Immersive redlining and annotation of 3D design models on the Web

Thomas Jung[†], Ellen Yi-Luen Do, and Mark D. Gross
Sundance Lab for Computing in Design & Planning, University of Colorado, Boulder, CO, USA [†] & C.R.A.I. Centre de Recherche en Architecture et Ingenierie, Nancy. France

Key words: VRML, immersive environment, virtual annotation, computer-aided design, building models

Abstract: The Web now enables people in different places to view three-dimensional models of buildings and places in a collaborative design discussion. Already design firms with offices around the world are exploiting this capability. In a typical application, design drawings and models are posted by one party for review by others, and a dialogue is carried out either synchronously using on line streamed video and audio, or asynchronously using email, chat room, and bulletin board software. However, most of these systems do not allow designers to embed annotations and proposed design changes in the three-dimensional design model under discussion. We present a working prototype of a system that has these capabilities and describe the configuration of Web technologies we used to construct it.

1. INTRODUCTION AND MOTIVATION

We are constructing a collaborative three-dimensional environment for design review, comment, and exploration of design alternatives. Our system offers two distinct mechanisms for collaborative review. In both mechanisms, a designer posts a model in VRML format for on line review. Reviewers can walk through and inspect the model using a standard Web browser that has VRML capability. In the first mechanism a reviewer annotates the design by using a Java applet to drop text notes and simple annotation symbols (spheres, arrows, call-out dots with identifiers) into the model. The annotations are objects in the VRML world, although they are

stored and loaded as separate files from the base world that is being discussed. In the second mechanism, a reviewer explores, selects, and make changes to the objects in the model by interacting with a Java controlled VRML browser. The reviewer can change the location, colour and texture of the design objects (e.g., furniture, window, walls) in the VRML world. As with the first mechanism, the new objects with changes are stored in a separate file from the base model, and are loaded in to the shared VRML world by other reviewers.

The paper outlines the motivations for this research; it reviews related work on 3D interaction in Virtual Reality both in laboratory settings and over the Web, design rationale, and on-line collaborative design; and describes our working system for interactive immersive redlining and annotation of VRML models.

We are interested in building, using and studying systems that support a group of people participating in a design process. Collaborative design means not only the collaborations among different designers, but also between designers and their clients. Often a design is carried out as a conversation and negotiation process among architects and their clients. The process is not merely acceptance and rejection of presented alternatives, but a collaborative negotiation of individual decisions about dimensions, positions, and selection of components.

We believe a virtual environment that provides designers and clients the opportunity to negotiate through annotations and changes to the design alternatives inside the design artefact will be useful. This environment could serve as a record of what people agreed to do, their design and use rationale, and what actions they have agreed to perform. An environment that enhances this communication and encourages co-operation, overcoming the barriers of space and time constraints will be valued. For example, when people collaborate in daily life, from commissioning a design project, to several iterations of the review and redesign process, scheduling is usually the hardest part besides communication. A meeting usually requires the architects, clients, and other concerned parties to be present in the same location at the same time. This is not always convenient or even possible. Furthermore, clients often have difficulty understanding design presentations of floor plans and cut sections. They may easily be seduced or confused by graphic representations of the design (e.g., water colour painting, airbrush rendering, pencil sketches) and fail to understand the basic elements of the design. Therefore, a simulated environment of the design presented in a format that allows people to access, review and even change the design from a distance can help transcend the boundaries of time and space.

A shared world of virtual reality offers a more sophisticated, interactive, stimulating and engaging environment for people—architects and clients alike—to communicate. Therefore, we built a working prototype that facilitates this kind of communication and collaboration in a VRML environment. We call it the "Immersive Redlining and Annotation" project, or Redliner for short.

In the following, section 2 describes how architects and their clients (or domain experts and consultants) might use our Redliner environment to discuss and interact about the decisions of a house design. Section 3 follows with a description of the implementation of our working prototype. Section 4 reviews related work in the

domain of VR and CSCW. Finally, section 5 concludes with discussion and future work directions.

2. SCENARIO

We present here a scenario showing how architects and clients can work and communicate in a shared virtual environment by leaving annotated objects and making changes in the VRML world of an architectural design.

Our story has three key players: the architect Bill who is hired by a couple, John and Mary, to design their new house. Bill works at his architecture firm in California, John and Mary live in Pennsylvania, and their new house is to be built in Georgia. John's profession as an executive in a successful Internet startup requires him to travel frequently around the country. All three have easy access to the World Wide Web (in the office, community library, or Internet cafes).

Our architect Bill, after initial discussions with his clients and the examination of the site proposed an initial design for the residence. He posted a Web address of a model of his newly designed house made in a 3D VRML format using the Redliner program. He calls and sends emails to John and Mary to inform them of his progress on the project and invites them to explore and review the design.

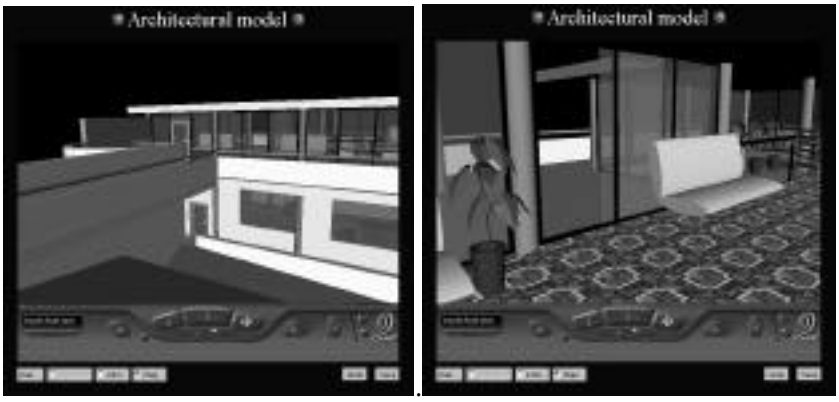

Figure 1. Mary parks her car, views the exterior of the building (left) and walks inside, finding herself in an entrance area with a tile floor (right). Note the annotation buttons for John, Mary, Architect, Undo, and Save.

Mary is the first to visit the virtual house on the site. She rolls up on the set in her virtual Mercedes (well, actually, the Cosmoplayer VRML browser plug-in for Netscape Navigator or Internet Explorer on PC or Mac). She sees the building, as it would appear from the driveway. The door opens as she approaches and the lights come up, bathing the interior of the entrance area with a rosy glow.

To her left, Bill (the architect) has made a place for the Possible Mondrian Painting they inherited from her parents, and he has illuminated it from above by a

lamp that turns on when she looks at the painting on the wall (Figure 2, left). Looking around, she hears a waltz by Chopin playing from the piano downstairs (Figure 2, right). She feels that the space is a little dark, though, and she begins to think about what changes she would like to see.

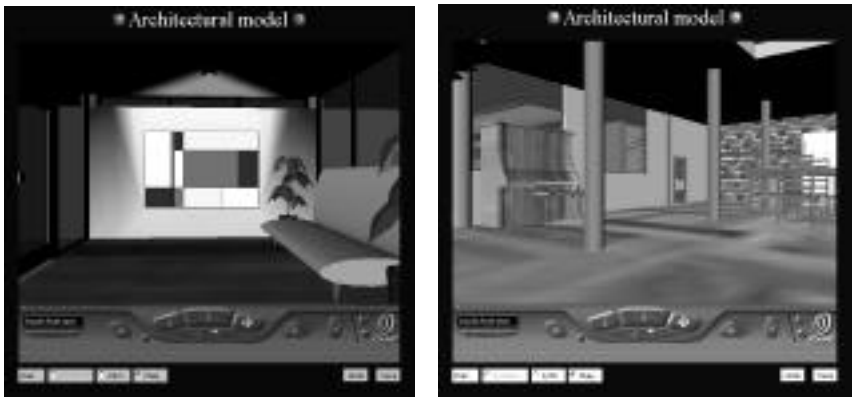

Figure 2. Mary looks around the entrance area. The light comes up over a painting (left) and music comes from the piano (right).

Mary decides to leave a note for Bill (Figure 3). She drops a green annotation marker (a small sphere, the VRML equivalent to a PostIt note) on the wall. A new green marker (numbered '1') sticks to the wall, and a text annotation window appears. She types, "The wall behind the piano looks too dark". She closes the annotation window and moves further into the house.

Figure 3. Mary drops a green annotation marker on the wall near the column (left) and types in a comment (right).

Later that day, John logs on from his hotel room in Seattle. He makes comments using red markers. Figure 4 shows comment #1, in which he asks about what appears to be a cut off column in the middle of the room.

Figure 4. John drops a note (#1) on the cut-off column asking whether it is part of the stairs.

After both John and Mary have visited the virtual design, Bill returns to the world to review their comments (Figure 5, left). He responds to each of their comments (Figure 5, right).

Figure 5. Left, Mary has left annotation markers inquiring about the wall materials and furniture arrangement; right, Bill responds to Mary's questions.

In some cases, he responds by creating several alternatives, for example, enabling Mary and John to view different colours and materials for the walls (Figure 6). He replaces their red and green annotation markers with new, yellow ones that link to a page of his design revisions in response to their comments.

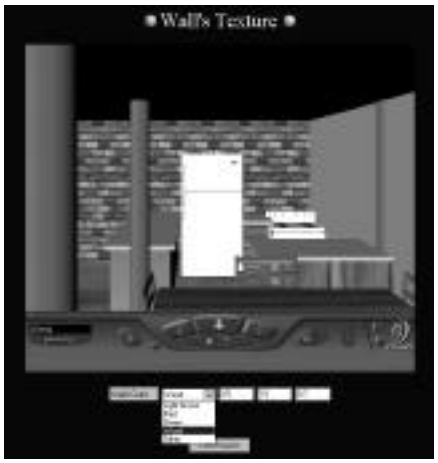

Figure 6. Bill has provided an 'options window' to try out different wall materials. Note the colour and materials choice menus and the colour co-ordinate menu on the bottom, as well as the comments button.

John returns the next day to review the revised design. He sees the design as it was before, with yellow markers in the locations where his and Mary's red and green spheres were before. When he touches one of the yellow markers, his browser pops up a window showing Bill's responses to their comments (Figure 5 right). This HTML page also links to a custom configured VRML browser that allows him to explore design alternatives Bill provided (Figure 6, 7).

Figure 7. Left, the wood panelling alternative in an options window. Right, John's response to the wood panelling inquires about the cost.

John tries the different wall material options (Figure 7 left). Lacking a subtle taste, he prefers the wood panelling and he leaves a note (Figure 7 right) for Bill and Mary, expressing his preference and asking how much this option would cost. (Each

change in this design negotiation process creates a new .wrl file with annotation objects and links to the comment pages.)

Also, Bill has provided several options for the kitchen and dining area. The original design (Figure 8, left) leaves the kitchen open to the rest of the house. One option (Figure 8, right) encloses the kitchen with a partial wall and places the dining area outside the wall. As with the choice of wall materials, John and Mary can explore these wall alternatives and record their preferences for Bill to complete the design.

Figure 8. Left, original configuration with no walls; right, new walls define the kitchen and the dining table is moved outside the kitchen.

We constructed this scenario to demonstrate several of Redliner's features. Designers, clients, and consultants browse a VRML world that is stored on a server. Beyond just browsing the model, they can post their comments about the design, by leaving markers (coloured spheres) on or near the parts of the design they are commenting on. Others who visit the VRML model later can see these markers and read the posted comments. With a little more effort (this is the designer's job), objects in the model can be assigned optional positions, sizes, and colours, and an interface is provided for clients and consultants browsing the model to explore these variations and record their preferences.

3. IMPLEMENTATION OF THE WORKING PROTOTYPE

We describe briefly the technologies we explored and employed in this project. There are four levels of implementation: 1) basic VRML modelling and behaviour, 2) VRML communication to Java through Script Node;. 3) VRML communication to Java applets through EAI, and 4) CGI scripts in the Perl language to support input, saving, and processing of forms, new world files and file uploads.

The implementation of Redliner consists of several modules. First, VRML models are stored on the server, and a CGI program manages the storage and retrieval of posted text. Client side Java applets handle user mouse events in the VRML world. The Java applets communicate with the VRML world in several ways: via VRML's internal object behaviour language, Script Nodes (in an earlier version of Redliner), using the External Authoring Interface (EAI), and using the CGI Perl script.

3.1 Using VRML's Scene Description Language

The first level implementation of Redliner uses VRML's scene description language to specify object behaviours. VRML provides a set of object-oriented graphics objects augmented by hypermedia links with geometry rendering. It is a file interchange format[1]. We use VRML to specify object behaviours and their response to user events. Using ROUTEs, we connect 3D object nodes and fields to behaviour driven sensors and timing in the VRML world. For example, when a user enters a room, the ProximitySensor can activate and turn on a SpotLight (DirectionalLight, PointLight) node to provide radiating light rays. The other object nodes in the scene that are visible will be illuminated and material and texture are properly shaded. The Sound node places an audio clip (.aif, .wave) at a certain location; it can be rendered spatially (e.g., music comes from the right speaker) and can be triggered on and off. Similar examples of Sensors (e.g., TouchSensor, ProximitySensor, PlaneSensor) are embedded in objects like doors and windows, and the switches for a television or an elevator.

A WWWAnchor node can be placed in the world to link to a new world or a Web site (with a URL, Uniform Resource Locator specified). For example, a high rise building can be divided into separate worlds (.wrl files) and connected through the anchor buttons in an elevator. The reason to break the high rise building into separate worlds is because of memory and file size concerns. Anchors have nothing to do with sensors but behave like TouchSensors. This is the basic level to specify object node behaviours, such as doors that open, or a loop event that rotates a globe. To achieve our goal of greater interaction with the users and letting users control the manipulation of the scene requires the next level of implementation.

Figure 9 shows how a simple event of activating a viewpoint by a click works. For example, when the user clicks on an elevator button, the TouchSensor becomes active (its value becomes True). ROUTE receives the isActive value and turns the set-bind field of viewpoint (an x,y,z co-ordinate value, i.e., a vector of a 2nd floor location) to True. The user viewpoint is then assigned to this preset value.

[1] VRML became an ISO specification (International Specification ISO/IEC 14772-1, http://www.vrml.org/) in December 1997.

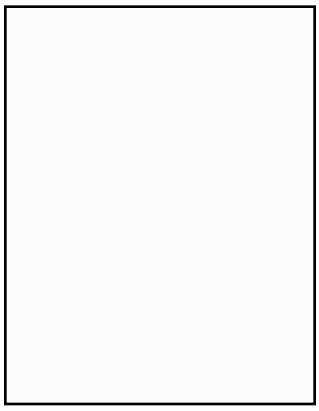

Figure 9. Simple events described in VRML

.

3.2 VRML to JAVA via Script Node

In Redliner's second level of interaction Java communicates to VRML through a Script node interface with APIs. The Applications Programming Interface (API) provides hooks from Java to any component of the object nodes in a scene. For example, one can use API scripts to access any field type in VRML. VRML provides ways to get and set the value of its own fields or get and set global information associated with the VRML browser. Script Nodes encapsulate Java (or JavaSscript, VRML script or any other language) functionality and connect (and convert) variables in the Java environment with VRML object field values. Script Nodes can contain a program, identify a change or a user action, receive events from other nodes, and send events to change others. For example, Figure 10 below shows our first prototype of touching an object to change its location according to user input. This approach is more interactive than the version above because a user can supply inputs based on specific, individual preferences.

For example, a sphere object that responds to a mouse click event by changing its colour, texture and location requires communication between a Java class or Java Script to access the object's fields and respond to events through Script Nodes. Figure 10 shows the process flow. First, when the TouchSensor isActive, its ROUTE sends an EventOut to the Script node. The Script node activates a JavaScript or Java class that takes user input and sends it as an EventOut (changePosition) back to the Script node. ROUTE takes this EventOut and sends its value to an exposedField that translates this object to a new location according to the x,y,z coordinate received by Java.

.

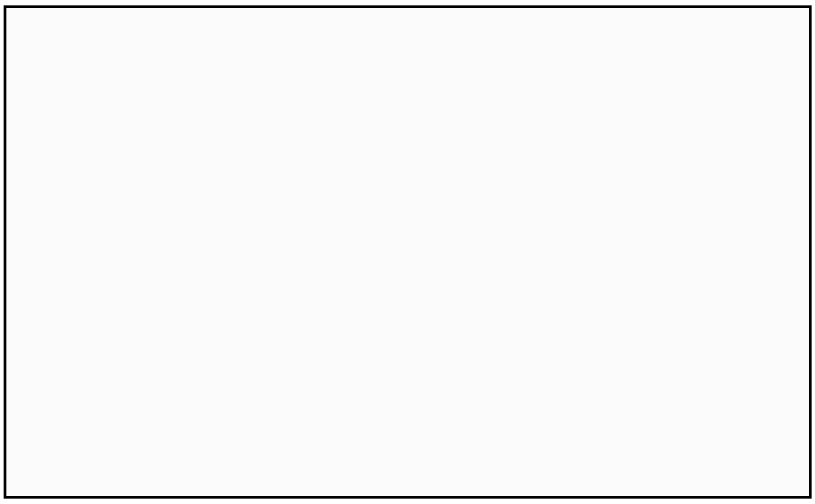

Figure 10. Communication between VRML and Java through a Script node.

3.3 JAVA to VRML via EAI

The above example works for simple interactions such as changing the colours of the object from red to blue or from one location to another. Basically, the Script Node provides a connection to Java from inside the VRML world. A more complicated modification of the scene from user input (e.g., type in RGB values or x,y,z co-ordinates) requires accessing several different Java classes and windows in order to make them interact with each other. For example, to accomplish the task of letting a user specify a change of view location, we need communication among a VRML browser, a Script Node containing a Java program and a Java window class with an interface for input. Any modification of the interaction requires changes to both the Java program and also the VRML file (fields of the Script nodes). Therefore, we investigated using the VRML External Authoring Interface (EAI). The EAI provides a Java interface that communicates with an external HTML Web browser. EAI applets can send information to and from VRML scenes embedded in an HTML page[2]. Using EAI, a Java applet can handle any new world from "outside" without going "inside" it to change its specification for events and fields.

The current version of Redliner has several modules. The first provides the functionality of adding annotation markers on the surface of objects as a person walking through the scene. A Java applet is always watching the VRML world and observes the user's actions (e.g., moving the cursor to a location, capturing mouse clicks). Once an event (e.g., a mouse-click) occurs, the program logs the location of the click, creates an annotation sphere in the VRML world, then passes these messages and values back to the VRML browser so the object appears with an ID number. The second module is the interface that enables the user to select design

[2] (EAI information can be found from the Web, www.vrml.org, and www.cosmosoftware.com/developer/eai_whitepaper.html).

options and put in changes of location, texture, and colour through the type-in boxes.

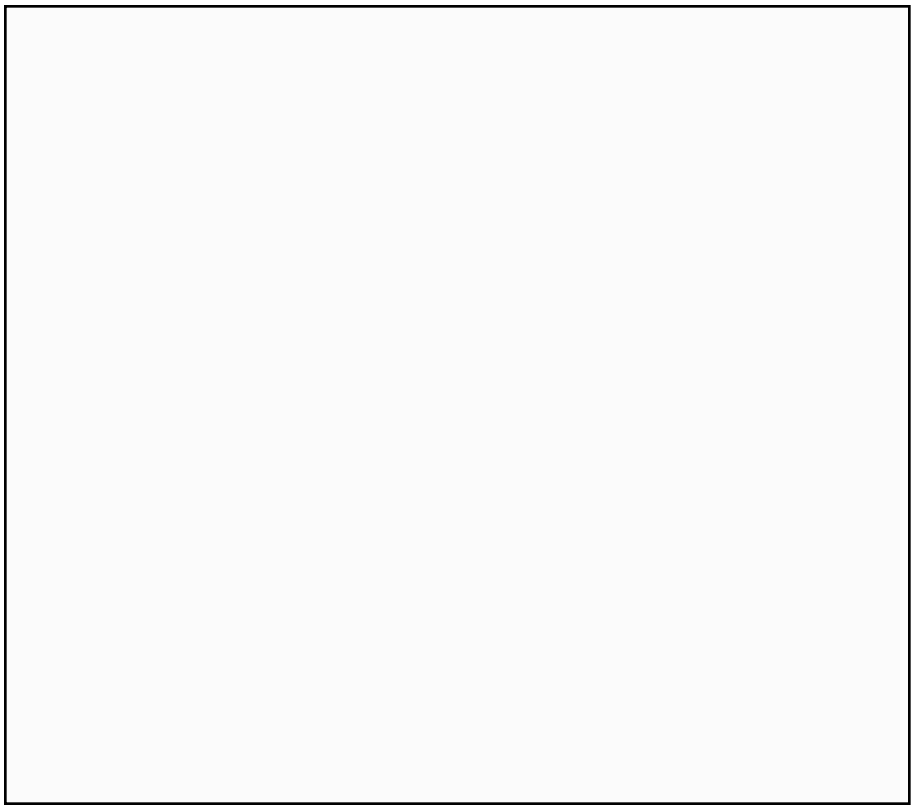

Figure 11. VRML and Java communicate via the Extended Authoring Interface (EAI).

The Java applet has four methods: 1) an **init** method, 2) a **start** method that looks for a VRML browser and special nodes as defined in the program, 3) an **action** method that waits for Java events, and 4) a **callback** method that implements the EventOutObserver class to watch for VRML events. For example, Figure 11 shows how a Java applet can control all the interaction of a classic VRML file. The Java applet and a VRML file are loaded at the same time in a HTML document. The first function of the applet (after initialising some instance variables) is to recognise the VRML world and some specific nodes that will be used and changed. This is done by the **start** method. Then the **callback** and **action** methods are called whenever something new happens. In Redliner, both these methods are used for dropping the annotation markers (spheres). The **action** method is used to recognize the user—John, Mary or Bill—and send to VRML the colour of the Note (red for John, green for Mary, yellow for Bill). The **callback** method, constantly watching for VRML event, is activated when the user clicks in the VRML world to drop a node. **Callback** records the position of the click and sends these co-ordinates back to the VRML files where a new sphere appears.

3.4 CGI Perl Access to Server to Facilitate Reviews

As designers and reviewers add annotations and make changes to the VRML world, these modifications must be stored so that other users can view them later. Saving new VRML object additions and the annotation texts is handled through a Perl CGI (Common Gateway Interface) script. A server side CGI script in Perl reads values (e.g., strings, numbers) set by a Web page form, does some processing on these values, and creates an output (e.g., result, generating a new HTML page) after closing the document. It also allows uploading a file, creating a log file back to the server.

In Redliner, a VRML browser and several buttons (to identify different users, activate comments and submission by Java applets and Perl scripts) are embedded in an HTML page. A user first identifies herself (Mary) by clicking on a button, then explores in the VRML world. She posts annotation nodes (green spheres) in the scene with her signature. When she clicks on the annotation markers a window pops up with a new HTML page that contains a form for her to enter remarks associated with the current location. The Perl script on the server handles this form. Comments entered in the form are written back to the page and redisplayed below the form. The new VRML (.wrl) file with the annotation objects are also saved as a new file back to the server. A later appendix to the comments at the same location can be entered through the same interface, and changes appended to the previous input in chronological order. The second HTML page where the users review and select design options, or enter a new specification of the locations and material of the design objects is handled in a similar way. This page has interfaces for selecting among the designer's proposed alternatives (colours: red, pink, orange). The page also offers type-in boxes for a user to specify a particular value (e.g., of location, or texture) if unsatisfied with the alternatives the designer provided. The result values and the world are saved back to the server and are ready for the next iteration of review.

3.5 Summary of Functionality

In summary, here is the basic functionality of our current Redliner prototype. Users can view a shared design model stored on a central server from their VRML-enabled Web browsers. They can add text annotation objects to the design model, which appear as small coloured spheres with numbers on them. The spheres are linked to text that is maintained in a log on the server, and the colours of the spheres indicate the authors' identities. The annotation objects (and all additions and changes made by users) are stored in separate VRML files on the server, which are loaded on top of the base model VRML file that contains the original design. User annotations are available for all to see and comment on.

Designers can post several alternatives for various elements in the design, allowing browsing users to change the colours, positions, and other parameters of the walls, windows, and floors, in the design. The users can record their preferences, which are saved back to the server as a VRML overlay file to the base model.

Designers and users can record preferred viewpoints in the model, thereby describing a path through the design. These viewpoints are stored on the server, and using a Java applet, users can select a navigation path and move through the model.

3.6 Optimisation of VRML models

In our demonstration project, we constructed the architectural model in Form•Z, a commonly used three-dimensional modelling application, and we used that program's export function to produce the VRML model. Form•Z (version 2.9.5) does not optimise the VRML files it produces, and therefore some effort was spent reducing the complexity of the VRML representation of the model by hand. Therefore, we are also developing a set of simple translation tools to help with these tasks. For example, one tool will convert a complex Form•Z model such as a cylinder with multiple facets to a single cylinder object node in VRML.

4. RELATED WORK

Visionary architects and designers have heralded the potential of virtual reality and immersive environments. Bertol's book, Designing Digital Space: an architect's guide to virtual reality (Bertol 1997) offers visions of possible applications, combined with more technical descriptions of recent virtual reality research. Donath and Regenbrecht (Donath and Regenbrecht 1996) have explored sketching in three-dimensional virtual reality spaces, using hand gestures to sweep out planes and solids and to manipulate the virtual objects in a space. Davidson and Campbell (Davidson and Campbell 1996) developed and experimented with Greenspace 2, a prototype virtual environment for architectural design review. Virtual reality systems have also been used to help architects assess the performance of their designs (Pinet 1997). Campbell (Campbell 1998) describes a real world test application, more pragmatically oriented, in which a design firm used VRML models to communicate construction information (e.g., dimensions) about the design of a stair with a fabricator. In the model, each object in the VRML was defined as an anchor for call-out notes and links to the specification document.

Web PHIDIAS, a system with aims somewhat similar to ours, has been under construction in our laboratory for the past two years. The principal differences between Redliner and Web PHIDIAS are as follows. First, Web PHIDIAS, which derives from the PHIDIAS system that has been under development for over a decade, is based on an issue-based information system (IBIS) approach, in which comments are structured in a hierarchy of design rationale (issues, answers, and responses). Our Redliner comment system is at present more loosely structured; users simply add to comment logs associated with the coloured spheres. Second, at least in the version described in (McCall, Holmes, Voeller and Johnson 1998) the objects in the VRML world had to be explicitly prepared to allow designers to attach a hyperlink to the PHIDIAS issue base. In Redliner, designers and clients add new annotation objects to the existing VRML world, attaching coloured spheres to the design elements in the model as they browse through the world scene. Third, we are developing a tool kit that makes it easy for a designer to modify VRML objects

in the model so that users can move them, change their colours, positions, or other properties.

Virtual Reality (VR) was defined originally as an advancement of technology that engages human senses (e.g., sight, hearing, touch) through input and output devices. As Heim pointed out, VR has "three I's" – immersion, interactivity, and information intensity (Heim 1998). Immersion is usually achieved through a head mounted display (HMD) or a CAVE environment (Cruz-Neira, Sandin, DeFanti, Kenyon et al. 1992). A desktop virtual reality ("fish tank VR") usually is equipped with stereoscopic glasses and manoeuvring device (e.g., data glove, joy stick or space mouse). Dille argues that VRML is an information management tool for the Internet that provides a capability for simulation and participation from multiple users (Dille 1996).

VR applications have included simulation of site (e.g., walkthrough, flight, rides) and object manipulation (e.g., product assembly, work operation). The Committee on Virtual Reality Research and Development's report to the US federal government agencies (National_Research_Council 1995) recommends that four areas show the most promise for synthetic environments (that is, virtual environments, teleoperation, and augmented reality). They are: 1) design, manufacturing, and marketing; 2) medicine and health care; 3) hazardous operations; and 4) training.

Examples of VR projects abound. In the category of design and manufacturing systems, simulation tools have been built to support configuration lighting control (Rygol, Dorbie, Worden, Ghee et al. 1995) and to support testing and evaluation of robot welding work cell design in virtual prototyping (Bickel 1998). Medical applications include a dynamic simulation connected with a force feedback arm to position a drug molecule in potential binding site (Ouh-young, Beard and Brooks 1989), remote surgery (Green, Hill and Satava 1991; Satava 1992), and a medical virtual workbench (Lawton, Poston and Serra 1995). Training systems for hazardous operations include flight simulators (Furness 1986; Vince 1995), Steve (Soar Training Expert for Virtual Environments) (Johnson and Rickel 1997), and Jack, a simulated human figure to carry out control and repair tasks (Badler, Phillips and Webber 1993), and remote control for atomic bomb factory waste disposal (TRW Space and Defense Systems, for the US Department of Energy).

A relevant arena of virtual environments is Computer Supported Collaborative Work (CSCW). This work focuses on managing and visualising information. For example, SeeSoft (Eick, Streffen and Sumner 1992) uses arrangements and colours to represent large bodies (millions of lines) of source code. Cone Trees (Robertson, Card and Mackinlay 1991; Robertson, Card and Mackinlay 1993), Perspective Wall (Mackinlay, Card and Robertson 1991) and Web Forager (Card, Robertson and York 1996) at Xerox PARC use physical metaphors such as tree, walls, and books to display hierarchically structured data as objects in a three dimensional virtual world.

Current VRML development follows the larger trends of VR development and has augmented VRML with extensions (see, for examples, the proceedings of the VRML 98 symposium). The human simulation system Jack (now distributed by Transom Technologies, Inc.) has been incorporated into the OpenWorlds (Eick, Streffen and Sumner 1992) toolkit as a base for creating applications of human

performance—a responsive actor in a story, or the simulation and analysis of an astronaut's body movement in space. A Showroomkit (Dauner, Landauer and Fraunhofer 1998) is under development to support adding predefined exhibition objects into a showroom. Another project explores integrating geographic information (Abernathy 1998) such as topography and GPS (Global Positioning System) information into VRML. A psychology application uses VRML with objects of different colors and shapes to test people's spatial cognition (Givaty, vanVeen, Christou and Bulthoff 1998) in a 3D world and records the memory test on a 2D plan view.

5. DISCUSSION AND FUTURE WORK

We have constructed a working prototype to demonstrate the potential of interactively, asynchronously annotating a three-dimensional architectural design model. Our scenario involves an architect and two clients interacting around the design of a house. The same techniques could be used to support design critiques in a 'virtual design studio' project conducted at several architecture schools around the world, or in a more professional context in which architects, consulting engineers, and other experts collaborate.

Sun released a 3D toolkit for Java called Java3D in 1998. Java3D is an API for creating and controlling 3D objects and behaviours[3]. Java3D programs are saved as Java bytecodes, not as modelling format. It's a useful approach but implementation is tedious. It requires the programmer to understand the structure of any imported object in order to apply behavioural control in the proper places. A VRML-Java3D working group is exploring the interoperability between Java 3D and VRML[4].

Our Redliner prototype is constructed from a diverse collection of current Web technologies including VRML scene description language and CGI scripting. These particular technologies are already being replaced by newer approaches; for example, Java3D is likely to replace VRML in the near future. We are less interested in the specific technologies than in the opportunities for a collaborative discussion around a three-dimensional design model.

We look forward to a number of enhancements and extensions to the current Redliner prototype. Some of these are described below:

We adopt a conservative (and inexpensive) approach to virtual reality: we do not employ head-mounted displays, CAVE (VR room) technology, haptic feedback, or immersive audio (we do not have access to this hardware). An obvious enhancement to the current system would simply place our Redliner system in a hardware immersive virtual reality environment. Along similar lines, our VRML models do not include constraints, such as fixed viewpoints at eye level, and interference of solids; any object can be placed in any location, and viewers can fly

[3] http://java.sun.com/products/java-media/3d/
[4] http://www.vrml.org/WorkingGroups/vrml-java3d/

around and through the model. These immediate and relatively simple enhancements are needed.

We have built our system on an asynchronous model, but it could also be used in a synchronous mode. We would then need to construct avatars to represent the various designers as they move through the model. It might prove valuable to embed the Redliner system in a larger online conferencing system with text chat, whiteboard, and audio-video conferencing.

Currently, designers interact with the building model by posting annotations, or by manipulating objects that have been pre-programmed (by the designer) to allow for variation. An interesting way (for a designer) to interact with a virtual world might be to sketch on top of the VRML model (in a kind of transparent window over the browser), to add elements to the model or to change the positions or attributes of existing elements. We have built an early prototype of this capability but additional work is required.

At present our VRML models are simply geometric; their elements are geometric surfaces and solids. A logical enhancement would link our Redliner system with online catalogues of building products so that the designer could access dimensionally accurate design elements that would store their cost, material, dimensional, and performance attributes. Reviewers could not only manipulate pre-built building components, but they could also select new ones from the product libraries and add them to the model. As with our current system, the 'base model' would be stored separately from the reviewers' annotations and additions.

The current Redliner system just logs text typed into the annotation screens, associating it with each coloured sphere. The text could also be automatically emailed to the designer(s) who are responsible for the design elements under discussion, or the logs could be structured as for example in the PHIDIAS system of McCall and his colleagues.

6. ACKNOWLEDGEMENTS

This research was supported in part by the National Science Foundation under Grant No. IIS-98-19856. The views contained in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. The authors also acknowledge support from the Chancellor's office at the University of Colorado at Denver, and a grant from the Undergraduate Research Opportunity Program (UROP) for Jimmy Davidson. Jimmy Davidson constructed the VRML building model and Dongqiu Qian and Peter Kappus helped with Java and Perl debugging. John and Mary's house is modelled loosely on a design by Bennett Neiman.

7. REFERENCES

Abernathy, M. (1998). Integrating Geographic Information in VRML Models. VRML 98 – Third Symposium on the Virtual Reality Modeling Language. S. N. Spencer. Monterey, CA, ACM**:** 107-114.

Badler, N. I., C. B. Phillips and B. L. Webber (1993). Simulating Humans: Computer Graphics, Animation, and Control, Oxford University Press.

Bertol, D. (1997). Designing Digital Space: an Architect's Guide to Virtual Reality. New York, Wiley.

Bickel, D. (1998). Virtual Reality for Industrial Applications. 3D Realtime Simulation and VR-Tools in the Manufacturing Industry. F. Dai. Berlin, Springer**:** 123-138.

Campbell, D. A. (1998). Architectural Construction Documents on the Web: VRML as a Case Study. Digital Design Studios: Do Computers Make a Difference? T. Seebohm and S. V. Wyk. Quebec City, Canada, ACADIA 98**:** 266-275.

Card, S. K., G. G. Robertson and W. York (1996). The WebBook and the WebForager: An information workspace for the World Wide Web. CHI '96 ACM Conference on HUman Factors in Software, ACM**:** 111-116.

Cruz-Neira, C., D. J. Sandin, T. A. DeFanti, R. V. Kenyon, et al. (1992). "The CAVE: Audio Visual Experience Automatic Virtual Environment." Communications of the ACM **35**(6): 65-72.

Dauner, J., E. Landauer and I. Fraunhofer (1998). 3D Product Presentation Online: The Virtual Design Exhibition. VRML 98 – Third Symposium on the Virtual Reality Modeling Language. S. N. Spencer. Monterey, CA, ACM**:** 57-62.

Davidson, J. N. and D. A. Campbell (1996). Collaborative Design in Virtual Space: Greenspace II: A Shared Environment for Design Review. Proc. 1996 National Conference Association of Computer Aided Design in Architecture (ACADIA '96): "Collaboration, Reasoning, and Pedagogy". P. McIntosh and F. Ozel. Tucson, AZ, 165-179.

Dille, E. (1996). Exploring Moving Worlds. Research Triangle Park, NC, Ventana.

Donath, D. and H. Regenbrecht (1996). Using Virtual Reality Aided Design Techniques for Three Dimensional Architectural Sketching. in Proc. 1996 National Conference Association of Computer Aided Design in Architecture (ACADIA '96): "Collaboration, Reasoning, and Pedagogy". P. McIntosh and F. Ozel. Tucson, AZ, ACADIA**:** 199-212.

Eick, S. G., J. L. Streffen and E. E. Sumner (1992). "SeeSoft – A tool for visualizing software." IEEE Transactions on Software Engineering **18**: 957-968.

Furness, T. A. (1986). The supercockpit and its human factors challenges. 30th Annual Meeting of Human Factors Society. Dayton, OH**:** 48-52.

Givaty, G., H. A. vanVeen, C. Christou and H. H. Bulthoff (1998). Tele-Experiment – Experiments on Spatial Cognition using VRML-based Multimedia. VRML 98 – Third Symposium on the Virtual Reality Modeling Language. S. N. Spencer. Monterey, CA, ACM**:** 101-105.

Green, P., J. H. Hill and R. M. Satava (1991). "Telepresence: Dextrous procedures in a virtual operating field." Surgical Endoscopy(57): 192.

Heim, M. (1998). Virtual Realism. New York, Oxford, Oxford University Press.

Johnson, W. L. and J. Rickel (1997). "Steve: An Animated Pedagogical Agent for Procedural Training in Virtual Environments." SIGART Bulletin **8**(1-4): 16-21.

Lawton, W., T. Poston and L. Serra (1995). Time-lag reduction in a medical virtual workbench. Virtual Reality Applications. R. A. Earnshaw, J. A. Vince and H. Jones. London, New York, Academic Press.

Mackinlay, J. D., S. Card and G. Robertson (1991). Perspective Wall: Detail and Context Smoothly Integrated. ACM SIGCHI '91 Conference on Human Factors in Computing Systems. New Orleans, LA, ACM**:** 173-179.

McCall, R., S. Holmes, J. Voeller and E. Johnson (1998). World Wide Presentation and Critique of Design Proposals with Web-PHIDIAS. Digital Design Studios: Do

Computers Make A Difference? ACADIA 98. T. Seebohm and S. V. Wyk. Quebec City, Canada, ACADIA:Association for Computer-Aided Design in Architecture**:** 254-265.

National_Research_Council (1995). Virtual Reality – scientific and technological challenges. Washington D.C., National Academy Press.

Ouh-young, M., D. Beard and F. Brooks (1989). Force display performs better than visual display in a simple 6D docking task. IEEE Robotics and Automation**:** 1462-1466.

Pinet, C. (1997). Design Evaluation Based on Virtual Representation of Spaces. ACADIA 97, Association of Computer Aided Design in Architecture**:** 111-120.

Robertson, G. G., S. K. Card and J. D. Mackinlay (1991). Cone Trees: Animated 3D Visualization of Hierarchical Information. CHI 91, ACM**:** 189-194.

Robertson, G. G., S. K. Card and J. D. Mackinlay (1993). "Information visualization using 3D interactive animation." Communications of the ACM **36**(4): 57-71.

Rygol, M., A. Dorbie, A. Worden, S. Ghee, et al. (1995). Tools and Metaphors for User Interaction in Virtual Environments. Virtual Reality Applications. R. A. Earnshaw, J. A. Vince and H. Jones. London, New York, Academic Press**:** 149-161.

Satava, R. M. (1992). "Robotics, telepresence and virtual reality: a critical analysis of the future of surgery." Minmally Invasive Therapy **1**: 357-363.

Vince, J. (1995). Virtual Reality Techniques in Flight Simulation. Virtual Reality Systems. R. A. Earnshaw, M. A. Gigante and H. Jones. London, New York, Academic Press**:** 135-141.