# UbiWorld: An Environment Integrating Virtual Reality, Supercomputing, and Design

Terrence Disz, Michael E. Papka, and Rick Stevens
Mathematics and Computer Science Division
Argonne National Laboratory
Argonne, IL 60439
{disz,papka,stevens}@mcs.anl.gov

**Abstract**

*UbiWorld is a concept being developed by the Futures Laboratory group at Argonne National Laboratory that ties together the notion of* ubiquitous computing *(Ubicomp) with that of using virtual reality for rapid prototyping. The goal is to develop an environment where one can explore Ubicomp-type concepts without having to build real Ubicomp hardware. The basic notion is to extend object models in a virtual world by using distributed wide area heterogeneous computing technology to provide complex networking and processing capabilities to virtual reality objects.*

## 1 Introduction

In the Futures Laboratory [1] in the Mathematics and Computer Science (MCS) Division at Argonne National Laboratory (ANL), our research agenda is driven partly by discussions of advanced computing scenarios. We find that by suspending disbelief momentarily and by engaging in serious discussion of such topics as off-planet infrastructure, green nomadic computing, and molecular nanotechnology, we are able to project beyond the current set of problems and to conceptualize innovative solutions.

UbiWorld is a result of this fertile ground where concepts converge and evolve. This convergence is evident in the off-planet infrastructure problem, or People to Mars scenario. It's a safe assumption that support for people on Mars will be primarily computing, that the computing will be ubiquitous and nearly invisible, and that "green" technology will be used to minimize power consumption will be important, as will the deployment of nanotechnology to manufacture devices. On Mars, computers will always outnumber people. Computers will undoubtedly be heterogeneous, it being hard to imagine a single architecture deployed in devices from gloves and boots to landers, flight control decks, and mining machines. Computers will need to be transparently interconnected; reliability and fault tolerance will be critical; and programming and code maintenance will be significant activities. Everything of value will be available on mars.net from anywhere on the planet. Immersive telepresence will be a critical capability to overcome the obstacle of distance.

These requirements push the boundaries of computing and networking as we know them and as we can imagine them in the near future. Today, we don't even have the tools to experiment with implementations of some of these ideas. We can, however, conduct experiments in a virtual world, creating and designing objects out of "pure thought-stuff," to borrow a phrase from Frederick Brooks. This is the concept behind UbiWorld.

## 2 Ubiquitous Computing

In the beginning, there were mainframe computers. Access to mainframes has historically been characterized by many people per computer, batch operations, text input, and paper output. Today, we are living in the era of the personal computer. Personal computer use is characterized by one person per computer, multithreaded interactive use, multimedia, windows, and mouse interface. The next wave of computing will be ubiquitous computing, characterized by many computers per person and a transparent interface, used to amplify one's powers, not replace them. Ubiquitous computing means computers will become as invisible to us today as text is [2]. There was a time when the written word was the sole province of the experts, guarded and used sparingly, much as computing has been. Text technology has undergone a transformation from being written on clay tablets, then coarse paper, up to today's refined paper and display technology. Believers in ubiquitous computing see a day when the same transformation will occur with respect to computing; users will not be any more aware of the computers in their lives than we are aware today of

the text in which this document is written. We are already beginning to see this happen with the integration of computers in automobiles: the driver is really unaware of the computer and its function.

The ubiquitous computing philosophy originated at Xerox Parc in 1988 [3], pioneered by Mark Weiser. He conceived of Ubicomp as nonintrusive, mobile, flexible computing, highly integrated into the working and living environment. Ubicomp is not virtual reality (VR). VR techniques, which put people into artificial worlds, primarily pose a computing and graphics horsepower problem. Ubicomp forces the computer to live in the real world with people. It is the integration of human factors, computer science, engineering, and social science. The human factors issues go well beyond yet another human computer interface problem and will not be solved with another windowing system. The computer science issues span all areas—networking, operating systems, distribution of memory and processing, naming, resource management, etc. A general discussion of these issues can be found in [4]. Engineering will have to make advances in nanotechnology to manufacture the devices we can foresee. No one knows the social implications of everyone's having full-time computing and network access.

An example of an Ubicomp object is intelligent curtains that contain light and temperature sensors and control room lighting and temperature. As developers of UbiWorld, we talk about binoculars with image processing software to detect and highlight objects in a scene, to track eyes and vary the focus, or to perform other image transformations. We imagine scrap paper that will be aware of the identity of the user, will auto connect to the user's environment, and will automatically store and retrieve all notes made in a personal database accessed via contact with a table or desk. We discuss many other examples, limited only by our imagination.

Research in ubiquitous computing conducted at Xerox Parc followed standard experimental science protocols [5]. Devices were conceived and prototypes constructed and tried out on willing subjects. There were three prototypes of note: the Xerox ParcTab, a palm-sized device; the Pad, a notebook-sized device; and the Liveboard, a wall hanging device. Applications were constructed to perform e-mail, take notes, schedule meetings, check weather, etc. [6]. The primary lesson we take from the work, however, is that technology today is nowhere near what is required to design and perform experiments in truly ubiquitous computing. A discussion of the many compromises that Xerox had to make in the development of the ParcTab can be found in [6].

## 3 UbiWorld

UbiWorld is an experimental system combining virtual reality, advanced networking, and supercomputing to explore the implications of ubiquitous computing. We use a virtual reality system as a design and evaluation environment. Instead of actually building devices, we use VR techniques to model the representation of devices. We use advanced networking to link VR objects with computational servers to represent the behavior of the objects. Using these techniques, we can explore devices that are not yet possible to build.

Today's hardware capabilities fall short of what ubiquitous computing will need in terms of power consumption, miniaturization, network bandwidth, and computing power. Until these capabilities can be met, we feel that experimenting in virtual spaces is a productive method of exploring the concepts in ubiquitous computing. The UbiWorld project builds on existing software and projects in MCS. It serves to focus those efforts and leverages our long-standing expertise in software engineering and our strong development environment.

Starting with the CAVE$^{TM}$ family of display devices we integrate tools for the construction of 3D objects into the existing library [7]. Using these objects as models, we can then imbed new information technology within them. These products might be handheld computers, intelligent paper, image-processing binoculars, desks, clothing, jewelry, cups, eyeglasses, or carpeting. The plan is to couple the virtual objects with remote computers via fine-grained heterogeneous computing technology and to provide Ubicomp behavior and functionality to the models. The 3D objects will be placed in virtual rooms, thereby creating a shared virtual world (in a collection of CAVEs, ImmersaDesks, or whatever) where users can experiment with using the virtual devices.

Each object in the world has behavior controlled by a program running on the network. The behavior could be one that, in the real object, would be provided by a local computer or by a combination of local computer and network connection to remote processors or databases. These "behavior" processes are able to communicate with each other by using a shared protocol (UbiWorldcomm). The objects also react and are influenced directly by interactions with the virtual world and its users. If someone picks up an object in the UbiWorld, that object knows it has been picked up and then "does the right thing," which may mean communicating with other objects or computing

something or displaying some network stream. For example, one's coffee cup could be displaying live video, or one could talk to an earring and make a phone call.

UbiWorld will let us debug Ubicomp years before we have the technology to build it. It will enable us to change specifications without changing hardware and to identify software requirements. Through UbiWorld we can explore the boundaries of embedded computing and network computing. It serves as a testbed for heterogeneous computing tools and systems such as fine-grained networking protocols, image composition mechanisms, and agent integration.

# 4 UbiWorld Design

A fundamental principle in the design of UbiWorld is the separation of an object's representation from its behavior. As shown in Figure 1, the user interacts with the representation of an object. The behavior of the object is specified independently of the representation. The actual computation of the behavior is performed on a simulation server, separate from the representation computing. Furthermore, as seen in Figure 2, the world in which the object interacts is viewed as an orthogonal issue from the object itself. The virtual world, its representation, and its properties (such as light and gravity) are computed, stored, and rendered separately from the objects we choose to place in the world. In Figure 2, we show a virtual world record-and-playback engine that builds on work in progress at MCS. This engine, which we call VR Voyager, can be thought of as a virtual world server, storing virtual worlds, serving them to clients, recording VR experiences, and playing them back on demand.

In addition, we introduce the concept of "distributed rendering." This is an attempt to overcome the bottlenecks introduced into today's VR systems by lack of graphics rendering power. By employing rendering engines in distributed machines, we can bring much greater rendering power to bear than would otherwise be possible. Also, it gives us the ability to bring the rendering closer to the source of the data generated in the simulations. Distributed rendering is a new research area introduced into the Futures Lab by the UbiWorld project. We are studying ways to composite the separately rendered images into a single image in the VR theater, by tapping into the OpenGL pipeline, for instance. Along with distributed rendering, we also introduce the concept of "aware networking." Aware networking means that networking resources are not precisely known at all times and implies an adaptive nature imbedded in the objects as they seek to join networks. Objects can join networks by using a variety of bandwidths and protocols.

UbiWorld is a classic example of using the power of virtual reality systems to prototype devices that are impossible or too expensive to build. To accomplish our goals, we must construct tools that simplify the connection of physical representations to computer simulations that are prototyping the hardware. This then gives us the ability to construct and test the usefulness of hardware that is impossible to build with today's technology.

# 5 UbiWorld Development Environment

The UbiWorld development environment is a combination of hardware and software environments. The hardware is a combination of supercomputers, networks, and display devices. The software covers all major areas of software development from the low-level network code all the way up to high-level scripting languages that allow users to configure the environment.

## 5.1 Hardware

The following two subsections will address the display environments in which the UbiWorld system will be tested, and the various hardware limitations.

### 5.1.1 Display Environments

The current display environment is comprised of three virtual reality devices.

- **CAVE**

  The CAVE$^{TM}$ (CAVE Automatic Virtual Environment) is a 10 x 10 x 9 foot room that uses rear-projected high-resolution projectors to produce an immersive 3D environment (Figure 3). The CAVE environment, originally developed by the Electronic Visualization Laboratory (EVL) at the University of Illinois at Chicago, produces a 3D stereo effect by displaying in alternating succession the left and right eye views of the scene as rendered from the viewer's perspective [7]. These views are then seen by the user through a pair of LCD shutter glasses whose lenses open and close forty-eight times a second in synchronization with the left- and right-eye views. The correct viewer-centered projection is calculated based on the viewer's position and orientation as determined by a electromagnetic tracking system. The position and orientation of a 3D wand are also tracked; this wand allows for navigation of and input into the virtual world. Along with the visual

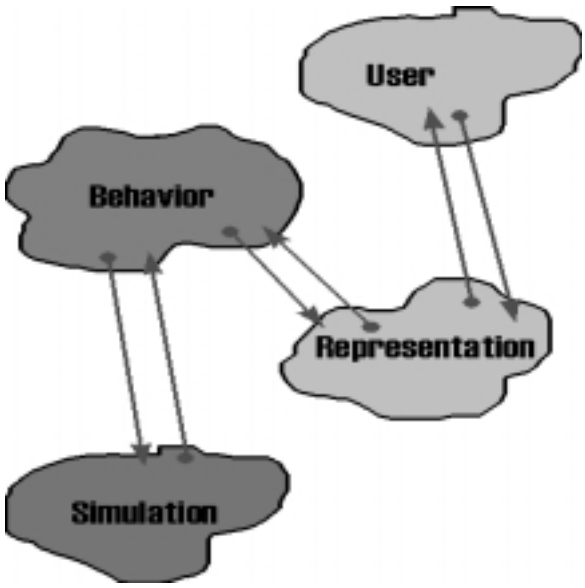


Figure 1: The Connections of Behavior, Representation, Simulation, and the User within the UbiWorld model.

feedback of the CAVE environment, a complete 3D audio environment is available to the user.

- **ImmersaDesk**

  The ImmersaDesk is based on the same rear-projection technology as the CAVE (Figure 4). It is a fully interactive, 3D immersive environment that is about the size of a large drafting table. The ImmersaDesk allows for one tracked viewer, along with two to three passive viewers.

- **InfinityWall**

  The InfinityWall is a large rear-projected system that is created from compositing four standard 1280 x 1024 screens together to create one large high-resolution screen. The InfinityWall can be used as a large ImmersaDesk, where the images are projected in stereo and the viewer is tracked, or can substitute for a large high-resolution workstation. The NII/Wall was developed by EVL, the National Center for Supercomputing Applications, and the University of Minnesota, with support from Silicon Graphics, Inc.

#### 5.1.2 Hardware Limitations

Although this development environment is satisfactory for early experiments, we believe that several improvements will need to be made in virtual technologies to fully realize the benefits of the UbiWorld project.

- **Resolution**

  The current resolution of the CAVE is 1280 x 768 on each wall. If we were to attempt a resolution close to the capabilities of the human eye, we would need a resolution of 4,800 x 3,800 [8]. That resolution is not available at this time, but we believe we can achieve that resolution on selected areas of the screen by using a "high-resolution window." By using a separate projector and rendering engine and by driving the location of the projector based on gaze direction, we can provide an area of high resolution at the place where the user is looking. This high-resolution window is another of the research projects in the Futures Lab that is motivated by the UbiWorld project.

- **Tracking**

  The resolution of the tracking system is another weakness of the current environment. Today, our effective sampling rate is approximately 100 ms, with a spatial resolution of approximately 1 inch. In the future, for fine-grained manipulation of objects, we expect a sampling rate closer to 1 ms and a spatial resolution of 1 mm.

- **Haptics**

  Today, our CAVE environment has no haptic devices. The UbiWorld project requires that we bring these devices into the CAVE and learn to register their actions with virtual representations.

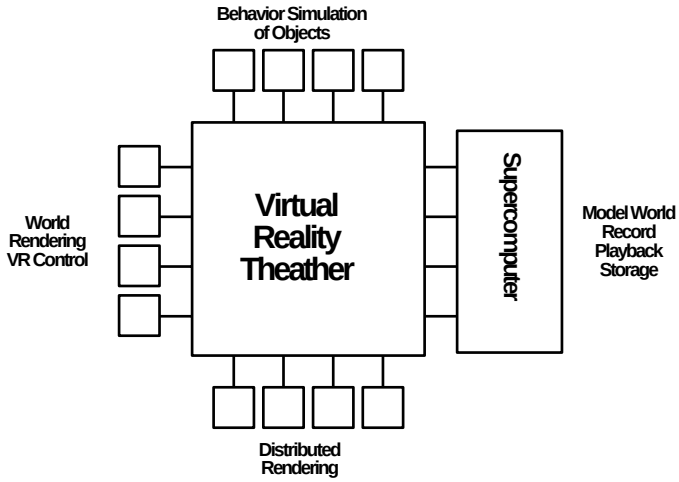


Figure 2: Separation of UbiWorld Spaces

Force and feedback will be required to fully prototype actions of devices in our virtual world.

- **Control Interfaces**

  Software in the CAVE today is programmed as an extension of current windows-based systems. We use menus, visual displays of pick lists, radio buttons, dials, etc. We believe these interfaces are wholly inadequate for use in UbiWorld. Devices and objects represented in UbiWorld may be activated by voice, by absolute position or proximity to other objects, by proximity to or by sensing features in the virtual world, or by any other means that we haven't thought of yet. We need a new paradigm to allow freedom and innovation in control interfaces. This is yet another active research project spawned by requirements of the UbiWorld project.

## 5.2 Software

Although more tools are required, we have a good group of software already available to us for use in the UbiWorld project.

### 5.2.1 CAVElib

The CAVE library [7], developed at EVL to work with the CAVE family of display devices, provides basic VR functionality and viewer-centered perspective transforms automatically. This frees the VR programmer to focus on the graphics of the problem at hand, not the viewer perspective problem. The CAVE library provides basic navigation functions, tracking of the user and wand, and interaction with the wand buttons and joystick.

### 5.2.2 CAVEcomm

The CAVEcomm library is a communications library that aids developers of virtual reality applications in the area of remote communications [9, 10]. The remote communications can be either virtual reality device to virtual reality device or virtual reality device to supercomputer. Using the CAVEcomm library, users register their virtual reality applications and/or supercomputing simulations with a broker. The broker process handles the connections of separate entities. The broker manages resources and connections but does not handle data traffic. Once the broker has set up the connection between the entities, they send the actual data traffic only between each other. CAVEcomm is specifically designed to work with the CAVE group of virtual reality devices, namely, the CAVE, the ImmersaDesk, the CAVE simulator, and the InfinityWall. The ideas of CAVEcomm can be extended, however, to any virtual reality-based system.

CAVEcomm has been used to connect virtual reality applications to supercomputing simualtions that are running in real time. It has been used to connect two CAVEs that are geographically separated, allowing the users to collaborate on a joint task or to demonstrate something in one CAVE to users in another.

### 5.2.3 CAVEav

The CAVEav library brings multimedia capabilities to the CAVE. Using the library, programmers can connect to video sources on the network and texture map the resulting video stream onto objects in the CAVE. Live video streams from other CAVE, for instance, can

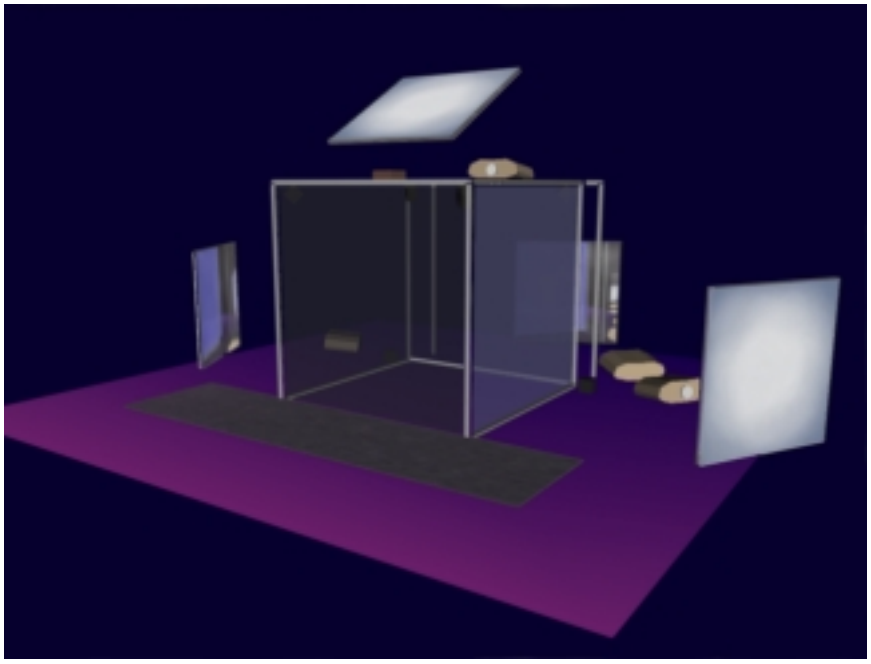

Figure 3: CAVE Virtual Environment (Milana Huang, EVL, 1994)

be texture mapped onto avatars showing remote users. Live streams from robots or instruments are used to provide an immersive telepresence capability. Prerecorded streams can be used to provide instruction or backgrounds.

A prototype system demonstrating these features has been developed at ANL. A small robot mounted with a variety of video and audio components is connected to the CAVE; from within the CAVE, a user can navigate the robot and interact with the environment within which the robot lives [11]. This system is being used to test the requirements and to expose the difficult problems within a toolkit of this nature.

#### 5.2.4 CAVE-VRML Modeler

Three-dimensional virtual computer environments such as the CAVE should have the capacity to be a working development environment, not just an interactive display environment. One aspect of a development environment is 3D modeling. Currently, no 3D modelers work well in conjunction with the CAVE. Often, in going from the modeler to the CAVE, "what you see is what you get" is not always true. Frequently, objects modeled on a workstation look quite different in the CAVE, particulary with respect to the object's scale, color, and lighting. One of the new projects in the Futures Lab is the development of an interactive modeling system to be used in the CAVE. This system will allow users to create objects in the native environment and will import/export VRML-based objects that can be used in or taken from other VR environments. Users will be able to affect the transformations (rotate, scale, translate) of the object as well as its material properties (ambience, diffusivity, shininess, specularity, etc.). The ability to edit materials and lighting is significant given the fact that many objects look very different in the CAVE due to the physical components of the CAVE, (i.e., projectors and screens). This CAVE modeling tool will also have the ability to edit the object's shape, not just its extraneous properties. Users will be able to edit the polygons of the object; adding, deleting, and moving vertices will give users the ability to redefine and combine existing shapes or create new objects from scratch. The created worlds and environments will be exportable to VRML.

#### 5.2.5 VR Voyager

The Voyager multimedia recording and playback system has been under development in the Futures Lab for the past two years [12]. It uses an IBM SP2 for multistream, multimedia record and playback of network-based sources. We propose to use the Voyager system as the basis for a new virtual world server,

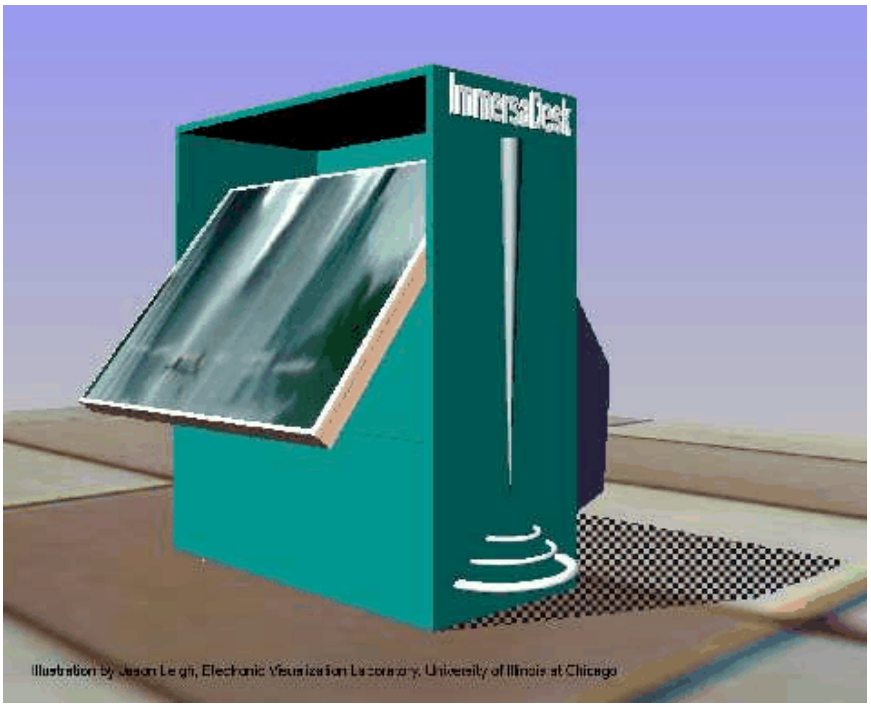


Figure 4: ImmersaDesk Virtual Environment (Jason Leigh, EVL, 1995)

providing record and playback of VR experiences.

## 6 UbiWorld Goals and Requirements

Our goal is to test UbiWorld objects and worlds in a task-based manner, under different scenarios. We will provide several different environments, for example, a home, office, airplane, hotel, car, field, restaurant, laboratory, and shop. Each of these environments must be rendered with high-resolution textures, complex spaces and lighting, and varying degress of scene complexity. Each of these models is different, but the requirements of ubiquitous computing span all of these environments. In each different environment, Ubicomp objects must behave appropriately. Scenario spaces will be used for experimenting with device functionality and interaction by requiring task-based explorations. For instance, users may be required to order a meal, prepare a talk, negotiate a contract, drive to an unknown destination, or buy groceries.

### 6.1 Virtual Device Requirements

To accomplish this goal, we believe that computing objects in the UbiWorld must possess or make use of at least the following features.

#### 6.1.1 Innovative Representational Design

Current thinking, as evidenced by the ParcTab, is too restricted by technology limitations to create truly innovative design. We would like to be able to take the best from advanced industrial design and apply it to the design of future Ubicomp devices in UbiWorld. The type of advanced design philosophy we have in mind is embodied in publications such as *Arbitare* magazine and the book *The Art Factory, Design in Italy Towards the Third Millenium* [13]. In this latter text, the author analyzes the birth of virtual industry, links between the fashion system and the media system, the rediscovery of art and craft traditions, and renewed ecological awareness of materials in terms of contemporary and New Wave design.

#### 6.1.2 Novel Information Technology Components

Using supercomputer and external multimedia servers and resources, designers in the UbiWorld will be able to specify and simulate behavior associated with advanced CPU capabilities, imaging technology, sensors, actuators, multimedia components, communications capabilities, etc.

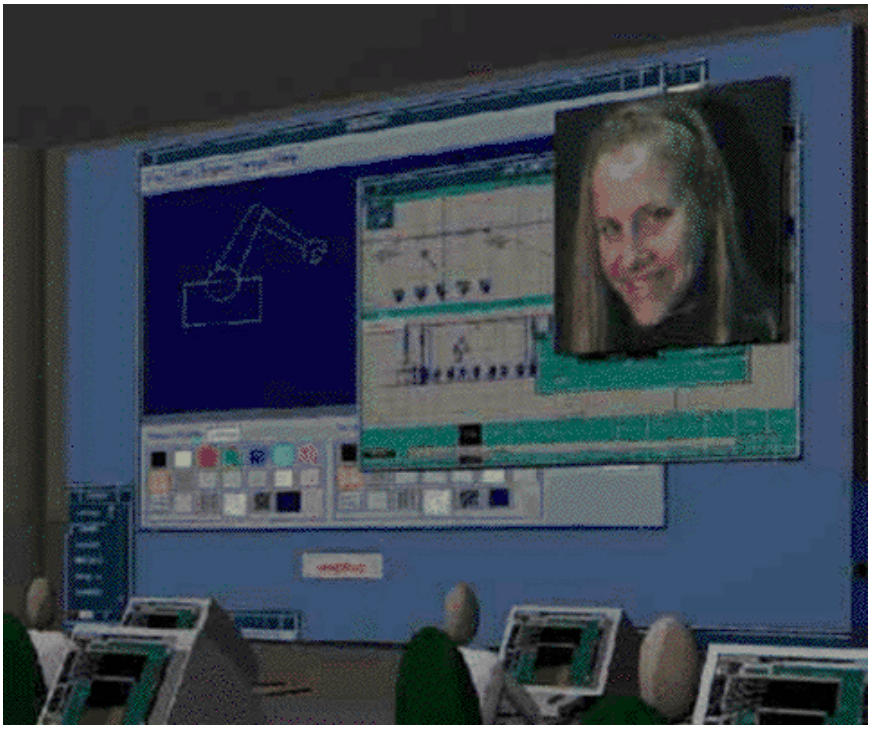

Figure 5: InfinityWall Virtual Environment (Jason Leigh, EVL, 1995)

#### 6.1.3 Transparent Networking

Transparent, or "aware," networking is assumed to be a fundamental capability in the UbiWorld. The network in this case is transparent to the user. UbiWorld devices automatically connect to whatever other device is appropriate, using mutually acceptable bandwidths and protocols.

#### 6.1.4 Device/Space Awareness

Devices in the UbiWorld must exhibit awareness of other devices and of the space in which they are operating. If a user carries a device to a different space, the device must automatically be aware of the new space context and take appropriate action.

#### 6.1.5 Reactive/Proactive/Proxy Behavior

UbiWorld devices should, of course, react properly to users requests, but beyond that, they should be proactive as necessary. An example is the loading of a new context when the user enters a new space or the proactive downloading of news or information known to be of interest to the user. Based on current research within the artificial intelligence community on agent based systems, one can already see a trend emerging. Agents are being constructed that passively view the behaviors of users and learn about their interests. Using methods similiar to these techniques, we hope to construct within the UbiWorld a framework that allows the Ubicomp devices to be reactive and proactive.

#### 6.1.6 Integration Functions

In the UbiWorld, we believe the network really is the computer. As devices near other devices or objects in the space, they should be able to interrogate the devices and perform acts of spontaneous integration if it is of benefit to do so. Benefit in this case could be defined as access to greater bandwidth, computing power or advanced capabilities. The intelligent scrap paper idea fully embodies this idea, since the scrap paper integrates with desktops for greater bandwidth or with imaging devices to capture multimedia information.

### 6.2 UbiWorld Design Problems

We see four critical design problems in the UbiWorld project:

- Object shape, form, and representation
- Computing and communications internals
- Functional behavior

- Integration with and awareness of environment

In the UbiWorld project, we believe it is important that these design problems be separated. We want to feel free to experiment with the form and shape of an object independent of its other attributes. Since the object is virtual, we can experiment with communications and computing internals without being encumbered by physical packaging or power problems. Functional behavior is simulated via the computational servers and can be varied at will, independent of the packaging or other factors. Finally, integration with and awareness of the environment are accomplished via simulated sensors and connection interfaces. For instance, when an object such as a piece of intelligent paper is placed on a table, we expect it to perform the above mentioned "spontaneous integration" with the table. Through its sensors, it must become aware of the table and its capabilities. By using an aware networking approach, the intelligent paper negotiates with the table to establish a connection, thereby integrating its functionality with that of the table.

For each of these design problems, appropriate tools are essential. Where existing tools are inadequate, it will be necessary to build or invent more robust tools.

### 6.3 Technical Challenges

The scope of the UbiWorld project pushes the bounds of current VR and networking technology and gives rise to a host of technical challenges. The following list is not meant to be comprehensive but serves to enumerate those problems we feel are most important to focus on now in the implementation of the UbiWorld.

- **Scalability**

  Adding objects into a UbiWorld environment stresses protocols, bandwidths, computing, and rendering power. Important research issues regarding scalability of these components and their interactions must be investigated.

- **Latency**

  We know that there are limits on the user-perceived latencies in an interactive system. The latency of the total system (including latency from the graphics system, the tracking systems, the networks and the computation engines) cannot exceed 100 ms - 1000 ms, depending on the user's experience level [14]. Taylor et al. studied this phenomenon in multiple-tracked, network-connected VR systems and offered data and research issues to be investigated to mitigate the latencies in the system [15].

- **Object Representation**

  The representation of objects within the UbiWorld model must be flexible, so that they can be changed easily, without affecting the underlying simulation of the objects. It is important to allow for a variety of different representations to be attached to the simulation, to allow users to experiment with different interfaces. This component is one of the true strengths of the connection of Ubicomp devices to VR, since expensive prototypes do not need to be built to try out a new device.

- **Behavior Specification**

  An open question is how best to specify the behavior of an object. For our purposes, there are three categories of behavior:

  - Representation Dynamics
  - Functional Mechanics
  - Computational and Communication

  It is not likely that the same tools will be appropriate for each of these tasks. A suite of tools will need to be developed to enable specification of each of these behaviors and their interactions.

- **Object Binding and Brokering**

  Resource brokering is the use of computationally enhanced entities to aid in the requesting, allocation, and management of remote capabilities. Much in the same way that data hiding is used in object-oriented programming to make the interfaces easier to use and understand, wide-area resource brokering can be used to simplify the user interaction with a large-complex set of computational and collaborative resources. One approach can be attacked by expanding on what we have learned from traditional system management software and current work in cluster management. Through the use of resource brokering, the level of complexity needed to configure and control a large virtual world is reduced to a manageable state for the user.

- **Process Mapping and Execution Control**

  The management and control of processes become important in the UbiWorld model. The ability to map a new process into an existing computational framework is an essential component of the UbiWorld model. As a new user enters the UbiWorld or a new Ubicomp device is introduced, the process controlling the simulation will have to be seamlessly integrated. At the computational level that will require mapping the new process onto a computational resource and then controling the execution from startup to termination.

- **Evaluation and Measurement**

  As in any scientific endeavor, we desire to measure, evaluate, and report on our work in a rigorous manner. Tools are required to enable instrumentation of all the computational and communications processes and their relationships. Evaluation and reporting tools are essential to reduce and analyze the data generated by instrumented codes.

- **Distribution**

  Distribution covers a whole set of problems related to using networked resources: the mapping of processes to processors; the distribution of databases over networked resources; the issues of redundancy, failure recovery, and the other problems usually associated with distributed computing — all are present in the UbiWorld project.

- **Naming and Identification**

  The current Internet provides mechanisms for naming computers and Web pages. Future environments will require the ability to name a much wider variety of objects with varying degrees of persistence and scope [16]. Mechanisms are also required for locating objects based on different criteria. Proposals for Universal Resource Names are a step in this direction but are designed for long-lived objects. We believe that a new class of naming mechanism will be needed that can refer to much shorter-lived objects that would be the topic of communication between user agents and simulation spaces. Brokering and name translations mechanisms are also needed. These will need to be high performance and scalable and to incorporate hooks for security and access rights.

- **Security**

  Fine-grained, scalable authentication, authorization, and accounting mechanisms will be required to control access to information and computational resources by both users and computational entities. Complex issues include secure communication of high-bandwidth multimedia data, access control of dynamically created entities, multi-entity interactions, security of archived data, object-level security in virtual environments, and delegation of authority between users and associated computational entities.

# 7 Tools

The underlying infrastructure for the construction of UbiWorld requires the combination of a wide area of computer science disciplines. Techniques and tools need to be borrowed from the fields of computer graphics, artificial intelligence, and systems to name but a few.

We have identified a set of existing tools for use in implementing the UbiWorld concept. These tools will not be sufficient in the long term, but most represent a suitable beginning for work on the UbiWorld requirements.

## 7.1 Representation

The physical representation of the objects within UbiWorld, such as intelligent paper or image-processing glasses, needs to modeled in such a way that the objects can be easily changed and modified. Currently one can use a wide variety of desktop modeling and CAD packages to physically design the objects. These packages are not sufficient to develop UbiWorld objects to final state. While it is important to separate the representation from the function, the modelers require the ability to provide hooks or connections to a behavior toolkit. We believe that the Open Inventor$^{TM}$ and VMRL modeling formats lend themselves most to this goal.

## 7.2 Behavior Specification

For the representational dynamics of an object, the tools such as VRML or OpenInventor come to mind. For the functional (or mechanical) behavior, procedural systems such as Java or C++ are available today, but they are too low level to use in the long term. For the specification of the computational and communications behavior, we can also use Java or C++, or other still too low-level systems such as nperl or Nexus.

### 7.3 Object Binding/Brokering

Binding of objects to resources can currently be accomplished by hard-wiring the connection or soft-wiring the connection through the use of a brokering system such as CORBA or the LabSpace broker.

### 7.4 Process Mapping and Execution Control

The design of UbiWorld calls for object behavior to be computed on separate computational servers. The mapping of many object processes to computational servers and control of the execution is a problem we have been examining for some time. For SC'95, a team at ANL developed custom software for scheduling and mapping processes to network-based processors. The system deployed at SC'95 was the I-WAY Point of Presence (I-POP) machine. The I-POP machine was specifically set up to manage security issues and was configured specifically for the I-WAY [17]. It allowed use of process mapping and control software such as Nexus and MPI. Subsequent to the I-WAY project, a team at ANL has been integrating the Adaptive Communication Environment (ACE) [18] system into a parallel message-passing toolkit.

Each of these solutions has shortcomings, and we fully expect that UbiWorld requirements will force the design or evolution of a new, more sophisticated system for process mapping and control.

### 7.5 Evaluation and Measurement

It is important to be able to measure and quantify the progress that is being made. Since UbiWorld requires the combination of such a wide variety of different aspects of the field of computing, we are required to connect various areas that are only now beginning to be tested. The connection of supercomputers to immersive virtual reality display devices is just one area. For the first time, issues such as latency, for example, need to be looked at outside their normal meanings to supercomputer users and to graphics programmers. Therefore, as UbiWorld is constructed, measurements and evaluation need to be done. Currently, we are using PABLO from the University of Illinois for instrumenting VR and simulation programs. We use MPI logging and the Upshot display system for tracing and evaluating MPI programs.

## 8 Conclusion

UbiWorld is a project that pushes beyond the bounds of current technology and forces us to think of heterogeneous computing in new terms. Rather than making incremental changes in existing technology, UbiWorld gives us a chance to leapfrog into a new problem space where heterogeneous computing is the norm, not the exception. In this space, we can more clearly see the technical challenges that await us, and we can proceed to invent solutions well ahead of technological progress that can implement these solutions.

The scope of UbiWorld is very broad and invites research on many fronts. We have identified some of these issues, such as network latency, scalability, object representation and specification, process and data mapping, object brokering and binding, security, and measurement. Each of these issues is deserving of a focused research effort in the futures-oriented ubiquitous computing scenario, and we invite the heterogeneous computing community to engage in discussions and research in this rich problem space.

## Acknowledgments

We thank Ian Foster, Ivan Judson, Bill Nickless, Bob Olson, and Matt Szymanski for all their help and input. This work was supported by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Computational and Technology Research, U.S. Department of Energy, under Contract W-31-109-Eng-38.

## References

[1] Terrence L. Disz, Remy Evard, Mark W. Henderson, William Nickless, Robert Olson, Michael E. Papka, and Rick Stevens, "Designing the future of collaborative science: Argonne's futures laboratory," *IEEE Parallel and Distributed Technology Systems and Applications*, vol. 3, no. 2, pp. 14–21, Summer 1995.

[2] Mark Weiser, "The computer for the 21st century," *Scientific American*, vol. 265, no. 3, pp. 94–104, September 1991.

[3] Mark Weiser, "The world is not a desktop," *Interactions*, vol. 1, no. 1, pp. 7–8, January 1994.

[4] Ian Foster, Michael E. Papka, and Rick Stevens, "Tools for distributed collaborative environments: A research agenda," in *Proceedings of the Fifth IEEE International Symposium on High Performance Distributed Compututing*. IEEE, 1996, pp. 23–29, IEEE Computer Society Press.

[5] Mark Weiser, "Some computer science issues in ubiquitous computing," *Communications of the ACM*, vol. 36, no. 7, pp. 74–83, July 1993.

[6] Roy Want, Bill N. Schilit, Norman I. Adams, Rich Gold, Karin Petersen, David Goldberg, John R. Ellis, and Mark Weiser, "The parctab ubiquitous computing experiment," Tech. Rep. CSL-95-1,

Xerox PARC, Xerox Palo Alto Research Center, March 1995.

[7] C. Cruz-Neira, D. J. Sandin, and T. A. DeFanti, "Surround-screen projection-based virtual reality: The design and implementation of the CAVE," in *Computer Graphics (Proceedings of SIGGRAPH '93)*. ACM SIGGRAPH, August 1993, pp. 135–142, Addison Wesley.

[8] M. McKenna and David Zeltzer, "Three dimensional visual display systems for virtual environments," *Presence*, vol. 1, no. 4, pp. 421–458, 1992.

[9] T. L. Disz, M. E. Papka, M. Pellegrino, and R. Stevens, "Sharing visualization experiences among remote virtual environments," in *International Workshop on High Performance Computing for Computer Graphics and Visualization*. 1995, pp. 217–237, Springer-Verlag.

[10] Terrence L. Disz, Michael E. Papka, Michael Pellegrino, and Matthew Szymanski, *CAVEcomm Users Manual v1.0*, Mathematics and Computer Science Division, Argonne National Laboratory, 1.0 edition, August 1996, ANL/MCS-TM-218.

[11] Ivan Judson and Rick Stevens, "Architecture of an immersive virtual reality telepresence system," Private Communication, Argonne National Laboratory, April 1997.

[12] Rick Stevens, Terrence Disz, Robert Olson, Michael E. Papka, and Remy Evard, "Voyager: A next generation hypermedia server to support the construction of virtual organizastions," 1995, Grant Proposal, Submitted to DOE 1995.

[13] Stefano Casciani, *The Art Factory Disgn in Italy Towards the Third Millennium*, Springer-Verlag, 1996.

[14] A. Liu, G. Tharp, L. French, S. Lai, and L. Stark, "Some of what one needs to know about using head-monunted displays to imporve teleoperator performance," *IEEE Transactions on robotics and Automation*, vol. 9, no. 5, pp. 638–648, 1993.

[15] Valerie E. Taylor, Milana Huang, Thomas Canfield, Rick Stevens, Daniel Reed, and Stephen Lamm, "Performance modeling of interactive, immersive virtual envrionments for finite element simulations," *The International Journal of Supercomputing Applications and High Performance Computing*, vol. 10, no. 2/3, pp. 141–151, November 1996.

[16] John Kunze, "Functional recommendations for Internet resource locators," Febuary 1995, Network Working Group, RFC 1738.

[17] Ian Foster, Jonathan Geisler, William Nickless, Warren Smith, and Steve Tuecke, "Software infrastructure for the i-way high-performance distributed computing experiment," in *5th IEEE Symposium on High Performance Distributed Computing*. IEEE, 1996.

[18] Douglas C. Schmidt, "The adaptive communication environment an object-oriented network programming toolkit for developing communication software," in *Sun Users Group Conference*, December 1993.