JIM CHRISTENSEN, JEREMY SUSSMAN, STEPHEN LEVY, WILLIAM E. BENNETT,
TRACEE VETTING WOLF, WENDY A. KELLOGG
IBM RESEARCH

Two applications reveal the key challenges in making context-aware computing a reality.

Too Much Information

As mobile computing devices and a variety of sensors become ubiquitous, new resources for applications and services—often collectively referred to under the rubric of context-aware computing—are becoming available to designers and developers. In this article, we consider the potential benefits and issues that arise from leveraging context awareness in new communication services that include the convergence of VoIP (voice over IP) and traditional information technology.

To ground the discussion, we describe two services that IBM has developed and deployed over the past few years to help people communicate more effectively. We examine techniques that have proven effective, as well as problems that remain unsolved. The two services have a somewhat different focus. The first, called Grapevine, helps a person communicate with another individual

using an aggregated and filtered set of contextual information. The second, the IBM Rendezvous Service, helps people meet and talk on the telephone. While people clearly do these things today without additional help from context-aware services, the goals of such services are to allow people to make better communication choices, engage in a richer and more valuable interaction, and waste less time in accomplishing their interactions, while providing significant cost savings to the enterprise.

THE PROMISE AND PERILS OF CONTEXT-AWARE COMPUTING

Many have noted that computing has long since expanded beyond the desktop. For example, Thomas P. Moran of IBM's Almaden Research Center and Paul Dourish of the University of California wrote in 2001:

Computation is now packaged in a variety of devices. Smaller and lighter laptop/notebooks, as powerful as conventional personal computers, free us from the confines of the single desk. Specialized devices such as handheld personal organizers are portable enough to be with us all the time. Wireless technology allows devices to be fully interconnected with the electronic world. Cameras and VCRs are being supplanted by digital equivalents, while we increasingly listen to music on devices that are digital and solid-state. Cell phones are really networked computers. The distinction between communication and computation is blurring, not only in the devices, but also in the variety of ways computation allows us to communicate, from email to chat to voice to video.¹

It is against this backdrop that the almost mesmerizing promise of context-aware computing emerges: cellphones that know when to be quiet; sensors that can judge an opportune time to interrupt. What's next? Perhaps a corporate credit card that can warn if an expense you are about to charge is not reimbursable? While the dream of intelligent devices has been alive for some time in the computer science community, it has not yet had a

profound effect on the applications and services we use to get our jobs done. Why not?

The simple answer is because it is hard to do well—or even well enough. After all, the desktop is a relatively controlled environment, whereas the real world is dynamic and complex. The gap between what technology can “understand” as context and how people understand context is significant. Indeed, some critics have asserted that context-aware computing makes a fundamental error in trying to remove the human from the control loop in creating intelligent autonomous devices.² A different tactic is to capture context but render its results unto humans to decide what actions to take. It is this tack that IBM research has taken in applying context awareness to some of the communications tools that IBM employees use every day. It has not yet had the profound effect that we think is possible (perhaps inevitable), and here we consider some of the reasons for this result.

IBM GRAPEVINE

The Grapevine service (originally described in the February 2004 issue of *ACM Queue*)³ was used inside IBM beginning in 2001, and the research project was completed in 2005. Its most useful features have been absorbed into IBM product offerings. In a nutshell, Grapevine revealed a user's context to observers through a new dynamic program element that was embedded in popular existing applications such as e-mail, instant messaging, enterprise directory, Web sites, etc. This program element was presented to end users as a business card with realtime information, called a Grapevine e-card. Figure 1 is an example of an e-card.

The e-card provided information about the owner's current context and activities. This allowed potential communicators to leverage contextual information in making decisions about when to initiate a contact and via which channel. Sources of context used by Grapevine were personal computers, mobile wireless PDAs, telephones, and motion detectors. The context extracted and derived from these devices included physical location, computer application activity (for example, was the user

currently using IM or e-mail?), and telephone activity.

Making a user's activity visible to others raises privacy and control issues, so Grapevine also provided a simple model for specifying a default level of context-awareness for one's e-card, and person- or group-specific mechanisms for allowing greater "translucence" of context. How did this help people communicate? How well did people cope with the potential issues?

LOCATION AWARENESS IS A GOOD THING

The current or last-known physical location of a person was the most useful information provided by the Grapevine service. Observers used a subject's location not only to select an appropriate means of communication (instant message, e-mail, walk to the office), but also on occasion to decide whom to communicate with. For example, if you find that the person you want to communicate with was last known to be on the other side of the country one hour ago, you might choose another person for an inquiry. Notice that the person's last known location was combined with the time that the subject was last known to be at that location. A timestamp can make "stale" location information useful.

Location was combined with other information to support a communication decision. For example, a colleague may observe that you are online from home in the early morning, and then you go offline. That colleague infers (correctly) that you are on your way to the office, and decides rather than calling to wait to speak with you in person when you arrive.

COMPUTER APPLICATION ACTIVITY IS A MIXED BLESSING

In launching the Grapevine service, we worked hard to determine what computer activity would be relevant to communication decisions. For example, the Grapevine context agent watched for activity on a user's laptop com-

puter in IM and e-mail applications, programming tools, presentation tools, etc. Whenever the agent observed a change in activity, it reported to the central context aggregation service. We believed that observers (with the owner's permission) would be able to use this information to make better-informed communication decisions.

To a limited extent, providing this context functioned as we had imagined. For example, people who were actively using their programming tools reported that they expected their colleagues to refrain from interrupting them with nonessential instant messages during that time. The majority of users, however, did not find application activity context useful for a variety of reasons.

First, users often were not comfortable with others knowing what they were doing. The Grapevine service provided complete control over who could observe which

Users often were not comfortable with others knowing what they were doing.

elements of context, and users commonly blocked *all* others from viewing their computer activity *all* of the time. Although the service allowed observer-by-observer blocking, it was rarely used. This is an area for further research.

Second, people use applications for more than one purpose. This made it difficult to determine a person's situation from looking at his or her use of computer applications. Two examples stand out. The Grapevine context could not only detect people using PowerPoint, but could also differentiate between the edit and presentation modes. Thus, observers could see that a user was "editing slides" vs. "presenting slides." While this promised to be a useful guide for communication decisions, in practice it was common for people to review slides in presentation mode (and not uncommon to present slides in edit mode); hence, observers could not reliably determine when a user was actually giving a presentation.

Even given this uncertainty, Grapevine's ability to automatically suppress incoming instant messages when the user was in presentation mode could be useful in preventing the potential embarrassment of a message

Jim Christensen's E-card

FIG 1

popping up on a projected screen during a presentation. Users might therefore tolerate missing a few IMs even while privately reviewing slides in presentation mode. Another example of multipurpose applications is the IBM corporate e-mail application, Lotus Notes. Because the Notes environment supports a variety of applications in addition to e-mail, when you observed that a Grapevine user was using Lotus Notes, you could not conclude reliably that the user was engaged in e-mailing.

A broader observation was that many people did not care what another person was doing when they wanted to communicate with that person. The most important questions were often, "Can I use instant messaging?" or "Can I call and what number should I use?" The data we gathered from Grapevine usage could not determine why this was the case. Perhaps it has less to do with technology and more with personal habits or culture, but more research is needed in this area.

LESSONS LEARNED

The Grapevine service went through three major revisions over its lifespan. We discarded features that were not used, and reinforced and introduced new features we thought would be useful. Some of the lessons learned are generally applicable to context-aware systems:

Do not expect users to do anything extra to provide context. Context that depends on manual user actions will not be reliable.

There must be a simple, powerful, and intuitive way of giving users peace of mind (and hence control) with respect to the visibility of their context information. If not, users simply will opt out of allowing their contextual information to be used. A substantial number of users expressed concern about the ability of others to track their physical location and/or computer-related activity. We provided a few simple and minimally invasive ways for users to control the visibility of their context information, but none seemed to catch on. People did not use them extensively, and they felt uncomfortable in anticipating and managing the consequences of making personal information available.

People look to instant messaging for realtime context.

While the Grapevine service allowed e-cards to be embedded in e-mail and Web pages, the majority of users found that glancing at their IM client to see enhanced context was the most convenient and useful way to quickly find out about another user's current context.

A substantial semantic gap exists between the information that low-level sensors and programs can detect and the high-level ability and willingness of a person to communicate with someone else.⁴ What computer scientists commonly call *context* often has more to do with technology than with work situations, people, or frames of mind. While low-level information is useful, it is only a rough indicator of a user's social context. Such ambiguity can be socially useful;⁵ nevertheless, care must be taken in presenting and labeling sensor data in the interface. With Grapevine we had a tendency to make assumptions about the overall or high-level activity of end users that were not warranted by the low-level sensor data. Builders of context-aware systems should consider carefully how they map sensor data into end-user views.

IBM RENDEZVOUS

The IBM Rendezvous service allows people to talk in small groups using telephones (to "rendezvous" on the phone) and/or computer applications that provide a telephone function. This is similar to audio-conference calls, and the service appears to be a layer on top of audio-conferencing. Instead of calling directly into an audio-conference, however, a user of the IBM Rendezvous service in effect phones his or her corporate calendar, selects a meeting from it, and enters into a multiparty conversation with the people invited to that meeting. This service is aimed at the highly mobile population of IBM employees who may often be working without convenient access to their normal IT resources (e.g., high-bandwidth access, laptops, corporate applications, etc.).

The Rendezvous project is more recent than the Grapevine project (at this writing, Rendezvous had been used by about 700 people, from corporate executives to technicians, for two to six months), so not as much is known

about the features users will find helpful.

The Rendezvous system incorporates two primary uses of personal context: one is a system that facilitates getting people into their conference calls, particularly when they are mobile; the other is a form of context-awareness that adds information about the other people on a call and the state of the call itself. Context-awareness is served through a visualization known as the *conference call proxy*.

First, because most people use Rendezvous via non-VoIP hardware (that is, via a “normal” telephone device consisting of a microphone, speaker, and standard keypad), context is used to get people into their meetings as quickly as possible and with as little interaction as possible with the Rendezvous IVR (interactive voice response) system (nicknamed “Mr. Menu”). Effective use of context gets people into their meetings in seconds and helps solve communication problems common in today’s corporate environment. Here are some of our techniques:

The credentials of the calling device help identify and authenticate the user. This may be as simple as the caller ID of the user’s personal telephone (for example, his or her cellphone), or more definitive information that can be presented by a call from a piece of software acting as a telephone (a so-called “soft phone”). Since it can take 20 seconds or more to identify and authenticate a caller using an IVR, this is a significant convenience to users.

The time of day is used to select the most appropriate starting point on a user’s calendar. For many users this is easy, as their calendars are sparse lists of time-ordered events. More senior-level employees tend to double- and triple-book meetings into the same time slot, and using the time of day as a starting point to allow scrolling through the day’s events appears to be an effective and understandable technique.

The calendar of the meeting chairperson is used to detect meetings that have been postponed or rescheduled. It is not uncommon for a mobile user to be surprised by a change in the time or date of a meeting. By calling into a context-aware calendar service, such users need not listen to music on hold for several minutes to discover that the meeting has been rescheduled. Instead, the caller can be told immediately that the chairperson has rescheduled the meeting without wasting time or interrupting others. (Interestingly, the data showed that a relatively high percentage of conference calls have only a single user. Such calls are expensive not only in cost to the organization but also in wasting the user’s time.)

The caller’s corporate or social network can resolve confusion about meetings or access problems. Users of the IBM Rendezvous service necessarily rely on their

corporate calendars as their links to meetings; indeed, as described previously, the context provided by the calendar allows for easier authentication and access to meetings. But what if for some reason the Rendezvous system fails to get the user into a phone meeting? The Rendezvous service allows such a user to send an instant message to a colleague who may be able to help resolve the problem. This service is known as iHelp (Instant Help), explained in more detail in the next section.

These techniques describe how the Rendezvous service is integrated with corporate resources such as caller ID, calendar, and instant messaging. By taking this approach, Rendezvous potentially realizes value for its users by enhancing the effectiveness of meetings run by conference call, but only if it can get people into their calls as efficiently as the usual method of looking up the conference call number and passcode and dialing. Preliminary evaluation data suggests that the system is meeting users’ expectations in this regard.⁶

Beyond getting into calls easily, much of Rendezvous’ added value is conveyed through the conference call proxy (figure 2), a visualization available to users with access to their laptops during a call. The conference call proxy provides information about the call and its participants. Specifically, the proxy shows: (1) how much time is remaining in the meeting; (2) authenticated users who are on the call, the meeting host (starred name), who is speaking (users with speech bubbles by their names), and who is on mute (grayed-out names, not shown); (3) which participants have not yet arrived; and (6) recent

The Rendezvous Conference Call Proxy

FIG 2

actions. In addition, users can access the corporate directory information for anyone on the call (4,5) or initiate instant messaging sessions (by double-clicking on a name). The meeting host can “lock” the meeting so that no one else may join.

INSTANT MESSAGING ON PLAIN OLD TELEPHONES

Rendezvous is targeted at busy corporate users who may not be willing to “peck out” instant messages on a cellphone. Thus, Rendezvous provides a way of using instant messaging from plain old telephones without a lot of “pecking.” The iHelp service uses a caller’s corporate network, social network, and calendar to determine a manageable set of people who might be able to help a caller get information on an upcoming (or currently underway) meeting on any given day. When a caller to the Rendezvous service asks to send an instant message for help getting into a meeting, the service determines a moderate-size set of people who may be able to help. The Rendezvous caller is then asked to press three keys on his or her telephone keypad corresponding to the first three letters of the last name of the IM recipient. In the rare instance where two or three names match the three-key sequence, an additional prompting is used to select the target name.

Once the target is identified, a precompiled instant message is sent to that person. (The obvious extension of letting the person requesting help attach a short voice clip to the instant message has been planned, but is not currently offered to users of the Rendezvous service.) The instant message identifies the person requesting help and provides for fixed actions on the part of the respondent that will result in communication. Those fixed actions allow one of the following to occur:

- The person requesting help is pulled into a telephone meeting already in progress.
- The person requesting help is transferred to the telephone of the person who received the instant message.
- A short textual response to the instant message is played back (after a text-to-speech conversion) to the person requesting help.

Although the Rendezvous service has not been used long enough to determine the value of iHelp to a large population, people have used it to resolve questions about pending meetings. In addition, this appears to be a feature that provides for communication between devices that are not often paired (i.e., plain old telephones and laptop computers) and could be used to solve communication problems not involving meeting scheduling. We expect to do more research in this area.

TELEPHONES, SMART CLIENTS, AND MOBILE DATA DEVICES

IBM Rendezvous is an early example of a service that allows IBM employees using a plain old telephone to access other corporate resources such as calendars and instant messaging. Since the user interface on plain old telephones is more constrained than on personal computers or PDAs, the use of context to simplify the user experience is all the more important. While more powerful PDAs and cellphones with richer data-based applications provide an alternative way of accessing the same resources, a solution that allows users to participate from plain old telephones is needed in the current deployment context, and for the foreseeable future. This allows Rendezvous to accommodate mobile users under the widest array of circumstances.

CONCLUSION

Working out the problems and promise of context-aware applications and services depends on a complex interplay in a moving landscape of technical, organizational, social, and cultural factors. These include what is technically feasible in terms of the kinds of contextual information available; what is practically feasible in terms of assumptions that can be made about the distribution and nature of devices, bandwidth, and cost; what is possible within the constraints of our imaginations; and what will be perceived by users as valuable, as well as socially and culturally appropriate. For this reason, experience with deploying real applications and services at a realistic scale is essential.

As Moran and Dourish eloquently put it:

One goal of context-aware computing is to acquire and utilize information about the context of a device to provide services that are appropriate to the particular people, place, time, events, etc....However, this is more than simply a question of gathering more and more contextual information about complex situations. More information is not necessarily more helpful. Further, gathering information about our activities intrudes on our privacy. Context information is useful only when it can be usefully interpreted and it must be treated with sensitivity....Context awareness is fine in theory. The research issue is figuring out how to get it to work in practice.

We couldn't agree more. Q

REFERENCES

1. Moran, T. P., Dourish, P. 2001. Introduction to this special issue on Context-Aware Computing. *Human-Computer Interaction* 16(2-4): 1-8.
2. Erickson, T. 2002. Some problems with the notion of context-aware computing. *Communications of the ACM*, 45(2): 102-104.
3. Richards, J., Christensen, J. 2004. People in our software. *ACM Queue* 1(10): 80-86.
4. Hudson, J., Christensen, J., Kellogg, W. A., and Erickson, T. 2002. "I'd be overwhelmed, but it's just one more thing to do:" Availability and interruption in research management. In *Human Factors in Computing Systems, the Proceedings of CHI 2002*. New York: ACM Press.
5. Erickson, T. 2003. Designing visualizations of social activity: Six claims. In *Human Factors in Computing Systems, Extended Abstracts of CHI 2003*. New York: ACM Press.
6. Sussman, J., Christensen, J. E., Levy, S., Bennett, W. E., Vetting Wolf, T., Erickson, T., Kellogg, W. A. 2006. *Rendezvous: Designing a VoIP conference call system*. Submitted manuscript.

LOVE IT, HATE IT? LET US KNOW

feedback@acmqueue.com or www.acmqueue.com/forums

JIM CHRISTENSEN joined IBM's Thomas J. Watson Research Center in 1983 where he has worked on projects in programming environments, tools for understanding program execution, digital imaging and federated multimedia libraries, and context-aware applications that help people communicate. He has held both management and engineering

positions with IBM and has received numerous awards.

JEREMY SUSSMAN is a research staff member at IBM's Thomas J. Watson Research Center, where he is part of the Social Computing Group. His current work involves designing, building, and evaluating CMC (computer-mediated communication) systems to support groups and organizations. He has an M.S. and Ph.D. in computer science from the University of California, San Diego, and an A.B. in computer science from Princeton University. Prior to graduate work, Sussman worked on an object-based repository for a CASE tool. His research interests include CSCW (computer-supported cooperative work), user-interface design, fault tolerance, and distributed computing.

STEPHEN LEVY is a consulting design product engineer at IBM's Thomas J. Watson Research Center. He joined IBM in 1979 as a systems engineer in the General Systems Division, developing telephony and voice store and forward products for the IBM Series/1. Currently he is working on developing new conferencing applications and user interfaces to support conferencing as part of the Rendezvous project.

WILLIAM BENNETT is a research staff member at IBM's Thomas J. Watson Research Center. He joined IBM in 1977 as an electrical engineer and has held numerous technical and management positions. His professional technical interests include use of context in personal communications systems, systems architecture and networking in distributed systems, and automated deployment and administration. Bennett holds three engineering degrees from the University of Cincinnati and Boston University.

TRACEE VETTING WOLF is a designer at IBM's Thomas J. Watson Research Center. She has 18 years of professional design experience in visual communication, architecture, and interaction design. She has been with IBM Research for nearly six years, during which she has been the lead designer for the Social Computing Group. Her research interests include understanding how to create a sense of place in online environments and how to support interactions appropriately within such spaces, emphasizing collaborative experiences in an online social setting. She recently joined the Learning Department at IBM Research to pursue research in adaptive learning simulations. She holds a B.S. in graphic design and an M.Arch from the University of Minnesota.

WENDY KELLOGG is manager of the social computing group at IBM's Thomas J. Watson Research Center, where her work focuses on computer-mediated collaboration systems for supporting work in organizations. She holds a Ph.D. in cognitive psychology from the University of Oregon. She is a member of IEEE and the *ACM Queue* editorial board, and was elected ACM Fellow in 2002.

© 2006 ACM 1542-7730/06/0700 \$5.00