HUMAN COMPUTER INTERACTION

A Brief Survey on Human Computer Interaction Technology

Dr. Debasis Samanta

INDIAN INSTITUTE OF TECHNOLOGY, KHARAGPUR

2.0 Introduction

Over the last three decades, computers have become an integral part of many people's jobs and even in their day-to-day lives. Most organizations have introduced computerized systems, expecting to improve productivity by speeding up work cycles, reducing the number of repetitive tasks, and increasing employee flexibility. More often than not, however, the introduction of computer systems did not meet these expectations. Even in organizations which planned the introduction of their systems carefully, many users of the new technology encountered problems. Whilst technology suppliers initially attributed many of these problems to *technophobia* or resistance to change, user organizations often discovered that employees had been left to struggle with systems which were difficult to learn, difficult to use, and did not provide appropriate support for their tasks.

Before the start of the microcomputer revolution of the seventies, most computer systems were designed by computer professionals and used by computer professionals. With the spread of personal computers (PCs), the user population expanded to include people who were experts at their own jobs and not computer professionals. This new generation of users had neither the time nor the motivation to acquire extensive background knowledge in electronics and programming: they wanted to use the computer as a tool to help them do their work more effectively and efficiently. The discrepancy between users' expectations and goals, and the usually frustrating experiences resulting from attempts to use computer systems, set the background for the emergence of a new research discipline: human-computer interaction (HCI).

The speed of hardware development and increasing competition amongst suppliers meant that the cost of hardware decreased in real terms. Perceived user requirements began to replace hardware constraints as the dominant influence on the software development process. Users' buying decisions shifted from being hardware-led (select hardware, then find or commission suitable software) to being software-led (select a suitable software package, then buys the hardware needed to run it). Terms such as *user friendly, easy to learn* and *easy to use* are important marketing tools today, and software companies attach much importance to the *look and feel* of the user interfaces of their products. The user interface has become a major determinant of the success or failure of computer systems. With both industry and users looking to HCI researchers and practitioners for guidance, HCI is very much an applied discipline, and its *purpose* is to provide knowledge and tools for the design of usable computer systems.

HCI in general and the design of "good" user interfaces in particular, has been an extraordinarily popular area of research over the last 20 years. The subject has attracted researchers and practitioners from mainly three established disciplines: psychology, computer science, and human factors or ergonomics. These disciplines have contributed to HCI from its early days. More recently, there has been research input from the fields of sociology, anthropology, linguistics, organizational behaviour, fine arts and various design disciplines. The number of publications in the field has soared, journals and book series specialising in HCI have appeared. The professional bodies of relevant disciplines have established specialist groups. HCI conferences are held regularly at national and international levels. HCI courses and options are

taught at higher education establishments all over the world. Whilst there can be little doubt that HCI has established itself as a discipline, there has been some argument over the nature of the discipline, and how its research should be conducted and communicated.

2.1 Research fields within HCI

The common definition of HCI is “Human-computer interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them. The main objective of any research in the field of computer science is to make the use of computer or computer like devices more efficient and effective. HCI views this objective from user’s point. The user sees a system as the presentation given by its interface. Designing a good interface is not an easy job. A lot of factors starting from computational efficiency to human psychology has to be considered for it. So the researches on this field are also very much diversified. The following list gives a brief overview of various research fields within HCI

- **Ubiquitous communication**

Computers will communicate through high speed local networks, nationally over wide-area networks, and portably via infrared, ultrasonic, cellular, and other technologies. Data and computational services will be portably accessible from many if not most locations to which a user travels.

- **High functionality systems**

Systems will have large numbers of functions associated with them. There will be so many systems that most users, technical or non-technical, will not have time to learn them in the traditional way (e.g., through thick manuals).

- **Mass availability of computer graphics**

Computer graphics capabilities such as image processing, graphics transformations, rendering, and interactive animation will become widespread as inexpensive chips become available for inclusion in general workstations.

- **Mixed media**

Systems will handle images, voice, sounds, video, text and formatted data. These will be exchangeable over communication links among users. The separate worlds of consumer electronics (e.g., stereo sets, VCRs, televisions) and computers will partially merge. Computer and print worlds will continue to cross assimilate each other.

- **High-bandwidth interaction**

The rate at which humans and machines interact will increase substantially due to the changes in speed, computer graphics, new media, and new input/output devices. This will lead to some qualitatively different interfaces, such as virtual reality or computational video.

- **Large and thin displays**

New display technologies will finally mature enabling very large displays and also displays that are thin, light weight, and have low power consumption. This will have large effects on portability and will enable the development of paper-like, pen-based computer interaction systems very different in feel from desktop workstations of the present.

- **Embedded computation**

Computation will pass beyond desktop computers into every object for which uses can be found. The environment will be alive with little computations from computerized cooking appliances to lighting and plumbing fixtures to window blinds to automobile braking systems to greeting cards. To some extent, this development is already taking place. The difference in the future is the addition of networked communications that will allow many of these embedded computations to coordinate with each other and with the user. Human interfaces to these embedded devices will in many cases be very different from those appropriate to workstations.

- **Group interfaces**

Interfaces to allow groups of people to coordinate will be common (e.g., for meetings, for engineering projects, for authoring joint documents). These will have major impacts on the nature of organizations and on the division of labor. Models of the group design process will be embedded in systems and will cause increased rationalization of design.

- **User Tailorability**

Ordinary users will routinely tailor applications to their own use and will use this power to invent new applications based on their understanding of their own domains. Users, with their deeper knowledge of their own knowledge domains, will increasingly be important sources of new applications at the expense of generic systems programmers (with systems expertise but low domain expertise).

- **Information Utilities**

Public information utilities (such as Compuserve, Prodigy, home banking and shopping, etc.) and specialized industry services (e.g., weather for pilots) will continue to proliferate. The rate of proliferation will accelerate with the introduction of high-bandwidth interaction and the improvement in quality of interfaces.

2.2 History of HCI

Human-computer interaction arose as a field from intertwined roots in computer graphics, operating systems, human factors, ergonomics, industrial engineering, cognitive psychology, and the systems part of computer science. Computer graphics was born from the use of CRT and pen devices very early in the history of computers. This led to the development of several human-computer interaction techniques. Many techniques date from Sutherland's Sketchpad Ph.D. thesis (1963) that essentially marked the beginning of computer graphics as a discipline. Work in computer graphics has continued to develop algorithms and hardware that allow the display and manipulation of ever more realistic-looking objects (e.g., CAD/CAM machine parts or medical images of body parts). Computer graphics has a natural interest in HCI as "interactive graphics" (e.g., how to manipulate solid models in a CAD/CAM system).

A related set of developments were attempts to pursue "man-machine symbiosis" (Licklider, 1960), the "augmentation of human intellect" (Engelbart, 1963), and the "Dynabook" (Kay and Goldberg, 1977). Out of this line of development came a number of important building blocks for human-computer interaction. Some of these building blocks include the mouse, bitmapped displays, personal computers, windows, the desktop metaphor, and point-and-click editors Work on operating systems, meanwhile developed techniques for interfacing input/output devices, for tuning system response time to human interaction times, for multiprocessing, and for supporting windowing environments and animation. This strand of development has currently given rise to "user interface management systems" and "user interface toolkits". The following list presents a brief look on the development history of HCI.

- In 1962 at the Stanford Research Lab, Engelbart proposed and later implemented a **word processor** with automatic word wrap, search and replace, user-definable macros, scrolling text, and commands to move, copy, and delete characters, words, or blocks of text.

- The first **graphical video game** was probably SpaceWar by Slug Russel of MIT in 1962 for the PDP-1 including the first computer joysticks.

- The first **direct manipulation interface**, where visible objects on the screen are directly manipulated with a pointing device, was first demonstrated by Ivan Sutherland in Sketchpad, which was his 1963 MIT PhD thesis.

- The first **CAD/CAM system** in industry was probably General Motor's DAC-1 (about 1963).

- Stanford's TVEdit (1965) was one of the first **CRT-based display editors** that was widely used.

- The **mouse** was developed at Stanford Research Laboratory (now SRI) in 1965 as part of the NLS project (funding from ARPA, NASA, and Rome ADC) to be a cheap replacement for light-pens.

- **Multiple tiled windows** were first demonstrated in Engelbart's NLS in 1968

- Engelbart's NLS system at the Stanford Research Laboratories in 1965 made extensive use of **linking documents** (funding from ARPA, NASA, and Rome ADC).

- The initial **spreadsheet** was VisiCalc which was developed by Frankston and Bricklin (1977-8) for the Apple II while they were students at MIT and the Harvard Business School. The solver was based on a dependency-directed backtracking algorithm by Sussman and Stallman at the MIT AI Lab.

- The **X Window System**, a current international standard, was developed at MIT in 1984

2.3 Factors in HCI

There are a large number of factors which should be considered in the analysis and design of a system using HCI principles. Many of these factors interact with each other, making the analysis even more complex. The main factors are listed in the table below:

<table>
<tr><td colspan="2">Organisation Factors
Training, job design, politics, roles, work organisation</td><td>Environmental Factors
Noise, heating, lighting, ventilation</td></tr>
<tr><td>Health and Safety Factors</td><td>The User
Cognitive processes and capabilities
Motivation, enjoyment, satisfaction, personality, experience</td><td>Comfort Factors
Seating, equipment, layout.</td></tr>
<tr><td colspan="3">User Interface
Input devices, output devices, dialogue structures, use of colour, icons, commands, navigation, graphics, natural language, user support, multimedia,</td></tr>
<tr><td colspan="3">Task Factors
Easy, complex, novel, task allocation, monitoring, skills</td></tr>
<tr><td colspan="3">Constraints
Cost, timescales, budgets, staff, equipment, buildings</td></tr>
<tr><td colspan="3">System Functionality
Hardware, software, application</td></tr>
<tr><td colspan="3">Productivity Factors
Increase output, increase quality, decrease costs, decrease errors, increase innovation</td></tr>
</table>

Fig. 2.1 Different factors in HCI design

2.4 Evolution of HCI Technology from Computer Side

2.4.1 Charles Babbage's computer

Babbage designed two types of computers, the Difference Engine and the Analytical Engine, making constant variations on both of them. Mechanical calculators had already been devised by the time Babbage had begun work on his difference engine, but none worked as independently or as accurately. Slide rules had also been invented by then, but they were accurate only up to a point. The intent of the Difference Engine was to find the common difference in a sequence of mathematical terms. For example, in the sequence "2, 4, 6, 8, 10…", the common difference is 2. By finding the common difference, the Engine could create a mathematical table. Astronomical tables for star positions at different times could then be printed out, as could tables in other disciplines. In designing the Analytical Engine, Babbage introduced various improvements and advances. Notably, he designed separate sections for doing computations, called the Mill, and a section where numbers were stored. Appropriately enough, this was called the Store, and is analogous to what we now call memory in a computer. The Analytical Engine was actually a machine that could be programmed using punch cards. This was not unlike the use of punch cards or computer tape used in 20th century computers. Babbage freely acknowledged that the idea of using punch cards came from their use in industrial looms, in which they were used to program patterns in making woven material. In addition to his work on computers, he published work on biology, geology, life insurance, astronomy, magnetism, religion, and economics. In 1847, he devised and built an opthalmoscope for examining the interior of the eye. His work in mathematics included probability, geometry, numbers theory, calculus, mathematical notation, and code breaking. Babbage discusses needle making, the regulation of power in a steam engine, determining the direction of a shock from an earthquake, the use of diamonds for cutting glass, and a variety of printing processes. The work, which is largely an outgrowth of his need for understanding precise manufacturing processes for the Difference Engine, also considers the issues of unionizing workers, factory size, and even the very modern-day issues of technology being exported to other countries, replacing coal as an energy source, and using ocean tides as a source of power. Babbage's work on computers also helped promote thought about how a physical mechanism could be related to thought and the mind.

2.4.2 Punch card

The **punch card** (or "**Hollerith**" card) is a recoding medium for holding information for use by automated data processing machines. Made of thin cardboard, the punch card represents information by the presence or absence of holes in predefined positions on the card. In the first generation of computing, from the 1920 into the 1950s, punch cards were the primary medium for data storage and processing. Eventually, during the late 1970s to early 1980s, the punch card was phased out as a medium for storage of computer data and replaced by huge floppy disks. Today, punch cards are long obsolete outside of a few legacy systems and specialized

applications. The punched card predates computers considerably. As early as 1725 Basile Bouchon used perforated paper loop in a loom to establish the pattern to be reproduced on cloth, and in 1726 his co-worker Jean-Baptiste Falcon improved on his design by using perforated paper cards attached to one another, which made it easier to change the program quickly. The Bouchon-Falcon loom was semi-automatic and required manual feed of the program. Joseph Jacquard used punched cards in 1801 as a control device for the more automatic Jacquard looms, which met with great success.

2.4.3 Command line

A *command line interface* is a method of interacting with a computer by giving it lines of textual commands (that is, a sequence of characters) either from keyboard input or from a script. It is occasionally also referred to as a *CLUE*, for Command Line User Environment. In its simplest form, the User types a Command after the computer displays a prompt. The computer then carries out the Command given. Programs that implement these interfaces are often called command line interpreters. Examples of such Programs include the various different Unix shells, VMS *DCL* (Digital Command Language), and related designs like CP/M and DOSs command.com, both based heavily on DEC's RSX and RSTS operating system interfaces (which were also Command line interfaces). Microsoft claims their next major operating system, code-named Longhorn, will include an enhanced Command line interface named MSH (Microsoft Shell, codename Monad), which combines the features of traditional Unix shells with the object-oriented framework.

Some applications provide Command lines as well. The CAD program AutoCAD is a prominent example. In Some computing environments like the Oberon or Smalltalk User interface, most of the text which appears on the screen may be used for giving commands.

2.4 Evolution of HCI Technology from Human Side

2.4.1 Human Factors

The field of Human Factor is the scientific discipline that attempts to find the best ways to design products, equipment, and systems so that people are maximally productive, satisfied, and safe. Historically, the term human factors have been used in the United States, and the term ergonomics has been used in Europe. Other terms used to describe the field are engineering psychology and applied experimental psychology. Whatever the name, Human Factor is the science that brings together psychology and engineering design.

The field is multidisciplinary and benefits from the input of experts from domains such as psychology, engineering, computer science, biomechanics, medicine, and others. Frequently, the Human Factor professional plays the role of mediator between divergent interests advocating for the human point of view in the design of products, equipment, and systems by championing

designs that make maximal use of the magnificent abilities that people possess and limiting the use of tasks where people could make errors.

Early contributions to the establishment of Human Factors included the analysis of time and motion of people doing work, and determining human capabilities and limitations in relation to job demands. Most people credit the beginning of the field with the military during World War II. Pilots were flying their airplanes into the ground, and eventually psychologists were called in to find out why. We'd call it "human error" today, and part of the reason for the aircraft crashes was the lack of standardization between different aircraft models. The growing complexity of military hardware during this time period was revealing for the first time in history that even highly selected individuals who were given extensive training could not do the tasks that they needed to do. Pilots were not able to control their aircraft under stressful emergencies. The machines outstripped people's capabilities to use them. Investigations revealed that pilots had certain expectations of how things should work (for example, the location of the landing gear control and how to activate it), and these were frequently violated by aircraft designers (who frequently knew very little about people's abilities and limitations). Before World War II, it was assumed that people could eventually learn whatever they were given if they were trained properly. Since World War II, the field has blossomed as is evident from the examples provided in the next section.

It is very telling that the word 'factors' in 'human factors' occasionally is equated with 'limitations'. From the Human Factors literature, one gets the impression that the 'factors' of computer systems were regarded at least fairly predictable, whereas the 'human factors' were not and hence attempted 'ruled out'.

An assumption of the Human Factors community was that insight into e.g. cognitive, sensory-motor, and perceptual aspects of human behavior could instigate the design of an optimum performance of 'man in concert with his machine', viewed as one united functional system/unit. With knowledge of 'generic' human factors or 'basic' human behavior and cognition, it was thought that an optimal coupling between man and machine could be found. However, as the discipline evolved, the hunt for 'human constants' proved problematic as well as it proved highly difficult to establish the design implications of such constants.

The stance of the Human Factors community toward this kind of 'singular ontology', i.e. that truth or meaning is 'out there' as in a natural science, came under heavy criticism by many, e.g. Suchman (1987) and Winograd and Flores (1987). In consequence, the HCI community, which had spawned from the Human Factors community, took a "contextual turn" toward more humanistic and sociological accounts of meaning from the late 80s and onward. Aspects like context were now given higher priority and meaning and truth was seeked in the interaction between man and machine or as a product of the wider context.

Human factors, or limitations, include:

- Impatience
- Limited memory

- Need analogies
- Limited concentration
- Changes in mood
- The need for motivation
- Prejudices
- Fears
- Make errors
- Misjudgement
- Prefer speech
- Process information non-linearly
- Near-sightedness
- Color-blindness
- Distraction
- Can only perform a limited number of concurrent tasks
- Short-term memory works differently than long-term memory
- Users are all different
- Think in terms of ideas composed of words, numbers, multimedia, and intuitions.
- Fatigue
- Must see and hear to understand
- Physical inability
- Need information presented in sets of threes
- Need complex information presented hierarchically
- Confined to one physical location at a time
- Require practice to become good at doing things
- Embarrassment can act as a limitation to accomplishing some tasks
- Tend to do things the easy way
- Resistance to change
- Can be physically harmed by some tasks
- Prefer to learn by doing than by explanation
- Have difficulty converting ideas into modes of communication
- Have difficulty converting modes of communication into ideas
- Act irrationally
- Sometimes affected adversely by stimuli such as colour and patterns
- Become nervous
- Miss details when tasks are memorized and performed cursorily
- Can be affected by socio/political climate.
- Prefer standard ways of doing things
- Constrained by time
- Incentive driven
- Work better in groups than individually

- Require tasks to be modularized in order to work in groups
- Use intuitions to construe information that is sometimes wrong
- Rely on tools to complete tasks (like spell checking) thus causing dependency
- Must delegate responsibility in order to free the mind of complexity
- Become addicted
- Associate unrelated things
- Sometimes do not trust what is not understood
- Death (typically a concern in trains or aero planes)

2.4.2 Ergonomics

Most people, if they even know the term ergonomics, might recognize it as dealing with chairs or possibly automotive displays. While the design of chairs and automobiles is within the purview of Human Factor/Ergonomics, the field is much broader than that. In fact, many Human Factor/Ergonomics professionals believe that nearly all aspects of daily activities are within the domain of Human Factor/Ergonomics. The field deals with the interface between people and things, whether it is a dial on a car dashboard or a control on a stovetop. The fundamental philosophy of Human Factor/Ergonomics is that all products, equipment, and systems are ultimately made for people and should reflect the goals of user satisfaction, safety, and usability.

Two specific examples might serve to illustrate Human Factor/Ergonomics considerations. The first is that as commonplace as automated teller machines (ATM) have become, many older adults do not use them even though they could benefit from their convenience. The goal of an Human Factor/Ergonomics specialist would be to ensure that the design of the machine was easy to use (including the design of the buttons, the wording, the spatial layout and the sequencing of the displays, etc.). Moreover, a Human Factor/Ergonomics person might suggest employing an outreach-training program to assist first-time users. The ultimate Human Factor/Ergonomics solution would be, however, to make the technology so obvious that training is not necessary. Many people can't program a VCR. You might know a statistics program that could be made easier to use and understand. These are the sorts of systems that could benefit from Human Factor/Ergonomics considerations.

The second example concerns pictorial symbols. Increasingly, symbols are being used to convey concepts to people who do not understand the primary language of the locale, and this is becoming increasingly important with people and companies involved with international travel and trade.

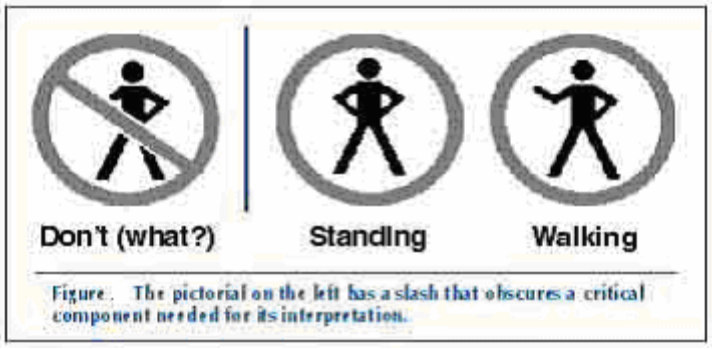

Fig. 2.2 Understanding in HCI

In Fig. 2.2, the pictorial on the left is from an actual sign on automatic doors like you might see at hospitals and airports. What does it mean? The slash obscures a critical feature of the underlying symbol. The pictorial could be interpreted as "Do not stand" or the opposite, "Do not walk." The interpretation the door manufacturer wanted to convey is the first one, because the doors sometimes close unexpectedly. Can you see that the pictorial symbol could be interpreted as the opposite of its intended meaning? The alternative interpretation was apparently missed by the designer. This is called a critical confusion because the meaning can create a hazard. Fortunately, most people probably do not have the chance to misinterpret this symbol. This is because whenever a person walks up to the door, the doors slide to the side, out of the way. The problem is (a) that the sensors sometimes do not pick up people standing at the threshold, and (b) that these people haven't seen the sign. People have been knocked to the ground by automatic doors that have closed unexpectedly, and for some fragile individuals that event has produced injury. An Human Factor/Ergonomics analyst would first want to "design out" the hazard (i.e., so it can't close on anyone) using, for example, better sensors and more reliable and better designed components and systems. If you can't design out the hazard, then at least you ought to guard against the hazard contacting and injuring people. When warnings are used, they ought to be designed so target audiences grasp the intended message quickly and readily with little time and effort.

2.4.3 Cognitive Psychology

Cognitive psychologists who work in the software industry typically find themselves designing and evaluating complex software systems to aid humans in a wide range of problem domains, like word processing, interpersonal communications, information access, finance, remote meeting support, air traffic control, or even gaming situations. In these domains, the technologies and the users' tasks are in a constant state of flux, evolution and co-evolution. Cognitive psychologists working in human-computer interaction design may try to start from first principles developing these systems, but they often encounter novel usage scenarios for which no guidance is available. For this reason, we believe that there is not as much application of theories, models, and specific findings from basic psychological research to user interface (UI)

design as one would hope. However, several analysis techniques and some guidelines generated from the literature are useful. In this paper we outline some efforts in human-computer interaction (HCI) research from our own industrial research experience, demonstrating at what points in the design cycle HCI practitioners typically draw from the cognitive literature. It is our goal to highlight opportunities for the two disciplines to work together to the benefit of both.

2.5 Building Real Bridges The

2.5.1 Millennium Bridge Example

A recent example of bridge building provides a nice example of the complex interactions between design, empirical evaluation, and iterative refinement. The Millennium Bridge was central London's first new bridge in over 100 years. The design was selected from hundreds of entries and built by leading architects and engineering firms. The stainless steel and aluminum pedestrian bridge formed a "blade of light" over the Thames River linking St. Paul's Cathedral to the new Tate Modern Art Gallery. On June 10, 2000 the bridge opened with much fanfare. Only a few hours later, however, the bridge closed briefly because it swayed violently in the wind under the weight of the many pedestrians who crossed it on opening day. Two days later on June 12, 2000 it closed semi-permanently for analysis and repairs. The bridge remains closed today, almost a year later. What happened? All suspension bridges move to some degree, and walking adds additional motion. Normal walking pace is about two steps a second, so a vertical force of 2 hertz is produced. Military commanders have long told their troops to break step when crossing bridges to avoid these vertical movements. Designers know this, standards require accommodating for it, and the Millennium Bridge engineers designed and tested for it. This worked fine. However, there is also a smaller horizontal movement. As we walk, one foot pushes left and the other pushes right, so there is an additional 1 hertz horizontal movement. When large numbers of people walked across the Millennium Bridge, an unintentional synchrony of stepping occurred. As the bridge moved slightly in the horizontal direction, people instinctively adjusted their steps in synchrony with the movement of the bridge. When large numbers of people do this at the same time the movements in the bridge become quite noticeable. The designers (and standards) missed the horizontal synchrony in walking, what they now call lock-in. Despite a good deal of experience building suspension bridges, standards that have been developed and improved over time, and complex mathematical models to predict and test how the bridge would behave, this one didn't quite work in the real world. Why? Actual behavior was outside expectations -- of standards, models and practice. Engineers are now revising the design with new simulation models to examine ways to dampen the lock-in effect. Hopefully the visibility of this failure will also lead to changes in standards and practice as well.

Much the same cycle of initial design from principles and practice, usability testing, careful failure analysis, and iterative design happens all the time in building complex human-computer system. One might even argue that designing HCI applications is much more complex that building bridges because of the many perceptual and cognitive processes involved. Regardless of where one stands on the relative complexity of design in these two arenas, the design cycle is much the same. Contributions from psychology can and often do influence all aspects of the HCI design cycle, some more successfully than others. We can provide examples from information retrieval and notification management. The former deals with how best to organize search results for efficient and accurate analysis, the latter focuses on the design of intelligent notification systems to convey important information so that users can efficiently make use of it.

Just as in bridge building, we cannot currently design complex HCI systems from first principles. Useful principles can be drawn from the sub-domains of sensation, perception, attention, memory, and decision-making to guide us on issues surrounding screen layout, information grouping, menu length, depth and breadth. Guidelines exist for how to use color in user interface design, how to use animation and shading, or even what parameters influence immersion in virtual worlds. Designers also often borrow ideas from best practices, such as very successful products that have been iteratively refined and accepted broadly in the marketplace. However, there will always be technology-usage scenarios for which the basic research simply does not exist to guide us during design. The complexity, learning and interaction effects seen in HCI usage scenarios are daunting. In addition, HCI systems are used by a very wide range of users for a large variety of tasks. Finally, information itself is an extremely complex material to work with, which makes the design task all the more challenging. So while there are a large number of areas within psychology from which HCI design can draw, theory falls far short of covering the entire design life cycle. Examining the areas of psychology have proven most useful to user interface design and describe what seems to work well during the design process and what areas could be improved.

2.5.2 Specific Findings and Guidelines

There are areas in which basic cognitive psychology research has helped the domain of human-computer interaction by providing specific findings and guidelines for design. Some of the best-known and often-cited examples of the applicability of results from basic psychology are the rather low-level perceptual-motor findings that have been used quite effectively in HCI design. For example, Fitt's Law and Hick's Law have been used to design and evaluate input devices for years. The Power Law of Practice, and the known limits on auditory and visual perception have often been leveraged in the design of interactive systems. Many other results are seen in guidelines for good control layout, the use of color and highlighting, depth and breadth tradeoffs in menu design, and many general abstractions have made their way into "libraries" of parameters describing typical response times, for example. Using findings and guidelines like

these allow designers to start with a good initial design, or prevent silly mistakes, but it doesn't guarantee a good system when all the variables are combined together into one design.

Even though specific findings and guidelines have proven useful in some cases, there also exist many problems in their use, which limits their effectiveness. Guidelines are often misused or misinterpreted. For example, using the rule of thumb of having only 7+-2 items in a menu, or applying visual search guidelines but not taking into account the frequency with which items occur or the quality of category cohesiveness or labels. In addition, guidelines are often written at too abstract a level to help with specific designs, or alternatively they are too specific for a given usage context. A designer often finds it very difficult to combine all of the recommendations from specific findings or guidelines without knowledge of the costs, benefits or tradeoffs of doing so. An example in managing interruptions illustrates some of these issues. The problem area of interruption while multitasking on the computer is one where much of what we have learned from basic psychological research has been applicable in system design. For example, 100 years of attention research has taught us about limited cognitive resources, and the costs of task switching and time-sharing even across multiple perceptual channels. But how does one go about designing an effective interface for alerts when the user may benefits from the information? And how does one design an interface that allows the user to easily get back to the primary task after a disruptive notification? In practice, specific findings guide our intuitions that flashing, moving, abrupt onset and loud audio heralds will attract attention. But how much attraction is too much? How does this change over time? Do users habituate to the alerts? How does the relevance of the incoming messages affect task switching and disruption? Some studies in the literature suggested relevance was influential, while others found that surface similarity was more important in terms of task influence. All of these studies used tasks that were not representative of typical computer tasks (i.e., they were too simple, demanded equal distribution of attention, etc

When, as part of the same notification system design, we can attempt to design a "reminder" cue that was visual and could help the user reinstate the original task context after an incoming notification, specific findings were again not useful. Using a "visual marker" as a spatial placeholder to get users back to a point in a primary task was not enough. Instead, users needed a "cognitive marker" that provided more of the contextual, semantic cues related to a primary task. No specific findings existed in the literature for exactly what visual anchors could be used in a display to help users tap into their mental representations of a task, or how many visual retrieval cues would be needed to reinstate a computing task context most efficiently. Again, prototypes needed to be designed and tested. In both the design of notifications, and the corresponding reminder system, paradigms from very basic cognitive research had to be utilized in new lab studies to examine the broader usage context and the specific user interface design chosen for the target task domains.

2.5.3 Analytical Models

Cognitive architectures and analytical models are sometimes used to evaluate and guide designs. GOMS, first proposed by Card, Moran & Newell (1983), is certainly among the most influential models in HCI. GOMS uses a general characterization of basic human information processing, and the GOMS technique (Goals, Operators, Methods and Selection rules) for task analysis. GOMS is most useful in evaluating skilled error-free performance in designs that have already been specified in detail, and can be costly in terms of time or evaluator training. In one of our experiences using GOMS to evaluate two user interfaces for trip planning, the GOMS model was not all that helpful. Using the GOMS analysis the two systems were equally usable and efficient. A subsequent laboratory study, however, showed that one design was more successful overall. Several more general cognitive architectures have been proposed as well, including EPIC, ACT and SOAR. HCI design has been a fertile testing ground for these efforts since complex tasks are a challenge for any such architecture.

2.5.3 Theory

Most psychological theories are descriptive, not prescriptive, and are tested using simple paradigms that may or may not scale up to real world HCI scenarios. Both of these factors limit their applicability. For example, many memory studies examine pairs of words, and attention studies examine the ability to quickly detect the onset of a letter either with or without a predictive cue. The point is not that these paradigms haven't proven to be valuable as basic researchers refine and evolve their theories. Insofar as these basic behaviors emulate components of complex task behaviors, they are extremely useful to practitioners as well as the basic cognitive scientist. The point is more that most psychological theories are developed by examining isolated phenomena in carefully controlled lab settings, with little or no guidance about complex interactions and tradeoffs, which are critical for design.

The details of cognitive theories often don't matter in practice. In system design, does it really matter whether the "bottleneck" in attention is early or late? Not really, designing to support attention, as a critical resource is what matters if the system is to be usable. But, few theories guide designers in how to do this. Wickens' Multiple Resource Theory (Wickens & Carson, 1995) is a good attempt toward this end. Does it matter if visual processing is serial or parallel? Again, not really, as simply knowing that visual scanning slopes are often linear with an increase in set size is what matters for most cases. So, from a practical perspective, much of what is enthusiastically debated in basic cognitive research has little practical implication to HCI designers trying to build useful, efficient interaction systems. Another point to reiterate here is that theories from cognitive psychology are often not predictive enough during the initial stages of design, and not effective for articulating tradeoffs later during design.

Human-computer interaction is complex, dependent on many unpredictable variables, and ever changing during interaction, and this probably is not going to change much over the next decade or so. Humans are complex, the tasks they perform and the information they work with are equally complex, the systems they interact with are varied, and the combined design problem

(which is what HCI is about) is daunting. Cognitive theory and empirical results are currently not up to the challenge.

2.5.4 Experimental Methods

Many of the methods that psychologists use during their academic careers are very useful to the HCI practitioners and researchers. This is often the first way in which a psychologist in industry can add value to a product team. For example, the team may want to design a educational computer application for children, but the programmers, designers and technical writers may not understand how to measure the performance of users of the software (small children), or may not understand the various developmental stages that need to be considered in the user interface design. The psychologist can work with the team to develop metrics of ease of use and satisfaction, including learning, efficiency and engagement. The psychologist will not only know the proper methods to use in the study of children's interaction with the early prototype of the system, but also which statistics to use in the analysis and how to interpret and communicate the findings from the study effectively. These are all skills that the programmers and other team members do not typically have, and they are therefore considered valuable by the team. These same skills are useful in an industrial research environment as well, where new technologies are developed. The inventors have ideas about the benefits of a particular idea, but they typically have little experience in designing the right experiments and tasks to study the questions of interest. Here we have used visual search tasks, dual tasks, reaction time and accuracy studies, deadline procedures, memory methods like the cued recall task and others to explore new technologies and interaction techniques. Problems with the proposed new technology are almost always identified, as rarely is a new design flawless in its first stages. And, iterative design-test-redesign works to improve the existing technology implementation from that point forward.

The downside of using traditional experimental designs and tasks in HCI work is that factorial designs with tight control and many subjects are simply not feasible given the time and resource constraints faced by design professionals. In addition, most real world designs consist of many variables and studying all possible combinations just isn't possible (or perhaps appropriate). Finally, as we saw in the bridge example, it is often the unanticipated uses of a system that are the most problematic, and these by their very nature are difficult to bring into the lab ahead of time. To understand some of the issues, think about how you would go about designing and evaluating a new voice-input word processor over a six-month period across multiple users, applications and languages. One might conjecture that important independent variables could be studied in isolation with the best of those being combined into a final design. In reality, this rarely works in human-computer interaction design. We have witnessed teams that have iteratively and carefully tested individual features of a software design until they were perfected only to see interactions and tradeoffs appear when all of the features were united in the final stages of design or when the system was used in the real world. Let's take a simple example from information retrieval. Highlighting the user's query terms in the listing of results allows for

more rapid identification of matching sections within documents. Semantic processing of query terms allows users to find more relevant information (e.g., normalization and spelling correction so that 'Susan Dumais" matches "Susan T. Dumais" or "Dr. Sue Dumais", or synonym expansion so that "HCI" matches "human-computer interaction" and perhaps even "computer-human interaction"). How do you combine these two features of highlighting and expansion? The more complex the processing that goes on, the harder it is to know what to highlight in the actual user interface to the information retrieval system.In addition to experimental methods, practitioners use a wide range of observational techniques and heuristics (e.g., field studies, contextual inquiry, heuristic evaluation, cognitive walkthrough, rapid prototyping, questionnaires, focus groups, personas and scenarios, competitive benchmarks tests, usage log collection, etc.) to better understand system usage and to inform design. These techniques are often borrowed from anthropology or sociology, and rarely would a system design be complete without these research techniques and tools. Often a qualitative description of the jobs people are trying to do, or how a system is used in the field by real people doing their real work is much more valuable than a quantitative lab study of one small system component. Observational techniques, task analysis, and related skills are much used in the practice of HCI but little represented in most cognitive psychology curricula. This is an area in which psychologists find their background and training lacking when first introduced to applied work. In addition, the methods themselves are not necessarily as well honed or useful as they could be in influencing user interface design. More research at the basic level is needed to better understand and extend these field methods. An important but often overlooked contribution is that psychologists often act as user advocates in informal but important ways. Being sensitive to individual differences, including the important realization that product managers and programmers are not "typical" users (for most systems) is a real contribution. Simply getting programmers to acknowledge that the user knows best will improve design. Evaluating and iterating on existing interfaces is only one aspect of the HCI design task. Generating a good design in the first place or generating alternatives given initial user experiences is equally important.

2.6 Ongoing Research in HCI

2.6.1. Mudibo: Multiple Dialog Boxes for Multiple Monitors

A general problem identified in recent research on multiple monitor systems is the placement of small windows such as dialog boxes and toolbars. These small windows could be placed on top of the application window or on a monitor next to the application window; different situations call for different placements. *mudibo*, a component of the window manager that alleviates this problem by initially placing a window in multiple locations simultaneously and subsequently allowing the user to easily interact with the window in a desired location. Additional important contributions of *mudibo* are that as a general technique it can be applied to a number of

situations and windows beyond simple dialog boxes, exploits the additional screen space that multiple monitors provide to solve a specific problem with dialog box interaction, and is among the first research prototype UIs that explicitly account for multiple monitor users.

Specific Application

There are two particular applications for which mudibo has found to be quite useful. One such application is instant messaging. In multiple-monitor systems, there is a danger that a user will miss the arrival of a new instant message if it appears on a monitor other than the one on which the user is currently interacting. Furthermore, knowing which monitor the user is viewing can be very difficult without an eye-tracker: even though the focus window might be located on one monitor, the user might be looking at another monitor. Even with an eye-tracker, the user might have risen for a moment to stretch, answer the telephone, *etc*. mudibo helps to ensure that the user sees the new instant message by presenting it on all of the monitors simultaneously (Figure). mudibo also allows the user to easily select which monitor is appropriate for the window. If the message will be dealt with in short order, the user might elect to interact with it on the same monitor as the window with which they are working. If the message will lead to a longer conversation, the user might elect to interact with it on an alternate monitor so as not to occlude the user's main task. A disadvantage of mudibo could be that a user does not want to deal with instant messages when the user is heavily focused on a task. Many IM clients also show a notification when a user comes online or goes offline and mudibo replicates those notification windows as well. If a multiple-monitor user is interested in seeing notifications, mudibo can decrease the chances that the user will miss them.

Fig. 2.3 Application of Mudibo

Figure. mudibo facilitates the appearance of a new incoming instant message on all three monitors of a multiple-monitor system. Monitor resolution has been decreased for visual effect. Typically, IM windows are much smaller and thus easier to miss.

2.6.2. Lumiere Project

The Lumiere Project centers on harnessing probability and utility to provide assistance to computer software users. The problems tackled in Lumiere research were

- The construction of Bayesian models for reasoning about the time-varying goals of computer users from their observed actions and queries,
- Gaining access to a stream of events from software applications,
- Developing a language for transforming system events into observational variables represented in Bayesian user models,
- Developing persistent profiles to capture changes in a user's expertise, and
- The development of an overall architecture for an intelligent user interface.

At the heart of Lumiere research and prototypes are Bayesian user models that capture the uncertain relationships among the goals and needs of a user and observations about program state, sequences of actions over time, and words in a user's query.

The high-level goals of the Lumiere project are captured by the influence diagram displayed in Figure which represents key aspects of user modeling and automated assistance in computing applications.

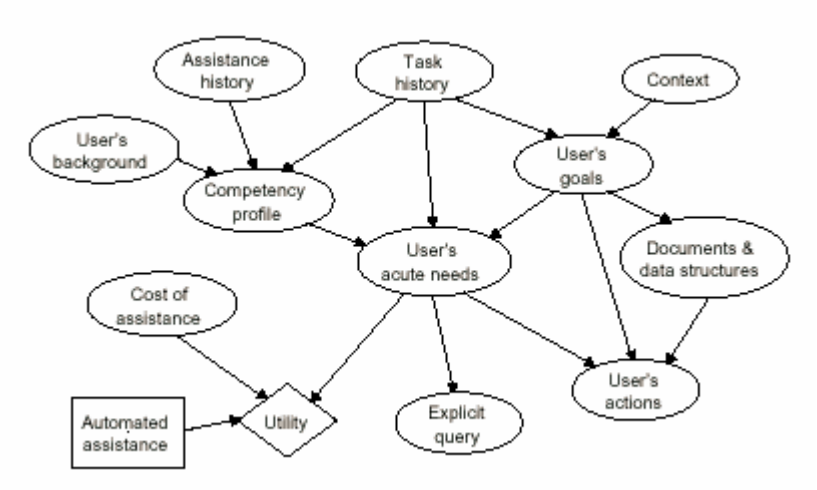

Fig. 2.4 The influence Diagram

Goals are target tasks or subtasks at the focus of a user's attention. Needs are information or automated actions that will reduce the time or effort required to achieve the goals.

In some situations, we can exploit a predetermined mapping between goals and needs. However, we typically must consider uncertainties and dependencies among these and related variables. As indicated by the variables and dependencies in the influence diagram,

A user's acute needs are influenced by the acute goals as well as the user's competency with using software. Prior assistance in the form of online help and the user's background representing experience with computing applications influence the user's competence.

A user's needs directly influence or cause patterns of activity that might be sensed by watching a user's activity. Such activity includes sequences of user actions recorded as the user interacts

with a mouse and keyboard as well as visual and acoustic clues that might be sensed by a video camera and microphone. A user's goals also influence the set of active documents and the presence, instantiation, and display of data structures (e.g., Does a user-authored chart exist in a spread- sheet project? Is the chart currently visible? Does it currently have system focus?).

At times, a user may explicitly request assistance. A user's acute needs influence the terms appearing in a user's explicit typed or vocalized queries. As indicated in the influence diagram, the overall goal is to take automated actions to optimize the user's expected utility. A system taking autonomous action to assist users needs to balance the benefits and costs such actions. The value of actions depends on the nature of the action, the cost of the action, and the user's needs. Assessing and integrating such user models would allow a system to compute a probability distribution over a user's informational needs in real time, given a set of observations about activity and explicit queries, when such queries are issued.

Architecture

The user interface agent of this project captures some atomic events like mouse clicks, keyboard events etc. Now each of these events get timestamped and Converted into higher level events like “mouse jitter’, “menu surfing” etc. These higher level events and explicit user queries (if provided) is used to decide an action with the help of the influence diagram. The following is a high level view of Lumiere architecture.

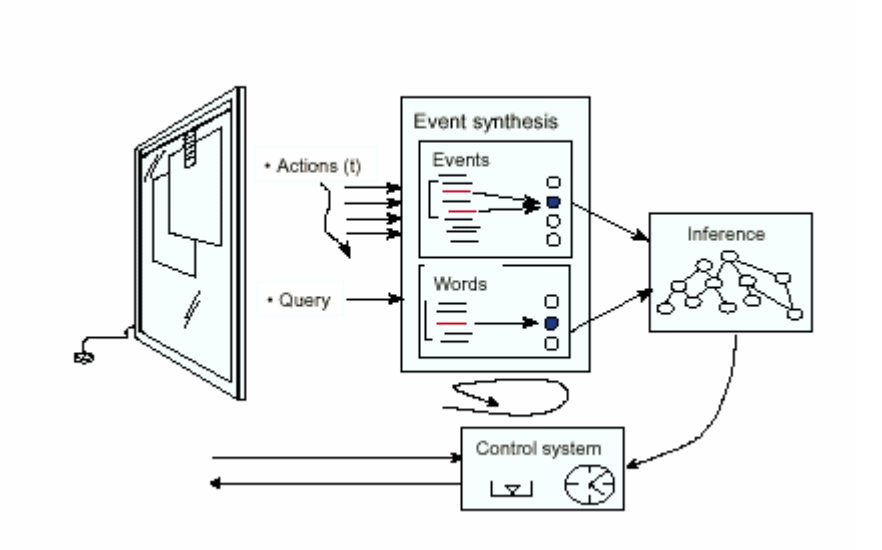

Fig. 2.5 High Level Architecture of Lumiere Project

The Lumiere project team worked closely with Microsoft Office development team. The system was first tested on MS Excel and the initial Lumiere/Excel prototype was first demonstrated to the Microsoft Office division in January 1994. Finally In January 1997, derivative of Lumiere research shipped as the Office Assistant in Microsoft Office '97 applications.

2.6.3. Voice-Operated Database Inquiry System (VODIS)

VODIS developed by British telecom, Logica ltd. And the university of Cambridge. The objective is to design a prototype that could eventually be used by ordinary telephone users to obtain train time table information about the departure and the arrival time of specific trains suitable for their personal travel requirements.VODIS uses continuous speech recognizer that accepts naturally spoken words and phrases and passes them to a dialogue controller which interpretes the received speech using knowledge of the application domain. Before building a prototype, studies of actual inquiries about rail travel were carried out and a model of an inquiry dialogue is developed. Constraints imposed by voice input and output technology also has to be taken into account.

2.6.4 Touch-Sensing Input Devices

The sense of touch is an important human sensory channel. In the present context, we use the term touch quite narrowly to refer to the tactile perception. During interaction with physical objects, pets or other human beings, touch (physical contact) constitutes an extremely significant event. Yet computer input devices, for the most part, are indifferent to human contact in the sense that making physical contact, maintaining contact, or breaking contact provokes no reaction whatsoever from most software. As such, touch-sensing input devices offer many novel interaction possibilities. Touch-sensing devices do not include devices that provide active tactile or force feedback. These are all output modalities that allow a device to physically respond to user actions by moving, resisting motion, or changing texture under software control. Touch sensing is an input channel; touch sensing allows the computer to have greater awareness of what the user is doing with the input device. Of course, certain input devices (such as touch pads, touch screens, and touch tablets) that require touch as part of their normal operation have been available for many years. In all of these devices, one cannot specify positional data without touching the device, nor can one touch the device without specifying a position; hence touch sensing and position sensing are tightly coupled in these devices. Yet once it is recognized that touch sensing is an orthogonal property of input devices that need not be strictly coupled to position sensing, it becomes clear that there are many unexplored possibilities for input devices such as mice or trackballs that can sense one or more independent bits of touch data (Fig. 2.6). There are two interaction techniques that match these new input devices to appropriate tasks. The On-Demand Interface dynamically partitions screen real estate depending on what the user is doing, as sensed by implicit interaction with touch sensors. For example, when the user lets go of the mouse, an application's toolbars are no longer needed, so we fade out the toolbars and maximize the screen real estate of the underlying document, thus presenting a simpler and less cluttered display. By contrast, touch sensors located above and below the wheel on the Scrolling Touch Mouse supports explicit, consciously activated interactions; the user can tap on these touch sensors to issue Page Up and Page Down requests.

Touch sensors allow this functionality to be supported in very little physical real estate and without imposing undue restrictions on the shape or curvature of the region to be sensed.

Fig. 2.6 The Touch Trackball and Scrolling touch

Fig. 2.6 shows the touch trackball (a modified Kensington Expert Mouse) senses when the user touches the ball and the Scrolling Touch Mouse (a modified Microsoft IntelliMouse Pro) senses when the user is holding the mouse by detecting touch in the combined palm/thumb areas. It can also sense when the user touches the wheel, the areas immediately above and below the wheel, or the left mouse button.

Now, let's consider the properties of touch sensors and touch sensing input devices in general. These are useful issues to consider when designing touch-sensing input devices and interaction techniques, and they may be suggestive of additional possibilities.

Although the touch sensors that we use do not sense positional information, since the geometric arrangement of sensors is known ahead of time, one can' potentially confer to the mouse properties that, in the past, have normally been associated with touch tablets. Thus touch sensors have some properties similar to those of touch tablets as enumerated by Buxton, Hill, and Rowley. For example:

- No moving parts: Touch sensors have no moving parts.
- No mechanical intermediary: Touch sensors require no mechanical intermediary to activate them.
- Operation by feel: Touch sensors can be arranged into regions that act like a physical template on a touch tablet. The user can feel the touch-sensing regions (e.g., the Page Up / Down controls on the Scrolling Touch Mouse) without looking at the device or at the screen. This can reduce the time that would be required to switch between devices or widgets on the screen.
- Feedback: Touch sensors differ from traditional pushbuttons in the amount and type of feedback provided. Compared to a mouse button, for example, the user does not feel or hear a distinct "click" when a touch sensor is activated. For cases where a touch sensor is being used in an implicit role and is not being used to simulate such devices, however, such feedback may not be needed or even desired.

2.6.5 AVANTI Project

The *AVANTI* AC042 project aims to address the interaction requirements of disabled individuals using Web-based multimedia applications and services. Along these lines, one of the main objectives of the work undertaken within the *AVANTI* project was the design and development of a user interface that would provide equitable access and quality in use to all potential end users. The User Interface (UI) of the *AVANTI* information system is a component which provides interactive views of adaptive multimedia Web documents. The distinctive characteristic of the *AVANTI* UI is its capability to dynamically tailor itself to the abilities, skills, requirements and preferences of the users, to the different contexts of use, as well as to the changing characteristics of users, as they interact with the system. The categories of disabled users supported in the current version of the system are: people with light, or severe motor disabilities, and blind people.

The *AVANTI* information system comprises five main modules:

1. A collection of multimedia databases which are accessed through a common communication interface (Multimedia Database Interface -MDI) and provide mobility information for disabled people;
2. The User Modeling Server (UMS),which maintains and updates individual user profiles, as well as user stereotypes;
3. The Content Model (CM),which retains a meta-description of the information available in the system;
4. The Hyper Structure Adaptor (HSA),which adapts the information content, according to user characteristics; and
5. The User Interface (UI) component, which is capable of tailoring itself to individual users.

The main concern here is the UI module. The UI module developed according to the Unified User Interface Design methodology (UUID), which has been proposed as an efficient and effective method for achieving the goal of user interfaces for all including disabled and elderly users. Following UUID, only a single unified user interface is designed and developed, which comprises alternative interaction components, appropriate for different target user categories.

The user interface first gets initialized according to some static characteristics of the user. These static characteristics include

- *Physical abilitie*s i.e. whether the user is able bodied, blind or motor-impaired;
- The *language* of the user (the system supports English, Italian and Finnish);

- *Familiarity* of the user with *computing networking hypermedia* applications, the *Web* and the *AVANTI* system itself;
- The overall *interaction target* speed, ease, accuracy, error tolerance
- *User preferences* regarding specific aspects of the application and the interaction; e.g. whether the user prefers a specific style for a given task; or the preferred speech volume when links are read; etc.

These data are collected either by questionnaire or by inserting smart card. After getting started the interface keeps changing its behavior depending on some dynamic characteristics of the user. These dynamic characteristics include

- *User familiarity with specific tasks* (capability to successfully initiate and complete certain tasks);
- *Ability to navigate* (move from one document to another in a consistent way);
- *Error rate*
- *Disorientation* (inability to cope with the current state of the system);
- *User idle time*
- *Repetition of interaction patterns* (commonly encountered sequences of interaction steps).

There is a decision mechanism which depending on these dynamic behaviors the user interface triggers certain actions which modify the interface. The interface also contains some additional features like

- Enhanced history control for blind users, as well as linear and non-linear (graph)history visualization for sighted users;
- Resident pages that enable users to review different pieces of information in parallel
- Link review and selection acceleration facilities
- Document review and navigation acceleration facilities
- Enhanced mechanisms for document annotation and classification
- Enhanced intra-document searching facilities

The *AVANTI* - UI is currently under evaluation by end users at three trial sites, in the context of the *AVANTI* system. The results of the evaluation will be used to refine the individual styles that comprise the interface, the interaction techniques developed specifically for disabled users, based on the supported special devices, as well as the adaptation rules used in the current version of the system. Future plans for the enhancement of the UI component include the integration of additional, non rule-based decision mechanisms, and the application of research results in the development of semantic level adaptation capabilities into the adaptation mechanism.

2.6.6 Projected Displays of Mobile Devices

Mobile devices (PDA, cellular phone etc.) have rapidly penetrated into our society and many people use them in their daily lives. For example, in Japan, the number of subscribers of cellular phones has amounted to 86 million, which is about three fourths of Japanese total population. One of the recent trends of cellular phones is multifunctional:

not only a phone to communicate with a person in a remote location but also a web browser, a digital video camera, a game machine, a music player, a television, a GPS, and so on. Although the growing trend toward more functions has been remarkably observed in cellular phones, the other mobile devices like PDAs also exhibit the similar tendencies, and various commercial accessories attachable to PDAs are available. This trend makes the differences between a mobile device and a personal computer smaller:

a cellular phone or a PDA is becoming a computer that has almost the same functionality of desktop/laptop computers retaining the feature of mobility. Actually, mobile devices have been used as a personal tool such as a personal scheduling assistant, and recently have begun to be used by multiple people in face-to-face or co-located situations.

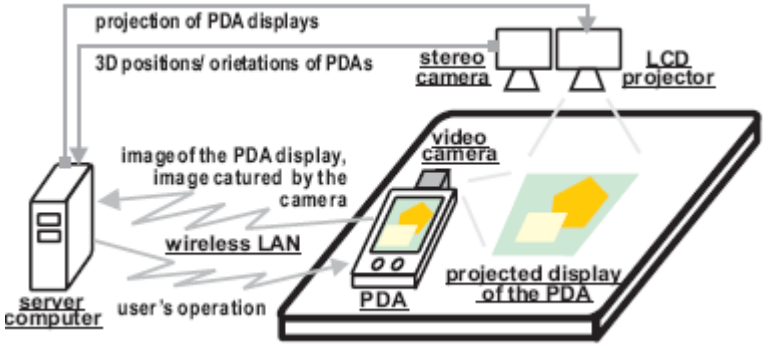

Fig. 2.7 An overview of the projected display

Let us take some examples. When you take a photo by using a digital camera mounted on your cellular phone, you may want to show the photo to (more than two) people around you. However, due to a problem of screen real estate of a cellular phone, it is not easy for multiple people to simultaneously look at the photo on your cellular phone. Moreover, when you send the photo to a person who has requested it, you need to conduct unintuitive and bothersome operations on your phone (e.g. selecting his mail address, attaching the photo to a mail, and send it through the user interface of the phone). Suppose if a display of your cellular phone can be projected on any wall in front of you. If your phone that mounts a projector is as light and small as the recent models of cellular phones, thanks to its mobility, you can make the projected display appear anywhere on a wall, a ceiling, a floor, or a desk, and easily look at a photo taken through your cellular phone with others. As your cellular phone is equipped with a digital video camera, it may also be possible to capture an image of its projected display. Therefore, if your cellular phone can recognize operations conducted on the projected display, for example, selecting a file or starting an application program with fingers, it allows multiple people in co-

located situations to interact with each other. Moreover, if displays of multiple cellular phones are projected on a wall and an image of their projected displays is captured by the cameras of the devices, it may be possible to allow people to conduct an intuitive operation, such as dragging a file from one phone to another by touching the projected displays with fingers. We generally believe that a mobile device that mounts a video camera and a projector has the possibility to become a new tool for supporting co-located collaboration in any location, retaining existing features of its personal use and mobility. Canesta Keyboard is a one-color projection system designed to be attached to a PDA and used as a personal virtual keyboard (therefore, inappropriate as a shared display for multiple people). Unfortunately, a projector mountable on a mobile device is currently not available due to its weight and power consumption. According to the recent news, however, such projectors will become available in near future.

2.6.7 iWork Project

The Stanford Interactive Workspaces project is exploring new possibilities for people to work together in technology-rich spaces with computing and interaction devices on many different scales. Within this project, an experimental research facility called the iRoom, located in the Gates Information Sciences Building at Stanford, was built.). The iRoom contains three touch sensitive white-board sized displays along the side wall, and a custom-built 9 mega pixel, 6' diagonal display with pen interaction called the interactive mural built into the front wall. In addition, there is a table with a built in 3' x 4' display that was custom designed to look like a standard conference room table. The room also has cameras, microphones, wireless LAN support, and a variety of wireless buttons and other interaction devices.
The main research objectives of this project are,

- Multi-device, multi-user applications
- Multimodal and fluid interaction
- Reusable, robust, and extensible system software for deploying COTS-based ubiquitous computing environments like our own
- integration of large (wall-sized) displays with advanced visualization capabilities into an iRoom
- integration of computing "appliances" including PDA's, scanners, digital cameras, etc. into an iRoom

To understand the application of this project let us compare a traditional conferencing scenario with an iRoom meeting.

In a traditional conference the speaker carries with him a removable storage medium (CD, Floppy etc.) containing his speech material. He copies it to the conference room computer and uses the projector to present it.In the case of iRoom, the speech material can be stored on the speaker's personal laptop or PDA. Upon entering the room he turns on projectors with the help of a touch screen. The speaker can present his materials directly from his laptop. He can also

redirect his laptops' pointers and keyboards to control the machines driving the displays when he need to directly control applications from where he is sitting. Additionally the system will provide a 3D viewer for better visualization and table displays for better collaboration. The following picture shows a photograph of the iRoom in use.

Fig. 28 A photograph of iRoom at work

The multi user, multi application distributed nature of iWork project is managed by a middleware system called iRoom Meta Operating System. The following picture shows the component architecture of the middleware system.

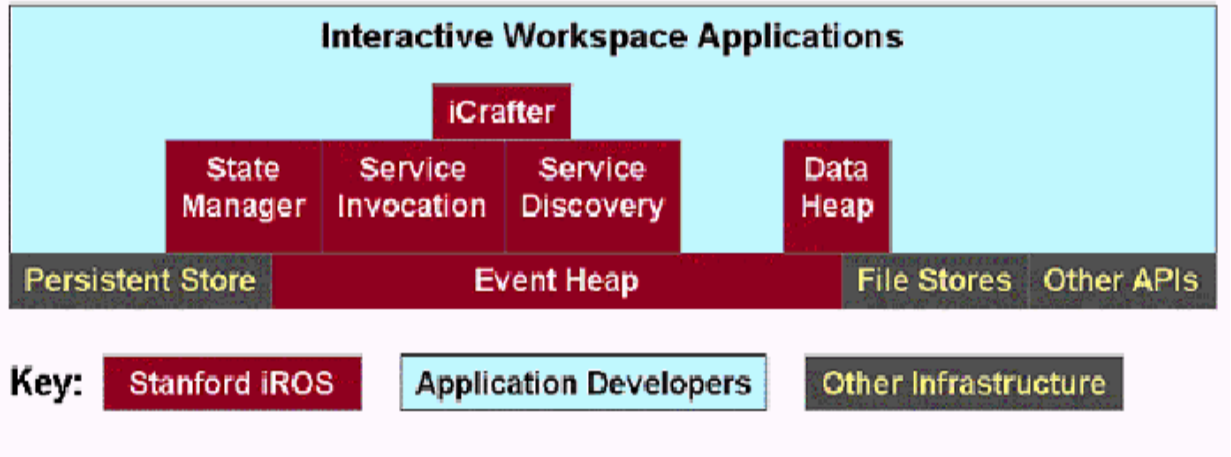

Fig. 2.9 iROS Component Architecture

The three iROS sub-systems are the Data Heap, iCrafter, and the Event Heap. They are designed to address the three user modalities of moving data, moving control and dynamic

application coordination, respectively. Figure above shows how the iROS components fit together. The only system that an iROS program must use is the Event Heap, which in addition to providing for dynamic application coordination is also the underlying communication infrastructure for applications within an interactive workspace.

The Event Heap

Given the heterogeneity in interactive workspaces and the likelihood of failure in individual devices and applications, it is important that the underlying coordination mechanism decouple applications from one another as much as possible. This encourages applications to be written which are less dependent on one another, thereby making the overall system less brittle and more stable. The Event Heap coordination infrastructure for iROS is derived from a tuplespace model (TSpace from IBM), which offers inherent decoupling.We are all accustomed with the Event Queue model. In Event queue the events are stored in FIFO order and consumed by an application. The event consumed by an application can not be reused by other application. In iWork by a single event we want to control more than one application e.g. by the Laptop mouse event we may want to control a PC that is controlling the projector. So the event queue model is not suitable for it.

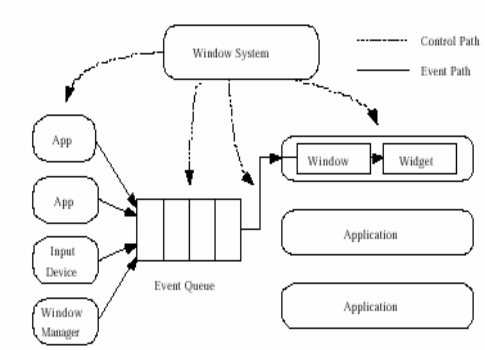

Fig. 2.9(a). Event Queue of a Window System

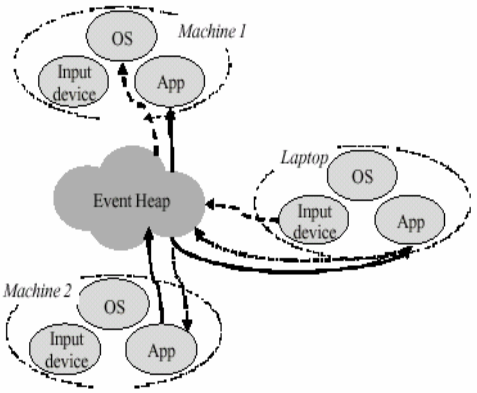

Fig. 2.9(b) . Event Heap

The Event Heap stores and forwards messages known as “events,” each of this is a collection of name-type-value fields. It provides a central repository to which all applications in an interactive workspace can post events. An application can selectively access events based on a pattern match over fields and values, and can retrieve either destructively or non-destructively. So when an event is consumed non-destructively it can be used by more than one application. There will not remain any unconsumed events because unconsumed events will be expired with time. This allows unconsumed events to be automatically removed, and provides support for soft-state through beaconing. . Applications can interface with the Event Heap through several APIs including web, Java, and C++. There is a standard TCP/IP protocol, making it easy to create

clients for other platforms. The Event Heap differs from tuplespaces in several other respects which make it better suited for interactive workspaces.

The Data Heap

The Data Heap facilitates data movement in an interactive workspace. It allows any application to place data into a store associated with the local environment. The data is stored with an arbitrary number of attributes that characterize it, and can be retrieved by a query specifying attributes that must be matched. By using attributes instead of locations, applications don't need to worry about which specific physical file system is being used to store the data. Format information is also stored in the Data Heap, and, assuming appropriate transformation plug-ins are loaded, data is automatically transformed to the best format supported by retrieving applications. If a device only supports JPEG, for example, a retrieved Power Point slide will automatically be extracted and converted into that image format.

iCrafter

The iCrafter system provides a system for service advertisement and invocation, along with a user interface generator for services. ICrafter services are similar to those provided by systems such as Jini, except that invocation is through the Event Heap, and soft-state beaconing is used instead of leases. The novel aspect of iCrafter is the interface manager service which provides a method for users to select a service or set of services to control, and then automatically returns the best interface to the service(s) for the user's device. The interface iCrafter generates communicates directly with the services through the Event Heap. When a custom-designed interface is available for a device/service pair, it will be sent. Otherwise, a more generic generator will be used to render into the highest quality interface type supported on the device. Generation is done using interface templates that are automatically customized based on the characteristics of the local environment. For example, if room geometry is available, a light controller can show the actual positions of the lights on a graphical representation of the workspace. Templates also allow multiple services to be combined together in a single interface— for example; all lights and projectors in a workspace can be put together.

2.6.8 Improving User Interaction with Spoken Dialog Systems via Shaping

Although speech recognition offers the promise of simple, direct, hands-free access to information, many factors, including environmental noise, non-native or regional speech patterns, or friendly yet "talkative" conversational agents, can make communication with spoken dialog systems inefficient. Inefficiencies in human-computer speech interaction are also commonly caused by the user having spoken beyond the bounds of what the computer
Understands. This situation leads to misunderstandings on the part of both the user and the system; recovering from such events can add extra turns and time to the overall interaction. There is one strategy for improving interaction efficiency with spoken dialog systems by encouraging users to adapt their interaction to match what the system understands best, thus

reducing the chance for misunderstandings. Because this adaptation should occur without explicit instruction from the user, this strategy is called *shaping*. Since shaping occurs during the interaction, it should reduce the need for intensive pre-use training.

General approach

The proposed approach to increasing interaction efficiency with spoken dialog systems has three main components. First, an *expanded grammar* will allow the system to accept more natural language input than is allowed by the target language, Speech Graffiti. The use of the expanded grammar will reduce training time and allow the system to be more forgiving for novice users, which should increase user satisfaction. Although the expanded grammar will somewhat broaden the range of utterances accepted by the system, its underlying purpose is not to make the system more natural, but to facilitate the processing of non-target language input in order to be able to help users eventually speak within the target language. *Shaping confirmation* will then provide an appropriate response for non-Speech Graffiti input that has been accepted by the expanded grammar, and will be designed to encourage users to use Speech Graffiti when speaking to the system. Finally, an error classification and response strategy will provide context-appropriate, *shaping help* for situations in which the recognized input string is accepted by neither the target nor the expanded grammar.

2.6.9 What You Look At is What You Get

It is all about the Use of Eye Movements in Human-Computer Interaction Techniques.

Current user-computer dialogues tend to be one sided, with the bandwidth from the computer to the user far greater than that from user to computer. A fast and effortless mode of communication from a user to a computer would help redress this imbalance. We therefore investigate the possibility of introducing the movements of a user's eyes as an additional input medium. While the technology for measuring eye movements and reporting them in real time has been improving, what is needed is appropriate interaction

techniques that incorporate eye movements into the user-computer dialogue in a convenient and natural way. It uses some of the human factors and technical considerations that arise in trying to use eye movements as an input medium.

General approach

The simplest eye tracking technique is electronic recording, using electrodes placed on the skin around the eye to measure changes in the orientation of the potential difference that exists between the cornea and the retina. However, this method is more useful for measuring relative eye movements than absolute position. Perhaps the least user-friendly approach uses a contact lens that fits precisely over the corneal bulge and is held in place

with a slight suction. This method is extremely accurate, but suitable only for laboratory studies. More practical methods use remote imaging of a visible feature located on the eye, such as the

boundary between the sclera and iris, the outline of the pupil, or the corneal reflection of a light shone at the eye. All these require the head to be held absolutely stationary (a bite board is customarily used), to be sure that any measured movement represents movement of the eye, not the head. However, by simultaneously tracking two features of the eye that move differentially (because of the difference in radius of curvature of the cornea and sclera), it is possible to distinguish head movements (the two features move together) from eye movements (the two move with respect to one another), and the head need not be rigidly fixed. This is currently the most practical method for use in a conventional computer-and-user setting, since the eye tracker sits several feet from the user, nothing contacts him or her, and the head need not be clamped. Figure shows the components of this type of eye tracker. It simultaneously tracks the corneal reflection (from an infrared light shining on eye) and the outline of the pupil (illuminated by same light). Visual line of gaze is computed from the relationship between the two tracked points.

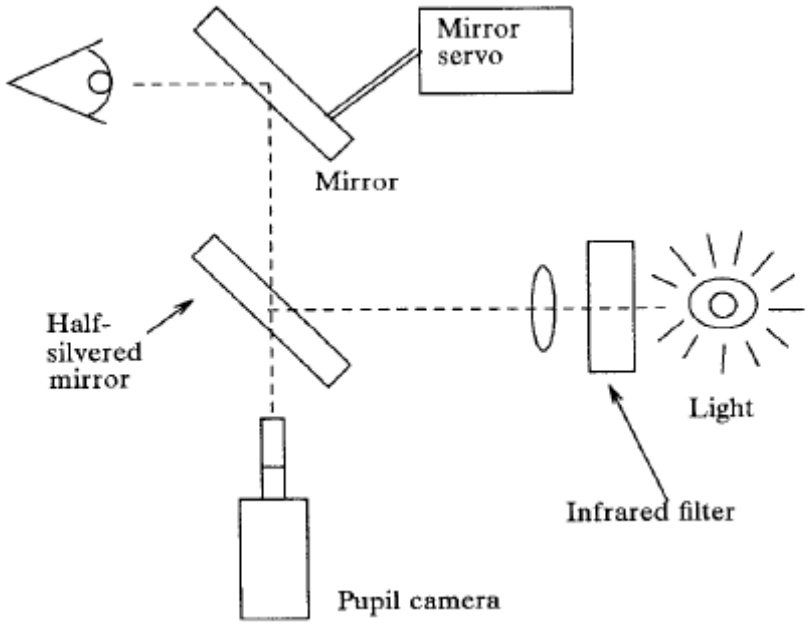

Fig. 2.10 Illustration of components of a corneal reflection-plus-pupil eye tracker the pupil camera and illuminator operate along the same optical axis, via a half-silvered mirror. The servo-controlled mirror is used to compensate for the user's head motions.

2.6.10 Oxygen Project

Let us consider we want to use a computer somewhere. So first we have to bring the machine there, have to install the required software, and have to learn the software and then we can start to operate it. Again to use the computer anytime we have to go to that air-conditioned room where it is placed. So we can say before getting any service from the machine we have to serve it lot.

The Oxygen project is an effort to revert the scenario. The name of the project implies that it will be made freely available as the natural gas, Oxygen. The system will enter the human world, handling our goals and needs and helping us to do more while doing less. We will not need to carry our own devices around with us. Instead, configurable generic devices, either handheld or embedded in the environment, will bring computation to us, whenever we need it and wherever we might be. As we interact with these "anonymous" devices, they will adopt our information personalities. They will respect our desires for privacy and security. We won't have to type, click, or learn new computer jargon. Instead, we'll communicate naturally, using speech and gestures that describe our intent ("send this to Hari" or "print that picture on the nearest color printer"), and leave it to the computer to carry out our will.

New systems will boost our productivity. They will help us automate repetitive human tasks, control a wealth of physical devices in the environment, find the information we need (when we need it, without forcing our eyes to examine thousands of search-engine hits), and enable us to work together with other people through space and time.

Challenges

To support highly dynamic and varied human activities, the Oxygen system must master many technical challenges. It must be

- *pervasive*—it must be everywhere, with every portal reaching into the same information base;
- *embedded*—it must live in our world, sensing and affecting it;
- *nomadic*—it must allow users and computations to move around freely, according to their needs;
- *adaptable*—it must provide flexibility and spontaneity, in response to changes in user requirements and operating conditions;
- *powerful, yet efficient*—it must free itself from constraints imposed by bounded hardware resources, addressing instead system constraints imposed by user demands and available power or communication bandwidth;
- *intentional*—it must enable people to name services and software objects by intent, for example, "the nearest printer," as opposed to by address;
- *eternal*—it must never shut down or reboot; components may come and go in response to demand, errors, and upgrades, but Oxygen as a whole must be available all the time.

Approach

Oxygen enables pervasive, human-centered computing through a combination of specific user and system technologies. Oxygen's user technologies directly address human needs. Speech and vision technologies enable us to communicate with Oxygen as if we're interacting with another person, saving much time and effort. Automation, individualized knowledge access, and

collaboration technologies help us perform a wide variety of tasks that we want to do in the ways we like to do them.

Oxygen's device, network, and software technologies dramatically extend our range by delivering user technologies to us at home, at work or on the go. Computational devices, called Enviro21s (E21s), embedded in our homes, offices, and cars sense and affect our immediate environment. Handheld devices, called Handy21s (H21s), empower us to communicate and compute no matter where we are. Dynamic, self-configuring networks (N21s) help our machines locate each other as well as the people, services, and resources we want to reach. Software that adapts to changes in the environment or in user requirements (O2S) help us do what we want when we want to do it.

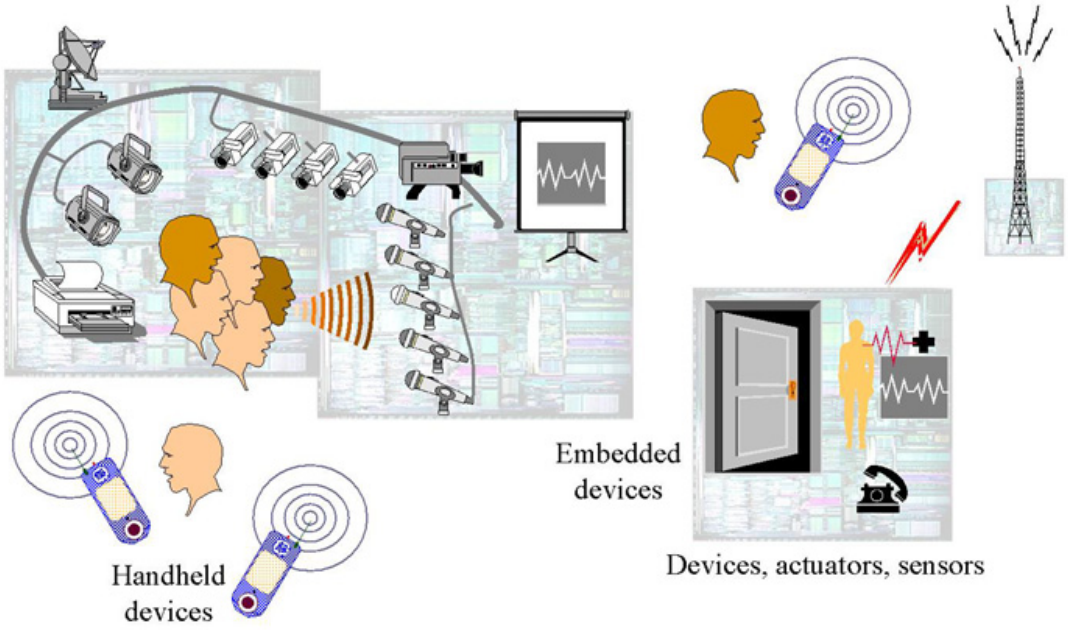

Fig. 2.11 An Overview of OXYGEN Project

Oxygen device technologies

Devices in Oxygen supply power for computation, communication, and perception in much the same way that batteries and wall outlets supply power for electrical appliances. Both mobile and stationary devices are universal communication and computation appliances. They are also anonymous: they do not store configurations that are customized to any particular user. As for batteries and power outlets, the primary difference between them lies in the amount of energy they supply.

Collections of embedded devices, called E21s, create intelligent spaces inside offices, buildings, homes, and vehicles. E21s provide large amounts of embedded computation, as well as interfaces to camera and microphone arrays, large area displays, and other devices. Users communicate naturally in the spaces created by the E21s, using speech and vision, without being aware of any particular point of interaction.

Handheld devices, called H21s, provide mobile access points for users both within and without the intelligent spaces controlled by E21s. H21s accept speech and visual input, and they can reconfigure themselves to support multiple communication protocols or to perform a wide variety of useful functions (e.g., to serve as cellular phones, beepers, radios, televisions, geographical positioning systems, cameras, or personal digital assistants). H21s can conserve power by offloading communication and computation onto nearby E21s.

Initial prototypes for the Oxygen device technologies are based on commodity hardware. Eventually, the device technologies will use Raw computational fabrics to increase performance for streaming computations and to make more efficient use of power.

Oxygen network technologies

Networks, called N21s, connect dynamically changing configurations of self-identifying mobile and stationary devices to form collaborative regions. N21s support multiple communication protocols for low-power point-to-point, building-wide, and campus-wide communication. N21s also provide completely decentralized mechanisms for naming, location and resource discovery, and secure information access.

Oxygen software technologies

The Oxygen software environment is built to support change, which is inevitable if Oxygen is to provide a system that is adaptable, let alone eternal. Change is occasioned by anonymous devices customizing to users, by explicit user requests, by the needs of applications and their components, by current operating conditions, by the availability of new software and upgrades, by failures, or by any number of other causes. Oxygen's software architecture relies on control and planning abstractions that provide mechanisms for change, on specifications that support putting these mechanisms to use, and on persistent object stores with transactional semantics to provide operational support for change.

Oxygen perceptual technologies

Speech and vision, rather than keyboards and mice, provide the main modes of interaction in Oxygen. Multimodal integration increases the effectiveness of these perceptual technologies, for example, by using vision to augment speech understanding by recognizing facial expressions, lip movement, and gaze. Perceptual technologies are part of the core of Oxygen, not just afterthoughts or interfaces to separate applications. Oxygen applications can tailor "lite" versions of these technologies quickly to make human-machine interaction easy and natural. Graceful interdomain context switching supports seamless integration of applications.

Oxygen user technologies

Several user technologies harness Oxygen's massive computational, communication, and perceptual resources. They both exploit the capacity of Oxygen's system technologies for change in support of users, and they help provide Oxygen's system technologies with that capacity. Oxygen user technologies include:

Automation technologies, which offers natural, easy-to-use, customizable, and adaptive mechanisms for automating and tuning repetitive information and control tasks. For example, they allow users to create scripts that control devices such as doors or heating systems according to their tastes.

Collaboration technologies, which enables the formation of spontaneous collaborative regions that accommodate the needs of highly mobile people and computations. They also provide support for recording and archiving speech and video fragments from meetings, and for linking these fragments to issues, summaries, keywords, and annotations.

Knowledge access technologies, which offer greatly improved access to information, customized to the needs of people, applications, and software systems. They allow users to access their own knowledge bases, the knowledge bases of friends and associates, and those on the web. They facilitate this access through semantic connection nets.

Some Current Applications Oxygen Project

Oxygen technologies are entering our everyday lives. Following are some of the technologies being tested at MIT and by the Oxygen industry partners.

Oxygen device technologies

A prototype H21 is equipped with a microphone, speaker, camera, accelerometer, and display for use with perceptual interfaces. RAW and Scale expose hardware to compilers, which optimize the use of circuitry and power. StreamIt provides a language and optimizing compiler for streaming applications.

Oxygen network technologies

The Cricket location support system provides an indoor analog of GPS. The Intentional Naming System (INS) provides resource discovery based on what services do, rather than on where they are located. The Self-Certifying (SFS) and Cooperative (CFS) File Systems provide secure access to data over untrusted networks without requiring centralized control. Trusted software proxies provide secure, private, and efficient access to networked and mobile devices and people. Decentralization in Oxygen aids privacy: users can locate what they need without having to reveal their own location.

Oxygen software technologies

GOALS is an architecture that enables software to adapt to changes in user locations and needs, respond both to component failures and newly available resources, and maintain continuity of service as the set of available resources evolves. GOALS is motivated, in part, by experience gained with MetaGlue, a robust architecture for software agents.

Oxygen perceptual technologies

Multimodal systems enhance recognition of both speech and vision. Multilingual systems support dialog among participants speaking different languages. The SpeechBuilder utility supports development of spoken interfaces. Person tracking, face, gaze, and gesture recognition utilities support development of visual interfaces. Systems that understand sketching on whiteboards provide more natural interfaces to traditional software packages.

Oxygen user technologies

Haystack and the Semantic Web support personalized information management and collaboration through metadata management and manipulation. ASSIST helps extract design rationales from simple sketches.

The Intelligent Room is a highly interactive environment that uses embedded computation to observe and participate in normal, everyday events, such as collaborative meetings.

2.6.11. FeelTip: Tactile input Device for Small Wearable Information Appliances

Miniaturization of information appliances is enhancing their mobility and wearability, but it is also introducing many new problems. One of the main problems is that their small formfactor does not allow the space for traditional input devices. Input operations, however, such as text entry and menu selection are essential part of interaction for any information appliances. The current input devices for small information appliances such as a small keypad, a touchpad, and a touch sensitive screen will not be viable any more as
miniaturization of information appliances continues to the point of a wristwatch or a pendant. A new input device that we need for such a small wearable device should be of course small and should not require a large working space. In addition to being small, a new input device should cope with a much more dynamic environment and even try to be
aesthetically pleasing because a small wearable device is not only small but also “wearable”. In this project we introduce a new input device that is ideal for small wearable information appliances such as a watch phone and a mini music player. It does not require more space than a miniature tact switch, but it is a free pointing device more efficient than an analog joystick and is also a text entry device comparable with the usual mobile phone keypad.

The concept of the FeelTip, the new input device that we introduce in this paper, is illustrated in Figure below. The main idea is to exchange the usual role of a finger and a touch sensitive surface of a touchpad. An input device now has a tip and a finger now provides a surface. The result is an input device requiring a minimal space but is potentially more efficient than a touchpad due to the tactile feedback of a tip on a finger.

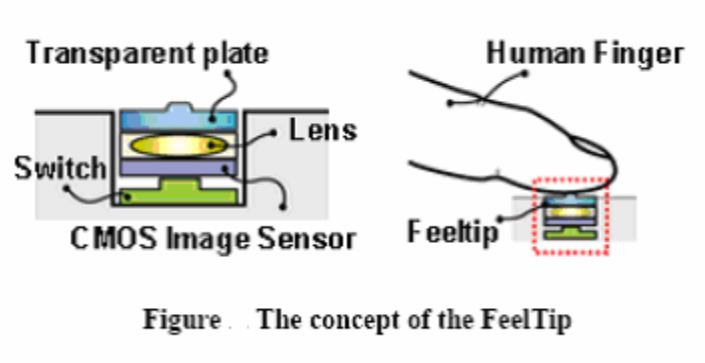

Fig. 2.12 The concept of FeelTip

The concept is simple but the realization of the concept required a technique to detect the position of a tip on a fingertip. One of the techniques that we could conceive was to use a small camera and track the fingerprint, as shown in Figure. As it turned out soon, what we need was already available off-the-shelf; the optical sensor chip designed for an optical mouse was found to be exactly to our purpose. The figure shows a prototype FeelTip that is built using the optical sensor chip, ADNS-2620 from Agilent. The chip has a small CMOS area for imaging and additional circuitry for image processing and communication with a host system. An accompanying miniature lens, which is of course designed for a mouse application, was good enough for our purpose. The only add-ons that we need to supply were a transparent sensor chip. A small program running on a host computer interfaces the prototype unit and applications. The program translates displacement data received from the optical sensor chip to a series of mouse messages. Also, it translates the button event from the prototype to a left-button mouse message.

2.6.12. Pervasive computers: context-based computing

Today we are facing the end of the dominance of the traditional personal computer. Computing is already embedded in more places than just our desktop computers. Computers make our cars run properly with antilock braking systems and power steering. These examples illustrate what seamless computing should be—it can provide wonderful functionality without requiring that the user understand its inner workings. We already know what it is like to live in a world where the physical is being inhabited by the digital. More and more, the digital will permeate the physical space in a seamless manner. We will expect computing to be everywhere. Computers will not only be increasingly mobile, but information will be accessible from any mobile position. We should not have to carry around devices containing our information. Rather, devices will recognize who we are and obtain information about us, through “remembrance agents” or adaptive user models, Internet information storage, or other means. Information appliances have human-computer interfaces. An information appliance should be easy for anyone to use and the interaction with the device should be intuitive. Careful design is critical for an intuitive inter action with the device. Although the desktop computer can do many things, this functionality can be separated into more appropriate devices. Some examples of successful

popular devices are cellular phones, pagers, televisions, wristwatches, and toasters. Of course, there can be times when these devices become difficult to use, but in their basic form, they meet the criteria for information appliances. Devices will become more “aware.” A device will be more aware of its user and more aware of its own environment. Devices will not only be able to sense the presence of a user but also be able to sense the user’s needs and goals. Devices will be aware of their position and their surroundings. Bio-sensing will become prevalent throughout the environment, not only for entertainment and medical reasons, but also to enhance person-to-person communication. When devices become more aware, they can be responsive and seem “smarter.” Computers will have the sensory devices analogous to human senses: sight, sound, speech, touch, and smell. Perhaps the best way for computers to really help humans is for computers to become more a part of the physical, human world. Maybe it is the nature of humans to create things with an image of themselves in mind.

If you walk up to an exercise machine and you are wearing some sort of identifier—a personal area network (PAN)—the exercise machine knows how to train you and to reward you. When a user sits down at an exercise machine, the electronic trainer speaks to him or her. It starts by recognizing the user and recounting what the user has done before on this machine. It may say, for example, “Hi Wendy, it’s been two days since you worked out. I think you’re ready to raise the weight by ten pounds. What do you say we do ten reps (repetitions) at 100 pounds?” A screen allows the user to interact with the machine and change its program. In any case, as the user works out, the exercise machine encourages the user to pull faster, or longer, or slower to get a better workout. In the example, going through the entire ten repetitions causes the system to say, “You’ve had a great workout.

Go over to the vending machine and get yourself a reward.” The exercise machine can then give an appropriate reward by activating the vending machine. Although not part of the demonstration, a foreseeable future includes bio-sensing. In this scenario, the exercise machine knows how many muscles are working with respect to the person’s capacity. It knows how much the person is perspiring with respect to a normal workout. An intelligent coach can give feedback on the workout. “If you keep working out like this, you’ll look like this (display of projected image) in one year.” The intelligent coach can display body fat count, weight, or whatever is the best motivator for that person, taking into account the person’s mood and personality.

This kind of information could be obtained using emotion technology.

1. This technology uses biosensors that are non intrusive and virtually invisible to the user. These sensors collect physiological information about the user so that, over a period of time, we can identify the person’s emotional state. With emotional state information, along with the current task, a model of a person’s personality can be developed over a period of years. Gaze tracking as another way to unobtrusively gain information about a user; the use of computers with the Global Positioning System (GPS); and natural interactions with computers through

multimodal input and output. Our initial efforts toward this last model are described by our BlueEyes project.

2 In addition, to incorporate gaze tracking, emotion detection, speech recognition, and gesture recognition in a “smart” computer. These have allowed us to explore the field of contextual or *in situ* computing. As computers become more a part of our lives, we all should have a say in how we interact with these computers. It is a place to look for provocative possibilities: social, cultural, and physical.

2.6.13 Perceptive Assistive Agents

Perceptual computing is concerned with using information from a variety of sensors such as video, audio and touch to make the computer aware of what a human is saying or doing via head, hand or body gestures. It also encompasses multimodal input and output. Though team spaces are supported by a variety of technologies from infrastructure to applications to devices, the combined use of these technologies is limited by physical context awareness. That is, the environment is not very adaptable to changes in group structure since elements of the environment (for example, noise level, location and number of participants, focus of activity, user technology savvy, and availability of resources) are not taken into account. Focus is on the ability of assistive agents to monitor, access, and manipulate elements of the physical context, such as locating speakers and adjusting audio input to enhance speech recognition and remote listening quality or enriching the collaborative experience of remote participants via sound localization, panoramic video and shared views. Through the use of perceptive team space agents managing interactions between humans and the collaboration environment we will enhance team interaction and performance in both local and remote group interaction.

Perceptual interfaces seek to leverage information about how humans interact in the physical world. Sensors and devices should be transparent and passive. Perceptual intelligence is key to interfacing with future generations of machines. These machines include both large-sized interfaces (e.g., large plasma displays) as well as wearable and other ubiquitous devices. Familiar WIMP (windows, icon, menus, pointer) interfaces are not ideally designed for these sorts of computing environments. If we are to assess the potential of advanced HCI technologies to impact the collaborative process, we require a means to experiment on actual users in a working environment. The MITRE Experimental Team Room (ETR) provides a platform for experimentation.

Two objectives of project:

1) Enhance the effectiveness of assistive agents by leveraging perceptual information accessible from the physical environment and

2) Provide an experimental foundation for evaluating sophisticated HCI concepts, specifically, for assessing the potential benefits of perceptive assistive agents in the context of team room interactions. These interactions range from small informal working group meetings, discussion

groups, instruction, and presentation. We are not only concerned with team activity when participants are co-present, but are also concerned with teams including remote participants – especially those connecting from the desktop or with mobile devices.

An initial ETR facility built on the existing state of the art MITRE team room facilities, which is envisioned as a future experimental facility for internal operational use of advanced collaboration technologies and human computer interfaces. Rather than being a static showcase of cutting edge technologies, it is seen as the first step in a "plug- and-play" user experimentation lab where technology and operational use emerge in an ongoing manner.

Fig. 2.14 3D model of the ETR

A prototype embodied conversational agent, Emma (Electronic MITRE Meeting Assistant) interfaces with the room's control system and provides the comprehensive set of functions currently available from the touchpad control panel sitting in every team room. A distributed software interface that enables Emma to control applications on team room PCs is installed. Users can communicate with Emma either via speech or through a context-sensitive graphical user interface.

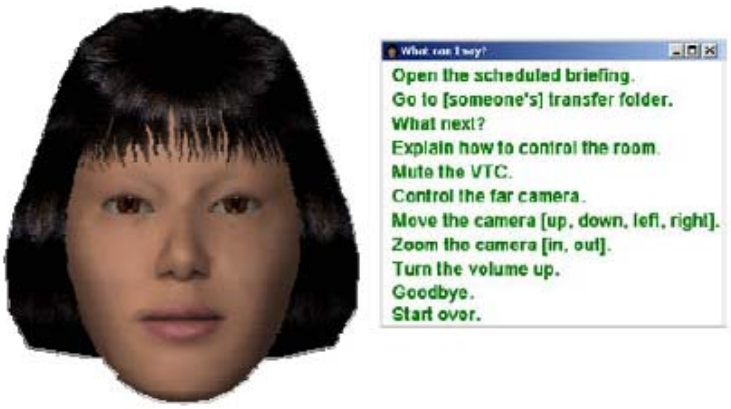

Fig. 2.15 Emma's head and "What Can I Say" window

Currently, Emma is able to control lighting, speaker volume and camera angles, turn on the displays, initiate video-teleconference connections, and navigate PowerPoint presentations. She can also access the corporate network to retrieve files and look up basic employee data such as phone number and user ID. Emma has both tutorial and help modes to assist all levels of potential users. She was designed using custom head animation software, COTS automatic generation of synthetic speech and a research dialogue engine from Mitsubishi Electronic Research Laboratory. Ultimately, Emma provides an alternative interface to the current touchpad interface and will gradually be equipped to deal with a greater number of resources as well as communicate with other "Emmas" residing on desktops interfacing with other team rooms.

Though Emma has great promise as a meeting assistant who manages devices in the room and accesses scheduling and other information for users, she is effectively "blind" and cannot see user actions except for very specific sorts of events. To remedy this, current work is in the process of creating a sensor-based environment, focusing particularly on microphone, video, wireless resource discovery, and identity detection (e.g., via RF emitters on badges). With these sensors in place, Emma will have access to information about *who* is in the room, *where* they are, *what* they are doing, and *how* they are doing it. She will use this information to enhance interaction in a number of ways. As an embodied agent, information about Emma's focus of attention and state are visual. That is, users can see if she is "awake" and listening. Given access to perceptual information such as speaker location, she will be able to integrate her physical activity more closely with the activity in the room. For instance, she can turn her head to look at different speakers, gesture toward people in the room, and use non-verbal cues to ground the conversation.Emma's knowledge of who is present can be used to automatically configure the room according to the preferences of those users, and to pre-fetch relevant resources (e.g. the meeting participants' shared network folders). Combining this with scheduling information, Emma can make the appropriate video teleconference connection and open the relevant PowerPoint file as soon as the meeting participants enter the room. If meeting participants are absent, Emma should be aware of this and be able to contact them upon request.

Another use of perceptual information is to dynamically improve audio quality. Current team rooms suffer from poor audio quality across remote connections. Speech recognition is also poor in these environments. A significant factor is the presence of multiple sources of noise in the environment including other speakers. Even if there were only one speaker, that speaker is often mobile. We could believe that a network of phased array microphones that is able to identify the most important signal in the room and dynamically filter out other noises may provide the key to this puzzle. This ability to identify the most relevant source of auditory information and focus on it is not too difficult for human participants, but very challenging for machines. Finally, with enhancing a sense of "near presence" among remote meeting participants. Ultimately, teams have members that are not physically co-present at some time

or another. Often they connect in via VTC or telephone. But they typically don't feel as included in the interaction as if they were actually present. Emma's perceptual information can be leveraged to support remote users sense of co-presence in a variety of ways. For instance, a panoramic camera that automatically adjusts its position to identify shared visual focus of attention might be desirable. But recognizing that individuals also maintain an individual focus of attention, remote users should be able to manipulate visual attention manually as well.

References

1. Human-Computer Interaction in the Control of Dynamic Systems:WILLIAM B. ROUSE
University of IUinois, Urbana, Illinois

2. A look at human interaction with pervasive computers by W. S. Ark ,T. Selker

3. Intuitive Manipulation Techniques for Projected Displays of Mobile Devices
Kosuke Miyahara1 Hiroshi Inoue1 Yuji Tsunesada1 Masanori Sugimoto

4. A Brief History of Human Computer Interaction Technology: Brad A. Myers, December, 1996

5. Goldberg, A. and Robson, D. "A Metaphor for User Interface Design," in *Proceedings of the 12th Hawaii International Conference on System Sciences.* 1979.

6. Henderson Jr, D.A. "The Trillium User Interface Design Environment," in *Proceedings SIGCHI'86: Human Factors in Computing Systems.* 1986. Boston, MA.

7. mudibo: Multiple Dialog Boxes for Multiple Monitors Dugald Ralph Hutchings, John Stasko

8. Touch-Sensing Input Devices *Ken Hinckley and Mike Sinclair* Microsoft Research,
Microsoft Way, Redmond.

9. Improving User Interaction with Spoken Dialog Systems via Shaping Stefanie Tomko
Language Technologies Institute Carnegie Mellon University

10. The Use of Eye Movements in Human-Computer Interaction Techniques:What You Look At is What You Get ROBERT J. K. JACOB, Naval Research Laboratory

11. Perceptive Assistive Agents in Team Spaces Lisa D. Harper, Abigail S. Gertner, James
A. Van Guilder The MITRE Corporation 7515 Colshire Drive

12. FeelTip: Tactile input Device for Small Wearable Information Appliances:
Sunyu Hwang, Geehyuk Lee,Buyong Jeong, Woohun Lee,Ilyeon Cho

13. Building Bridges from Theory to Practice Susan Dumais and Mary Czerwinski
Microsoft Research, One Microsoft Way, Redmond, WA 98052