Advanced Information Systems Engineering, 13th International Conference (CAiSE2001), Interlaken, Switzerland, June 2001, in Lecture Notes in Computer Science, Vol. 2068, Springer-Verlag, 2001.

Project Oxygen: Pervasive, Human-Centric Computing – An Initial Experience

Larry Rudolph

MIT Laboratory for Computer Science
rudolph@lcs.mit.edu

1 Introduction

For the past six months, I have been integrating several experimental, cutting-edge technologies developed by my colleagues at MIT as part of the MIT LCS/AIL Oxygen project. This paper gives a snapshot of this work-in-progress.

Project Oxygen is a collaborative effort involving many research activities throughout the Laboratory for Computer Science (LCS) and the Artificial Intellegence Laboratory (AIL) at the Massachusetts Institute of Technology (MIT). The Oxygen vision is to bring an abundance of computation and communication within easy reach of humans through natural perceptual interfaces of speech and vision so computation blends into peoples' lives enabling them to easily do tasks they want to do – collaborate, access knowledge, automate routine tasks and their environment. In other words, *pervasive, human-centric computing.*

At first blush, this catch-phrase appears vacuous. Today, computers are certainly pervasive; it is likely, at this moment, you are within 100 meters of a computer. Computers are certainly human-centric; what else can they be? On the other hand, computers are not yet as pervasive as is electricity or water. Although computers perform jobs required by humans, they do not feel human-centric – humans must conform to an unnatural way of communicating and interacting with computers. Finally, the tasks described have little to do with computation; computer-mediated functions is a more accurate term but sounds awkward.

The vision and goals of the Oxygen project are described in detail elsewhere [11, 2, 1], the purpose here is to show how many maturing

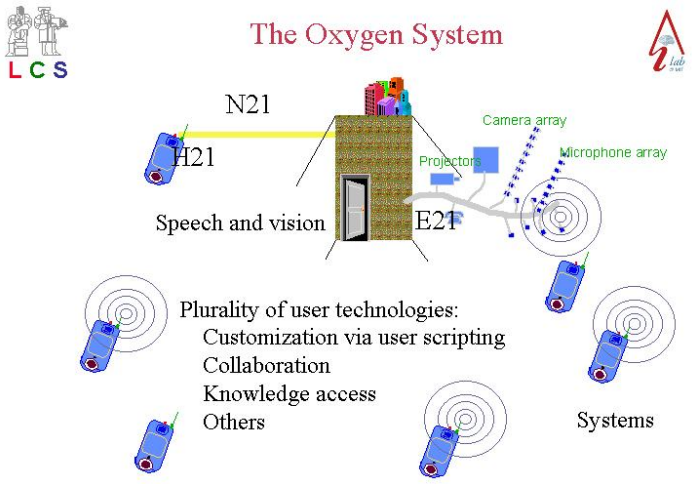

Fig. 1. An overview of the Oxygen Infrastructure, showing the division into three parts: H21, a handheld digital device, N21, the network infrastructure, and E21, the environment infrastructure.

technologies can be integrated as a first step towards achieving the Oxygen vision. There are research efforts at other universities and research institutions that roughly share the same vision, however, each institution focuses on integrating their own maturing technologies. Oxygen has a three-pronged approach by dividing the space into three broad categories: the H21, a hand-held device, the N21, an advanced network, and the E21, a sensor-rich environment (see Figure 1).

In what follows, an Oxygen application is described in terms of its human-centric features as well as the required technologies. It is important to keep in mind that this is just one of many applications and that it is merely a vehicle to explain how many technologies can be integrated and how to create the infrastructure necessary to enable the introduction of context into applications making them more "natural" to use.

The sample application is that of a seminar presentation support system. The next section gives an overview of the application. Section 3, reviews many of the technologies that will go into this application. Section 4, shows how they integrate to form the application. A preliminary programming language and middleware support is described in Section 5.

2 A computer-mediated, seminar presentation system

This section describes a computer-mediated seminar presentation system. As you read through the description, compare it to how presentations are given today. Although a laptop with programs like Powerpoint or Freelance attached to an LCD projector is a vast improvement over the old days of foils or 35mm slides, the human has given up a degree of control, freedom, and naturalness. The system described below provides for a more natural human interface.

Alice is to give a seminar about her $O_{2.5}$ project. As she walks into the seminar room, she allows herself to be be identified as the speaker. She does not need to carry a laptop with her slides on it – all of her files are globally accessible. Alice tells the system how to find her talk by simply supplying enough keywords to uniquely identify the file she wants. Her files are well indexed and so she merely describes the file in human terms and not with some bizarre syntax. The system knows where she is and marshals all the physical components that may be needed for her to control the display.

Alice wants to control the display so that it matches her current desires. A seminar is a live event and the dynamics depend on the audience and speaker. Although it is crucial that she control the presentation, this control should be of minimal distraction. The same is true for the audience – they should be able to see the visual content, hear her commentary, and take notes at the same time. Moreover, unexpected events should be handled in a natural way.

Even today, Oxygen technologies can make a computer-mediated presentation a more natural experience. In particular, three natural ways to control the slides are provided, as opposed to the traditionally way where Alice either clicks the mouse or hits the enter key. It is computer-centric to force the speaker to always walk over to the laptop in order to advance the slide. A wireless mouse is only a partial solution as it requires that something be held in a hand. For Bob this might be fine, but Alice likes to use a laser pointer to highlight objects on the screen and she finds holding two objects to be very awkward. An integrated pointer/mouse is no better since it now requires attention to find the correct button.

Alice can use her laser pointer to highlight words, draw lines and sketches, as well as to switch between slides. Holding the pointer in the bottom right corner means to advance to the next slide. A camera looking at the screen interprets Alice's laser pointer gestures. But not all humans like to use laser pointers. Some people, especially when they

are continuously engaged in speaking, like to use verbal commands to control the presentation. This is done with a microphone and software that continuously tries to understand commands. All three modes of control will always be active, allowing the speaker to use whatever is convenient.

There is more to a presentation than just advancing slides. Alice may want to see her notes, see the next slide before the audience, skip a slide or present the slides in a different order. A laptop, handheld, or any other personal communication device can be used by the speaker. To skip to a different slide without anyone knowing it, is a task that is easily performed by simply clicking on a different slide image on her personal display. The personal display must remain consistent with the public display. So, whether Alice says "Next Slide," chooses a slide from her private computer (handheld or laptop), or uses the laser pointer, both displays are updated.

The audience should also have a choice of ways to observe the presentation. They can look at the large projection screen in the front of the room, as is usually the case, or they may choose to view the presentation on their own personal digital device. The output is simultaneously broadcast to these devices. Some people in the audience might like to take notes and have them correlate with the presentation itself. We propose broadcasting a URL or some other identification symbol for the current contents. This can be either used to display the slide on the laptop, or be inserted into their notes. Later on in the privacy of their own room, these notes can be merged with an archived version of the talk. The archived version will match the presentation rather than the original file. Alice may have many "emergency" slides prepared that will be shown only in response to a question.

To summarize, there are several output modalities: the LCD projection, a broadcast of the current content, an archival copy that can be accessed afterwards, and the ability to correlate the public presentation with her own personal view of the presentation.

Lastly, Alice also has "meta" operation control - e.g. switching to a different presentation package, such as a browser or Mathematica, or even to the contents of another presentation. She should also be able to control whether or not content is broadcast or archived.

3 Technology overview

Research into many technologies that support the above scenario being pursued as part of Project Oxygen. Once again, we wish to emphasize that there are many competing technologies being developed elsewhere.

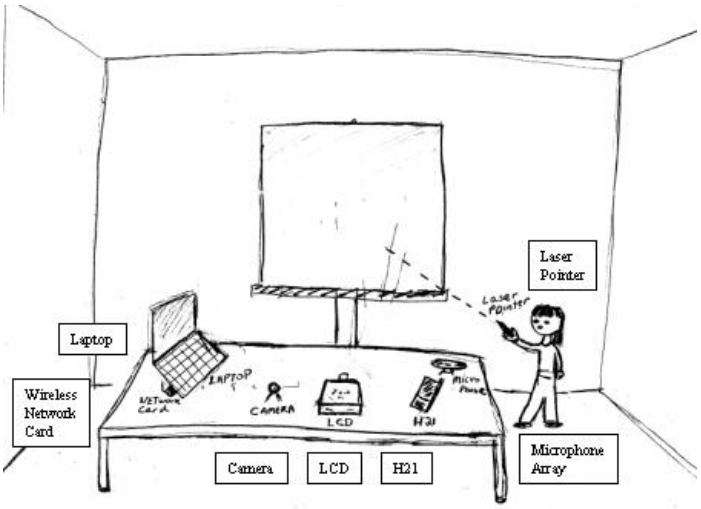

Fig. 2. The seminar room can be assembled from off-the-shelf components. The laptop controls the LCD, camera, microphone, and the networking parts of connecting to the file system, broadcasting, and archiving. The H21 is used by the speaker for personal notes and skipping slides. This application makes use of many of emerging technologies being pursued at the Lab. for CS and AI Lab at MIT.

We deliberately ignore them for several reasons[1]. First, to do justice to them all would make this article too large. Second, close physical proximity is usually required when making use of experimental, research systems. While it is possible to do this remotely for one component, it is nearly impossible do this for a number of research projects. We wish to provide feedback to these other research projects before they are ready for prime-time and we deliberately try to use them in some unintended way. While there are similar efforts in many of the intelligent or instrumented rooms, our example is simply geared towards exposing how components interact even with commodity hardware. As fun as it is, the particular demo of an oxygenated presentation is not the goal.

3.1 The Handy 21 (H21)

Although the commercial sector has been cranking out all kinds of hand-held devices, there is still much research to be done. The H21 should replace the plethora of communication gadgets with a single portable device. In particular, it should combine at least the functions of a cellular phone, wireless Internet connection, pager, radio, as well as a music and video player/recorder. Packing all this functionality into a single device appears to make it too heavy to be portable. So, industry strives to find the right set of combinations and to then sell add-ons to fill-in the missing pieces. The Oxygen approach is different: all that is needed is a minimal set of components built into the hardware with software and reconfigurable hardware used to provide whatever functionality is needed.

The SpectrumWare project [13] is developing a multipurpose communications system that can be programmed to receive and transmit many different types of signals in order to form a "communications chameleon." One can program the H21 to be a radio, cell phone, or television receiver. To fit in a small space it will need configurable hardware.

The RAW project [3] is developing a chip that will deliver unprecedented performance, energy efficiency and cost-effectiveness because of its flexible design, by exposing its wiring to the software system, the chip itself can be customized to suit the needs of whatever application is running on it. The Raw chip could be incorporated into a single de-

[1] The author wishes to apologize to all those who do not agree with these reasons. In a future, expanded version of this paper, many competing technologies will be cited. The author would be happy to learn about an relevant research.

vice that could perform a wide variety of applications: encryption or speech recognition, games or communications.

The commercial sector also understands the need for low-power devices, especially handheld ones. However, to make substantial improvements, it is important to re-examine computer architecture from basic principles. The SCALE project [6] is aimed at doing just that.

Rather than waiting for this research to come to fruition, the Oxygen project will make use of commercial handheld computers. In fact, it is doubtful that we will ever build our own device. More likely, we will continue to modify and adapt commercial products that at a minimum, support Linux, audio and visual I/O, and multiple communication channels [4]. Although in an ideal world one will have the right devices for the job, in reality that is usually not the case. It is thus important to be able to make use of what is available. Users want to get the job done and so we expect to support a wide range of devices.

3.2 Networking, naming and location management

One's personal data should be easily and universally accessible. Having a multitude of digital devices, each with some possibly inconsistent set of data, is neither natural nor geared towards the needs of the human. Having to remember a set of arbitrary names just to use a physical device sitting in plain sight is also demeaning to the human user.

The self-certifying file system, SFS [14], is a universal, secure filesystem that has a single global namespace but no centralized control. Other similar filesystems require the users to use a particular key management system to provide security and authentication. SFS separates key management from file system security, thereby allowing the world to share the filesystem no matter how individuals chose to manage their keys.

Within a building it is useful to know where things, including one's self, are physically located. The traditional approach is to have all things periodically broadcast their identity and to have sensors spread throughout the building that detect these things. To provide a degree of privacy, among other reasons, the Cricket [7] location-support system takes the opposite approach. Spread throughout the building are a set of beacons. The beacons are a combination of RF and infrared signals that broadcast physical and logical location information. Things in the environment sense these beacons. Thus, a handheld knows where it is located rather than the system knowing it. A person has the freedom to reveal his or her location – usually when some service or resource

is required. All sorts of devices need to be integrated into the system having network connectivity and location awareness [12].

Knowing the location of things enables one to name digital devices by their intended use rather than by some specific URL or IP address. The Intensional Naming System [8] does just that by maintaining a database of currently active devices along with attributes describing their intended use. Devices periodically check-in to avoid having their entries time-out. INS supports early binding for efficiency and late binding for mobility. With INS, it is possible to route packets to the LCD projector that is located in the same room as the speaker or to route messages to whatever display device is near the intended recipient.

3.3 Security and correctness

As evident by the central place of this subsection, the Oxygen Project considers security and privacy as a central component of a human-centric system. We are developing a personal identification device that has two interesting features. It has a very simple interface, perhaps only a single button to distinguish between identification and authorization [12, 19]. The simpler the interface the easier it is to make the device secure. The second feature is that identification mechanisms provide privacy. A guiding philosophy is that privacy is the right to reveal information about oneself. When one chooses to make use of public system resources one is choosing to reveal information about one's self. Various schemes for secure, private group collaboration are being developed [19] as well.

As computers continue their infestation of human activities, their reliability becomes more important. Specifying the behaviors of interacting systems is particularly challenging. Research efforts, I/O Automaton (IOA) [16] and Term Rewriting Systems (TRS) [5], aimed at proving the appropriateness of collective behaviors, have focused on precise and concise specifications.

3.4 Human interfaces

There is no question that verbal and visual interfaces to computers are rapidly maturing and are already being successfully deployed. However, speech and natural language systems need to extend beyond simple dialog systems. The approach is to gather information from the speaker in a number of ways, to fuse this with information from other sources and to carry out tasks in an off-line fashion as well, so as to optimize the users time [22]. This effort is also trying to make it easy to develop

domain-specific speech interfaces; moving the creation of a interface from an art to a science.

The focus on visual input system that recognize a range of gestures tries to leverage multiple input devices. Just like stereo vision makes it easier to differentiate objects, collaboration between multiple monitoring devices simplifies many recognition tasks. For example, a microphone array along with a array of cameras can be used to do speech processing of all the conversations going on in a meeting, even when several people talk at once [9]

On the output side, there is research aimed at building very large displays. The challenge is to overcome the bandwidth problem of getting all the pixels out of a computer. The approach is to embed processing, connectivity, and display all in one package so that the pixel generator is close to the pixel display, thereby creating a sufficiently rich environment to mimic the Holodeck of Star Trek fame [18]. A related effort is to develop an economical method for displaying 3D using an auto-stereoscopic display [17].

3.5 Collaboration

There is much to be done in the way of supporting computer-mediated human collaboration. Teleconferencing has made strives in allowing collaboration between people who are widely spatially disjoint, but it is still difficult to collaborate when people are temporally disjoint [21]. Much of this work is going on in the Artificial Intellegence Laboratory at MIT and unfortunately, I only know a little bit about it. The seminar presentation scenario described in this paper is just the beginning.

4 Implementing the seminar presentation system

We can now relate the technologies described in the previous section with the needs of our seminar presentation system. The explanation roughly follows the description in Section 2. We ignore traditional issues like authorization and allocation of resources and application code written in traditional ways.

When Alice enters the seminar room, she must be identified and the presentation manager must be initiated. A simple tag broadcasts her public key to her H21 or, if she does not have one, then to the room computing E21 infrastructure. The H21, with Alice's permission, will initiate the seminar presentation manager application as an extension of Alice's computing environment. Rather than having applications run

on machines with only a loose connection to the owner, they are all under direct control of the initiator, who has the ability to interact with them from any Oxygen supported I/O device..

Access to Alice's presentation files is via the secure, global filesystem, SFS. Advanced indexing systems, such as Haystack [15], will be used to find files or slides within a file. The Cricket location management system is being used to know where the presentation is occurring. It is possible for the speaker and the audience to seamlessly move to a larger seminar room without losing any content. The intentional naming system, INS, is used to route packets between the components of the system and provide for fault tolerance and mobility. If one component crashes, INS will help in finding an alternate or reconnect when the component comes back online.

For the input modes, speech and vision processing is used. The speech project, Galaxy, has been developed mostly for dialogs and is being adapted to an active monitoring mode. The vision system [9] is used for the initial laser pointer and later for human gesture recognition. A microphone array combined with a vision system that precisely located the position of the speakers mouth is being developed to allow the speaker more mobility. Recognition of drawing gestures, makes use of technology underlying the Rational Capture project [10].

The presentation itself will be controlled in a conventional manner. For powerpoint presentations running under windows, we use visual basic to connect to the rest of the application middleware as well as to control the presentation itself. For the speaker's note view, a stripped down web-browser is used with the application code written in Java.

The output side, at the moment, is the least sophisticated. We hope to make use of the Auto-stereoscopic Display work that will enable 3D image rendering and the Holodeck research that will support very large active displays. In addition, capturing the experience for later review will make use of the research in collaboration [21].

Finally, the presentation manager application is written in a special "communication oriented" language and middleware, described in the next section. Such communication oriented languages, along with IOA and TRS research will lead to the development of correct distributed systems that work first time out, and allow one to focus on performance as the system scales.

5 The language overview

This section highlights the core of a communication oriented language used to program some Oxygen applications. The work described in this

section is preliminary and so the description is deliberately sketchy[2]. Although, Java could be used, especially since most of the underlying technologies provide a Java interface, our language lets the application writer to focus on what is relevant, permits very aggressive optimizations and is more concise and precise.

At a high level of abstraction, there are only a few components that need to be manipulated: nodes, edges, messages, and actions. Nodes are just about anything that can be named and communicated via sockets. An edge is a directed connection between nodes. A message is an entity that flows along an edge. An *event* is the creation or destruction of one of these components; thus there are only six different types of events. Rules (or actions) make up the final component of the language. An *action* consists of a trigger and a consequence. A trigger is an event, such as the creation of a node or the destruction of an edge. A consequence is also an event. For example, the existence of a message on an edge can trigger a set of edges to be disconnected.

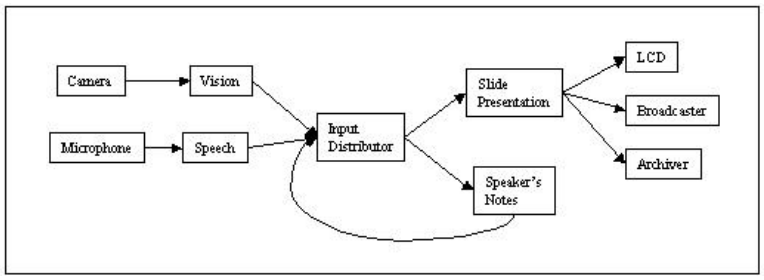

Fig. 3. A graphical view of the components and their connections

Nodes are named in a way that is compatible with the intentional naming system [8] and consists of a collection of key/value pairs. When a node is named, these pairs are matched against a database of existing, functioning devices or services. For simplicity, any node can be created or destroyed. In actuality, it is only the connection that is created or destroyed when the node is a physical device or an enduring software service. Connections are IP/Port specifiers; it is assumed that all devices and third-party services have some kind of wrapper that converts input and output to the appropriate formats. Rather than have a special case to handle the case when a named node does not exist, it is

[2] This work is so preliminary that the language has yet to be named.

assumed that such nodes are created and then immediately destroyed. The code to handle a non-existent node is exactly the same as the code to handle a node that was connected but becomes disconnected or destroyed.

Messages are self-describing and self-typed. They can be named, as with nodes, by a set of key/value pairs but must include a location, either a node or an edge. Very large messages or streams, such as audio, video, or screen images are conceptionally the same as text messages, but the implementation treats them different to ensure sufficient performance. As each message moves through the system, it is assumed to be created when it arrives at a location and destroyed when it leaves that location. This permits actions to treat message events and creation/destruction of node and edge events in the same uniform way.

All the action is, of course, with the actions. Actions are simple yet powerful rules. Actions can be created or destroyed, just like all other objects in the language, and are thus events. So, some event trigger can create new actions or remove current ones. Actions are needed to control what happens in a dynamic, sensor rich environment. When one enters a room from the hallway, two events happen: the link to the hallway is destroyed and the link to the room is created. Either of these events can serve as triggers for a whole slew of actions.

Figure 4 shows the specification of the presentation manager. The nodes are named using key/value pairs. The variable *owner* is a parameter of the system and is passed-in when the application begins. The location specifier could be done in the same way, but in the code in the Figure it is hardwired. Nodes, messages, and edges are all named to make it easier to read the code. Two sets of actions are specified. One disconnects the I/O devices. This, presumably is useful for situations in which the speaker wishes to temporarily pause the current presentation and to switch to a different one[3]. The first action is invoked whenever there is a "pause" message on the dialog-in port. The consequence of this action is to destroy the four edges that connect to the camera, microphone, LCD, and broadcast process. These will be used by the other application. The archiver is dedicated to this application and so can remain connected. A second set of actions show another example of disconnecting only the edges to the broadcaster and archiver nodes.

There is a middleware system that executes the language [20]. Initially it executes on a single machine, but soon will be made fault tolerant and decentralized. Nearly all actions performed by the mid-

[3] Hopefully, the speaker is not checking her mail during the presentation itself!

Application Name:
Seminar Presentation Manager (owner)

Nodes:
microphone : ["Device", "microphone", "Location","NE43-518"]
camera : ["Device", "camera", "Location","NE43-518"]
input : ["Process", "input-collector", "OS","Unix","Owner", owner]
ppt : ["Process", "powerpoint-displayer", "OS","Windows","Owner", owner]
ppt' : ["Process", "speaker-notes", "Platform","H21","Owner", owner]
lcd : ["Device", "lcd", "Location","NE43-518"]
broadcaster : ["Process", "broadcast-slides","OS","Unix","Owner", owner]
archiver : ["Process", "archive-slides", "OS","Unix","Owner", owner]

Edges:
m_{pause} : ["Message", "pause", "Location", dialog-in]
m_{resume} : ["Message", "resume", "Location", dialog-in]
$m_{confidential}$: ["Message", "confidential", "Location", dialog-in]
m_{public} : ["Message", "public", "Location", dialog-in]

Messages:
e_{ms}: (microphone , speech) , e_{si}: (speech , input)
e_{cv}: (camera , vision) , e_{vi}: (vision , input)

e_{ip}: (input , ppt) , $e_{ip'}$: (input , ppt')
e_{pl}: (ppt , lcd) , e_{pb}: (ppt , broadcaster)
e_{pa}: (ppt , archiver) ,

$e_{p'i}$: (ppt' , input)

Actions:
(m_{pause} , ($!e_{cv}$, $!e_{ms}$,$!e_{pl}$, $!e_{pb}$))
(m_{resume} , (e_{cv} , e_{ms} ,e_{pl} , e_{pb}))
($m_{confidential}$, ($!e_{pb}$, $!e_{pa}$))
(m_{public} , (e_{pb} , e_{pa}))

Fig. 4. Part of the communications program that expresses the connections. There are always two implicit nodes: dialog-in and dialog-out. The actions disconnect and reconnect the I/O devices on a pause or resume command. Presumably this is used to switch to another presentation. Similarly, the speaker may want to go "off-the-record" and show slides that are not archived nor broadcast. Going "public" reestablishes these links.

dleware corresponds to an event and events can be triggers for other events.

Although, we expect that the language will be compiled for optimum performance and reliability, it is also possible to interpret commands during run time. A user can modify an application during run-time to adapt to changing needs. The simple structure makes this easy to do provided there is a way for a user to easily name nodes, edges, and messages.

6 Conclusion

Scientific endeavors have always alternated between periods of deep and narrowly focused research activities and periods of synthesis across many fields. I believe we are in the midst of a new computer revolution. The relentless doubling of performance every 18 months, the even faster exponential growth of the web and its communication infrastructure, and the maturing of many human-computer interface technologies means that things will not stay the same. While industry is doing some of this work, the emphasis is on producing products that are good at one thing. Oxygen is not producing products, rather it is exploring what is possible when one synthesizes the fruits of research across many fields.

This paper describes a work in progress. Not only is this system still under development, but many of the technologies that it exploits are also under development. The presentation manager is simply a data point. It gives insight into the tools that will be needed in the future, gives feedback to those researchers developing the components, and is just one of several parallel efforts. These efforts will create the infrastructure necessary for the next decade. There is much to be done but we must keep the goal in sight – computers must become easier and more natural to use.

7 Acknowledgements

The author would like to thank John Ankorn for his help with all aspects of the integration effort as well as with this text. Thanks are also due to Hilla Rogel for her drawings and to Anant Agrawal for involving me in the Oxygen Project. This work is supported in part by our industrial partners,: Acer, Delta, Hewlett Packard, NTT, Nokia, and Philips, as well as by DARPA through the Office of Naval Research contract number N66001-99-2-891702.

References

1. Mit. http://www.oxygen.lcs.mit.edu.
2. Scientific american. August Issue, 1999.
3. A. Agarwal and S. Amarasinghe. RAW. http://www.cag.lcs.mit.edu/raw.
4. J. Ankorn. EMS. http://www.oxygen.lcs.mit.edu/ems.
5. Arvind and L. Rudolph. TRS. http://www.csg.lcs.mit.edu/synthesis.
6. K. Asanovic. SCALE. http://www.cag.lcs.mit.edu/scale.
7. H. Balakrishnan. Cricket. http://nms.lcs.mit.edu/projects/cricket.
8. H. Balakrishnan. INS. http://nms.lcs.mit.edu/projects/ins.
9. T. Darrell. Visual interface. http://www.ai.mit.edu/ trevor/vision-interface-projects.html.
10. R. Davis. Rational capture. http://www.ai.mit.edu/people/davis/davis.html.
11. M. Dertouzos. *The Unfinished Revolution*. Harper-Collins Publishers, New York, 2001.
12. S. Devadas. Aries. http://caa.lcs.mit.edu/caa/projects.html.
13. S. Garland and J. Guttag. Spectrumware. http://nms.lcs.mit.edu/projects/spectrumware.
14. F. Kaashoek. SFS. http://www.fs.net.
15. D. Karger. Haystack. http://haystack.lcs.mit.edu.
16. N. Lynch. Input output automata. http://theory.lcs.mit.edu/tds/i-o-automata.html.
17. L. McMillan. Auto-steroscopic display. http://www.graphics.lcs.mit.edu/ mcmillan.
18. L. McMillan. Image based rendering. http://www.graphics.lcs.mit.edu/ mcmillan/IBRwork/.
19. R. Rivest. CIS. http://theory.lcs.mit.edu/ cis/cis-projects.html.
20. L. Rudolph. Project oxygen. http://www.org.lcs.mit.edu.
21. H. Shrobe. Knowledge-based collaboration. http://kbcw.ai.mit.edu.
22. V. Zue. GALAXY. http://www.sls.lcs.mit.edu/sls.