Cambridge University Press, Cambridge, England (1996).
3. Lieberman, H. Letizia: An agent that assists Web browsing. In *Proceedings of the International Joint Conference on Artificial Intelligence.* (1995), 924–929.
4. Licklider, J.C.R. Man-machine symbiosis. *IRE Transactions on Human Factors in Electronics HFE-1.* (1960), 4–11.
5. Maes, P. Agents that reduce work and information overload. *Commun. ACM 37,* 7 (July 1994), 31–40.
6. Maglio, P.P. and Barrett, R. Intermediaries personalize information streams. *Commun. ACM 43,* 8 (Aug. 2000), 96–101.
7. Maglio, P.P. and Campbell, C.S. Tradeoffs in the display of peripheral information. In *Proceedings of the CHI 2000.* (The Hague, The Netherlands, Apr. 2000). ACM Press, NY, 241–248.
8. Maglio, P., Campbell, C.S., Barrett, R., and Selker, T. An architecture for developing attentive information systems. *Knowledge-based Systems 14,* (2001), 103–110.
9. Maglio, P.P., Matlock, T., Gould, S.J., Koons, D., and Campbell, C.S. On understanding discourse in human-computer interaction. In *Proceedings of the 24th Annual Conference of the Cognitive Science Society* (Fairfax, VA, 2002). Lawrence Erlbaum, Mahwah, N.J. 602–607.
10. McCrickard, D.S., Catrambone, R. and Stasko, J.T. Evaluating animation in the periphery as a mechanism for maintaining awareness. In *Proceedings of the IFIP Conference on Human-Computer Interaction (INTERACT'01).* IOS Press, Tokyo, Japan, 148–156.
11. Vertegaal, R., Slagter, R., Van der Veer, G., and Nijholt, A. Eye gaze patterns in conversations: There is more to conversational agents than meets the eyes. In *Proceedings of CHI 2001.* (Seattle, Apr. 2001). ACM Press, N.Y., 301–308.

Paul P. Maglio (pmaglio@almaden.ibm.com) is a manager and researcher at the IBM Almaden Research Center, San Jose, CA.
Christopher S. Campbell (ccampbel@almaden.ibm.com) is a researcher at the IBM Almaden Research Center, San Jose, CA.