

Context-Aware Communication

Anand Ranganathan, University of Illinois at Urbana-Champaign
Hui Lei, IBM T.J. Watson Research Center

Until the invention of writing about 5,000 years ago, people primarily communicated using speech and nonverbal gestures. Since then, technological advances such as the telegraph, the telephone, radio, television, and the Internet have made it possible to communicate increasingly complex ideas faster and across greater distances. Recent developments in computing and mobile networking technology have spawned numerous communication devices, each with its own features and properties, that provide unprecedented flexibility.

Nevertheless, personal interaction remains the most effective means of communication. When face-to-face, people use many subtle cues—changes in posture, facial expressions, hand and arm motions, vocal inflections—to enrich their conversation. Also, because they occupy the same physical space, persons engaging in a conversation share an intuitive awareness of relevant circumstances, objects, or conditions in the surrounding environment.

Much of this context is lost in device-mediated communication. For example, text-based interactions such as e-mail and instant messaging (IM) do not convey emotion well. Providing appropriate contextual information would greatly enhance such communication as well as prevent misunderstandings. It would also help determine



Augmenting device-mediated communication with contextual information would enrich user interaction.

the best type of device to use at a given time—a cell phone is useful when you're on the move, but using it in a theatre or while participating in a meeting is inappropriate.

With the advent of pervasive computing, sensors are now available that can detect a wide range of information about the user's

- physical context such as location and time;
- environmental context such as weather, lighting, and sound;
- personal context such as health, mood, and schedule;
- social context such as group activity and whether others are present; and
- application context such as e-mail sent or received and Web sites visited.

Researchers have proposed a number of architectures for helping applications obtain such data through, for example, synchronous query or asynchronous callback options. We have created two

systems, Mercury and ConChat, that demonstrate the usefulness of augmenting device-mediated communication with contextual information.

MERCURY

Each of the many communication devices available today has advantages and disadvantages depending on the user's preferences and situation. Contextual information would therefore be helpful in determining both the most appropriate device—cell phone, pager, laptop—and the preferred communication format—e-mail, facsimile,

Short Message Service—for a particular time.

Unlike prior systems that allowed only unified one-way messaging, Mercury supports unified two-way communication. A person can initiate a conversation with another party using any available device. The system routes the call based on which device the other party prefers to use in a given context. For example, a user can specify that if he is in a meeting, the system should route all incoming calls to an IM client. The two parties then talk to each other, possibly on heterogeneous devices.

Mercury uses the session initiation protocol as the underlying mechanism for creating, maintaining, and terminating sessions. As Figure 1 shows, each device type has a SIP-addressable entity called a *device agent* that can send and receive SIP requests. These device agents relay conversational messages to and from other device agents and, if necessary, translate the messages into different formats or languages. All session creation messages pass through the *Mercury engine*. This

component obtains the current user context from the *context service* and, based on the user's preferences, routes the call to the appropriate device agent.

Mercury supports both two-way devices such as cell phones and one-way devices such as pagers. If a user is not reachable via a two-way device, the system can route an incoming call to a one-way device through the *soft ring* feature. For example, if the call recipient is away from her office and not running an IM client, Mercury can page her. If the recipient wants to accept the call, she can make herself available on a two-way device by, say, logging onto Lotus Sametime or supplying an alternate phone number. Mercury will then redirect the call to this two-way device.

To reduce the burden on the caller, Mercury lets one party determine whether the other is currently reachable on a two-way device; if the person receiving the call is not available, the caller can request to be notified of a change in status. The system also supports automatic migration of a call to a different device based on the user's context, which can change during a communication session. For example, a person talking on a cell phone while on the way to work might want to continue the conversation on a desktop IM client after arriving at the office.

Upon notification of any changes in context by the context service, the Mercury engine checks the user's preferences to determine whether the change warrants a switch in devices and, if so, queries the user for confirmation. If the user approves the switch, Mercury places the original call on hold, initiates a new call to the other device, and terminates the original call.

The system can modify other session properties based on context. For example, if the user receives a highly anticipated call from another party during a communication session, Mercury can automatically place the original call on hold and patch through the more important one. In addition, it can allocate more or less bandwidth to a video

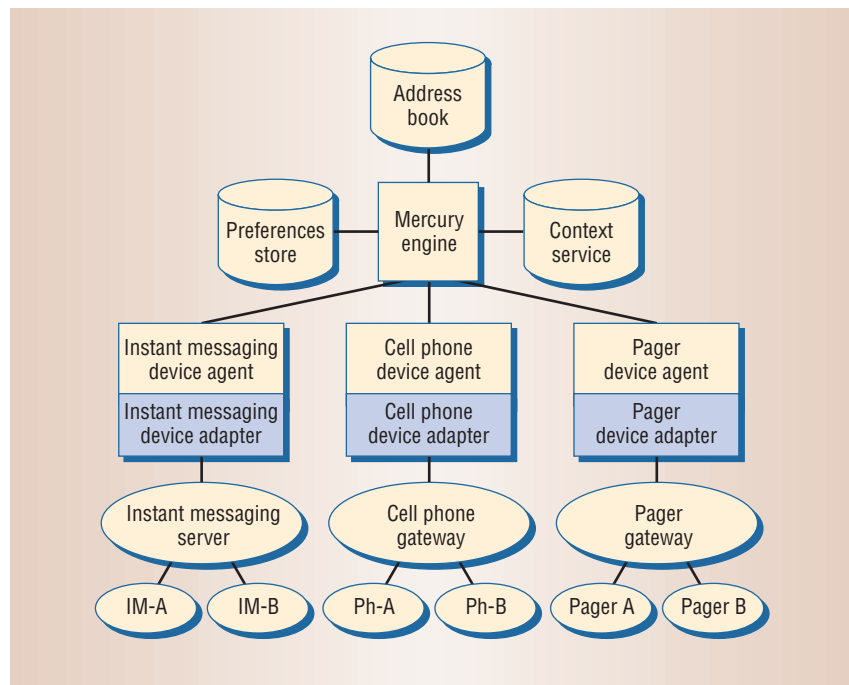


Figure 1. Mercury architecture. The Mercury engine obtains the current user context from the context service and, based on the user's preferences, routes the call to the appropriate device agent.

call based on changing resource requirements.

To protect users' privacy, Mercury does not reveal the actual device the parties are using while communicating. In addition, the context service reveals contextual information only to authorized entities such as the Mercury engine.

CONCHAT

When people use devices to communicate, they are usually unaware of the other party's context. Building on previous research such as the work on affective computing at MIT's Media Laboratory (<http://affect.media.mit.edu/>), we have developed ConChat, a context-aware chat application.

In addition to exchanging text messages, ConChat lets two users communicate a broad range of contextual information such as

- their location;
- the number and identities of other people in either room;

- their rooms' ambient temperature, lighting, and sound;
- other applications and devices running in either room;
- their mood—for example, happy, sad, or excited;
- their personal status—for example, on the phone or out to lunch; and
- any activity in the room—for example, a meeting, lecture, or presentation.

Users can query for their chat partner's context through a side channel as well as subscribe for notifications when the other party's context changes. To protect their privacy, users can modify an access control table that determines whether or not a particular user can view a specific context.

ConChat uses a contextual model based on first-order predicates: It can infer abstract contexts from sensed data and uses Prolog-like mechanisms to evaluate queries and perform reasoning. For example, it uses various

cues such as the number of people, applications that are running, and ambient sound level in the room to deduce activity through rules written in first-order logic.

The system tries to ensure that both users have the same understanding of what one party says by automatically tagging certain words or expressions based on context. For example, if two users are in different time zones, the application can tag any reference to time with the time zone of the party typing the message. Thus, if the sender types "10:00," ConChat can tag this as "10:00 (! -8:00 (PST) !)" to let the receiver know that the sender's time zone is two hours ahead. In this way, the parties can recognize any differences and, if necessary, take proactive measures to minimize misunderstandings that may arise from those differences.

The field is ripe for further research in context-aware communication, particularly in developing better user interfaces, using more varied contexts, and in extending these functionalities to multiple parties. An ideal communication mechanism would combine elements from both Mercury and ConChat by using context to set up and maintain sessions as well as facilitating the exchange of contextual information between parties to enrich interaction.

Because users may want to restrict access to information about their location, mood, or activity at a given time, any such system must have mechanisms to prevent the disclosure of contextual information to unauthorized third parties. It should therefore be translucent rather than fully automated, using contextual information

as well as individual preferences to proactively make certain decisions on behalf of the user but without removing final user control. ■

Anand Ranganathan is a graduate research assistant in the College of Engineering's Department of Computer Science at the University of Illinois at Urbana-Champaign. Contact him at ranganat@uiuc.edu; <http://choices.cs.uiuc.edu/~ranganat>.

Hui Lei is a research staff member at the IBM T.J. Watson Research Center in Yorktown Heights, N.Y. Contact him at hlei@us.ibm.com; www.research.ibm.com/people/h/hlei/.

Read articles on these diverse topics in *Computer* in 2003

outlook: looking ahead to future technologies
january

commercial workload evaluation
february

grid computing
march

hardware/software codesign
april

the changing face of networking
may

agile software development
june

nanotechnology
july

piracy and privacy
august

mobile systems
september

web services
october

safety-critical systems
november

power-aware computing
december

To submit an article for publication in *Computer* on these or other topics, read our author guidelines at <http://computer.org/computer/author.htm>.

Innovative Technology for Computer Professionals
Computer

IEEE
COMPUTER SOCIETY