Application Design for Wearable and Context-Aware Computers

To address mobile-application design challenges, the authors created four user interface models that map problem-solving capabilities to application requirements.

Asim Smailagic and Daniel Siewiorek, *Institute for Complex Engineered Systems and Human Computer Interaction Institute, Carnegie Mellon University*

Pervasive or ubiquitous computing succeeds when it helps people access and use information on or off the job. Even at work, many people don't have desks, or spend most of their day on the run. A mobile system must therefore make information readily available at any place and time. The computing system should also be aware of the user's context so that it can respond appropriately to users' cognitive and social state and anticipate their needs.

Carnegie Mellon's Wearable Computers project (www.wearablegroup.org) is developing small-footprint computing systems that people can carry or wear to interact with computer-augmented environments.[1] We've designed and built more than two dozen wearable computers over the past decade and have field-tested most of these. The application domains range from inspection, maintenance, manufacturing, and navigation to on-the-move collaboration, position sensing, and real-time speech recognition and language translation.

We've developed a taxonomy of problem-solving capabilities for wearable and context-aware computers developed from our iterative design methodology with a wide variety of end users, mainly mobile workers. Here we illustrate the taxonomy with wearable systems whose capabilities range from basic stored-information retrieval to synchronous or asynchronous collaboration to context-aware platforms with proactive assistants. Example evaluation methods show how user tests can quantify wearable systems' effectiveness.

Wearable computing challenges

Today's computer systems aren't as effective as they could be because they distract users in many ways and can easily overwhelm users with data. Effective human–computer interaction design therefore requires interfaces that preserve human attention and avoid information overload.

By bringing wearable computers into a growing number of application areas, we've improved their interfaces to better meet this need. The family tree of CMU wearable computers (see Figure 1), modeled after the US National Science Foundation family tree of early computers, classifies wearable computers into application categories and presents their development over the past decade. Each name represents a distinct wearable computer placed under its corresponding application domain. The four starred designs (VuMan 3, MoCCA, Digital Ink, and Promera) have earned international design awards.

Figure 2 shows CMU wearable computers, ranging from proof of concept, to customer-driven systems based on a task specification, to visionary designs predicting wearable computers' future form and functionality.

We use techniques such as user-centered design, rapid prototyping, and in-field evaluation to identify and refine paradigms that will prove useful across many applications.[2,3] These paradigms build

1536-1268/02/$17.00 © 2002 IEEE

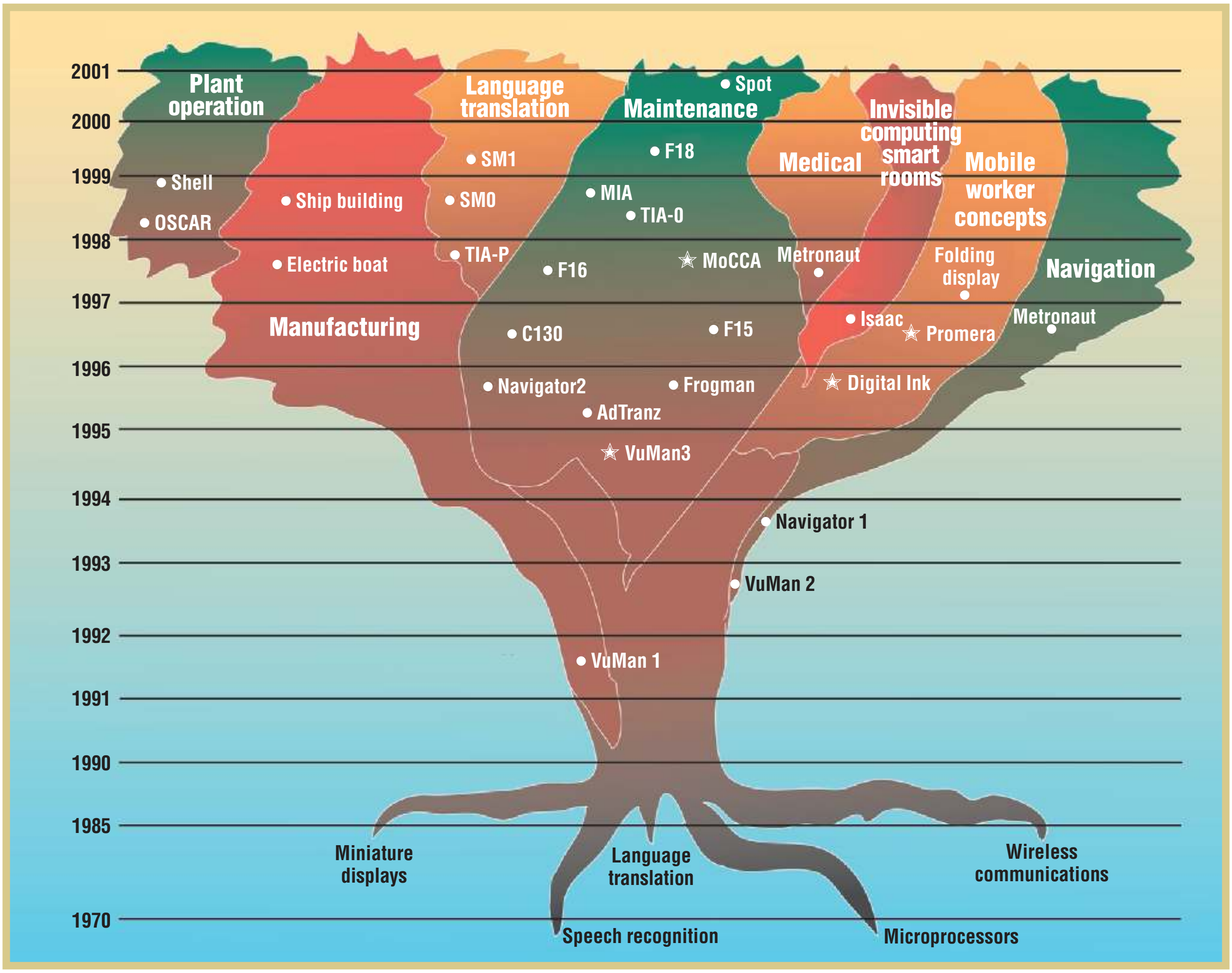

Figure 1. Family tree of Carnegie Mellon wearable computers (F15, F16, F18, and C130 are airccraft programs).

on the notion that wearable computers should merge the user's information with his or her workspace, blending seamlessly with the user's existing environment and providing as little distraction as possible. This requirement often leads to replacements for the traditional desktop paradigm, which generally require a fixed physical relationship between the user and devices such as a keyboard and mouse. Identifying effective interaction modalities for wearable computers and accurately modeling user tasks in software are among the most significant challenges in wearable-system design.

Table 1 shows that a new user interface takes about 10 years to become widely deployed. Defining and refining user interface models requires extensive lab and end-user experimentation to filter out technology bugs, reduce costs, and adapt applications to the new user interfaces. Speech and handwriting recognition are approaching mainstream status after years of development. In the near future, position sensing, eye tracking, and stereographic audio and visual output will enhance 3D virtual reality information, but these technologies won't be fully developed and deployed for at least another decade.

Mobile system design principles

Mobile systems must balance resource availability with portability and usability. We've identified four design principles for mobile systems.

User interface model. What metaphors can we use for mobile information access—what is the next "desktop" or "spreadsheet"? These metaphors typically take over a decade to develop (that is, the desktop metaphor started in early 1970s at Xerox PARC and more than a decade passed before it was widely available to consumers). Extensive experimentation with end-users is required to define and refine these user interface models.

Input/output modalities. Several decades of computer science research on modali-

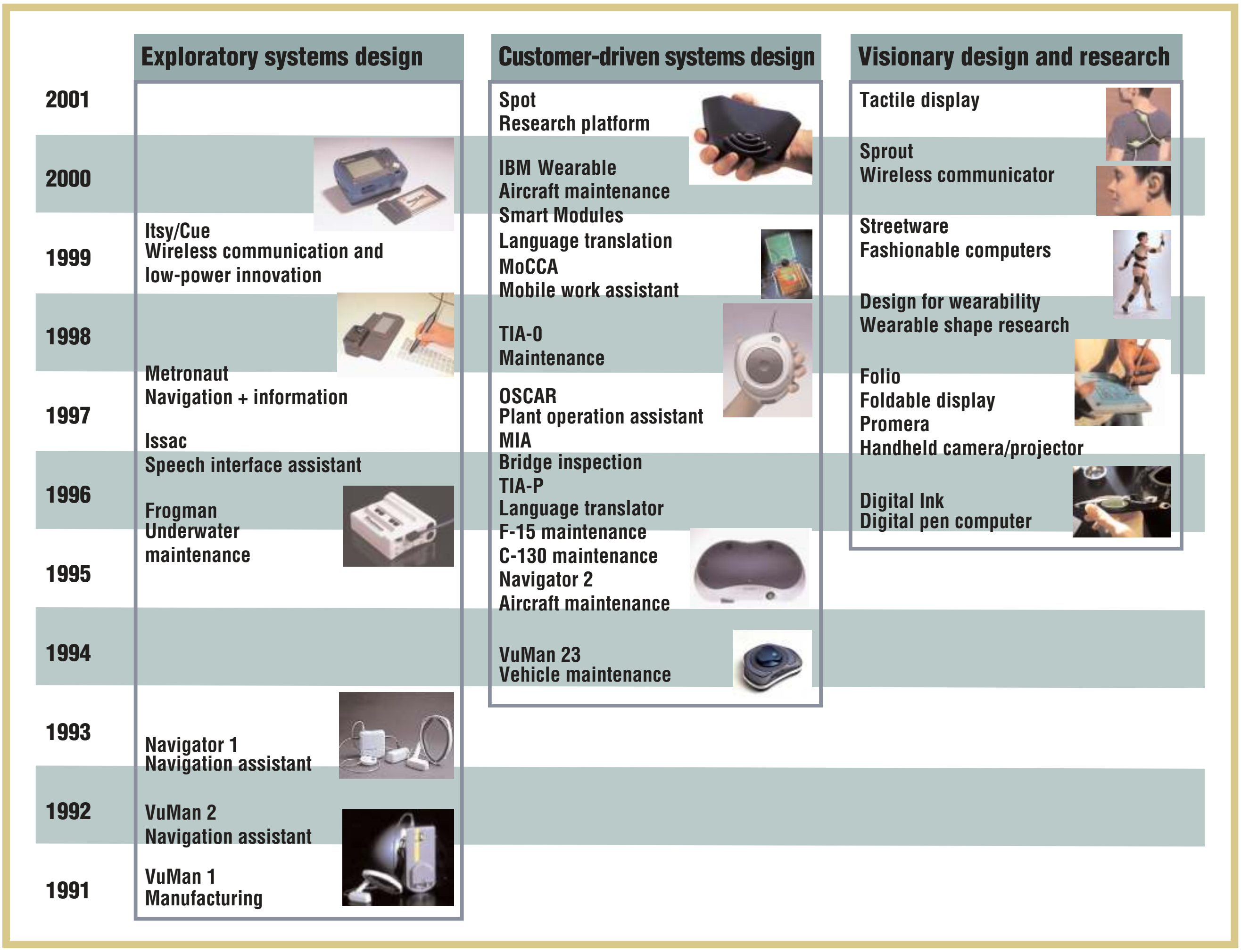

Figure 2. Ten years of wearable computing at Carnegie Mellon (1991–2001).

TABLE 1
User interface evolution.

Input, output, information	Introduction	Mainstream acceptance
Keyboard, alphanumeric display, text	1965	1975
Keyboard and mouse, graphic display, icons	1980	1990
Handwriting, speech recognition, speech synthesis, multimodal	1995	2005
Position sensing, eye tracking, stereo audio and video, 3D virtual reality	2010	2020

ties mimicking the human brain's I/O capabilities haven't yet produced acceptable accuracy and ease of use. Many current modalities require extensive training periods, and their inaccuracies frustrate users. Most of these modalities also require extensive computing resources not available in lightweight, low-power wearable computers. New, easy-to-use input devices such as CMU's dial, developed for list-oriented applications,[4] could prove useful.

Matching capabilities with application requirements. Many mobile systems attempt to pack as much capacity and performance in as small a package as possible. However, users don't always need high-end capabilities to complete a task. Enhancements such as full-color graphics not only require substantial resources but also might compromise ease of use by generating information overload. Interface design and evaluation should focus on the most effective information access means and resist the temptation to provide extra capabilities simply because they're available.

Quick interface evaluation. Currently, human–computer interface evaluation requires elaborate procedures with many subjects. Such evaluations might take months and don't provide an appropriate reference for interface design. New evaluation techniques should especially focus on decreasing human errors and frustration.

Functionalities

Over the past decade, in building wearable computers for diverse application areas, we've observed that several functionalities prove useful across multiple applications. These functionalities form the basis for our four user interface models, each with unique user interfaces, I/O modalities, and capabilities.

Procedures: text and graphics. Maintenance and plant operation applications draw on a large volume of information that changes slowly. For example, even simple aircraft might have more than 100,000 manual pages. Operational changes and upgrades, however, make half of these pages obsolete every six months for even mature aircraft. Rather than distribute CD-ROMs to each maintenance person and risk a procedure being performed on the basis of obsolete information, maintenance facilities usually maintain a centralized database that personnel consult for the relevant manual sections. The centralized information base can change as needed.

Also, as manufacturing becomes more customized, no two aircraft on an assembly line are identical—they might belong to different airlines or be configured for different missions. When manufacturing or maintenance personnel arrive for a day's work, they receive a list of job orders describing the tasks and including documentation such as text and schematic drawings. Because it's centrally maintained, even if this information changes daily or hourly, workers still get accurate information.

Master–apprentice help desk. Sometimes a worker requires assistance from experienced personnel. Historically, apprenticeship programs let novices learn by observing and working with experienced workers. More recently, help desks have evolved to provide audio and visual access to experienced people for help with problem solving.

Team maintenance and collaboration. The help desk can service many field workers simultaneously. Today, downsizing and productivity improvement efforts compel even geographically distributed teams to pool their knowledge to solve immediate problems. In an extension of the help desk idea, a team of field service engineers, police, firefighters, and others trying to resolve an emergency situation must have reliable access to information that will change minute by minute or even second by second.

> Today, downsizing and productivity improvement efforts compel even geographically distributed teams to pool their knowledge to solve immediate problems.

Context-aware collaboration with a proactive assistant. Distractions pose even more of a problem in mobile environments than in desktop environments because mobile users often must continue walking, driving, or taking part in other real-world interactions. A ubiquitous computing environment that minimizes distraction should therefore include a context-aware system able to "read" its user's state and surroundings and modify its behavior on the basis of this information. The system can also act as a proactive assistant by linking information such as location and schedule derived from many contexts, making decisions, and anticipating user needs. Mobile computers that can exploit contextual information will significantly reduce demands on human attention.

Example systems

How do these four application paradigms address the design principles of user interface, I/O modalities, and functional capability requirements? We primarily use CMU wearable computer systems to illustrate the paradigms (see Figure 3), but other organizations' systems also use these models.

For the procedures model, we look at VuMan 3, which provides text-based inspection of heavy military vehicles.[4] Other applicable examples include Navigator 2, which assists graphical-based inspection of Boeing aircraft,[1] and Georgia Tech's wearable computer for quality assurance inspection in food-processing plants.[5]

We illustrate the master–apprentice help desk paradigm using TIA-P (Tactical information Assistant Prototype), used for CMU's C-130 help desk.[6] Netman also lets field technicians and office-based experts collaborate in real time using audio and video.[7]

For team collaboration, we look at MoCCA (Mobile Communication and Computing Architecture), which supports collaboration of geographically distributed field engineers.[8] Another system in this model, Land Warrior (www.fas.org/man/dod-101/sys/land/land-warrior.htm), is an integrated infantry soldier system for close combat designed to avoid information overload. (See "The Evolution of Army Wearable Computers" article in this issue).

To illustrate context-aware collaboration we discuss a context-aware cell phone, a part of the contextual car and driver interface that proactively helps drivers manage information and communication. Another example, Touring Machine, combines 3D augmented-reality graphics with mobile computing to help users navigate while traveling.[9] Synthetic-assistant technology developed at CMU[6,10] lets a computer-modeled expert interact conversationally, provide advice, read procedures, and answer questions.

Table 2 summarizes the four user interface paradigms with respect to the first design principle and I/O modalities, and presents each model's knowledge sources.

Table 3 evaluates how the user interface

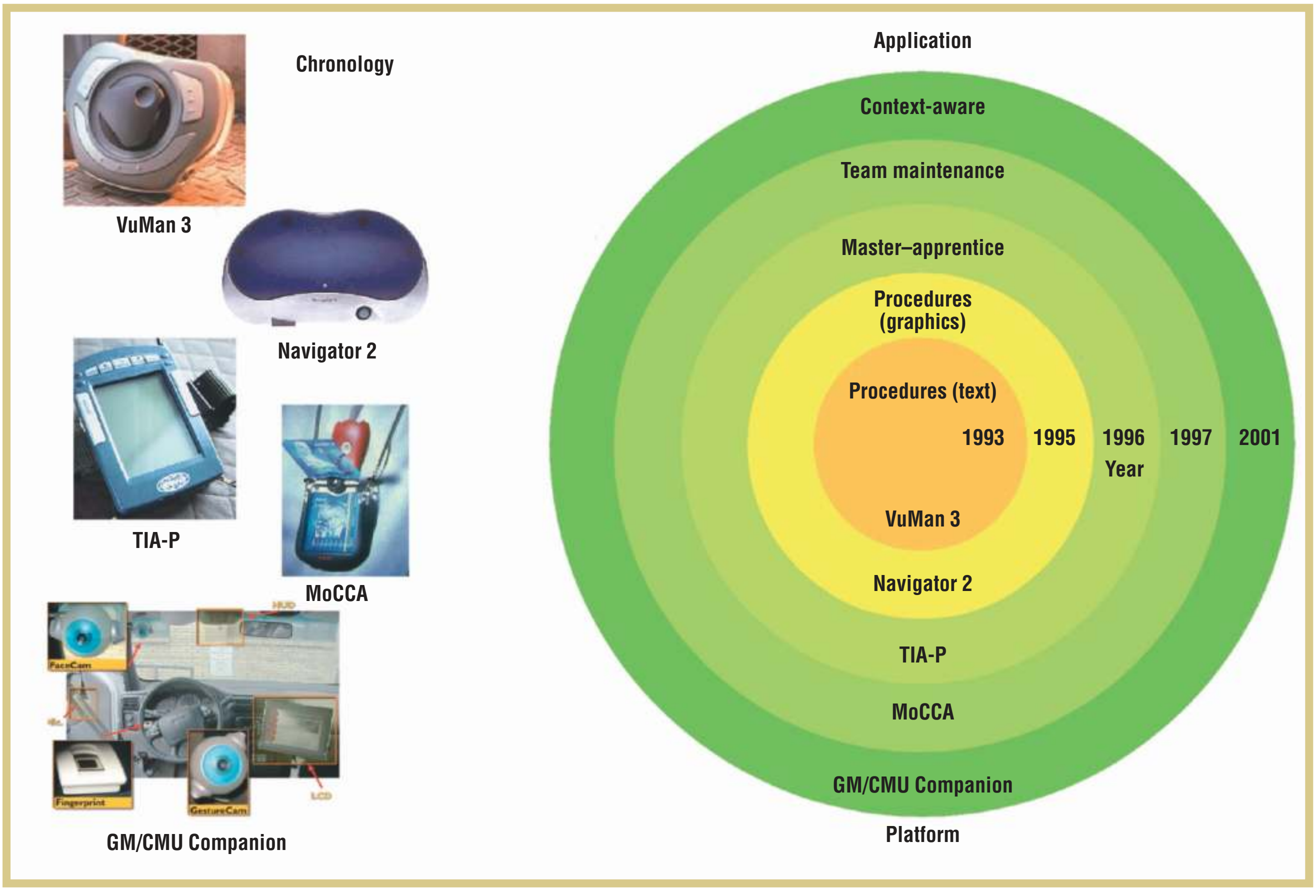

Figure 3. Wearable computer platform examples.

TABLE 2
Input/output modalities and information sources for interface models.

User interface model	Data representation	Knowledge source	Input	Output
Prestored procedures: text	Text	Task-specific, prestored procedures, menu selection, input	Buttons	Alphanumeric
Prestored procedures: graphics	Bitmap	Task-specific, prestored procedures, menu selection, input	Mouse	Graphical user interface
Master–apprentice help desk	Speech synthesis	Archival	Speech	Multimedia
Team collaboration	Pictures	Team “corporate” memory	Archival data	Group collaboration
Synthetic collaboration	3D animation	Real-time physical and social context	Contextual information	Proactive, context-appropriate

models deploy the third design principle, ability to fulfill requirements. For example, the master–apprentice model employs static and synchronous expert functionality. Figure 4 depicts the four problem-solving capabilities in a state diagram.

Procedures: Prestored text and graphics

Navigator 2 assists Boeing inspectors with the sheet metal inspection of a military aircraft. An average 36-hour inspection identifies about 100 defects. The user begins by selecting an aircraft body region and proceeds with object inspection, manual information referencing, archival observation storage, and status recording. During inspection, the field of interest narrows from major features such as the plane’s left

TABLE 3
User interface models and problem-solving capabilities.

User interface model	Problem-solving capability			
	Static	Synchronous expert	Asynchronous expert	Proactive assistant
Prestored procedures	Yes	No	No	No
Master–apprentice help desk	Yes	Yes	No	No
Team collaboration	Yes	Yes	Yes	No
Synthetic collaboration	Yes	Yes	Yes	Yes

wing or right tail to specific details such as individual cockpit windowpanes or aircraft body polygons (see Figure 5). The user records the area each defect covers and its type using a "how malfunctioned" code such as *corroded*, *cracked*, or *missing*. To maximize usability, the system lets the user select each item or control simply by speaking its name or, for more complicated phonemes, a designated numeral. Boeing aircraft inspectors at McClellan Air Force Base in California have praised this 2D selection method, which specifies defect locations on a planar region, and the overall user interface design.

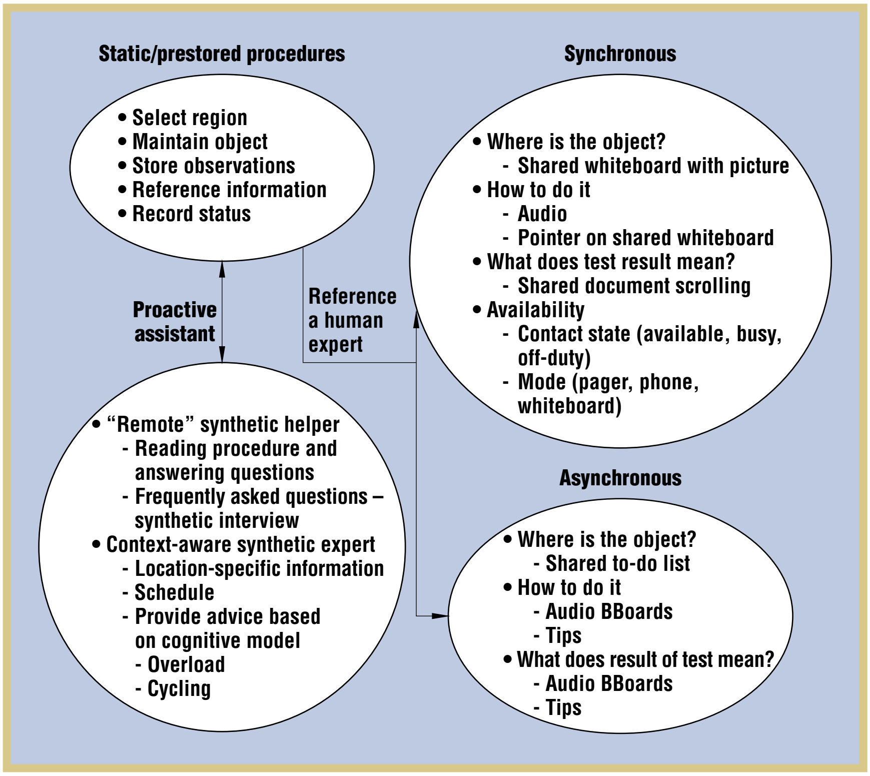

Figure 4. State diagram of problem-solving capabilities.

Master–apprentice (live expert) help desk

The C-130 project uses collaboration to facilitate training and increase the number of trainees per trainer. Remotely located trainers teach inexperienced users to perform a cockpit inspection. The trainee loads the inspection procedures and performs the inspection. A desktop system manages the normal job order process and lets instructors observe trainee behavior. In collaboration, the instructor looks over the trainee's shoulder (through a small video camera attached to the top of the trainee's head mounted display) and offers advice. In addition to a two-way audio channel, the instructor can provide advice using a cursor to indicate areas on a captured video image shared through a whiteboard. The instructor manages the sharing session and whiteboard; the trainee can only observe the whiteboard.

The synchronous communication bubble in Figure 4 shows the master–apprentice paradigm's capability. Synchronous communication facilitates answering questions such as, "Where is the object?" (annotating captured image), "How do I do this?" (audio guidance through prestored material), and "What does the test result mean?" (audio discussion). This model also uses the static and prestored capability.

Team collaboration

MoCCA supports a group of geographically distributed *field service engineers*.[8] The FSEs spend 30 to 40 percent of their time driving to customer sites, and half of what they service is third-party equipment for which they might not have written documentation. MoCCA developers therefore sought to provide a system that let FSEs access information and get advice from other FSEs while commuting or at customer sites. The system supports synchronous and asynchronous collaboration (see Figure 4) for both voice and digitized information.

User interviews yielded another challenge: FSEs wanted laptop computer functionality, including a larger color display, with an operational cycle of at least eight hours. The system had to be very light, preferably less than one pound, and able to access several legacy databases. Further discussions with the FSEs indicated that they most often used text-oriented databases and only rarely accessed graphical

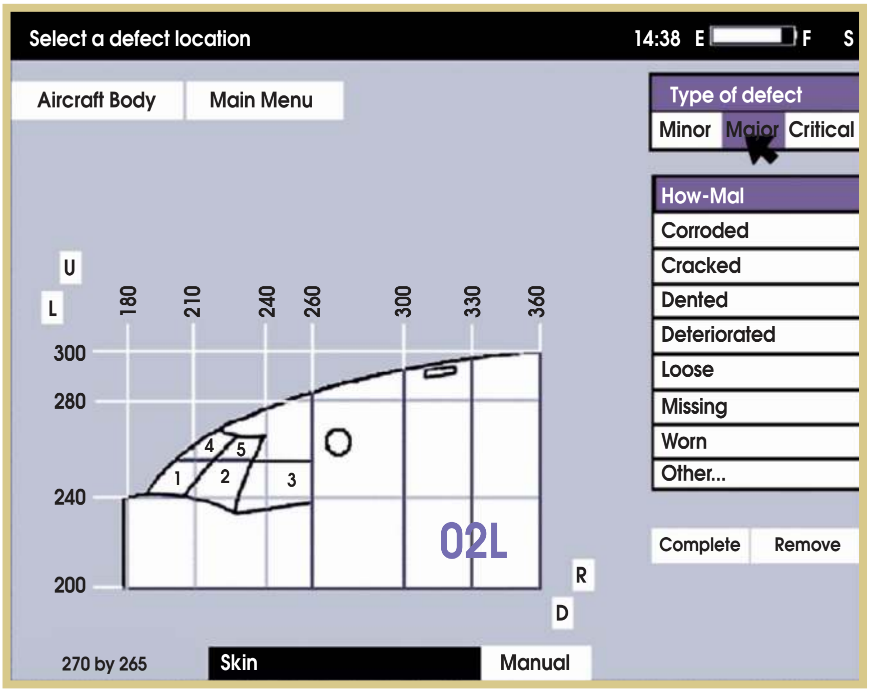

Figure 5. Sample user interface screen for static and prestored information.

databases. The system's architecture combined a lightweight alphanumeric satellite computer with a base unit that FSEs could carry into any customer site and gain instant access to the global infrastructure.

MoCCA's asynchronous capabilities for team problem-solving (see Figure 4) include audio bulletin boards and tips for shared collaboration space between remote FSEs and their colleagues. The audio bulletin board compares to a storehouse of audio clips describing problems that FSEs encounter on the job. Each "trouble" topic contains a list of audio responses from other FSEs with possible solutions. Figure 6 shows the integrated user interface that starts with the call list, list of available FSEs, and information about the incoming service request.

Context-aware collaboration

We designed a context-aware cell phone application to give the remote caller feedback on the current context of the person being called. It uses time (via a calendar), location, and audio environment sensing and interpretation to derive the callee's context. It derives location from a GPS unit on the car (some newer Ericsson and Motorola cell phones also have GPS capability).

If it determines that the callee is driving, the system must let the caller interact with a driver as if he or she were a passenger in the car. For example, in a particularly difficult driving situation with a high cognitive load (such as passing a truck on a downhill curve at night in the rain), a passenger would be sensitive to the situation and suspend conversation until the driving situation has passed. With contemporary cell phones, however, the caller is unaware of the driver's context and will continue talking, perhaps causing the driver to enter a state of cognitive overload.

Evaluation

We used laboratory prototypes to evaluate four applications' performance for each user interface model. Metrics included time on task and time required to achieve high accuracy.

Prestored procedures

We field-tested MoCCA at the Digital Equipment Corporation facility in Forest Hills, Pennsylvania. Five FSEs performed a set of typical troubleshooting and repair operations on computing equipment including printers, motherboards, and networks. Our prototype saved FSEs considerable time—35 to 40 percent over the system they normally use (see Figure 7). The FSEs used our system for the first time during these tests, and we would expect greater efficiency with continued use. MoCCA also let FSEs immediately fix some problems that otherwise would have required return trips to find and bring back manuals.

Master–apprentice help desk

For this paradigm, we measured performance on a bicycle repair task when working alone compared to working with expert guidance. Workers and experts communicated through a video link and an audio connection. Study participants consisted of 60 CMU students (69 percent male) and two bicycle repair experts. Study participants used a help desk collaborative system with head-mounted display and a small CCD camera mounted on the display.

We found that workers performed substantially better with collaborative help, and we used a repeated-measures analysis of variance test[11] to examine the results' statistical significance. Workers with remote expert guidance took, on average, half the time to complete the repair tasks as those working solo (7.5 versus 16.5 minutes, respectively; $p < .001$). They also performed higher-quality repairs (79 percent of quality points for the collaborative condition versus 51 percent for the solo condition; $p < .001$). While access to an expert dramatically improved performance, having better communication tools did not improve the number of tasks completed, the average time per completed task, or performance quality. In particular, neither video (comparison of full-duplex audio/video with full-duplex audio/no video) nor full-duplex audio (comparison of full-duplex audio/video condition with half-duplex audio/video) helped workers perform more tasks, perform tasks more quickly, or perform them better.[12]

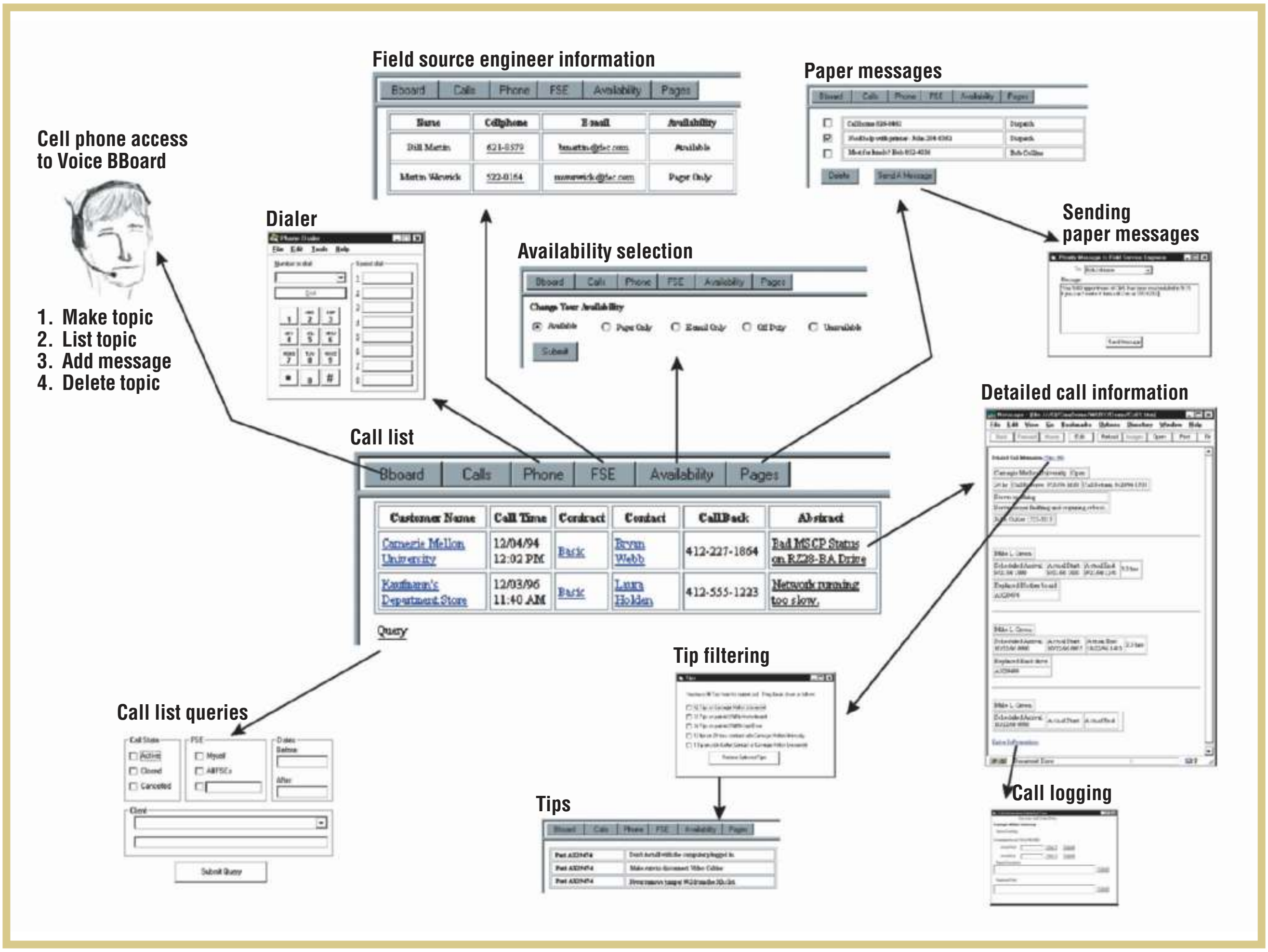

Figure 6. The Mobile Communication and Computing Architecture integrated interface.

Team collaboration

Idealink provides a virtual space for groups to manipulate and share observations about graphical objects related to their work task.[13] Asynchronous audio tags let users record an audio explanation or annotation of a particular object or procedure. The system records and archives each session, making the knowledge contained within them available for later reference.

We designed an experiment to compare how effectively users communicated concepts with Idealink versus a traditional whiteboard. Eight groups of four CMU

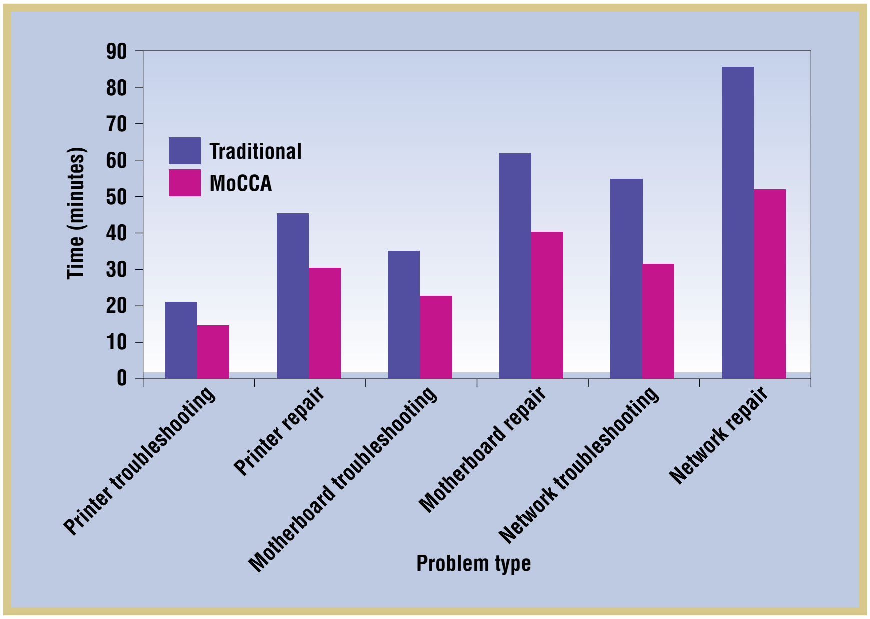

Figure 7. Improvement in problem solving with prestored knowledge versus traditional approach.

TABLE 4
Idealink user evaluation.

Collaboration event	Whiteboard	Idealink
Explicit communication events	54	33
Errors and difficulties	3	0
Implicit communication events	40	7
Errors and difficulties	2	0

students from different majors performed a group problem-solving task to design a remote control for a stereo system; four groups used Idealink, and the other four used a standard whiteboard. We videotaped all sessions and examined the recordings to determine how frequently users from each group requested clarification (see Table 4). We observed two event types. *Explicit communications* offered explicit verbal references to a particular object or region of the drawing area ("Look at the box on the left"). *Implicit communication* indirectly referred to an object or drawing area region ("Look at that one"). Errors occurred when a team member misunderstood another team member's reference. Idealink not only reduced the number of communications but also reduced the number of communication errors by providing smaller regions in which collaborators could more easily focus their attention.

Context-aware collaboration

We designed two experiments to test the hypothesis that a context-aware cell phone could change caller and driver behaviors. Experiment 1 tested whether remote cell-phone callers would slow or stop their conversation with a driver when signaled. Experiment 2 tested whether a driver's performance while speaking on a cell phone would be improved by slowing or stopping the remote callers' conversation.

We asked 24 participants to role-play a person seeking to rent an apartment. Each participant made successive cell phone calls while driving to three "landlords," played by the experimenter. We gave participants a list of questions to ask the landlord about each apartment (for example, how many bedrooms the apartment had). At a prespecified point in each call, the landlord would unexpectedly pause for 10 seconds. Results showed that the callers spoke fewer than half the number of sentences during the pause when they were sent a signal compared to when the driver remained silent. The spoken message, "The person you have called is busy; please hold," was the most effective signal.

The second experiment used a driving simulator composed from a virtual-reality authoring environment that let users navigate a vehicle through a test track. Before beginning the experiment, the 20 participants practiced using the driving simulator until they said they felt comfortable. Participants then completed one circuit on the driving simulator under each of three conditions: control (no phone call), call without pause, and call with pause. Results showed that talking on the cell phone caused people to crash more (6.8 crashes) compared to driving without a call (3.55 crashes). Inducing pauses during the call caused the driver to crash less (3.65 crashes) when using the cell phone.

Our results show that a driver using a context-aware cell phone that interrupts the caller during dangerous driving conditions could make driving while talking on the cell phone safer. We are building a context-aware cell phone that will recognize these conditions and induce such a pause to effectively direct the driver's attention to the driving task.

To effectively integrate wearable computers into ubiquitous computing environments, we must address several important challenges. How do we develop social and cognitive application models? How do we integrate input from multiple sensors and map them to users' social and cognitive states? How do we anticipate user needs? How do we interact with users? Our four paradigms break these challenges into manageable design and evaluation tasks to ensure that applications developed for specific domains best meet users' needs.

Our future work will focus on developing a *virtual coach* that will deploy a wearable augmented-cognition platform and software application. This system will monitor users' cognitive load, assess cognitive performance online, and route tasks to underloaded users. Providing immediate suggestions to users for cognitive augmentation and arbitration of resource redeployment will further enhance performance.

ACKNOWLEDGMENTS

We thank the US National Science Foundation, DARPA, and the Pennsylvania Infrastructure Technology Alliance (PITA) for funding support.

REFERENCES

1. A. Smailagic and D.P. Siewiorek, "The CMU Mobile Computers: A New Generation of Computer Systems," *Proc. IEEE COMPCON 94*, IEEE Press, Piscataway, N.J., 1994, pp. 467–473.

2. A. Smailagic et al., "Benchmarking an Interdisciplinary Concurrent Design Methodology for Electronic/Mechanical Design," *Proc. 32nd ACM/IEEE Design Automation Conf.* (DAC 95), ACM Press, New York, 1995, pp. 514–519.

3. D.P. Siewiorek, A. Smailagic, and J.C. Lee, "An Interdisciplinary Concurrent Design Methodology as Applied to the Navigator Wearable Computer System," *J. Computer and Software Eng.*, vol. 2, no. 3, Oct. 1994, pp. 259–292.

4. A. Smailagic et al., "Very Rapid Prototyping of Wearable Computers: A Case Study of Custom versus Off-the-Shelf Design Methodologies," *J. Design Automation for Embedded Systems*, vol. 3, nos. 2–3, Mar. 1998, pp. 217–230.

5. L. Najjar, J.C. Thompson, and J.J. Ockerman, "A Wearable Computer for Quality Assurance in a Food-Processing Plant," *Proc. 1st Int'l Symp. Wearable Computers*, IEEE CS Press, Los Alamitos, Calif., 1997, pp. 163–164.

6. A. Smailagic, "An Evaluation of Audio-centric CMU Wearable Computers," *ACM J. Special Topics in Mobile Networking*, vol. 6, no. 2, 1998, pp. 59–68.

7. M. Bauer et al., "A Collaborative Wearable System with Remote Sensing," *Proc. 2nd Int'l Symp. Wearable Computers*, IEEE CS Press, Los Alamitos, Calif., 1998, pp. 10–17.

8. A. Smailagic et al., "MoCCA: A Mobile Communication and Computing Architecture," *Proc. 3rd Int'l Symp. Wearable Computers*, IEEE CS Press, Los Alamitos, Calif., 1999, pp. 64–71.

9. S. Feiner, B. MacIntyre, and T. Höllerer, "A Touring Machine: Prototyping 3D Mobile Augmented Reality Systems for Exploring the Urban Environment," *Proc. 1st Int'l Symp. Wearable Computers*, IEEE CS Press, Los Alamitos, Calif., 1997, pp. 74–81.

10. D. Marinelli and S.M. Stevens, "Synthetic Interviews: The Art of Creating a 'Dyad' between Humans and Machine-Based Characters," *Proc. 4th IEEE Workshop Interactive Voice Technology for Telecommunications Applications*, IEEE Press, Piscataway, N.J., 1998.

11. E.R. Girden, *ANOVA Repeated Measures*, Sage Publications, Thousand Oaks, Calif., 1992.

12. R.E. Kraut, S.R. Fussell, and J. Siegel, "Situational Awareness and Conversational Grounding in Collaborative Bicycle Repair," submitted for publication, 2001; www.andrew.cmu.edu/user/sfussell/Manuscripts_Folder/Bicycle_Repair.pdf.

13. D. Garlan et al., "Project Aura: Toward Distraction-Free Pervasive Computing," *IEEE Pervasive Computing*, vol. 1, no. 2, Apr.–Jun. 2002, pp. 22–32.

For more information on this or any other computing topic, please visit our Digital Library at http://computer.org/publications/dlib.

the AUTHORS

Asim Smailagic is the director of the Laboratory for Interactive Computer Systems and Wearable Computers and a research professor at the Institute for Complex Engineered Systems at Carnegie Mellon University's College of Engineering. His research interests include wearable computers, mobile and pervasive computing, reliability, and audio and visual interfaces. He served as program committee chairman for the 2002 IEEE Computer Society Symposium on VLSI 2002. Contact him at Inst. for Complex Engineered Systems, Carnegie Mellon Univ., 5000 Forbes Ave., Pittsburgh, PA 15213; asim@cs.cmu.edu; www-2.cs.cmu.edu/~asim.

Daniel Siewiorek is the director of the Human–Computer Interaction Institute and professor of electrical and computer engineering at Carnegie Mellon University. His research interests include wearable computing, fault-tolerant computing, and reliability. Contact him at the Human–Computer Interaction Inst., Carnegie Mellon Univ., 5000 Forbes Ave., Pittsburgh, PA 15213-3891; dps@cs.cmu.edu; www-2.cs.cmu.edu/~dps.