INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

**Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA**

UMI[®]
800-521-0600

Running Head: ROOM DESIGN UTILITY

**Development & Usability Test
of a Room Design Utility**

**Brian Paynter (B.A.(Hon))
Carleton University, Department of Psychology
Advisor: Dr. Richard F. Dillon**

**Submitted in partial fulfillment
of the requirements for the degree of
Master of Arts in Psychology,
December 1998**

**National Library
of Canada**

**Acquisitions and
Bibliographic Services**

395 Wellington Street
Ottawa ON K1A 0N4
Canada

**Bibliothèque nationale
du Canada**

**Acquisitions et
services bibliographiques**

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-43321-8

Canada

The undersigned recommend to the
Faculty of Graduate Studies and Research
Acceptance of the thesis:

“Development and Usability Test of a Room Design Utility”

submitted by

Brian Paynter

in Partial fulfillment of the requirements for
the degree of Master of Arts

Chair

Thesis Supervisor

Carleton University
January, 1999

Abstract

A User Centered Design methodology was used to develop a fully functional Room Design Utility for use in the control of video conferencing environments. Area Network. The steps included a task analysis, user definition, issue identification, a prototyping session with five subjects, functional prototype development, a heuristic evaluation and the development and application of a formal usability test. The usability test involved 20 subjects over two iterations of test, analyze and modification of the interface. The utility passed all usability criteria established to determine ease of use and effectiveness. Outstanding modifications are identified.

Table of Contents

Telepresence, Telelearning and the Device Area Network.....	1
User Centered Design.....	10
Design Guidelines.....	13
Method.....	16
Task Analysis.....	16
User Definition.....	17
Issue Identification.....	18
Issues Prototype	
Procedure.....	21
Results and Discussion.....	22
Usability Goals and Measures.....	28
Design of a Functional Prototype.....	30
Heuristic Evaluation.....	33
Design of the Usability Test.....	34
Formal Usability Test	
Procedure.....	37
Results.....	38
Discussion.....	52
Short Test of Prototype Changes	
Procedure.....	55
Results.....	55
Discussion.....	57
Second Iteration of Usability Test	
Procedure.....	58
Results.....	58
Discussion.....	65
General Discussion.....	67
Conclusion.....	71
References.....	72
Appendices	
A - Prototype Questions (Test #1).....	A1
B - Formal User Test (Test #2) - Tasks and Questions.....	B1
C - Follow-up Test (test #3) - Tasks.....	C1
D - Room Design Utility Source Code.....	D1

List of Tables

Table 1. List of Tasks for the Room Design Utility.....	17
Table 2. Wizard Storyboard.....	20
Table 3. Room Description Results.....	23
Table 4. Room Transitions.....	25
Table 5. Rotation Buttons.....	26
Table 6. Usability Test Task Mapping.....	34
Table 7. Self-Rated Subject Profile.....	38
Table 8. Usability Test Results.....	39
Table 9. User Ratings by Exposure.....	40
Table 10. User Results - Question 2: Itemizing the Room.....	40
Table 11. Noted Errors.....	42
Table 12. Frequency of Errors.....	43
Table 13. User Responses - Task 3: Wall Objects.....	44
Table 14. Post-Session Questionnaire.....	46
Table 15. Suggested Improvements.....	50
Table 16 Other Subject's Comments.....	52
Table 17. User Performance.....	56
Table 18. User Ratings.....	56
Table 19. Self-Rated Subject Profile.....	58
Table 20. Usability Test Results (Test 2 and Test 4).....	59
Table 21. Results - Question 2: Itemizing the Room (Test 2 and Test 4).....	60
Table 22. Noted Errors During Test 4.....	60
Table 23. Frequency of Errors (Test 2 and Test 4).....	60
Table 24. User Results - Question 3: Wall Objects.....	61
Table 25. Post-Session Questionnaire (Test 2 and Test 4).....	62

List of Figures

Figure 1. DAN screen showing control of local VCR.....	3
Figure 2. DAN screen showing a view.....	4
Figure 3. DAN screen showing camera controls and preset views.....	6
Figure 4. Generated east view based on user designed north view.....	7
Figure 5. Room Design Utility Mock-up.....	9
Figure 6. Icon Palette.....	22
Figure 7. Possible Rotation Buttons.....	26
Figure 8. Screen Shot of the Room Design Utility.....	31
Figure 9. Conceptual Layers of the Room Image.....	32
Figure 10. RDU Screen Shots Showing Room Rotations.....	45

The purpose of this study was to develop and perform usability tests on a Room Design Utility (RDU) for the video conference room control derived from the Device Area Network (DAN). The DAN originated in the Telepresence project and continues as part of a research project for the Telelearning Network of Centers of Excellence. The RDU was developed using a User Centered Design methodology and enables different conferencing environments to be added to the DAN.

Telepresence, Telelearning and the Device Area Network

The Ontario Telepresence Project was a research venture undertaken between 1991 and 1994 by the Information Technology Research Centre (ITRC) and the Telecommunications Research Institute of Ontario (TRIO), both part of the Government of Ontario, in collaboration with four European countries, three universities and several high technology companies. The universities included Toronto, Carleton and Ottawa. The companies included Corel, Newbridge, Hewlett-Packard, Bell Ontario Network Systems, Bell Canada and Xerox. The goal of the Telepresence project was to conduct applied, user-centered research into the design of computer supported collaborative work (CSCW) systems (Riesenbach, 1995). It was intended to integrate computers, multimedia databases, audio-visual and telecommunications equipment to enable people to work together over long distances (ITRC, 1991; Narasimhan, 1993). This was done by designing and implementing a networked, common technology platform where video, text, graphics, images and voice were handled in a uniform and coherent manner (ITRC, 1991). The

participating companies were encouraged to take this research and develop it into marketable products.

The Telelearning project is a current research initiative sponsored by the federal government and Canadian industry, involving over 130 researchers at more than 29 universities (Dillon, 1998). The Computer User Research and Evaluation (CURE) laboratory at Carleton University is involved and is primarily interested in applying computer communications technology to distance education and more specifically, in investigating user interface and ease of use issues as they relate to the technology.

The Device Area Network is one application that resulted from the Telepresence project. It allows users to connect to and control multiple audio-visual (AV) devices from a computer (Narasimhan, 1993; Riesenbach, 1995). A typical DAN office configuration has two connections to the network; one from the computer to the LAN and the other, an audio-video cable, connecting AV equipment to the AV switch/server (ITRC 1991). Devices available in an office may include cameras, television monitors, VCRs, microphones and speakers. A key concept of the original research (ITRC, 1991, P18) was to provide “integration of the many services and facilities into an environment usable by people who have little or no understanding [of the] computer or communication technologies which are being deployed and who want to complete the shared task without being seriously distracted by the control requirement of the technology.” The idea was to hide the technology and control structures and let users proceed with their work.

This earlier Device Area Network control program used at Carleton University had an intimidating interface usable only by trained engineers. A new DAN interface developed by the CURE laboratory at Carleton University as part of the Telelearning project is nearing completion. The new interface actually replicates the VTel video conferencing system which is a subset of the Device Area Network. Eventually the system will be expanded to encompass the full DAN's capabilities. The new program replaces the older system with a Graphic User Interface (GUI) that most people can learn with a minimum of instruction. Screen shots of the new interface are shown in Figures 1, 2 and 3. It is a dramatic improvement over the original system.

Figure 1. DAN screen showing control of local VCR

The screen as shown here displays abstract representations of both the local and remote sites. These representations provide the minimum information necessary to allow spatial orientation by the user (Dillon, 1998). For example, the table, door and, if necessary, a window, are all that are needed to allow users to orient themselves. Other objects shown, such as VCRs, computer and document cameras, are devices that can be controlled by clicking on them. In Figure 1 the VCR has been selected. When a device is selected, the controls to manipulate it are displayed underneath the room image. These controls are modeled after industry standard controls and users generally require no instruction to use them.

Figure 2. DAN screen showing a view

Video cameras are not directly selected in the Device Area Network. Instead users select views of what they would like to see. In Figure 2 the user is about to click on the rectangle outlining the front of the room and the lecturer or instructor. This corresponds to a view by a camera of the front of the room. Similarly, clicking on the audience at the back of the room would provide a view of them. The remote site provides the same sort of camera views.

In Figure 3 the lecturer view for the local room has been selected. Similar to the VCR control in Figure 1, the controls to adjust the camera are displayed underneath the room. Preset camera views are displayed in a column beside the room. These preset views are specific combinations of the camera's zoom, pan and focus settings. Clicking on a preset automatically adjusts the camera to that view. The camera can be adjusted by using the pan, zoom and focus control buttons at the bottom of the screen. New presets can be created by selecting the 'Save new preset' button and selecting one of the camera shots to replace. Display of the controls and presets can be toggled by selecting the appropriate 'Hide...' buttons on the outside edge of the screen.

Other controls that are always on the screen are the PIP, speakers, microphone and disconnect buttons. The PIP (picture in picture) buttons toggle the PIP feature on the monitor on or off and swap the PIP and main screen images. The speaker buttons adjust the volume of the audio signal. The microphone button toggles the lecturer's microphone on or off. 'Disconnect' terminates the connection with the remote site.

Figure 3. DAN screen showing camera controls and preset views

The focus of this study is the room image. As the system stands now, it is not possible to change the room when installing at another site or change the user's orientation within the room. It is desirable to allow the user to reorient the room representation to correctly reflect the user's location within the room with respect to the equipment. A user may not be at the front of a room, but instead may be sitting along the table using perhaps a portable computer connected to the Device Area Network. The DAN user interface should reflect this and be able to change orientation and still understand what objects are in the room that it can control. The user would then correctly see the equipment in a room in a spatially correct orientation around the user.

Based on this, the Room Design Utility needs to address the Device Area Network's ability to change orientation in a room as well as design new rooms. The Room Design Utility must allow the user to layout a room and add audio-visual and computer equipment to it. The Room Design Utility must also be able to edit or reload the original image so that the user can modify the contents of a room when it changes. All of this information must be saved in such a way as to be useful to the Device Area Network. This would include saving the image from the four cardinal points (N, S, E, W) and storing information about the equipment in the room. The DAN would need to know both what is in a room and where it is in each of the four orientations of the image.

Figure 4. Generated east view based on user designed north view

Figure 4 illustrates the orientation requirements of the Room Design Utility. A user would design a room, the 'normal view', which we arbitrarily call the north view. The Designer must then take that information and generate the other three views which we call east, west and south. It should be able to rotate the wall, equipment

details and audience so that they appear in the correct places within the image. In this example the eastern window has been rotated to the end of the room and the equipment has been moved from around the speaker to the right of the speaker. This raises some perceptual and cognitive issues regarding how to present the transition to the user. Should it simply change instantaneously or should there be many visual cues of the room rotating?

From a technical perspective all of this can be accomplished by storing the detailed information about the room objects in each orientation in a small database. The images can then be interpreted correctly by the DAN through access to the database. The Room Design Utility would need to keep additional information regarding the layering of objects as well as how to render the room from the different perspectives. This would allow it to reload a room image for editing.

From an interface perspective, the Room Design Utility must be easy to use. Users will be novices to the utility since most will use it only occasionally. For this reason, its use must be obvious with little or no training required. A wizard may be useful to help a user design a new room and give an overview of the process. A wizard is a subprogram that provides conceptual information about an interface or activity and guides the user through an initial use of an application. Conceptually, the Room Design Utility can be broken down into a workspace and the tools that support designing a room as shown in Figure 5. To encourage user recognition, the Room Design Utility will model a generic paint or draw program. Tool objects for the room will be placed on the workspace using a drag-and-drop approach from a tool or object palette. This tool palette would be free-floating to allow the user maximum flexibility

in its placement. A view control to allow the user to shift the view to other orientations would also be available. Other utilities such as file saving and loading would be available through a menu. One method to ensure that the user interface meets this ease of use is to use a User Centered Design approach.

Figure 5. Room Design Utility Mock-up

User Centered Design

Gould & Lewis' (1987) and Booth's (1989) definition of a usable system includes three elements: it must be easy to learn, easy to use and useful. User Centered Design attempts to facilitate usability by placing the emphasis of the design process on the user. Instead of concentrating on the system requirements, such as internal representations and structures or efficiency, the needs of the user become the central focus. How tasks are to be completed and what information the user requires to complete them are determined first. The system is then designed to meet those needs. It is an attempt to match the system functionality to the requirements of the user. Care is taken to make the system behave in the same manner that the user thinks when completing tasks.

During the initial work of User Centered Design, predesign definitions of usability goals and tasks are established and less formal assessment procedures such as prototyping and heuristic evaluations are carried out.

Usability goals are targets that are established to determine at what point an interface will be considered usable. They must be attainable and usually take the form of proportions. For example, a goal may be that 80% of users can complete a certain task with no more than one error. Identifying tasks before any design work is done ensures that the development effort will be properly focused. Not only are the various tasks the user will carry out identified and explained but also the user characteristics are examined. This early focus on the users and tasks helps answer two questions: who will use the system and what will they do with it (Baecker, 1987). Gould & Lewis (1987) stress interaction with the users rather than just profiling them and

suggest that if subject domain experts are not available, more emphasis should be placed on the simulation and assessment procedures and the iterative design process.

Prototypes may be paper based, screen mockups or partial or fully functional designs. These designs will be modified and refined throughout the process. Heuristic evaluations are a process whereby usability specialists, working independently, judge the elements of the interface against established usability principles, the heuristics (Nielsen, 1998). Nielsen recommends three to five evaluators and where enough usability specialists are not available, software engineers can be used with a bit of pretraining.

After the initial steps are complete, focus shifts to more formal usability testing. During usability testing, predetermined measurements are collected that allow the designer to determine whether or not the usability goals are being met. These measurements may contain both objective and subjective elements. The objective measures may include performance characteristics such as speed, number of tasks completed and number of errors. Subjective measures may include user's thoughts and attitudes, such as satisfaction level and impressions or their understanding of the system model. If the goals are not met these results are examined to try to determine what parts of the interface may be causing the problems. The prototype is modified and new usability tests are carried out on the tasks that failed to meet the goals. These iterations of prototype-test-analysis are carried out until the goals are met.

By completing informal evaluations prior to the formal usability tests, the likelihood of meeting the usability goals is high. This usually translates into fewer iterations of design and test and a shorter formal test process.

Design Guidelines

Many authors express various guidelines or principles for the development of good computer user interfaces. Baecker (1987) recommends four cardinal principles: know thy user, minimize cognitive load, optimize operations and engineer for errors. *Knowing thy user* ties in with User Centered Design as outlined above. *Minimizing cognitive* load refers to keeping the amount of information a user must retain to complete a task to a minimum. This means making additional or complex information available on the screen and possible actions obvious rather than forcing the user to remember bits of data as well as the process required to complete a task. At the same time the user should not be overwhelmed by confusing or busy screens. *Optimizing operations* refers to enabling rapid execution of frequent operations through shortcuts and minimizing screen differences so that users do not have to study every screen before using it. The final principle, *engineer for errors*, simply means that mistakes or errors will be made and should be anticipated. Error messages should be understandable and, if possible, suggest solutions to the problem. ‘Fatal error: 3009’ is unacceptable. Common errors should be trapped and handled appropriately. Actions should be reversible.

Norman (1983) suggests five maxims when engaged in system design. First, there are no simple answers, only trade-offs. Designers need to realize that different approaches have strengths and weaknesses both of which must be weighed within the design context. Second, there are no errors: all operations are iterations towards a goal. Actions of a user are attempts to reach that goal. Designs should try to address this and help the user towards the goal. Third, low-level protocols are critical. There

should be consistency between modes or screens so that keys, buttons or certain actions are the same throughout the system. This standardisation facilitates usability since the user only has to learn the general process once. Fourth, activities are structured. Users should be guided in logical sequence through subgoals to attain higher goals or activities. Fifth, information retrieval dominates activity. All stages of using a system require a user to examine the state of the system to determine possible actions. Norman (1983; 1988) breaks out these actions into four stages of activity: 1) forming an intention (what to do), 2) choosing an action (how to do it), 3) specifying that action to the system (doing it) and 4) evaluating the outcome (did it work).

Booth (1989) identified seven design problems with a high frequency of occurrence in current systems. First, cognitive load is often too much. Users frequently have to write bits of information down or refer to manuals to complete a task. Second, systems are intolerant of minor errors. Incorrect keystrokes may cause the loss of entire sets of data or full disks may cause systems to halt or crash with rather unpleasant results. Third, systems are often very confusing to new users. Systems don't provide sufficient information to allow processes to be completed and worse don't provide ready access to explanations of what is needed. They also don't help the user develop a mental model of the system and its structures which aids future interactions with the system. All of this results in remedial measures, such as extensive user training, being undertaken to compensate for poorly designed systems. A fourth problem that occurs often is inappropriate interaction techniques, such as requiring commands or numeric entry when graphic manipulation may be more appropriate. An example would be requesting numeric co-ordinates in a CAD (Computer Aided

Design) program used by engineers. Booth's fifth identified problem is that systems often are not useful. They don't provide the functions that users are looking for.

Some examples include the wrong data such as summary information only when details are needed, or the wrong format such as sorting by date when an alphabetical sort is required or completely missing functions such as no subtotals on a long list of numbers. Another problem is forcing users to do tasks in an undesirable way. An example is invoicing systems that force users to gather a company's name from one screen and then force them to traverse several other screens to get the contact information. The final problem Booth identifies is systems that force unacceptable changes to structures or practices of an organization. This can cause dissatisfaction and conflict amongst user organizations as power shifts occur or workloads increase as a result of system requirements.

This design effort incorporated the above guidelines throughout the process.

Method

The study was broken down into discrete subtasks. The initial design work consisted of a task analysis, a user definition, identification of issues, a prototype to resolve the issues, establishment of usability goals and measures, the design of a functional prototype, a heuristic evaluation and the design of the usability test.

After the initial work was completed, the formal usability testing was started. This is an iterative process whereby the prototype is formally tested by users to determine if it is usable. It is iterative in that should it fail to meet the established usability goals, the reasons for the failure are identified, the prototype is modified and it is submitted for retesting. This process continues until it meets the goals. There were two iterations of the current prototype consisting of three sets of usability tests being performed.

A total of twenty five subjects were used during this study. Five were used during the initial steps to help resolve the identified issues. The other twenty participated in the usability tests.

Task Analysis

The task analysis identified that there are two major tasks for which a user would use this software. These are creating new rooms and editing existing rooms. These were broken down into smaller tasks that a user would typically carry out.

Adding a room consists of adding orientation objects such as wall objects and furniture, adding and describing devices or equipment, verifying orientations by rotating the room and saving the room image. Editing a room consists of retrieving a

stored room image, adding or removing items such as wall or room objects or devices, moving objects to different locations, verifying orientations by rotating the room and saving the room image.

Many similarities were noted between the two tasks and so the subtask lists were combined to produce a single list of tasks that the Design Utility would be required to provide. These are listed in Table 1.

Table 1. List of Tasks for the Room Design Utility

1. save and retrieve room
 2. add orientation objects (wall and furniture objects)
 3. add and describe devices or equipment
 4. move objects to different locations
 5. remove items (wall or room objects or devices)
 6. verify orientations (rotate room)
-

User Definition

It was established that there will be two major types of users of the RDU, technicians and presenters. The technicians will be technical support people who have a high level of experience with different programs including some sort of drawing program. The presenters will, by and large, be instructors, teachers and lecturers and will have a full spectrum of computer experience.

All of them will be occasional users, using the Room Design Utility once or a few times a year. Most, but probably not all, will understand how the DAN works having seen it at least once.

The RDU should guide less experienced or first time users through the general steps to assemble a room. It was determined that this could probably be best addressed through the use of a wizard. More experienced users should be able to bypass the wizard and work in a free-form manner.

Issue Identification

Following the initial review of the tasks and user population, eight issues that required further examination were identified. These issues concerned how to implement the various requirements of the interface.

The first issue was to identify appropriate icons for the object palette. It was determined that this could be answered by assembling a sample of icons and asking users during a prototyping session to identify them. The list of orientation objects included doors, windows, pictures, blackboard/whiteboards/screens, round and square tables and audience markers. Active devices or objects included VCRs, overhead projectors/document cameras and video camera views.

The next issue concerned the minimum information about a room that would be required to allow room orientation. This could be examined by explaining the concept to the subjects and then asking them to quickly describe pictures of rooms. The nouns they use and the order mentioned would be noted and used to guide what objects should be made available.

The third issue identified was visual cues during room transitions. When a room is rotated, how much information does the user need to understand what is happening? Will fast transitions (a flip) be acceptable or should intermediate steps be

shown? It was decided that this could be determined by having users examine a computer mockup of both types of transitions and see if they understand what is happening.

The fourth issue was “how should the room orientation mechanism be presented”? It was felt that asking users to identify the best navigator icons from a selection in a room mockup would provide the information. Initial options included a diamond button pad labeled North, South, East and West or a pad labeled Front, Back, Left and Right. Others possibilities were a button pair labeled Left and Right and a single button labeled ‘Rotate Room’ that would rotate the room in one direction only.

The next issue identified was how to deal with the camera view. In the DAN, devices are shown as objects on the screen. The user controls them by clicking on them. The exception to this is the video camera. The camera is represented on the screen by its view (see Figure 2). This is a large, transparent button that outlines what a camera is looking at. The camera controls are activated by clicking on this view button. Since it is desirable to maintain consistency between the DAN and the RDU to aid in the transfer of user skills and knowledge, it was decided the RDU should follow this approach. The question was how to explain this concept to the users so that they would understand what would happen after they placed a camera on the screen and indicated its direction of orientation. It was decided to explain the concept and the planned solution and ask users to comment on the solution.

Another issue identified was ‘smart’ icons. Some of the room objects would need specific behaviours such as doors relocating themselves onto walls if they are

dragged and released in a room. These objects didn't need to be prototyped but they did need to be identified. Objects that were eventually given intelligence included all the wall objects, including doors, windows, pictures and projector screens. These objects will snap back to the closest wall if dragged and released in the room. The camera view also was designed to 'know' when to switch between a vertical and horizontal orientation.

The seventh issue was the content of the Room Wizard. Should it try to encompass every activity possible or just a basic set of typical activities? The trade-off of being all encompassing is that the wizard can become awkward and frustrating if the user doesn't want to create a complex room. It was decided that the wizard should provide enough information to get a new user oriented and productive without trying to present every part of the utility. The wizard was storyboarded as appears in Table 2.

Table 2. Wizard Storyboard

Screen	Content
1	Wizard Welcome
2	Introduction - process to create room
3	Place Doors - introduce orientation and object dragging
4	Place Windows
5	Place Whiteboard/Blackboard/Screen
6	Place Pictures
7	Rotate Room - explanation of concept
8	Place Square Table
9	Place Round Table
10	Place Audience Markers
11	Place Video Camera View
12	Place Document Cameras/Overhead Projectors
13	Place VCRs

The final identified issue was whether the room orientation control should be visible at all times or only when requested? It was decided to keep this control visible because users are occasional and they may not even detect that they can re-orient the room unless it can be seen.

It was decided that the first five issues could best be answered by interviewing representative users, showing them alternatives and getting their opinions. An educated guess was made on the last three issues which would be confirmed during the usability tests. With the issues identified, the next step was to develop a small set of mockups to help gather information for the RDU.

Issues Prototype

Procedure

Five subjects participated in individual sessions lasting less than 20 minutes each. Each subject completed five tasks dealing with the five previously identified issues. The script used for these sessions is attached as Appendix A.

The equipment for these sessions included a Pentium 200 computer running Windows NT4 for the screen mockups and three photographs of boardrooms.

Upon arrival, the subject was briefed verbally about the session and was shown a standard DAN screen. After any questions were answered, the tasks were provided verbally to the subjects. A description of each task and the results are provided below.

Results and Discussion

Figure 6. Icon Palette

The first task presented 11 icons, shown in Figure 6, to users and asked them to identify them. All but three were identified correctly by all five subjects. The Round Table (third icon, left side) was identified as a stool by one subject. The Picture (second icon, right side) was identified as a sailboat by two subjects and the VCR (fifth icon, right side) was not identified as anything by two subjects.

Three users, independently, suggested adding an ornate picture frame around the picture to help identify it. This seemed reasonable and was included in the full prototype. The VCR was a little more of a problem. Two users suggested making the 12:00 display larger and removing the control buttons. A concern there was that it may end up looking more like an alarm clock than a VCR. The 12:00 clock display is stereotypically linked to the VCR in our culture as one of those things one sees on a VCR. The VCR was modified to make it as large as possible, making it both wider

and taller. The 12:00 display was made larger and the control buttons were maintained at the same size.

The second task explained the concept of minimal orientation information. It asked the subjects to describe the important orientation elements in three photographs. The first photograph showed a bare tutorial room containing a U-shaped table, chairs and two doors in opposite corners. The second one showed a meeting room containing a straight, rectangular table with chairs, full length windows with a door in one corner and a small table along one side. The third picture showed a straight rectangular table with chairs, an exterior window, a TV and VCR in a stand, a wallboard on the right hand side and a rolled up projector screen in front of the windows. The results from this question are displayed in Table 3 in the order that the subjects mentioned them.

Table 3. Room Description Results

Picture	Items	Subject 1	Subject 2	Subject 3	Subject 4	Subject 5
1	table chair doors	table chairs lights doors	doors table chairs	doors table	table chairs doors	doors table
2	table chairs window door sidetable	table chairs window door sidetable	door table chairs	door table window	door window table chairs sidetable	table chairs
3	table chairs window TV/VCR wallboard screen	table chair wall plug	table window TV	wallboard d window	window table chairs wallboard TV	wallboard screen TV

The items in the second column, Items, were the items in the room that were felt to be useful for orientation. The results indicate that the subjects understood the idea of minimal description and that they identified much the same objects that we had expected when describing rooms. There were other objects in the rooms that could have been identified but weren't with the exception of a wall plug. We proceeded with the orientation items, comfortable that most users would not have a problem describing rooms using the minimal set of objects provided.

The third task dealt with visual cues provided for room transitions. Users were presented with an instantaneous transition, a 'flip', where one view of a room was substituted for another view without any intermediate cues. In this case, a side view of a room was substituted for a view from the end of a room. They were asked what happened and then presented with the same transition but with intermediate pictures, an animation, to create the image of the room rotating to the side. They were asked to describe what happened and rate their preference of the two options on a 7-point Likert scale with 1 meaning "Very much prefer first transition", 7 meaning "Very much prefer second transition" and 4 meaning "No preference". The results are presented in Table 4.

Table 4. Room Transitions

Question	Subject 1	Subject 2	Subject 3	Subject 4	Subject 5
Describe 'flip'	"Something moved" - clicked 3 times & described direction	"View changed...a rotation"	"Door moved & table is different or he moved around the room"	"Room turned"	"Moved people & things - 90 degree change"
Describe animation	"Rotated room"	"Orientation changed...the flow"	"He went to the side of the table"	"Same thing"	"90 degree turn"
Preference	2 - prefer flip	2 - prefer flip	2 - prefer flip	2 - prefer flip	7 - Strongly prefer animation
Was the 'flip' Transition obvious?	No	Yes	Yes	Yes	No
Animation needed?	No, it doesn't do anything for me	No, it was too fast	No, there was too much movement	No, too fast	Yes, Shows transition - sense of movement

All of the subjects were able to determine what happened on the first transition. Four of the five detected and identified the change within a few seconds. The fifth detected the movement immediately and took three extra clicks on the mouse to repeat the 'flip' before announcing exactly the direction of change. Everyone described the second transition without hesitation but this was expected since they had been primed by the first exposure. Four of the five subjects preferred the 'flip' transition and three felt the transition was obvious. Only one subject felt the animation

is needed to help the transition; three of the others felt the transition was too fast to be useful. Overall, we concluded that the animation cues are not needed to assist users since they understood and preferred the 'flip' approach.

One interesting side note is that the fifth subject is an avid computer game player whereas the other subjects predominantly use computers in business and programming environments. Whether this experience would cause the game player to both better understand and prefer the animation is a question for someone else, but it is an interesting conjecture none-the-less.

The fourth task presented the users with four sets of buttons, presented one at a time. These are displayed in Figure 7. The subjects looked at the buttons and were then asked to describe their purpose. They were then asked to identify the most effective one. The results are presented in Table 5.

Figure 7. Possible Rotation Buttons

Table 5. Rotation Buttons

	Subject 1	Subject 2	Subject 3	Subject 4	Subject 5
Purpose?	"Rotate room"	"Moves room around"	"Allow room rotation"	"Rotate view & room"	"Moves things around"
Best Button	2-Button	2-Button	2-Button	Compass	2-Button

All the subjects understood the purpose of the buttons without any problems. Two subjects mentioned that the Direction buttons 'Front' and 'Back' should allow them to zoom in and out or allow them to move about the room like a 'first-person' perspective in a video game such as Doom. Four of the five subjects felt the 2-Button arrangement was the most effective.

The fifth task was more of a discussion than a laid-out question. The use of 'views' to control a camera as opposed to the camera itself was explained. A screen shot of the DAN with 'views' was shown as well as a screen shot of the planned process of placing and orienting a camera in the Room Design Utility.

Three of the five subjects felt that the concept made sense and that the use of information dialog boxes would help. These dialog boxes would provide step-by-step instructions. When asked how else the concept could be implemented two alternate ideas were suggested. The first suggestion was to rotate the camera button by dragging the corners of the button to the angle of view rather than drag an arrow. This, in effect, would feel like aiming the camera. The second suggestion was to simply have the 'view' frame appear on the screen when the camera icon is released over the drawing area. The user would then manipulate the 'view' to place it where they want it.

Both suggestions have merit. Rotating the button is intuitive as long as the user understands that is what they are supposed to do. Likewise, if the user understands that the large transparent button is the camera view then moving it about the room is very intuitive. All three ideas, the original plus the two suggestions, hinge

on the user understanding that they are setting the camera's view. The second suggestion, the transparent button, is the most useful because it gets the user into the idea of manipulating the view immediately, teaches them how to manipulate camera views in future use of either the DAN or RDU and reinforces the idea of how views are represented within the DAN interface. All of this leads to a consistency throughout the interface in that learning in one program can be carried over to the other one.

The issues prototype provided many clarifications and some new ideas to apply to the full Room Design Utility and are summarized here. A large, ornate frame was added to the picture icon. The clock on the VCR icon was expanded to make it more visible. No new objects were added to describe rooms. A simple 'flip' approach would work for the room transitions. The '2-Button' (Left & Right) approach for the room rotation would be sufficient. And finally, the camera orientation would be handled by using large, transparent buttons the same as currently used in the DAN interface.

Usability Goals and Measures

Having examined the issues, it was possible to establish the usability goals and how they would be measured. Given that usability is defined as a combination of ease of use, ease of learning and usefulness, the Room Design Utility would be considered usable if the users were satisfied with it and were able to complete the tasks with little or no help or errors. These goals were operationalized as follows. For each task and

overall, 70% of users should rate ease of use as easy (2 or less) on a 5-point scale with 1 equal to very easy and 5 equal to very difficult. Fifty percent or more of the users should be able to complete the tasks without any assistance or errors and 80% should be able to complete them with no more than one hint and one error. Since users will be occasional and no training will be provided the pass criteria for no errors or hints has been set a little lower than normal at 50%. However, since the interface should be easy to use most subjects should be able to complete the tasks with few errors so that goal was set fairly high at 80%.

A hint was recorded when the experimenter provided any sort of suggestion or clue. An example of this is 'Try dragging the object around the walls' when the subject persisted in trying to drag wall objects across the room. An error was recorded when the subject became irrevocably committed to an incorrect or inappropriate action. An example is trying to drag an object out of the room or pressing the key to delete the object.

For each task, the number of hints and errors as well as a user rating was collected. Throughout the session, the subject's thoughts and opinions were recorded. After the tasks were completed, the subject provided an overall rating and rated several aspects of the interface. The subject's thoughts and opinions provided insights to determine what aspects of the interface caused problems and might suggest solutions.

It was determined that there was no need to collect any completion time information since speed isn't an issue with this interface as all users are occasional. An

upper limit on Task Completion Time was not required since users that took an excessive amount of time were also likely to make some errors and require assistance which would result in a task failure.

Design of a Functional Prototype

With the goals and measures established, the Room Design Utility prototype was built. The prototype was designed using Delphi 3 with an additional Transparent Button object. An object oriented programming methodology was used and the Transparent Button was extended as a Room Object. This means that the properties or data and the methods or behaviours of the Transparent Button object were enhanced with specific information required for a Room object. Enumerated types, discrete ordinals, were created for the walls (Left, Right, Back, Front) and the room object (i.e. Door, Window, VCR, etc.) as well as new behaviours including, for example, RoomObjectLeft, WallChange and MouseUp. The full source code for the RDU, over 1500 lines, is provided in Appendix D.

Figure 8 is an example screen of the RDU. The central object of the utility is the room image. It contains the room objects such as doors, tables and document cameras. To the left is an object palette that contains new room objects that can be placed in the room. When an object in the palette is clicked, a new copy of it is placed on the room image. It can then be moved to the desired location in the room. Above the room image are the room rotation buttons. Clicking on one of these will 'rotate' the image. Positions of objects are recalculated based on where they should appear if

the room was actually turned to the left or right. To the right of the room image is the Wizard. It displays information and helps guide the user in designing a room from scratch. It can be turned on or off by selecting it from the options menu.

Figure 8. Screen Shot of the Room Design Utility

The room image itself, was designed using a layered approach as shown in Figure 9. The layers above the Background are all transparent. The Background is a fixed bitmap of the walls of the room. The object layers, Level 1, 2 and 3, are all a single bitmap. Objects were assigned a logical level when they were created. Wall objects were assigned to Level 1 and included doors, windows, pictures and screens.

These objects 'snap' to the walls when released in the room. Furniture was assigned to Level 2 and included the two types of tables. The third level was reserved for devices that users might frequently move. These included the audience, VCR, document camera and the camera view. The purpose for this division was to control the drawing order of the objects and to assign extra behaviours to certain objects. The room objects were contained in a list but rather than present them in any order, they were inserted into the list based on their level. This helps determine the order in which they are created and hence drawn on the screen and prevents the placement of background objects on top of foreground objects. This works fine for saving and retrieving objects but still has some problems with the objects created during a current session.

Figure 9. Conceptual Layers of the Room Image

The second reason for the layering, as already mentioned, was to allow extra behaviours. Level 1 objects belong to a specific wall. When a room is rotated these objects stay with their wall and therefore rotate also. If a wall becomes invisible in the foreground position so do the objects. Level 2 and 3 objects do not belong to walls and hence, do not snap to walls nor do they become invisible. Level 2 objects should be resizable. This was not implemented for the prototype but the level was added to accommodate future development.

Heuristic Evaluation

The last stage before preparing the usability test was an informal Heuristic Evaluation. Eight of Nielsen's general rules (http://www.useit.com/papers/heuristic/heuristic_evaluation.html) were applied to the interface to help identify problems. Specifically, they were:

1. Simple natural dialog
2. Use user's language
3. Minimize cognitive load
4. Consistency / follow standards
5. Provide feedback or status
6. Provide exits
7. Aim for recognition rather than recall
8. Help and documentation

The evaluation proved useful in that it identified the following important errors. There was no general system status feedback mechanism. This was overcome by adding a status bar on the bottom of the screen. Each method provides its current status through this mechanism. Additionally, there were no exit methods available. Both a Delete and Undo function were added.

Design of the Usability Test

With a functional prototype and task analysis complete, and usability goals and measures established, enough information had been assembled to proceed with the design of the usability test. The usability test was designed to examine the two major tasks identified in the task analysis, adding and editing rooms. Seven tasks were developed; four dealing with editing room objects and three with adding rooms. The Task List is available in Appendix B. Table 6 indicates which tasks were examined by which questions in the usability test.

Table 6. Usability Test Task Mapping

Task	Test Question
Save and retrieve room	1. Load image 3. Save image
Add orientation objects	5, 6, 7. Full room design
Add and describe devices	1. Add overhead 2. Describe objects 3. Add screen, 5, 6, 7. Full room design
Move objects to different locations	1. Move overhead 5, 6, 7. Full room design
Remove objects	4. Delete VCR
Verify orientations by rotating room	3. Verify screen 5, 6, 7. All require rotation to complete

The three tasks dealing with creating a room (Tasks 5, 6, 7) increased in complexity sequentially. Task 5 required the subject to use the Room Wizard to layout the room from a sketch. Task 6 asked the subject to layout the room from a sketch without the Wizard. Task 7 required the subject to layout the room using either method from a photograph.

Order effects were controlled by reversing the order of questions for half the subjects. Odd numbered subjects were provided with the questions in the order Edit then Add (1,2,3,4,5,6,7). Even numbered subjects completed the tasks in the order Add then Edit (5,6,7,1,2,3,4). This was necessary to control for exposure to the Room Wizard (Task 5), since it explains some of the basic system concepts and, in real use, not all users can be expected to have been exposed to them. This is especially the case if the user is only editing a room.

During the usability testing subjects, completed the tasks and then provided a rating of the difficulty of the task. The measures collected to determine user satisfaction included a 5-point Likert scale and verbal questioning. To measure usefulness, the number of tasks completed, and error and hint counts and descriptions were collected.

In addition to rating tasks during the test, subjects completed brief pre- and post-session questionnaires. The pre-test questions gathered demographic information and computer experience. The post-test questions focused on the subject's overall

ratings of the Room Design Utility plus asked for ratings of specific aspects of the interface.

Pre- and post-session briefings were designed. The pre-session briefing explained the process the test follows, advised the subject they may withdraw at any time and gave the subject an overview of the Device Area Network. The post-session briefing provided an overview for what the results were to be used and gave the subject the chance to ask any final questions or raise any concerns.

Before full testing was undertaken the test underwent a pilot test with two subjects to fine tune it. This resulted in the first question being reworded to remove a large amount of confusing, technical jargon that had slipped in. The original question was worded as:

“... The current DAN for the room doesn’t have an overhead but you’ve managed to acquire one. You want to add the overhead to the DAN image. You know the image is stored in the file...”

It was reworded as:

“... The current room doesn’t have an overhead but you’ve managed to acquire one. You want to add the overhead to the room image. You know the room is stored in the file...”

Formal Usability Test

Procedure

Upon arrival, subjects were read the briefing which outlined the process to be followed, stressed the interest in evaluating the interface only and encouraged subjects to think aloud as they completed the tasks. Subjects then completed a consent form and the pre-session questionnaire, regarding their computer experience and educational level. Subjects were provided with a screen shot of the DAN and were read a short explanation of the DAN interface. This may be viewed in Appendix B.

Subjects completed the set of tasks in the order Edit/Add or Add/Edit. They were asked to think aloud as they completed the tasks. While subjects worked, they were observed and any actions of interest were noted along with any errors committed or hints provided. In addition, any comments were noted. After each task, the subject was asked to rate how easy or difficult the task was on a 5-point scale.

After the tasks were finished, subjects completed a post-session questionnaire. This asked them to provide an overall rating of the interface as well as ratings about specific issues of interest. They were also asked about any improvements that could be made to the interface.

The final step in the session was a short debriefing that provided subjects with information concerning the results of the study and a chance to ask further questions.

Results

Ten subjects participated in the usability test. Five completed it in the Edit/Add order and five in the Add/Edit order. The sessions lasted on average 45 minutes. The tests were completed on either the subjects' computer which were all Pentium level machines running either Microsoft Windows NT4 or Windows 95 or on a portable 486/40 running Windows 95.

Table 7. - Self-Rated Subject Profile (N=10)

Computer experience:	Novice	Intermediate	Expert	
	1	4	5	
Frequency of use:	Almost Never	Monthly	Weekly	Daily
	-	-	-	10
Educational Level:	Secondary School	Some post- secondary	Completed Undergraduate	Completed Post-Graduate
	-	4	4	2

Table 7 is a summary of the pre-session questionnaire. It provides an overview of the experience levels of the subjects. There is a selection of intermediate and expert users. All subjects indicated that they use computers daily. At first this seemed a little puzzling. How could you use a computer daily and be a novice. The subject explained that she only used the computer to retrieve her mail. She knew exactly how to do that and nothing else. This subject performed better on the tasks than some self-identified experts. From an education point of view, the subjects all had at least some college or university exposure.

Table 8 summarizes the usability goal results for each task. The first column identifies the tasks that the user was asked to complete. Note that Task 2, describing the room, was not rated. The next column indicates the ease of use ratings that users assigned to the task. A 70% rating of Easy or Very Easy was required for the task to pass. All the tasks passed though two were close. The three subjects who rated Task 3 as less than easy all belonged to the Edit/Add group. This was their first exposure to rotating the room. All the subjects that had been exposed to this concept already in the Add/Edit group rated this task as Easy or Very Easy. The same sort of effect occurred with the first task for the Add/Edit group. Their first task was the 'Add room using the Room Wizard' (Task 5). Two of the five rated this task as less than easy. This is an order effect. If it had not been controlled then the first task that rotated a room would have failed.

Table 8. - Usability Test Results (N=10)

Task	User Rating: Easy or Very Easy	Completed: 0 Hints / 0 Errors (a)	Completed: 1 Hint / 1 Error (b)	Completed: 0 or 1 Hint / 0 or 1 Error (a + b)	Pass /Fail
1. Edit - Open file and add an object	90%	90%	10%	100%	Pass
3. Edit - Rotate room and add object	70%	70%	10%	80%	Pass
4. Edit - Delete object	100%	50%	50%	100%	Pass
5. Add - using Room Wizard	70%	50%	40%	90%	Pass
6. Add - unaided	100%	70%	30%	100%	Pass
7. Add - using photograph	100%	90%	10%	100%	Pass

Notes: User Rating requires 70% to pass, Completion with zero hints / zero errors (a) needs 50% and Completion with zero or one hint / zero or one error (a+b) needs 80%.

To illustrate, Table 9 restates the user ratings for each task by time of exposure. User ratings on their first exposure to room rotations failed to attain the minimum required, 70%, however after repeated exposures the ratings climbed into an acceptable range. This is a good example of an order effect. By reversing the order of tasks for half of the subjects, this effect was controlled.

Table 9. - User Ratings by Exposure (Percent Rating Task as Easy or Very Easy)

Task	First Exposure	After Repeated Exposures
3	40%	100%
5	60%	80%

The third column of Table 8 identifies the number of subjects who completed the task without any errors or hints. As defined in the usability goals, a 50% was required to pass the task. All tasks passed though two were on the border. The fourth column indicates the number of subjects who completed the task with one hint or one error or both. A cumulative value 80% of columns 3 and 4 was required to pass. All tasks passed.

Table 10. - User Results - Question 2: Itemizing the Room

Number of Items Mentioned	Number of Users (N=10)	Missed Items (Number of Omissions)
5	2	Camera View(2), Picture(1), Window(1), Door(1), Table(1)
6	2	Camera View(1), Table(2), Overhead(1)
7	2	Camera View(1), Door(1)
8	4	

Table 10 presents the results from Task 2 which asked the subjects to describe the room. Subjects were asked to complete this task to see if there were any objects that caused confusion or tended to be overlooked. The room contained eight objects. The numbers in parentheses in the Missed Items column are the number of subjects who missed that item. Six of the ten subjects missed at least one object. In order of frequency the Camera View was missed by four subjects, the Table by three, the Door by two and the Overhead, Picture and Window were each missed once. No subjects mislabeled any objects.

Table 11 is a list of the 21 errors and hints recorded during the usability test organized by the Tasks in which they occurred. Every subject made at least one error and no subject made more than four errors. Table 12 summarizes the errors into broad categories to help identify the problem areas. The most common problem was reorienting after a room rotation. Subjects seemed to have difficulty distinguishing between the walls whenever the room rotated, even with objects on the walls. This accounted for a full 1/3 of the errors. The next most common problem concerned deleting objects. Users either pressed the key or tried to drag the object out of the room. Providing a quick key would probably eliminate half of these errors. The next most common error was switching the Camera View from either a horizontal position to a vertical or vertical to horizontal position. This is done when the Camera View is moved from one part of the room to another, for example, from the back of the room to the side. To make the View switch the user drags part of the View button outside the boundaries of the room image. When 25 percent of the button's height or

Table 11. - Noted Errors

Task	Description of Error or Hint
1. Place Object	Clicked on Document Camera twice before realizing it had been placed in the room
3. Rotate Room	<p>Created three screens before realizing they had been placed in the room</p> <p>Tried to delete, found undo. Ended up reloading and drawing room</p> <p>Problem understanding room rotation</p> <p>Problem moving objects, Hint - move objects along walls</p> <p>Hint - asked why not rotate when subject started to rearrange room</p>
4. Delete Object	<p>Pressed key and tried to drag object out of room</p> <p>Pressed key and tried to drag object out of room</p> <p>Pressed key</p> <p>Tried to drag object out of room</p> <p>Tried to drag object out of room</p>
5. Create Room Using Wizard	<p>Re-oriented paper to reflect room orientation after room rotation</p> <p>Problems moving camera view to side of room</p> <p>Problems moving camera view to side of room - rotated room then placed camera and rotated room back</p> <p>Hint - when using wizard and moving screen to invisible wall - told to wait a bit for rotation information to be presented</p> <p>Problems moving camera to side - Hint drag view outside of room boundary</p> <p>Hint - showed how to move camera to side view</p>
6. Create Room Without Wizard	<p>Problem understanding results of room rotation</p> <p>Problem understanding results of room rotation</p> <p>Hint - try rotating the room</p>
7. Create Room From Photograph	<p>Placed all required objects then added door for invisible wall. Rather than rotate room and place door, moved all objects to other walls and then rotated room</p>

Table 12. - Frequency of Errors (N=21)

Number of Errors	Description	Possible Solution
7	Problems understanding results of rotating room (reorientation)	Enable way of distinguishing between walls (i.e. - label walls 1,2,3,4)
6	Problems deleting objects	Enabled key
4	Problem moving camera view to side (vertical) position	Increase area where switch occurs in code
2	Problems moving wall objects	
2	Multiple clicks on object palette before realizing objects where placed	

width is outside of the room, the button 'switches' either horizontally or vertically.

The problem is that frequently the user does not drag enough of the button outside the boundary so the switch fails to occur. The forth category of errors was the problem that subjects experienced when moving wall objects. They tried to drag the objects across the room. On the back wall the objects snap to their position and therefore only allow horizontal movement. This creates the effect of sliding the objects along the walls. The last category of problems noted was multiple clicking on the object palette before realizing the object had been placed in the room. There is a slight delay between clicking on an object on the palette and it appearing in the room. It is machine dependent but generally does not exceed one second. During this time the status bar at the bottom of the screen indicates that it is creating the object. Subjects do not always notice this and believe that nothing is happening and so click on the palette again.

Table 13. - User Responses - Task 3: Wall Objects (N=10)

Question	Yes	No	If No then problem (Number of subjects)
Does Door look correct?	4	6	too short on sidewall(6), doorknob(3)
Are wall objects an adequate size?	4	6	Picture too big(2), Picture too small(2), Window too small(3), Window too low on wall(2)

As part of Task 3, subjects were questioned about the wall icons. These included the door, window, picture and screen. The results are included in Table 13. They were asked whether the door image looked correct. All of them felt it looked acceptable when on the back wall, however, six felt it was too short when displayed on a sidewall. Three felt a doorknob would make it more identifiable. When asked about the other objects the results were somewhat ambiguous. Six subjects indicated that there were some problems with the objects. Two felt that the picture was too big and two felt it was too small. Three felt the window was too small and two felt it was too low on the wall. None felt the picture was too low on the wall. All that can be said with any certainty is that the door image should be heightened when on a sidewall.

There were other problems and issues noted that were not directly raised with the subjects through questioning. These problems and issues were noted by observation and user comment. The most frequent was that six subjects tried to drag icons from the palette into the room rather than clicking on them when trying to place new objects in the room. Four subjects mentioned that the room objects did not move correctly when the room was rotated. This tends to become more apparent if there are several objects in the room and the room is rotated two or three times in the same

direction. Figure 10 displays an example of this problem. This problem lies in the fact that, for rotation purposes, the shape of the room is wider at the front than at the back. This creates a trapezoid shape while the formula to recalculate the object position is based on a square. This is a problem that needs to be corrected at a technical level and is beyond the scope of this study.

Figure 10. RDU Screen Shots Showing Before and After Four Room Rotations in One Direction.

Other comments provided by users as they worked included two who felt the Room Wizard should be available under the Help menu, two who observed that it was much faster to design a room without the Wizard, the icon palette layout was not logical, there should be better highlighting when objects are selected and there is no TV stand. The remainder of the comments are covered by the questions and ratings gathered during the test and in the post-session questionnaire.

Table 14. - Post-Session Questionnaire (N=10, 5-Point Likert)

Rating	Very Easy	Easy	Neither	Difficult	Very Difficult	Mean
Overall	4	6	-	-	-	1.6
Selecting Room Objects	8	2	-	-	-	1.2
Placing objects in the room	5	4	1	-	-	1.6
Rotating the room	4	4	1	1	-	1.9
Loading the room	9	1	-	-	-	1.1
Using the "camera view"	3	1	2	2	2	2.9

Questions	Yes		Neutral		No	Mean
Did Room Wizard help?	5	1	2	1	1	2.2
Would resizing objects help with room rotation?	2	4	2	-	2	2.6
Do room objects need more detail?	2	2	-	2	4	3.4
Would a more rectangular room help orientation?	2	-	3	1	4	3.4
Did you notice problem with background object overlap?	1	1	1	1	6	4.0

Table 14 summarizes the results from the post-session questionnaire. This provides us with user feedback on specific aspects and questions of interest. The first column indicates the aspect that the subject was asked to rate. The next five columns indicate the number of subjects who provided that rating. The last column indicates the mean value of the subject's ratings. A mean below 2.5 indicates either Easy or Yes depending on the question. A value above 3.5 indicates a Difficult or No rating. A value between 2.5 and 3.5 indicates a Neutral response.

Overall, the subjects rated the RDU easy to use which means the prototype passed the last usability goal that had been established.

When various aspects of the prototype were rated on ease of use, the prototype did very well. When asked how easy or difficult it was to select room objects, place objects or load a room image, the subjects indicated that they found it easy. Even rotating the room was rated as easy which is puzzling since it caused the most errors. An examination of when the rotation errors occurred during the tests indicates that only two of the seven occurrences happened when the subject first encountered a task requiring room rotation. That tends to discount the possibility that rotation errors occurred early in the subject's experience and that they had gotten comfortable with room rotation by the end. However, it is also noted that two of the subjects each had two rotation errors during the tasks which accounts for four of the seven errors. One of these subjects had an error on their first exposure. Between these two sets of errors five of the seven errors are accounted for. For now, the conclusion has to be that while some subjects had problems with room rotation, they didn't view the process as difficult. The only aspect that did poorly was manipulating the camera view. It received a neutral rating.

Subsequent to the rating questions in the post-session questionnaire, subjects were asked several questions concerning potential problem areas. The first questions asked if the Room Wizard helped with the room design. Six answered Yes while two answered No, resulting in a mean score of 2.2. This means that the Wizard did provide some assistance.

When asked if the ability to resize objects would help with room orientation, six of ten subjects responded Yes, however the mean score was 2.6. While this falls into the neutral category, the fact that six subjects felt the ability to resize objects

would help orientation should be accepted. This ability to resize was not included in the prototype and the test was designed to require only square or round tables except in Task 7. Task 7 had a rectangular table and it was noted that at least two subjects placed two tables together in the room to represent it. For an example, see the Figure 10 “Before” image on page 42.

To the question of whether or not more detail on room objects would help identify them, six of ten responded No with a mean of 3.4. However, the results are bimodal with four subjects answering Yes and no neutral responses. Subjects obviously have no problem forming an opinion on this topic. There is no clear answer to this question at this time and the question remains open.

When asked whether orientation would be easier if the room was more rectangular, six of ten responded No with a mean of 3.4. This indicates that changing the room on rotation from long to wide and back is not necessary. It should be noted that several subjects thought the room was already rectangular. It is possible that the subjects did not understand the implications of a more rectangular room. It would provide stronger orientation cues after a rotation. The question may need to be reworded with an explanation. Another solution may exist. If resized objects, such as rectangular tables, maintain the correct perspective on a room rotation (i.e. depth becomes width), then modifying the walls may be unnecessary. This issue should remain open until object resizing has been implemented.

The final potential problem raised was whether the subjects had noticed an overlapping of background objects over foreground objects. Seven of ten subjects answered No with a mean response of 4.0. There is a problem with the order that

objects are redrawn on the screen at this point (see Figure 10), but it didn't seem to interfere or bother the subjects during the test. It may not have been apparent since it generally takes a couple of room rotations before objects really get out of alignment and many subjects only rotated the room once to place an object. This may not be the case in later tests or in production and this problem should be fixed.

The final three questions were open ended and focused on potential enhancements for the RDU. Eight of the ten subjects responded to the question about any objects that would help room orientation. Five explicitly responded No. The three other responses were "smaller side tables", "all four walls" and a "directional indicator". It was noted in task 7 which required subjects to draw a room based on a photograph that a few placed a side table along the wall. If the objects were resizable this would not cause any sort of a problem. Another request was for all four walls to be displayed. To accommodate this, the perspective of the room would have to be shifted to above the room. This would require a redesign of both the DAN and the RDU and would probably result in a higher cognitive load when using the DAN in a presentation. It would also result in the use of labeling or standard architectural images to represent objects such as doors and windows from above. The request for the directional indicator means that there should be some sort of icon that indicates the compass direction that the room is oriented towards (North, South, etc.). While this may help some users, it may just as well confuse many others. The idea of using a compass was examined in the early prototype exercise and was abandoned in favour of using a pair of rotation buttons.

Table 15. - Suggested Improvements

Doorknobs on doors
 Raise the position of the window
 Stretching of objects (resizing)
 Highlight icons when selected
 More icons for furniture
 During room rotation know where N,S,E,W is.
 Help to indicate which wall is being viewed - a method to differentiate walls
 More help when using camera view
 A presenter icon
 Should be at least one fixed point in a room
 Dimmed (greyed) area outside of camera view when selected
 Quicker response on room rotations

Table 15 presents the suggestions that the subjects made when asked what improvements could be made to the RDU. Some of these have been dealt with already. The doorknobs and window position were discussed with Table 13. Object resizing was discussed as part of Table 14 and should be implemented. Highlighting of the icons will become more apparent when resizing is implemented since the object will have to be framed by the resize mechanism. There is no need for more furniture icons at this point. This issue was eliminated during the Issues prototype. Orientation help through wall identification and the Camera View improvements were discussed after Table 11.

During the test several users wanted to place an icon representing the presenter's location. Since none was available some used an audience marker for the representation. Consideration should be given to providing such an icon on the palette. An issue that this raises is whether the presenter remains fixed in the foreground or rotates with the room. Keeping it fixed would represent the viewer's

position in the room and would provide a fixed point of reference. Moving it along with other objects would probably confuse the user especially if the “You are here” text stays with it (see Figure 4). Using the presenter icon as another method of rotating the room is a possibility. If the user drags the presenter into another quadrant and releases it, the room could rotate so the presenter comes back to the foreground position.

Another suggestion was to dim the room area outside of the camera view when it is selected. This would certainly help in defining the view from the user’s perspective. Technically this would be difficult to do and may well slow down the redrawing of the screen as the view is moved. An alternative, however, might be to grey the view when it is selected so that it is partially opaque.

The final suggestion to provide quicker response when rotating the room is a technical problem. One solution might be to suspend redrawing of the screen until all the objects have been plotted. Currently, each object is drawn as it is plotted.

Table 16 - Other Subject's Comments

Interesting
 Interesting and straightforward
 Pretty good for a first release. Future releases could be voice activated
 Overall once we use the Room Design enough it becomes easier to use
 Very easy to use after going through it once
 Rotating objects in room was difficult
 Camera view blocks access to objects beneath it
 Wouldn't use Room Wizard unless it could be designed to help with more complicated views or designs

The last question asked subjects if they had any other comments to make regarding the RDU. Eight of them provided comments, five of which were positive. The mention of the camera view blocking objects was interesting in that several users experienced the problem but only one mentioned it. The camera view is a large button. If it is placed last or nearly last on the screen it is on top of the other room objects and they can not be accessed without moving the camera view out of the way. This problem will be addressed when the redraw order for the objects is resolved. The last comment about the Wizard helping with complex designs runs contrary to the general purpose of a wizard. Wizards traditionally are used to help a novice user understand the general processes of an interface. Trying to increase its ability to guide complex designs may make it unwieldy for a novice user.

Discussion

All the usability goals were met during the usability test. All individual tasks and the overall assessment were rated as easy or very easy by at least 70% of the

subjects. Over 50% of the subjects completed the tasks without any errors or hints and 80% did so with only one error and one hint or without errors or hints.

While all the goals were met, there were some re-occurring problems identified. Two thirds of the subjects that failed to mention one or more items when describing the room, missed the camera view. One third of all errors made involved rotating the room. Six of the ten users had some minor difficulty deleting an object. Four users displayed difficulty manipulating the camera view plus it only received a neutral rating for ease of use.

It was decided that the usability of the RDU could be increased if these main problems could be addressed. If the visibility of the camera view was increased, for instance, the likelihood of it being overlooked would decrease. To this end it was decided to put a camera icon in the middle of the transparent view button.

Likewise, several subjects had mentioned they had problems determining which wall was which especially after a room rotation. If individual walls could be differentiated it might be easier. How the walls were differentiated didn't matter as long as they were. It was decided to number the walls and track the numbering as the room rotated.

Three of the six subjects that had problems deleting objects had tried pressing the as their first action. Adding the as a quick key would eliminate half of these errors.

Manipulating the camera view required that a quarter of the view button be dragged outside the room boundary before it switched from a vertical to a horizontal or a horizontal to a vertical orientation. Much of the difficulty that the subjects seem

to have was not in knowing what to do but in not dragging the 'view' far enough outside the room. Most of this problem could be eliminated by increasing the sensitivity of the view button. To do this moving a sixth of the button outside the room should trigger the switch.

It was hoped that making these four changes would reduce the number of errors recorded and increase the user ratings for the tasks involved.

There were other changes indicated that need to be addressed. The door icon when on a sidewall needs to be taller and doorknobs could be added. Windows need to be higher on the walls. Some objects, such as the tables and perhaps the windows and camera views, need to be resizable. The order in which objects are redrawn needs to be resolved. Objects on a background layer need to be drawn before objects are drawn on the foreground. The possibility of showing a presenter needs to be examined. It was originally thought that this would only be needed in the DAN but it seems that it may be useful in the RDU. The room rotation functions need to be corrected for positioning and optimized for speed. A drag-and-drop feature for the object palette should be enabled. None of these problems were addressed in the remaining tests though in hindsight some of them should have been. At the time, they were viewed as either cosmetic or part of other technical corrections. For example, fixing the object redraw order will eliminate object overlaps and the problem with the Camera View covering other objects. However, they were identified to be addressed before the RDU went into production.

To test the four changes made it was decided to run another set of tests. Two sets of tests were performed, a short test and a full test. The short test would involve

previous subjects and would only test the changes to see if the subjects felt the changes were an improvement based on their prior exposure. The full test would involve using the complete usability test on novice users to ensure adequate exposure to the interface. The full test would be carried out if the short test indicated an improvement in the interface.

Short Test of Prototype Changes

Procedure

The short test was designed to examine only the changes that had been made. It consisted of three tasks, 1) rotate the room and place an object, 2) delete an object and 3) move a camera view. Subjects were to be asked simple yes or no questions and their performance was to be measured in the same manner as Test 2, namely, errors and hints were to be noted. The Task List is available in Appendix C.

Results

Five subjects who had participated in the previous usability test, participated in the short test. The session lasted less than 15 minutes. The test was completed on either the subject's own computer which was a Pentium level machine running either Windows NT4 or Windows 95 or on a portable 486/40 running Windows 95.

Table 17. - User Performance (N=5)

Task	0 Errors or Hints	1 Error or Hint
1. Rotate room & add object	5	0
2. Delete object	4	1 ^a
3. Move camera view	4	1 ^b

Notes: a - Error: User tried to drag object out of room

b - Hint: Explained camera as opposed to a camera view

Table 17 summarizes the results of the short test. As can be seen subjects experienced little or no difficulty in completing the tasks. They did however have to pause at the start of the test to remember the interface and in one instance needed an explanation of camera views again. Since it had only been a week since their previous exposure, this does raise a question regarding memory retention of interface information. Perhaps that could be a thesis topic.

Table 18. - User Ratings (N=5)

Question	Yes	No
1. a) Rotation -Was it easier to orient your perspective within the room?	4	1
b) Rotation -Did the numbering on the walls help?	4	1
2. a) Delete -Did subject note key was now available?	3	2
b) Delete -Will the key make deleting easier?	5	0
3. a) Move View -Did the camera view seem easier to switch than before?	5	0
b) Move View - Did the camera icon help identify the view?	5	0

The subject's responses to the task questions are summarized in table 18.

They are generally positive and indicate that the changes made help the interface.

Four out five subjects felt that the room orientation was easier and that the numbering on the walls helped.

In the Delete task the subject was not asked whether the key was available. It was noted either by them using it or mentioning it when selecting 'Delete Object' from the menu. The three subjects that had pressed the key in the original test were included in this test session. They all either noted the quick key or used it. When asked if having the key enabled would help delete objects, they agreed unanimously.

All the subjects found the camera view easier to identify and work with. In the original design the view was a transparent button with only the edges containing any colour to identify it as a standard 3D button. The new button design added a video camera image to the middle of the button (see Figure 8 on page 29). The View button was made more sensitive to changing its orientation by reducing the amount it had to be moved outside the room boundary from a quarter of its width or height to a sixth.

Discussion

The results of the short test indicate that the changes made to the interface improved it. Therefore, validation of the improvements was sought by rerunning the original usability test with new subjects.

Second Iteration of Usability Test

Procedure

The same process that was used for the original usability test (Test 2, page 34) was followed for this iteration.

Results

Five subjects participated in the usability test. Three completed it in the Edit/Add order and two in the Add/Edit order. Like the first test the sessions lasted about 45 minutes. The test was completed on a portable 486/40 running Windows 95.

Table 19. Self-Rated Subject Profile

Computer experience:	Novice	Intermediate	Expert		
Test 2 (N=10)	1	4	5		
Test 4 (N=5)	3	1	1		
Total (N=15)	4	5	6		
Frequency of use:	Almost Never	Monthly	Weekly	Daily	
Test 2 (N=10)	-	-	-	10	
Test 4 (N=5)	1	1	-	3	
Total (N=15)	1	1	-	13	
Educational Level:	Secondary School	Some post- secondary	Completed Undergraduate	Completed Post-Graduate	
Test 2 (N=10)	-	4	4	2	
Test 4 (N=5)	2	3	-	-	
Total (N=15)	2	7	4	2	

Table 19 shows a summary of the pre-session questionnaire. It also includes the data from the original test as a comparison. It shows that the second group is more inexperienced and has a lower educational level than the first group. When the

Total column is examined we can see that we now have the full spectrum of experience represented along with a good selection of educational levels.

Table 20 summarizes the usability goal results. The results from the original usability test have been included under the Test 2 columns to allow easy comparison between the two iterations. All tasks exceeded the 70% required to pass the User Ratings. They also exceeded the performance requirements needed for 'Completed 0 Hints / 0 Errors' and 'Completed 0 or 1 Error and 0 or 1 Hint'. In addition, the second iteration, Test 4, posted better results and ratings than the first in all tasks where a ceiling effect did not preclude it.

Table 20. - Usability Test Results (Test 2 (N=10) and Test 4 (N=5))

Task	Test	User Rating: Easy or Very Easy		Completed: 0 Hints / 0 Errors (a)		Completed: 1 Hint / 1 Error (b)		Completed: 0 or 1 Hint / 0 or 1 Error (a + b)		Pass /Fail
		2	4	2	4	2	4	2	4	
1. Edit - Open file and add an object		90	100	90	100	10	0	100	100	Pass
3. Edit - Rotate room and add object		70	80	70	80	10	20	80	100	Pass
4. Edit - Delete object		100	100	50	60	50	40	100	100	Pass
5. Add - using Room Wizard		70	100	50	60	40	40	90	100	Pass
6. Add - unaided		100	100	70	100	30	0	100	100	Pass
7. Add - using photograph		100	100	90	100	10	0	100	100	Pass

Notes: User Rating requires 70% to pass, Completion with zero hints / zero errors (a) needs 50% and Completion with zero or one hint / zero or one error (a+b) needs 80%.

Table 21. - Results - Question 2: Itemizing the Room (Test 2 (N=10) and Test 4 (N=5))

Number of Items Mentioned	Number of Users Test 2/Test 4	Missed Items - Test 4 Only
5	2 / 0	
6	2 / 0	
7	2 / 1	Camera View(1)
8	4 / 4	

Table 21 displays the results from Task 2, describing the room. The second column indicates the performance of both the first and second iterations. Overall, subjects in the second iteration did better at itemizing the room. Only one subject missed an item as opposed to six in the first. Unfortunately the subject failed to mention the camera view.

Table 22. - Noted Errors During Test 4

Task	Description of Error or Hint
3. Rotate Room	Hint: Try dragging objects along walls
4. Delete Object	Accidentally deleted camera view, not VCR Created extra VCR when trying to delete it
5. Create Room Using Wizard	Created two screens before noticing it in the room Created extra view while trying to figure out how to adjust view

Table 23. - Frequency of Errors (Test 2 (N=21) and Test 4 (N=5))

Test 2	Test 4	Description
7	-	Problems with rotating room
6	2	Problems deleting objects
4	1	Problem moving camera view to side (vertical) position
2	1	Problems moving wall objects
2	1	Multiple clicks on object palette before realizing objects where being placed

Table 22 is a list of the five errors and hints recorded during the second iteration. Table 23 summarizes the errors against the categories established during the original test. In contrast to the first test the number of errors is down substantially. The original test averaged 2.1 errors per user. The second test only averaged one error per user. What is even more important is that the modifications to the interface have reduced the related errors in the second test. There were no rotation errors. The problems deleting objects are greatly reduced and only one error was noted with moving camera views.

Table 24. - User Results - Question 3: Wall Objects (N=5)

Question	Yes	No	No - problem
Door look correct?	4	1	too short on sidewall, doorknob
Wall objects an adequate size?	1	4	Window too small(3), Window too low on wall(1)

During Task 3 some questions were asked about the wall icons. The results are in Table 24. These are similar results to the original iteration except that a larger percentage felt that the window icon is too small.

Table 25. - Post-Session Questionnaire (Test 2 (N=10) & Test 4 (N=5), 5-Point Likert)

Rating	Test	Very Easy	Easy	Neither	Difficult	Very Difficult	Mean
Overall	2	4	6	-	-	-	1.6
	4	4	1	-	-	-	1.2
Selecting Room Objects	2	8	2	-	-	-	1.2
	4	5	-	-	-	-	1.0
Placing objects in the room	2	5	4	1	-	-	1.6
	4	3	2	-	-	-	1.4
Rotating the room	2	4	4	1	1	-	1.9
	4	4	1	-	-	-	1.2
Loading the room	2	9	1	-	-	-	1.1
	4	4	1	-	-	-	1.2
Using the "camera view"	2	3	1	2	2	2	2.9
	4	3	1	1	-	-	1.6

Questions		Yes		Neutral		No	Mean
Did Room Wizard help?	2	5	1	2	1	1	2.2
	4	3	1	1	-	-	1.6
Would resizing objects help with room rotation?	2	2	4	2	-	2	2.6
	4	2	1	-	1	1	2.6
Do room objects need more detail?	2	2	2	-	2	4	3.4
	4	1	-	-	2	2	3.8
Would a more rectangular room help orientation?	2	2	-	3	1	4	3.4
	4	-	-	1	2	2	4.2
Did you notice problem with background object overlap?	2	1	1	1	1	6	4.0
	4	2	1	1	-	1	2.4

Table 25 summarizes the results from the post-session questionnaire as Test 4. The results from the original usability test have been included as Test 2 for comparison purposes. As before, the first column indicates the aspect that the subject was asked to rate. The next five columns indicate the number of subjects that provided that rating. The last column indicates the mean value of the subject's ratings. A mean below 2.5 indicates either Easy or Yes depending on the question. A value above 3.5 indicates a Difficult or No rating. A value between 2.5 and 3.5 indicates a Neutral response.

Overall, the subjects rated the RDU easy to use which means the prototype again passed the last usability goal that had been established.

All the various questions rating ease of use of the prototype did very well. The first four ratings, selecting room objects, placing objects, rotating the room and loading a room image, were all rated as easy or very easy. Even using the camera view, which received a neutral rating in the original test, received an easy rating from four of the five subjects with a mean of 1.6.

The next set of questions dealt with potential problem areas. When asked if the Room Wizard helped with the room design, four of the five said Yes with a mean score of 1.6. This is a little stronger than the first test which had a mean of 2.2 and it supports the original conclusion. This means that the Wizard did provide some assistance.

When asked if the ability to resize objects would help with room orientation, three of the five subjects responded Yes, however the mean score was 2.6. This falls

into the neutral category and is identical to the original test result. So the result has to be interpreted again as supporting the ability to resize objects.

To the question of whether or not more detail on room objects would help identify them, four subjects responded No with a mean of 3.8. This falls into the No category which is a more extreme answer than in the original and indicates that additional object details are not necessary.

When asked whether orientation would be easier if the room was more rectangular, four of the five responded No with a mean of 4.2. This again indicates that changing the room on rotation from long to wide and back is not necessary. However, as mentioned after Test 2 this is an issue that should be revisited after object resizing has been implemented.

The final potential problem raised was whether the subjects had noticed an overlapping of background objects over foreground objects. Three subjects answered Yes with a mean response of 2.4. This is a reversal of the original results which had a mean of 4.0. Why these subjects noticed the overlap problem when the original subjects did not is unclear.

The final three questions were open ended and focused on potential enhancements for the RDU. Four of the five subjects responded to the question about any objects that would help room orientation. Three explicitly responded No. The other subject asked for a TV icon and a Computer icon.

There were only two suggestions made when asked what improvements could be made to the RDU. The first was that objects should not overlap. This has already

been identified. The second was “Colour Changes.” I’m not sure what this means and can not expand upon it in any way.

Only one subject responded to the last question asking for any additional comments. The comment was “Very easy and simple to utilize”.

Discussion

Again all the tasks passed the usability goals established for them. Subject performance and ratings were better during the second iteration indicating that the changes made did have a positive effect. The number of errors per user was halved. The camera view is more visible and easier to use. Orientation during and after a room rotation is easier. Deleting objects, while still not perfect, is easier to accomplish.

The other problems noted after the first test still remain. Added to that list is the problem that the Window object appears to be too small.

While the results between the Short Test using five repeat subjects and the second iteration of the usability test with five new subjects, are generally not directly comparable, there were two tasks that are identical. Task 1 in the Short Test and Task 3 in the regular test both require the user to rotate the room and place an object. All the repeat users, 100%, completed it without any errors while 80% of the new users completed it without errors. Similarly, Task 2 in the Short Test and Task 4 in the regular test required the users to delete an object. Eighty percent of the repeat users completed it without any errors while 60% of the new users did so. This shows that,

as one would expect, experienced users outperform novice users. It also tells us that there is a learning or practice aspect to the interface and that in any future rating tests involving repeat users, the required pass level should be increased.

General Discussion

This study provided several important results. It has demonstrated that the Room Design Utility is a usable product. The RDU consistently met the usability goals that were established at the outset. Users rated it as easy to use both overall and on each of the individual tasks that they completed. The interface demonstrated its effectiveness in that the users were able to complete typical tasks that can be expected with the utility with an acceptable level of errors.

Another result of this study is the Room Design Utility itself. As it stands, it is fully functional and could be used to design rooms for the DAN.

Finally, the study yielded, through the User Centered Design process, a collection of results that can be used to further improve the RDU. During the development of the RDU, after the issues prototype and before the design of the usability test, some new issues were identified.

The order in which objects are redrawn on the screen was noted as a technical problem. Objects on background layers should be drawn before objects are drawn on the foreground layers. Solving this will eliminate the effect of background objects overlapping foreground objects.

The ability for the user to resize objects was identified as a potential problem. Some objects, such as the tables and perhaps the windows and camera views, should be resizable by the user. This will allow them to better layout the room image to match the actual room.

Finally, the display of room objects when on a sidewall, was noted as another possible problem. The wall objects, such as doors and windows, did not appear the correct size or in the correct location when presented on an angle on the sidewalls. The door icon when on a sidewall should be taller and a doorknob added. The window icon should be larger and higher on the wall.

It was decided that confirmation for these problems would be sought from the usability test rather than by further refinement of the RDU. To this end, questions were added to the test to get feedback from the users whether or not these problems interfered with the use of the utility. Confirmation of the redraw issue was not indicated until after the second iteration. The other two issues were confirmed after the first iteration but were not deemed to be major detractors from the usability of the utility and were put aside to be addressed later.

Observations during the first iteration of the usability test revealed three new issues. The first was that there was a technical problem correctly positioning objects during room rotations. This was thought, rightly or wrongly, to be more of an irritation than a problem for the RDU and so it was left to be corrected at a later time.

The second issue was that a presenter icon should be available. This raised some new issues regarding the role such an icon would play. It could be representational only, either rotating with the room or remaining fixed in the foreground. Keeping it fixed would represent the viewer's position and would provide a fixed reference point. Moving it along with other objects may confuse the user especially if the "You are here" text stays with it. Alternatively it could actively be used to aid room rotation by being dragged and released within the room. Upon

release, the room would rotate so the presenter icon would return to the foreground position.

The third issue was that new icons should be selectable from the object palette in a drag-and-drop fashion. The original design places the new object in the room when it is clicked on the palette. Several users tried dragging objects into the room.

Neither of these two issues were addressed in the second iteration because they were not detected until after it. It was during a second review of the results that these two issues became apparent.

Of the six new problems identified only two require further analysis and testing. These are resizing of room objects and the role of the presenter icon.

The other four problems can be corrected without further testing. These include the object redraw order, the positioning of objects on a room rotation, the drag-and-drop feature and the icon size and position on the sidewalls.

On the whole, UCD is a robust approach that yields very high dividends for the effort involved. One error that was performed in this study was insufficient analysis of the observations after the first iteration. Future studies should ensure that a formal write up is completed rather than just a tabulation of the data.

This study demonstrated the impact order effects can have on the results of usability testing. Had this not been controlled by reversing the task order for half of the subjects, the first task exposing the subjects to room rotations would have failed. By controlling for this, the first exposure was spread over two tasks resulting in a more fair evaluation of the tasks.

Conclusion

The RDU will enable the DAN system to be deployed in other sites and facilities. It will be used as a separate product to design and edit room layouts. Portions of its code and interface will be incorporated into the DAN to load room images and provide the ability to rotate rooms.

Through the use of a User Centered Design methodology, this study has developed a fully functional, usable Room Design Utility. The interface has been validated through a series of formal usability tests. The use of the UCD methodology allowed problems to be identified and overcome in an incremental fashion. It is recommended that further development efforts use this design methodology to guide them.

References

- Baecker, R. & Buxton, W. (1987). Design Principles & Methodologies. In R. Baecker & W. Buxton (Eds.), Readings in Human-Computer Interaction, A Multidisciplinary Approach (pp. 483-489). San Mateo, CA: Morgan Kaufmann Publishers Inc.
- Booth, P. (1989). An Introduction to Human-Computer Interaction. London: Lawrence Erlbaum Associates.
- Dillon, R. (1998). Telelearning Research Network. From the World Wide Web: <http://www.carleton.ca/~ddillon/cure/telelearn.htm>, 1998
- Gould, J. & Lewis, C. (1985). Designing for Usability: Key Principles and What Designers Think, In R. Baecker & W. Buxton (Eds.), Readings in Human-Computer Interaction, A Multidisciplinary Approach (pp. 528-539). San Mateo, CA: Morgan Kaufmann Publishers Inc.
- Information Technology Research Centre - Telecommunications Research Institute of Ontario (ITRC-TRIO). (1991, March). Telepresence Project Detailed Project Proposal. Ontario: (ITRC-TRIO).
- Narasimham, P. (1993). The Design of an Integrated Telepresence Applications Graphical User Interface. Unpublished thesis, Carleton University, Ottawa, Ontario.
- Nielsen, J. (1994). Guerrilla HCI: Using Discount Usability Engineering to Penetrate the Intimidation Barrier. In R. Bias & D. Mayhew (Eds.). Cost-Justifying Usability. Boston: Academic Press.
- Nielsen J. (n.d., Retrieved May 15, 1998). How to Conduct a Heuristic Evaluation. From the World Wide Web: http://www.useit.com/papers/heuristic/heuristic_evaluation.html
- Norman, D. (1983) Design Principles for Human Computer Interfaces. In R. Baecker & W. Buxton (Eds.), Readings in Human-Computer Interaction, A Multidisciplinary Approach (pp. 492-501). San Mateo, CA: Morgan Kaufmann Publishers Inc.

Norman, D. (1988). The Psychology of Everyday Things. USA: BasicBooks, Harper Collins.

Riesenbach, R. (1995, March). Ontario Telepresence Project Final Report. Ontario: Information Technology Research Centre - Telecommunications Research Institute of Ontario (ITRC-TRIO).

Scherba, T. (1993). User Centered Design of an Interface to Control Multimedia Devices. Unpublished thesis, Carleton University, Ottawa, Ontario.

Appendix A - Issues Prototype Questions

RDU Prototype

14/08/98

The CURE lab at Carleton University is working on a program to control audio visual devices via the computer called the DAN. (Show DAN screens).

One of the things that is lacking is the ability to change perspectives within a room or to change rooms. The research I'm working on will allow these limitations to be overcome. I'm building a utility that allows a user to design a room.

To do that, some preliminary questions need to be answered. That is where this session comes in. I'm going to ask you some questions. The answers will help guide me when designing the program. There are no right or wrong answers. I am not testing you in any way. I only want to see if some ideas are likely to work.

This whole session should not take more than 20 minutes.

Do you have any questions?

1. Please name each icon in the palette on the screen.

2. One of the important concepts of the program being built at Carleton is that room pictures only need a minimal amount of information to allow a person to orient the picture to the actual room. Major objects are all that are needed. The picture below shows this. Only the essential information that is required is presented.

(include one of the RDU screens)

Keeping this minimal information requirement in mind, please describe the important elements of the following pictures that would be needed to orient yourself within the room.

(Present each picture in turn (2-3) and note the nouns used to describe each.

Indicate the order mentioned:

Blackboard/Screen	_____	_____	_____
Chair/Audience	_____	_____	_____
Door	_____	_____	_____
Table	_____	_____	_____
VCR	_____	_____	_____
Window	_____	_____	_____
Other	_____	_____	_____
Other	_____	_____	_____
Other	_____	_____	_____
Other	_____	_____	_____
Other	_____	_____	_____
Other	_____	_____	_____

(Note any discussion about lack of correspondence to reality.)

3. When ready click the button labelled 'Click Me'.

What happened? (They should mention something about the image of the room having rotated. This will be the simple flip approach)

Go to the next screen and when ready click the 'Click Me'. (This will contain transition cues). Describe what happened. (They should definitely mention that the room changed.)

On a scale with the following ratings, indicate your preference:

Very much prefer the first transition 1	Prefer the first transition 2	Slightly prefer the first transition 3	No preference 4	Slightly prefer the second transition 5	Prefer the second transition 6	Very much prefer the second transition 7

Was the transition obvious in the first example? Are the transition cues needed?

4. Given the following buttons in the top, right corner, what do they do?

Which one was most effective?

Compass	_____
Direction	_____
2 Button	_____
Rotate Button	_____

5. When a room is designed most of the devices appear on the screen. In the final program the user will be able to click on the devices to control them. The exception to this is the video camera. The camera is controlled by modifying its view. The view is represented on the screen by a large, transparent button. (Show typical DAN shot.)

In the RDU, when a camera is placed on the screen a short dialog will determine what part of the room it will be aimed at. This will include the user dragging and pointing the direction arrow and then selecting the OK button.

Once this is completed the camera will disappear and be replaced by a large transparent button like the DAN. Further modification of the camera will be done by clicking on its view.

Does this concept make sense? Will dialogs help get the point across? How else can we explain this concept to the room designer?

Appendix B - Formal User Test - Tasks and Questions

Session Script

1. Read session briefing to subject
2. Have subject complete Consent Form
3. Have subject complete Pre-Session Questionnaire
4. Read DAN/RDU Overview (Show DANShot)
5. If odd numbered subject provide them with Tasks - Odd Number
Else use Tasks - Even Number
Record results on Task List - Master form
6. Have subject complete Post-Session Questionnaire
7. Read debriefing note.

===== New Page =====

Session Briefing & Consent Form

The Computer User Research & Evaluation lab (CURE) at Carleton University has designed a prototype for a Room Design Utility to be used with a bigger program known as the Device Area Network or DAN. I'll explain more about it later.

Because this is a prototype, some parts of the interface may not be in their final form.

During this session, which should be less than one hour, you will be asked to perform several tasks using the Room Design Utility. I am interested in your impressions, so feel free to ask questions and make comments both positive and negative, throughout the tasks. It is important to note that we are not testing you. Rather we are evaluating the prototype. Also, in order to help me understand your experience, please say out loud what you are thinking as you complete each task.

At certain points during the evaluation, I will stop you to ask your opinion. Please remember that I am evaluating the utility and not you. Your comments will help me evaluate and improve the design.

Thank you for your participation. Do you have any questions?

Informed Consent Form - Room Design Utility

The purpose of an informed consent form is to ensure that you understand the reason for a study and the nature of your involvement. The informed consent must provide sufficient information such that you have the opportunity to determine whether you wish to participate in the study.

Research Personnel: The following people are involved in this research project and may be contacted during normal business hours: Brian Paynter (principle investigator) at 258-2079 and Dr. R.F. Dillon (faculty advisor) at 520-6629.

Purpose and Task Requirements: The purpose of the study is to evaluate the usability of the Room Design Utility interface. You will be asked to complete two short questionnaires and a set of tasks using a personal computer and the Room Design Utility. Problems completing the tasks and the your opinions will be collected.

Duration: The evaluation session should last less than one hour.

Potential Risk/Discomfort, Anonymity, Right to Withdraw: There are no potential physical or psychological risks. The data collected in this session is confidential and your name will not be associated with it in any way. You have the right to withdraw from the session at any time without prejudice.

I have read the above description of the Room Design Utility study and understand the conditions of my participation. My signature indicates that I agree to participate in the study.

Participant's Name: _____

Participant's Signature: _____

Researcher's Name: Brian Paynter

Researcher's Signature: _____

Date: _____

Pre-Session Questionnaire

Subject#: _____

In order to tailor our design we need a little bit of demographic and experience information. Please complete the following questions.

1. How do you rate your computer experience?

Novice / Intermediate / Expert

2. How often do you use a computer?

Almost never / Monthly / Weekly / Daily

3. What is your educational level?

Secondary School	_____
Some University or College	_____
Completed University or College	_____
Completed Post-Graduate Studies	_____

DAN/RDU Overview

The CURE lab at Carleton University is working on a program to control audio visual devices via the computer. This program is called the DAN or Device Area Network.

(Show DANShot)

This is a picture of a typical DAN control screen. One of the important concepts of a DAN screen is the abstraction of room details. In the picture of the room, unimportant details of the room have been eliminated. Only enough room details to allow a person to orient themselves are included. For example, the window and door.

Devices that a user can control appear as transparent buttons. In this picture we can see an overhead, a computer and a VCR tape. There is also a "camera view" which is indicated by the large transparent button on the top half of the picture. When the user clicks on one of these devices the bottom part of the screen displays the controls appropriate to the device. In this example, the buttons to control the camera view are displayed. If the VCR was picked we would see a standard set of VCR controls.

One of the things that is lacking in the current DAN is the ability to change perspectives within a room or to change rooms. The Room Design Utility will overcome these limitations.

Task List - Master Copy**Subject #:** _____**Odd:** 1234567**Even:** 5671234

1. You are going to do a presentation in the boardroom. For this you need an overhead projector. The current room doesn't have an overhead but you've managed to acquire one. You want to add the overhead to the room image.

You know the room is stored in the file ROOM222.RDD. Get the room and add an overhead button on the table in the room.

Steps: Select File | Open... & open ROOM222.RDD
Click on Doc Camera button and drag to table.
Select File | Save... & save.

Please indicate how easy or difficult it was to complete this task.

Very Easy		Neither		Very Difficult
1	2	3	4	5

Do you have any comments or suggestions?

Errors or special actions: _____

Hint: Yes / No

Describe Hint _____

Comment or Thoughts : _____

2. Please describe Room 222.

Objects: Picture _____
 Window _____
 Door _____
 Square Table _____
 3 Audience _____
 VCR _____
 Camera _____
 Doc Camera _____

Comment or Thoughts : _____

New Page

3. After you saved your new room, it occurred to you that the person presenting immediately after you will be doing so from the left side of the room. Since he is from out of town and will be arriving just before the meeting, you decide to help him out by ensuring that the room will look correct from the left side. There should be a projector screen on the wall behind you. If it is not there, please add one.

Steps: Click on Rotate Left Button (1 or 2 times)
 Click on Screen button and locate on wall.
 Select File | Save... & save.

Please indicate how easy or difficult it was to complete this task.

Very Easy		Neither		Very Difficult
1	2	3	4	5

Does the door look correct on the wall?

Are the wall objects (picture, window, door) an adequate size?

Do you have any comments or suggestions?

Errors or special actions: _____

Hint: Yes / No
 Describe Hint

Comment or Thoughts : _____

4. The VCR in Room 222 is no longer available. Remove it from the room.

Steps: Click on VCR

Select Options | Delete Object

Please indicate how easy or difficult it was to complete this task.

Very Easy

Neither

Very Difficult

1

2

3

4

5

Errors or special actions:

Hint: Yes / No

Describe Hint

Comment or Thoughts :

5. Your unit has just received a new DAN system for your boardroom. Just as the technician finished setting up the hardware, she was called away to attend to some problem the company president was having. Your boss wants you to show off the new system at the staff meeting in an hour. All that is left to do is design the room. Luckily, the technician had made a sketch of the room elements before she left. With her parting words, 'Use the room wizard', still ringing in your ears, you decide to forge ahead.

Steps: Select Options | Room Wizard
 Follow wizard.
 Select File | Save... & save as Room 223.

Please indicate how easy or difficult it was to complete this task.

Very Easy		Neither		Very Difficult
1	2	3	4	5

Do you have any comments or suggestions?

Errors or special actions: _____

Hint: Yes / No

Describe Hint _____

Comment or Thoughts : _____

6. Having finished Room 223 using the wizard and with some time to spare before the staff meeting, you decide you'd like to do another room without the wizard. Room 224 will be getting a DAN shortly so you decide you'd like to draw it. You rough out a sketch and off you go.

Steps: From top to bottom place

Door
Door
Window
Rotate room - screen - rotate room
Table
Audience
Audience
Audience
Overhead

Please indicate how easy or difficult it was to complete this task.

Very Easy		Neither		Very Difficult
1	2	3	4	5

Do you have any comments or suggestions?

Errors or special actions: _____

Hint: Yes / No

Describe Hint

Comment or Thoughts :

7. As you finish off Room 224, a coworker stops by to let you know the staff meeting is delayed for half an hour. Having some time to kill, you decide to draw up the room you're now sitting in, Room 225. Other than the door behind you, the room appears as you now see it in the picture.

Steps: Select File | New

From top to bottom place

Window

Screen

Screen or picture

Rotate room - door - rotate room

Table

Audience

Audience

VCR

Select File | Save... & save.

Please indicate how easy or difficult it was to complete this task.

Very Easy

Neither

Very Difficult

1

2

3

4

5

Do you have any comments or suggestions?

Errors or special actions: _____

Hint: Yes / No

Describe Hint

Comment or Thoughts :

Post-Session Questionnaire

Subject #: _____

Now that you've finished the tasks there are just a few more questions to answer.

1. Overall, how easy or difficult did you find the Room Design Utility to use?

Very Easy		Neither		Very Difficult
1	2	Easy or Difficult	4	5

2. Please rate the following aspects of the interface:

	Very Easy		Neither Easy or Difficult		Very Difficult
a) Selecting Room Objects	1	2	3	4	5
b) Placing Objects in the Room	1	2	3	4	5
c) Rotating the Room	1	2	3	4	5
d) Loading/Saving the Room	1	2	3	4	5
e) Using the "camera view"	1	2	3	4	5

	Yes		Neutral		No
3. Did the Room Wizard help with designing the room?	1	2	3	4	5
4. Would the ability to resize objects like the tables and windows help with the room orientation, especially after a room rotation?	1	2	3	4	5
5. Would more detail on the room objects, such as doorknobs on doors, help in understanding them?	1	2	3	4	5
6. Would orientation be easier if the room was rectangular rather than square?	1	2	3	4	5
7. Did you notice a problem with background objects (e.g. door) overlapping foreground objects (eg.table)?	1	2	3	4	5

8. Are there other objects that would help with room orientation?

9. What improvements, if any, could be made to the Room Design Utility?

10. Are there any other comments you would like to make about the Room Design Utility?

Debriefing

Thank you for participating in this study. The results should be available by mid-December or early January. If you are interested in receiving them please contact me at 258-2079 or at bpaynter@comnet.ca.

Are there any questions or comments you'd like to make?

Appendix C - Short Test - Tasks

Task List - Master Test #3**Subject #:** _____

1. Since last week someone accidentally deleted the new room file for ROOM222.RDD. However, you have the original. Re-add the projector screen on the wall behind you.

Steps: Click on Rotate Left Button (1 or 2 times)
Click on Screen button and locate on wall.

Was it easier to orient your perspective within the room?

Did the numbering on the walls help?

Errors or special actions: _____

Hint: Yes / No
Describe Hint

Comment or Thoughts : _____

2. The VCR in Room 222 is no longer available. Remove it from the room.

Steps: Click on VCR

Select Options | Delete Object or key.

Did the subject note that the Del key was now available (Observe - don't ask)?

Will the key make this task easier in the future?

Errors or special actions: _____

Hint: Yes / No

Describe Hint

Comment or Thoughts :

3. The default camera view has been rearranged to center on the door. Please change the camera view on the computer to capture the left side of the room.

Did the camera view seem easier to switch than it did before?

Did the camera icon help identify the view?

Errors or special actions: _____

Hint: Yes / No

Describe Hint

Comment or Thoughts :

Appendix D - Room Design Utility Source Code

There are two main units to the code, rdumain and roomobj. RDUMain contains the main program for the RDU. RoomObj contains the Room Object class definition. There are some other minor units that have not been included here.

Unit RDUMain

```
unit rdumain;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
  Dialogs, Menus, ExtCtrls, StdCtrls, ERombBtn, Grids, ComCtrls,
  ToolWin, printers, Buttons, TransBtn, RoomObj;

type
  TForm1 = class(TForm)
    MainMenu1: TMainMenu;
    File1: TMenuItem;
    New1: TMenuItem;
    Open1: TMenuItem;
    Save1: TMenuItem;
    N1: TMenuItem;
    Exit1: TMenuItem;
    Options1: TMenuItem;
    RoomWizard1: TMenuItem;
    Help1: TMenuItem;
    About1: TMenuItem;
    PHelpStatus: TPanel;
    PImagePanel: TPanel;
    Image1: TImage;
    ToolBar1: TToolBar;
    TBDoor: TToolButton;
    ImageList1: TImageList;
    TBWindow: TToolButton;
    TBScreen: TToolButton;
    TBPicture: TToolButton;
    TBRTTable: TToolButton;
    TBSTable: TToolButton;
    TBPeople: TToolButton;
    TBDoccam: TToolButton;
    TBVCR: TToolButton;
    TBCamera: TToolButton;
    pbLeft: TBitBtn;
    pbRight: TBitBtn;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
```

```

    pCanvas: TPaintBox;
    DoorImages: TImageList;
    WindowImages: TImageList;
    PictureImages: TImageList;
    ScreenImages: TImageList;
    TableImages: TImageList;
    ButtonImages: TImageList;
    SaveDialog1: TSaveDialog;
    OpenDialog1: TOpenDialog;
    pWizard: TPanel;
    txWTitle: TLabel;
    txWizard: TLabel;
    cbWLeft: TButton;
    cbWRight: TButton;
    cbPlace: TButton;
    Undol: TMenuItem;
    N2: TMenuItem;
    DeleteObject1: TMenuItem;
    stBack: TLabel;
    stLeft: TLabel;
    stRight: TLabel;
    procedure Exit1Click(Sender: TObject);
    procedure TBWindowClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure TBDoorClick(Sender: TObject);
    procedure New1Click(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure pbLeftClick(Sender: TObject);
    procedure pbRightClick(Sender: TObject);
    procedure TBPictureClick(Sender: TObject);
    procedure TBScreenClick(Sender: TObject);
    procedure TBRTTableClick(Sender: TObject);
    procedure TBSTableClick(Sender: TObject);
    procedure TBPeopleClick(Sender: TObject);
    procedure TBDoccamClick(Sender: TObject);
    procedure TBVCRCClick(Sender: TObject);
    procedure TBCameraClick(Sender: TObject);
    procedure SavelClick(Sender: TObject);
    procedure Open1Click(Sender: TObject);
    procedure RoomWizard1Click(Sender: TObject);
    procedure cbWLeftClick(Sender: TObject);
    procedure cbWRightClick(Sender: TObject);
    procedure cbPlaceClick(Sender: TObject);
    procedure About1Click(Sender: TObject);
    procedure UndolClick(Sender: TObject);
    procedure DeleteObject1Click(Sender: TObject);
private
    { Private declarations }
    procedure AddtoRoomList(aRoomObj : TRoomObj) ;
    function  GetNextEntry(var BigString : string):string ;
    function  GetWizardText : string ;
public
    { Public declarations }
    ActiveObject      : string ;
end;

var
    Form1              : TForm1;
    RoomList           : TList ;
    vg_WPage           : integer ;

```



```

    LastObject      : integer ;
    LoadFromFile   : boolean;

implementation
    uses wait, about, camdial;
    {$R *.DFM}

procedure TForm1.Exit1Click(Sender: TObject);
begin
    close;
end;

procedure TForm1.AddtoRoomList(aRoomObj : TRoomObj) ;
var CurrObj : TRoomObj ;
    index   : integer ;

begin
    LastObject := 0 ;
    ActiveObject := aRoomObj.Name ;
    CASE aRoomObj.Level OF
        1 : RoomList.Insert(0,aRoomObj);
        2 : BEGIN
            index := 0 ;
            REPEAT
                CurrObj := RoomList.Items[index] ;
                inc(index) ;
            UNTIL (CurrObj.Level = 2) OR (index = RoomList.count) ;
            IF index = RoomList.count THEN BEGIN
                RoomList.Add(aRoomObj);
                LastObject := RoomList.count -1 ;end
            ELSE BEGIN
                RoomList.Insert(index,aRoomObj);
                LastObject := index;
            END ;
        END ;
        3 : BEGIN
            RoomList.Add(aRoomObj) ;
            LastObject := RoomList.count -1 ;
        END;
    END;
    Undo1.Enabled := true ;
    DeleteObject1.enabled := true ;
end;

function TForm1.GetNextEntry(var BigString : string):string ;
var aString   : string ;
    aPosition : integer ;

begin
    aString := '' ;
    aPosition := pos(';',BigString);
    IF aPosition <> 0 THEN BEGIN
        aString := copy(BigString,1,aPosition -1);
        BigString := copy(BigString,aPosition +1,length(BigString));
    END;
    GetNextEntry := aString ;
end;

procedure TForm1.TBWindowClick(Sender: TObject);
var NewWindow : TRoomObj ;

```

```
    vd_Bitmap : TBitmap ;
    TempString : string ;

begin
    pHelpStatus.caption := 'Creating Window' ;
    pHelpStatus.repaint ;
    NewWindow := TRoomObj.create(PImagePanel) ;
    vd_bitmap := TBitmap.create ;
    WindowImages.GetBitmap(1,vd_bitmap) ;
    str(roomList.count,tempString) ;
    tempString := 'Window' + tempString ;
    WITH NewWindow DO BEGIN
        parent := PImagePanel;
        Initialize(roWindow);
        glyph := vd_bitmap ;
        Name := tempString ;
    END;
    AddtoRoomList(NewWindow) ;
    vd_bitmap.free ;
    pHelpStatus.caption := 'Ready' ;
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
    pHelpStatus.caption := 'Ready' ;
    RoomList := TList.create ;
    Left := 0 ;
    Top := 0 ;
    LoadFromFile := False;
end;

procedure TForm1.TBDoorClick(Sender: TObject);
var NewDoor : TRoomObj ;
    vd_Bitmap : TBitmap ;
    TempString : string ;

begin
    pHelpStatus.caption := 'Creating Door' ;
    pHelpStatus.repaint ;
    NewDoor := TRoomObj.create(PImagePanel) ;
    vd_bitmap := TBitmap.create ;
    DoorImages.GetBitmap(1,vd_bitmap) ;
    str(roomList.count,tempString) ;
    tempString := 'Door' + tempString ;
    WITH NewDoor DO BEGIN
        parent := PImagePanel;
        Initialize(roDoor);
        glyph := vd_bitmap ;
        Name := tempString ;
    END;
    AddtoRoomList(NewDoor) ;
    vd_bitmap.free ;
    pHelpStatus.caption := 'Ready' ;
end;

procedure TForm1.NewlClick(Sender: TObject);
var index : integer ;
    CurrObj : TRoomObj ;

begin
```

```

    IF RoomList.count > 0 THEN BEGIN
        IF messageDlg('Are you sure you want to clear the room?',
            mtConfirmation,[mbYes,mbNo],0) = mrYes THEN BEGIN
            FOR index := 0 TO RoomList.count -1 DO BEGIN
                CurrObj := RoomList.Items[index] ;
                CurrObj.free ;
            END ;
            RoomList.clear ;
            Undol.Enabled := false ;
            DeleteObject1.enabled := false ;
            ActiveObject := '' ;
            stLeft.caption := '1';
            stBack.caption := '2';
            stRight.caption := '3';
        END;
    END;
end;

procedure TForm1.FormClose(Sender: TObject;
    var Action: TCloseAction);
var index : integer ;
    CurrObj : TRoomObj ;

begin
    IF RoomList.count > 0 THEN BEGIN
        IF messageDlg('Do you want to save the room before quitting?',
            mtConfirmation,[mbYes,mbNo],0) = mrYes THEN BEGIN
            Save1Click(Form1) ;
        END ;
        FOR index := 0 TO RoomList.count -1 DO BEGIN
            CurrObj := RoomList.Items[index] ;
            CurrObj.free ;
        END ;
    END;
    RoomList.free ;
end;

procedure TForm1.pbLeftClick(Sender: TObject);
var Index      : integer ;
    NewWall     : TWall;
    CurrObj     : TRoomObj ;
    code        : integer;

begin
    pHelpStatus.caption := 'Rotating Room Left' ;
    pHelpStatus.repaint ;
    FOR Index := 0 TO RoomList.count - 1 DO BEGIN
        CurrObj := RoomList.Items[Index] ;
        WITH CurrObj DO BEGIN
            IF Level = 1 THEN BEGIN
                CASE Wall OF
                    wBack,wRight,wFront : BEGIN
                        NewWall := Wall;
                        dec(NewWall);
                        wallChange(NewWall) ;
                        wallObjReposition ;
                    END ;
                    wLeft : wallChange(wFront) ;
                END; end
            ELSE BEGIN

```

```

        RoomObjectLeft;
    END ;
END;
END ;
Val(stLeft.caption, Index, Code);
CASE Index OF
  1 : BEGIN
        stLeft.caption := '2';
        stBack.caption := '3';
        stRight.caption := '4';
    END;
  2 : BEGIN
        stLeft.caption := '3';
        stBack.caption := '4';
        stRight.caption := '1';
    END;
  3 : BEGIN
        stLeft.caption := '4';
        stBack.caption := '1';
        stRight.caption := '2';
    END;
  4 : BEGIN
        stLeft.caption := '1';
        stBack.caption := '2';
        stRight.caption := '3';
    END;
END;
pHelpStatus.caption := 'Ready' ;
end;

procedure TForm1.pbRightClick(Sender: TObject);
var Index      : integer ;
    NewWall    : TWall;
    CurrObj    : TRoomObj ;
    code       : integer;

begin
    pHelpStatus.caption := 'Rotating Room Right' ;
    pHelpStatus.repaint ;
    FOR Index := 0 TO RoomList.count - 1 DO BEGIN
        CurrObj := RoomList.Items[Index] ;
        WITH CurrObj DO BEGIN
            IF Level = 1 THEN BEGIN
                CASE Wall OF
                    wBack,wLeft : BEGIN
                        NewWall := Wall;
                        inc(NewWall);
                        wallChange(NewWall) ;
                        wallObjReposition ;
                    END ;
                    wRight : wallChange(wFront) ;
                    wFront : BEGIN
                        wallChange(wLeft) ;
                        wallObjReposition ;
                    END ;
                END; end
            ELSE BEGIN
                RoomObjectRight;
            END ;
        END;
    END;
end;

```

```
END ;
Val(stLeft.caption, Index, Code);
CASE Index OF
  1 : BEGIN
    stLeft.caption := '4';
    stBack.caption := '1';
    stRight.caption := '2';
  END;
  2 : BEGIN
    stLeft.caption := '1';
    stBack.caption := '2';
    stRight.caption := '3';
  END;
  3 : BEGIN
    stLeft.caption := '2';
    stBack.caption := '3';
    stRight.caption := '4';
  END;
  4 : BEGIN
    stLeft.caption := '3';
    stBack.caption := '4';
    stRight.caption := '1';
  END;
END;
pHelpStatus.caption := 'Ready' ;
end;

procedure TForm1.TBPictureClick(Sender: TObject);
var NewObject      : TRoomObj ;
    vd_Bitmap      : TBitmap ;
    TempString     : string ;

begin
  pHelpStatus.caption := 'Creating Picture' ;
  pHelpStatus.repaint ;
  NewObject := TRoomObj.create(PImagePanel) ;
  vd_bitmap := TBitmap.create ;
  PictureImages.GetBitmap(1,vd_bitmap);
  str(roomList.count,tempString) ;
  tempString := 'Picture' + tempString ;
  WITH NewObject DO BEGIN
    parent := PImagePanel;
    Initialize(roPicture);
    glyph := vd_bitmap ;
    Name := tempString ;
  END;
  AddtoRoomList(NewObject) ;
  vd_bitmap.free ;
  pHelpStatus.caption := 'Ready' ;
end;

procedure TForm1.TBScreenClick(Sender: TObject);
var NewObject      : TRoomObj ;
    vd_Bitmap      : TBitmap ;
    TempString     : string ;

begin
  pHelpStatus.caption := 'Creating Screen' ;
  pHelpStatus.repaint ;
  NewObject := TRoomObj.create(PImagePanel) ;
```

```
vd_bitmap := TBitmap.create ;
ScreenImages.GetBitmap(1, vd_bitmap);
str(roomList.count, tempString) ;
tempString := 'Screen' + tempString ;
WITH NewObject DO BEGIN
    parent := PImagePanel;
    Initialize(roScreen);
    glyph := vd_bitmap ;
    Name := tempString ;
END;
AddtoRoomList(NewObject) ;
vd_bitmap.free ;
pHelpStatus.caption := 'Ready' ;
end;

procedure TForm1.TBRTTableClick(Sender: TObject);
var NewObject : TRoomObj ;
    vd_Bitmap : TBitmap ;
    TempString : string ;

begin
    pHelpStatus.caption := 'Creating Table' ;
    pHelpStatus.repaint ;
    NewObject := TRoomObj.create(PImagePanel) ;
    vd_bitmap := TBitmap.create ;
    TableImages.GetBitmap(1, vd_bitmap);
    str(roomList.count, tempString) ;
    tempString := 'Table' + tempString ;
    WITH NewObject DO BEGIN
        parent := PImagePanel;
        Initialize(roRTTable);
        glyph.assign(vd_bitmap) ;
        Name := tempString ;
    END;
    AddtoRoomList(NewObject) ;
    vd_bitmap.free ;
    pHelpStatus.caption := 'Ready' ;
end;

procedure TForm1.TBSTTableClick(Sender: TObject);
var NewObject : TRoomObj ;
    vd_Bitmap : TBitmap ;
    TempString : string ;

begin
    pHelpStatus.caption := 'Creating Table' ;
    pHelpStatus.repaint ;
    NewObject := TRoomObj.create(PImagePanel) ;
    vd_bitmap := TBitmap.create ;
    TableImages.GetBitmap(0, vd_bitmap);
    str(roomList.count, tempString) ;
    tempString := 'Table' + tempString ;
    WITH NewObject DO BEGIN
        parent := PImagePanel;
        Initialize(roSTable);
        glyph.assign(vd_bitmap) ;
        Name := tempString ;
    END;
    AddtoRoomList(NewObject) ;
    vd_bitmap.free ;
```

```
pHelpStatus.caption := 'Ready' ;
end;

procedure TForm1.TBPeopleClick(Sender: TObject);
var NewObject : TRoomObj ;
    vd_Bitmap : TBitmap ;
    TempString : string ;

begin
    pHelpStatus.caption := 'Creating Audience' ;
    pHelpStatus.repaint ;
    NewObject := TRoomObj.create(PImagePanel) ;
    vd_bitmap := TBitmap.create ;
    ButtonImages.GetBitmap(0,vd_bitmap);
    str(roomList.count,tempString) ;
    tempString := 'Audience' + tempString ;
    WITH NewObject DO BEGIN
        parent := PImagePanel;
        Initialize(roAudience);
        glyph.assign(vd_bitmap) ;
        Name := tempString ;
    END;
    AddtoRoomList(NewObject) ;
    vd_bitmap.free ;
    pHelpStatus.caption := 'Ready' ;
end;

procedure TForm1.TBDoccamClick(Sender: TObject);
var NewObject : TRoomObj ;
    vd_Bitmap : TBitmap ;
    TempString : string ;

begin
    pHelpStatus.caption := 'Creating Document Camera' ;
    pHelpStatus.repaint ;
    NewObject := TRoomObj.create(PImagePanel) ;
    vd_bitmap := TBitmap.create ;
    ButtonImages.GetBitmap(2,vd_bitmap);
    str(roomList.count,tempString) ;
    tempString := 'DocCam' + tempString ;
    WITH NewObject DO BEGIN
        parent := PImagePanel;
        Initialize(roDocCam);
        glyph.assign(vd_bitmap) ;
        Name := tempString ;
    END;
    AddtoRoomList(NewObject) ;
    vd_bitmap.free ;
    pHelpStatus.caption := 'Ready' ;
end;

procedure TForm1.TBVCRClick(Sender: TObject);
var NewObject : TRoomObj ;
    vd_Bitmap : TBitmap ;
    TempString : string ;

begin
    pHelpStatus.caption := 'Creating VCR' ;
    pHelpStatus.repaint ;
    NewObject := TRoomObj.create(PImagePanel) ;
```

```

    vd_bitmap := TBitmap.create ;
    ButtonImages.GetBitmap(3,vd_bitmap);
    str(roomList.count,tempString) ;
    tempString := 'VCR' + tempString ;
    WITH NewObject DO BEGIN
        parent := PImagePanel;
        Initialize(roVCR);
        glyph.assign(vd_bitmap) ;
        Name    := tempString ;
    END;
    AddtoRoomList(NewObject) ;
    vd_bitmap.free ;
    pHelpStatus.caption := 'Ready' ;
end;

procedure TForm1.TBCameraClick(Sender: TObject);
var NewObject  : TRoomObj ;
    vd_Bitmap  : TBitmap ;
    TempString : string ;

begin
    pHelpStatus.caption := 'Creating Camera' ;
    pHelpStatus.repaint ;
    NewObject := TRoomObj.create(PImagePanel) ;
    vd_bitmap := TBitmap.create ;
    ButtonImages.GetBitmap(1,vd_bitmap);
    str(roomList.count,tempString) ;
    tempString := 'Video' + tempString ;
    WITH NewObject DO BEGIN
        parent := PImagePanel;
        Initialize(roVideo);
        glyph.assign(vd_bitmap) ;
        activeObject := tempString ;
        IF NOT LoadFromFile cameraDialog.show ;
        Name := activeObject ;
    END;
    AddtoRoomList(NewObject) ;
    vd_bitmap.free ;
    pHelpStatus.caption := 'Ready' ;
end;

procedure TForm1.Save1Click(Sender: TObject);
var Output      : textFile;
    aLine       : string;
    index       : integer ;
    CurrObj     : TRoomObj ;

begin
    IF RoomList.count > 0 THEN BEGIN
        IF SaveDialog1.Execute THEN BEGIN
            AssignFile(Output, SaveDialog1.FileName);
            Rewrite(Output);
            FOR index := 0 TO RoomList.count -1 DO BEGIN
                CurrObj := RoomList.Items[index] ;
                IF CurrObj.Enabled THEN BEGIN
                    aLine := CurrObj.PackAsRoomString ;
                    Writeln(Output,aLine) ;
                END;
            END ;
            CloseFile(Output);
        END;
    END;

```



```

        END; end
    ELSE BEGIN
        ShowMessage('There are no room objects to save.');
```

END ;

end;

```

procedure TForm1.Open1Click(Sender: TObject);
var Input      : textFile;
    anEntry    : string;
    anEntryVal : integer ;
    anEntryReal: single ;
    Code       : integer ;
    aLine      : string;
    index      : integer ;
    CurrObj    : TRoomObj ;

begin
    IF RoomList.count > 0 THEN BEGIN
        IF messageDlg('Do you want to save this room first?',
            mtConfirmation,[mbYes,mbNo],0) = mrYes THEN BEGIN
            Save1Click(Form1) ;
        END;
        FOR index := 0 TO RoomList.count -1 DO BEGIN
            CurrObj := RoomList.Items[index] ;
            CurrObj.free ;
        END ;
        RoomList.clear ;
    END;
    IF OpenDialog1.Execute THEN BEGIN
        stLeft.caption := '1';
        stBack.caption := '2';
        stRight.caption := '3';
        LoadFromFile := true;
        FWaitBox.show;
        FWaitBox.repaint ;
        AssignFile(Input, OpenDialog1.FileName);
        Reset(Input);
        WHILE NOT EOF(Input) DO BEGIN
            Readln(Input, aLine);
            anEntry := getNextEntry(aLine) ;
            IF anEntry = 'Door' THEN TBDoorClick(Form1) ;
            IF anEntry = 'Windo' THEN TBWindowClick(Form1) ;
            IF anEntry = 'Pictu' THEN TBPictureClick(Form1) ;
            IF anEntry = 'Scree' THEN TBScreenClick(Form1) ;
            IF anEntry = 'Round' THEN TBRTableClick(Form1) ;
            IF anEntry = 'Squar' THEN TBSTableClick(Form1) ;
            IF anEntry = 'Audie' THEN TBPeopleClick(Form1) ;
            IF anEntry = 'Camer' THEN TBCameraClick(Form1) ;
            IF anEntry = 'Docum' THEN TBDocCamClick(Form1) ;
            IF anEntry = 'VCR' THEN TBVCRClick(Form1) ;
            FOR index := 0 TO RoomList.count -1 DO BEGIN
                CurrObj := RoomList.Items[index] ;
                IF CurrObj.JustAdded THEN BEGIN
                    CurrObj.JustAdded := false ;
                    anEntry := getNextEntry(aLine) ;
                    val(anEntry,anEntryVal,Code);
                    CurrObj.Left := anEntryVal ;
                    anEntry := getNextEntry(aLine) ;
                    val(anEntry,anEntryVal,Code);
                    CurrObj.Top := anEntryVal ;
                END;
            END;
        END;
    END;
end;
```

```

        anEntry := getNextEntry(aLine) ;
        val(anEntry, anEntryVal, Code);
        CurrObj.Width := anEntryVal ;
        anEntry := getNextEntry(aLine) ;
        val(anEntry, anEntryVal, Code);
        CurrObj.Height := anEntryVal ;
        anEntry := getNextEntry(aLine) ;
        val(anEntry, anEntryReal, Code);
        CurrObj.PercentPos := anEntryReal ;
        anEntry := getNextEntry(aLine) ;
        val(anEntry, anEntryVal, Code);
        CASE anEntryVal OF
            0 : CurrObj.WallChange(wLeft) ;
            1 : CurrObj.Wall := wBack ;
            2 : CurrObj.WallChange(wRight);
            3 : CurrObj.WallChange(wFront);
            4 : CurrObj.Wall := wNo ;
        END;
    END;
END ;
END;
CloseFile(Input);
FWaitBox.Hide;
LoadFromFile := false;
END ;
end;

procedure TForm1.RoomWizard1Click(Sender: TObject);
begin
    pWizard.Visible := NOT pWizard.visible ;
    RoomWizard1.checked := NOT RoomWizard1.checked ;
    vg_WPage := 1 ;
    txWizard.caption := getWizardText ;
    cbPlace.Enabled := false ;
end;

procedure TForm1.cbWLeftClick(Sender: TObject);
begin
    CASE vg_Wpage OF
        1 : RoomWizard1.click ;
        2 : BEGIN
            vg_WPage := 1;
            cbWLeft.caption := '&Close' ;
            txWTitle.Caption := 'Wizard' ;
            txWizard.caption := getWizardText ;
        END;
        3 : BEGIN
            vg_WPage := 2;
            cbPlace.enabled := false ;
            cbPlace.caption := '&Place Object' ;
            txWTitle.Caption := 'Introduction' ;
            txWizard.caption := getWizardText ;
        END;
        4 : BEGIN
            vg_WPage := 3;
            cbPlace.caption := '&Place Door' ;
            txWTitle.Caption := 'Door' ;
            txWizard.caption := getWizardText ;
        END;
        5 : BEGIN

```

```

        vg_WPage := 4;
        cbPlace.caption := '&Place Window' ;
        txWTitle.Caption := 'Window' ;
        txWizard.caption := getWizardText ;
    END;
6 : BEGIN
    vg_WPage := 5;
    cbPlace.caption := '&Place Screen' ;
    txWTitle.Caption := 'Screen' ;
    txWizard.caption := getWizardText ;
    END;
7 : BEGIN
    vg_WPage := 6;
    cbPlace.enabled := true ;
    cbPlace.caption := '&Place Picture' ;
    txWTitle.Caption := 'Picture' ;
    txWizard.caption := getWizardText ;
    END;
8 : BEGIN
    vg_WPage := 7;
    cbPlace.enabled := false ;
    txWTitle.Caption := 'Rotate Room' ;
    txWizard.caption := getWizardText ;
    END;
9 : BEGIN
    vg_WPage := 8;
    txWizard.caption := getWizardText ;
    END;
10 : BEGIN
    vg_WPage := 9;
    cbPlace.caption := '&Place Table' ;
    txWTitle.Caption := 'Table' ;
    txWizard.caption := getWizardText ;
    END;
11 : BEGIN
    vg_WPage := 10;
    cbPlace.caption := '&Place Audience' ;
    txWTitle.Caption := 'Audience' ;
    txWizard.caption := getWizardText ;
    END;
12 : BEGIN
    vg_WPage := 11;
    cbPlace.caption := '&Place Camera' ;
    txWTitle.Caption := 'Video Cam.' ;
    txWizard.caption := getWizardText ;
    END;
13 : BEGIN
    vg_WPage := 12;
    cbPlace.caption := '&Place DocCam' ;
    cbWRight.caption := '&Next...' ;
    txWTitle.Caption := 'Doc. Camera' ;
    txWizard.caption := getWizardText ;
    END;
END;
end;

procedure TForm1.cbWRightClick(Sender: TObject);
begin
    CASE vg_WPage OF
        1 : BEGIN

```

```
        vg_WPage := 2;
        cbWLeft.caption := '&Back...' ;
        txWTitle.Caption := 'Introduction' ;
        txWizard.caption := getWizardText ;
    END;
2 : BEGIN
    vg_WPage := 3;
    cbPlace.caption := '&Place Door' ;
    cbPlace.enabled := true ;
    txWTitle.Caption := 'Door' ;
    txWizard.caption := getWizardText ;
    END;
3 : BEGIN
    vg_WPage := 4;
    cbPlace.caption := '&Place Window' ;
    txWTitle.Caption := 'Window' ;
    txWizard.caption := getWizardText ;
    END;
4 : BEGIN
    vg_WPage := 5;
    cbPlace.caption := '&Place Screen' ;
    txWTitle.Caption := 'Screen' ;
    txWizard.caption := getWizardText ;
    END;
5 : BEGIN
    vg_WPage := 6;
    cbPlace.caption := '&Place Picture' ;
    txWTitle.Caption := 'Picture' ;
    txWizard.caption := getWizardText ;
    END;
6 : BEGIN
    vg_WPage := 7;
    cbPlace.enabled := false ;
    txWTitle.Caption := 'Rotate Room' ;
    txWizard.caption := getWizardText ;
    END;
7 : BEGIN
    vg_WPage := 8;
    cbPlace.caption := '&Place Table' ;
    cbPlace.enabled := true ;
    txWTitle.Caption := 'Table' ;
    txWizard.caption := getWizardText ;
    END;
8 : BEGIN
    vg_WPage := 9;
    txWizard.caption := getWizardText ;
    END;
9 : BEGIN
    vg_WPage := 10;
    cbPlace.caption := '&Place Audience' ;
    txWTitle.Caption := 'Audience' ;
    txWizard.caption := getWizardText ;
    END;
10 : BEGIN
    vg_WPage := 11;
    cbPlace.caption := '&Place Camera' ;
    txWTitle.Caption := 'Video Cam.' ;
    txWizard.caption := getWizardText ;
    END;
11 : BEGIN
```

```

        vg_WPage := 12;
        cbPlace.caption := '&Place DocCam' ;
        txWTitle.Caption := 'Doc Camera' ;
        txWizard.caption := getWizardText ;
    END;
12 : BEGIN
        vg_WPage := 13;
        cbPlace.caption := '&Place VCR' ;
        txWTitle.Caption := 'VCR' ;
        cbWRight.caption := '&Close' ;
        txWizard.caption := getWizardText ;
    END;
13 : BEGIN
        cbWRight.caption := '&Next...' ;
        cbWLeft.caption := '&Close' ;
        txWTitle.Caption := 'Wizard' ;
        RoomWizard1.click ;
        txWizard.caption := getWizardText ;
    END;
END;
end;

procedure TForm1.cbPlaceClick(Sender: TObject);
begin
    CASE vg_WPage OF
        3 : TBDoor.click ;
        4 : TBWindow.Click ;
        5 : TBScreen.Click ;
        6 : TBPicture.Click ;
        8 : TBSTable.Click ;
        9 : TBRTTable.click ;
        10 : TBPeople.Click ;
        11 : TBCamera.Click ;
        12 : TBDoccam.click ;
        13 : TBVCR.Click ;
    END;
end;

function TForm1.GetWizardText : string ;
begin
    CASE vg_WPage OF
        1 : GetWizardText := ' Welcome to the Room Design wizard. '+
            'The wizard will guide you through the general '+
            ' steps to create a room.'+#13#10#13#10+
            'Click on the buttons below to complete your design.' ;
        2 : GetWizardText := 'You should normally place the room '+
            'objects such as doors && tables and then the devices'+
            ' like the VCR && Document Camera.'+#13#10#13#10+
            'Since the video camera view occupies such a large '+
            'portion of the screen. The earlier it can be '+
            'placed the better. This will make access to other '+
            'objects easier.';
        3 : GetWizardText := 'Click on the <Place Door> button below '+
            'to put a door in the room.'+#13#10#13#10+
            'Assume that you are standing at the front of the '+
            'room and move the door to the correct wall.'+
            #13#10#13#10+'This can be done by dragging the door.'+
            ' Don''t worry if it should be on the wall that '+
            'currently isn''t visible. It can be placed later.';
        4 : GetWizardText := 'If there are any windows in the room, '+

```

```

        'click on the <Place Window> button below, then drag '+'
        'the window to the place where it should be.' ;
5 : GetWizardText := 'If there are any projection screens, '+'
    'whiteboards or blackboards, click on the <Place '+'
    'Screen> button then drag the screen to its '+'
    'location.'#13#10#13#10+'You should only place '+'
    'a screen in the room if it aids orientation.';
6 : GetWizardText := 'If there are any pictures, paintings or '+'
    'charts and they aid room orientation then click on '+'
    'the <Place Picture> button and locate the '+'
    'picture.'#13#10#13#10+'The objects on three walls'+
    ' have been placed, it is time to place objects if '+'
    'needed on the final wall.';
7 : GetWizardText := 'Move any objects for the fourth wall to '+'
    'one of the side walls, left or right. Click twice on '+'
    'the left or right buttons under the Rotate Room. The '+'
    'room will turn. You are now looking at the wall that '+'
    'was behind you. Drag any desired objects to that wall.';
8 : GetWizardText := 'Clicking on one of the rotate buttons '+'
    'twice more will reorient the room to its original '+'
    'position.'+#13#10#13#10+'If there is a square or '+'
    'rectangular table in the room, click on the <Place '+'
    'Table> button and move it to its location.';
9 : GetWizardText := 'If there is a round or oval table in the'+
    ' room, click on the <Place Table> button and move it '+'
    'to its location.'+#13#10#13#10+'Remember, it is not '+'
    'necessary to place every table in the room, only the '+'
    'major ones.';
10 : GetWizardText := 'Place a few audience markers in the room '+'
    'to represent where they normally sit.';
11 : GetWizardText := 'If a video camera is available then click '+'
    'on the <Place Camera> button.'+#13#10#13#10+'Unlike the'+
    ' other objects the camera is not placed as a device. '+'
    'Instead the camera's view is represented && the camera'+
    ' can be named. You indicate what the camera is '+'
    'pointing at by adjusting the view rectangle.';
12 : GetWizardText := 'If a Document Camera or an Overhead '+'
    'Projector is available then place it in the room.';
13 : GetWizardText := 'If a VCR is available in the room then '+'
    'locate it in the room';

    END;
end;

procedure TForm1.About1Click(Sender: TObject);
begin
    form2.show;
end;

procedure TForm1.Undo1Click(Sender: TObject);
var    CurrObj : TRoomObj ;

begin
    IF (LastObject >= 0) AND (LastObject < RoomList.Count) THEN BEGIN
        CurrObj := RoomList.Items[LastObject] ;
        CurrObj.Visible := false ;
        CurrObj.enabled := false ;
        Undo1.Enabled := false ;
        DeleteObject1.Enabled := false ;
        ActiveObject := '' ;
    END;

```

```

end;

procedure TForm1.DeleteObject1Click(Sender: TObject);
var tempString : string ;
    index      : integer ;
    CurrObj     : TRoomObj ;

begin
    IF length(activeObject) <> 0 THEN BEGIN
        tempString := copy(activeObject,1,length(activeObject) -1);
        IF messageDlg('Are you sure you want to delete the '+tempString+
            '?', mtConfirmation,[mbYes,mbNo],0) = mrYes THEN BEGIN
            FOR index := 0 TO RoomList.count -1 DO BEGIN
                CurrObj := RoomList.Items[index] ;
                IF CurrObj.Name = activeObject THEN BEGIN
                    CurrObj.Visible := false ;
                    CurrObj.enabled := false ;
                    IF index = lastObject THEN Undo1.Enabled := false ;
                END;
            END ;
            DeleteObject1.enabled := false ;
            ActiveObject := '' ;
        END; end
    ELSE BEGIN
        showMessage('Select an object to delete first.');
```

Unit RoomObj

```

unit RoomObj;

interface

uses windows, extCtrls, graphics, TransBtn, classes, controls;

type
    TRoomObject = (roDoor, roWindow, roPicture, roScreen,
        roTable, roSTable, roAudience, roDocCam, roVCR,
        roVideo) ;
    TLevel = 1..3 ;
    TWall = (wLeft, wBack, wRight, wFront, wNo) ;
    TRoomObj = class(TTransparentButton)
    procedure Initialize(aRoomObject :TRoomObject);
    procedure click; override ;
    procedure MouseMove(Shift: TShiftState; X, Y: Integer);override;
    procedure MouseUp(Button: TMouseButton; Shift: TShiftState;
        X, Y: Integer);override;
    procedure RoomObjectRight;
    procedure RoomObjectLeft;
    procedure WallChange(NewWall : TWall) ;
    procedure WallObjReposition ;
    function PackAsRoomString : string ;
```

```

    public
        BackTop      : integer ;
        PercentPos   : single ;
        Wall         : TWall ;
        RType        : TRoomObject;
        Level        : TLevel ;
        JustAdded    : boolean ;
    END;

Const
    cLeftWall   = -10 ;
    cBackWall   = 74  ;
    cRightWall  = 330 ;
    cLeftLength = 61  ;
    cBackLength = 266 ;
    cRightLength = 58 ;
    cDefaultBackTop = 9 ;
    cDefaultSideTop = 43 ;
    cTotalWidth   = 420 ;
    cTotalHeight  = 349 ;

implementation
    uses rdumain ;
    procedure TRoomObj.Initialize(aRoomObject : TRoomObject);
    begin
        FrameStyle := fsRegular ;
        JustAdded   := true ;
        Movable     := true ;
        Transparent := true ;
        visible     := true ;
        NumGlyphs   := 1 ;
        BackTop     := 15 ;
        ShowHint    := true ;
        RType       := aRoomObject ;
        TextAlign   := ttaBottom ;
        CASE aRoomObject OF
            roDoor : BEGIN
                FrameStyle := fsNone ;
                Left      := 160 ;
                Top       := 9 ;
                BackTop   := 0 ;
                Width     := 44 ;
                Height    := 120 ;
                PercentPos := 0.327 ;
                Wall      := wBack ;
                Level     := 1 ;
                Hint      := 'Door' ;
            END;
            roWindow..roScreen : BEGIN
                FrameStyle := fsNone ;
                Left      := 110 ;
                Top       := 24 ;
                Width     := 44 ;
                Height    := 90 ;
                PercentPos := 0.139 ;
                Wall      := wBack ;
                Level     := 1 ;
                IF aRoomObject = roWindow THEN Hint := 'Window' ;
                IF aRoomObject = roPicture THEN Hint := 'Picture' ;
                IF aRoomObject = roScreen THEN

```



```

                                Hint := 'Screen/Whiteboard' ;
                                END;
                                roRTable..roStable : BEGIN
                                    FrameStyle := fsNone ;
                                    Left := 115 ;
                                    Top := 85 ;
                                    BackTop := 0 ;
                                    Width := 150 ;
                                    Height := 150 ;
                                    PercentPos := 0.0 ;
                                    Wall := wNo ;
                                    Level := 2 ;
                                    IF aRoomObject = roRTable THEN Hint := 'Round Table' ;
                                    IF aRoomObject = roStable THEN Hint := 'Square Table';
                                END;
                                roAudience..roVCR : BEGIN
                                    Left := 115 ;
                                    Top := 250 ;
                                    BackTop := 0 ;
                                    Width := 50 ;
                                    Height := 50 ;
                                    PercentPos := 0.0 ;
                                    Wall := wNo ;
                                    Level := 3 ;
                                    IF aRoomObject = roAudience THEN Hint := 'Audience' ;
                                    IF aRoomObject = roDocCam THEN
                                        Hint := 'Document Camera' ;
                                    IF aRoomObject = roVCR THEN Hint := 'VCR' ;
                                END;
                                roVideo : BEGIN
                                    Left := 0 ;
                                    Top := 0 ;
                                    BackTop := 0 ;
                                    Width := cTotalWidth ;
                                    Height := cTotalHeight div 2 ;
                                    BorderWidth := 2 ;
                                    PercentPos := 0.0 ;
                                    TextAlign := ttaCenter;
                                    Wall := wNo ;
                                    Level := 3 ;
                                    Hint := 'Camera View' ;
                                END;
                                END;
                                end;

                                procedure TRoomObj.Click;
                                begin
                                    inherited Click;
                                    form1.ActiveObject := Name ;
                                    form1.DeleteObject1.Enabled := true ;
                                end;

                                procedure TRoomObj.MouseMove(Shift: TShiftState; X, Y: Integer);
                                begin
                                    inherited MouseMove(Shift,X,Y);
                                    CASE RType OF
                                        *
                                        roDoor..roScreen : BEGIN
                                            CASE Wall OF
                                                wBack : BEGIN
                                                    IF Top <> (cDefaultBackTop + BackTop) THEN

```

```

        Top := cDefaultBackTop + BackTop ;
    IF Left < 51 THEN BEGIN
        wallChange(wLeft);
        Top := cDefaultSideTop + BackTop ;
        Left := 41 ; end
    ELSE IF Left > 317 THEN BEGIN
        wallChange(wRight);
        Top := cDefaultSideTop + BackTop ;
        Left := 328 ;
    END;
    END ;
wLeft : BEGIN
    IF Top < 43 THEN BEGIN
        Top := cDefaultSideTop + BackTop ; end
    ELSE IF Top > 217 THEN BEGIN
        Top := 217
    END ;
    IF (Left > 51) AND (Top < 130) THEN BEGIN
        wallChange(wBack);
        Top := cDefaultBackTop + BackTop ;
        Left := 73 ; end
    ELSE BEGIN
        IF Left < -10 THEN Left := -10 ;
    END;
    END ;
wRight : BEGIN
    IF Top < 43 THEN BEGIN
        Top := cDefaultSideTop + BackTop ; end
    ELSE IF Top > 217 THEN BEGIN
        Top := 217 ;
    END ;
    IF (Left < 317) AND (Top < 130) THEN BEGIN
        wallChange(wBack);
        Top := cDefaultBackTop + BackTop ;
        Left := 273 ; end
    ELSE BEGIN
        IF Left > 380 THEN Left := 380 ;
    END;
    END ;
    END;
    END;
roRTable..roVCR : BEGIN
    IF Top < 0 THEN Top := 0;
    IF Left < 0 THEN Left := 0 ;
    IF Top > 350 - Height THEN Top := 350 - Height ;
    IF Left > 410 - Width THEN Left := 410 - Width ;
    END;
roVideo : BEGIN
    IF Left < -(cTotalWidth div 6) THEN BEGIN
        Left := 0 ;
        Top := 0 ;
        Width := cTotalWidth div 2 ;
        Height := cTotalHeight ;
    END ;
    IF (Left > (cTotalWidth div 5)) AND
        (Left < (cTotalWidth div 3)) AND
        (Top < -10) THEN BEGIN
        Top := 0 ;
        Width := cTotalWidth ;
        Height := cTotalHeight div 2 ;
    END ;

```

```

        END ;
        IF Top > (cTotalHeight div 2 + 10) THEN BEGIN
            Top := cTotalHeight div 2 ;
            Width := cTotalWidth ;
            Height := cTotalHeight div 2 ;
        END ;
        IF Left > (cTotalWidth div 2 + 10) THEN BEGIN
            Left := cTotalWidth div 2 ;
            Top := 0 ;
            Width := cTotalWidth div 2 ;
            Height := cTotalHeight ;
        END ;
    END ;
END;
end;

procedure TRoomObj.MouseUp(Button: TMouseButton;
                           Shift: TShiftState; X, Y: Integer);
begin
    inherited MouseUp(Button, Shift, X, Y);
    CASE RType OF
        roDoor..roScreen : BEGIN
            IF Wall = wLeft THEN BEGIN
                IF Top <> (round(-3.41 * Left + 184 - BackTop)) THEN
                    Left := round((Top - 184 + BackTop) / -3.41) ;
                PercentPos := (Left - cLeftWall) / cLeftLength ;
            END ;
            IF Wall = wRight THEN BEGIN
                IF Top <> (round(3.6 * Left - 1144 + BackTop)) THEN
                    Left := round((Top + 1144 - BackTop) / 3.6) ;
                PercentPos := (Left - cRightWall) / cRightLength ;
            END ;
            IF Wall = wBack THEN
                PercentPos := (Left - cBackWall) / cBackLength ;
            END ;
        roVideo : BEGIN
            IF Width = cTotalWidth THEN BEGIN
                Left := 0 ;
                IF Top < 0 THEN BEGIN
                    Top := 0 ; end
                ELSE BEGIN
                    IF Top > (cTotalHeight div 2) THEN
                        Top := cTotalHeight div 2 ;
                    END;
                    PercentPos := Top / cTotalHeight ; end
                ELSE BEGIN
                    Top := 0 ;
                    IF Left < 0 THEN BEGIN
                        Left := 0 ; end
                    ELSE BEGIN
                        IF Left > (cTotalWidth div 2) THEN
                            Left := cTotalWidth div 2 ;
                        END;
                        PercentPos := Left / cTotalWidth ;
                    END ;
                END ;
            END ;
        end;
    end;

    procedure TRoomObj.WallChange(NewWall : TWall) ;

```

```

var vl_Bitmap : TBitmap ;

begin
  vl_Bitmap := TBitmap.Create;
  CASE RType OF
    roDoor : BEGIN
      CASE NewWall OF
        wLeft : BEGIN
          IF enabled THEN BEGIN
            form1.DoorImages.GetBitmap(0,vl_bitmap);
            IF NOT visible THEN visible := true ;
          END ;
        END;
        wBack : form1.DoorImages.GetBitmap(1,vl_bitmap);
        wRight : BEGIN
          IF enabled THEN BEGIN
            form1.DoorImages.GetBitmap(2,vl_bitmap);
            IF NOT visible THEN visible := true ;
          END;
        END;
        wFront : visible := false ;
      END; END;
    roWindow : BEGIN
      CASE NewWall OF
        wLeft : BEGIN
          IF enabled THEN BEGIN
            form1.WindowImages.GetBitmap(0,vl_bitmap);
            visible := true ;
          END;
        END;
        wBack : form1.WindowImages.GetBitmap(1,vl_bitmap);
        wRight : BEGIN
          IF enabled THEN BEGIN
            form1.WindowImages.GetBitmap(2,vl_bitmap);
            visible := true ;
          END;
        END;
        wFront : visible := false ;
      END; END;
    roPicture : BEGIN
      CASE NewWall OF
        wLeft : BEGIN
          IF enabled THEN BEGIN
            form1.PictureImages.GetBitmap(0,vl_bitmap);
            visible := true ;
          END;
        END;
        wBack : form1.PictureImages.GetBitmap(1,vl_bitmap);
        wRight : BEGIN
          IF enabled THEN BEGIN
            form1.PictureImages.GetBitmap(2,vl_bitmap);
            visible := true ;
          END;
        END;
        wFront : visible := false ;
      END; END;
    roScreen : BEGIN
      CASE NewWall OF
        wLeft : BEGIN
          IF enabled THEN BEGIN

```

```

        form1.ScreenImages.GetBitmap(0,vl_bitmap);
        visible := true ;
    END;
    END;
    wBack : form1.ScreenImages.GetBitmap(1,vl_bitmap);
    wRight : BEGIN
        IF enabled THEN BEGIN
            form1.ScreenImages.GetBitmap(2,vl_bitmap);
            visible := true ;
        END;
    END;
    wFront : visible := false ;
    END; END;
END;
Wall := NewWall ;
glyph := vl_Bitmap ;
vl_bitmap.free ;
end;

procedure TRoomObj.WallObjReposition ;
begin
    CASE Wall OF
        wLeft : BEGIN
            Left := round(PercentPos * cLeftLength + cLeftWall) ;
            CASE RType OF
                roDoor..roScreen : Top := round(-3.41 * Left +
                                                184 - BackTop) ;
            END; END;
        wBack : BEGIN
            Left := round(PercentPos * cBackLength + cBackWall) ;
            CASE RType OF
                roDoor..roScreen : Top := cDefaultBackTop + BackTop ;
            END; END;
        wRight : BEGIN
            Left := round(PercentPos * cRightLength + cRightWall) ;
            CASE RType OF
                roDoor..roScreen : Top := round(3.6 * Left - 1144 +
                                                BackTop) ;
            END; END;
    END;
end;

procedure TRoomObj.RoomObjectRight;
var oldY      : integer;
    vlOffset : integer ;
begin
    IF RType = roVideo THEN BEGIN
        IF Width = cTotalWidth THEN BEGIN
            Width := cTotalWidth div 2 ;
            Height := cTotalHeight ;
            Left := round((cTotalHeight div 2 - Top) * (cTotalWidth /
                                                         cTotalHeight));

            Top := 0 ; end
        ELSE BEGIN
            Width := cTotalWidth ;
            Height := cTotalHeight div 2 ;
            Top := round(cTotalHeight * (Left / cTotalWidth)) ;
            Left := 0 ;
        END; end
    
```

```

ELSE BEGIN
    oldY := Top ;
    vOffset := 0 ;
    IF (Top < 150) AND (Left > 300) THEN vOffset := 25 ;
    IF (Top > 260) AND (Left > 300) THEN vOffset := -50 ;
    IF (Top > 260) AND (Left < 125) THEN vOffset := 25 ;
    Top := Left ;
    Left := cTotalWidth - oldY - height + vOffset ;
    IF Left > (cTotalWidth - width) THEN Left := cTotalWidth - width ;
    IF Left < 0 THEN Left := 0 ;
    IF Top > (cTotalHeight - Height) THEN
        Top := cTotalHeight - height ;
    IF Top < 0 THEN Top := 0 ;
END ;
end;

procedure TRoomObj.RoomObjectLeft;
var oldX      : integer;
    vOffset : integer ;

begin
    IF RType = roVideo THEN BEGIN
        IF Width = cTotalWidth THEN BEGIN
            Width := cTotalWidth div 2 ;
            Height := cTotalHeight ;
            Left := round(cTotalWidth * (Top / cTotalHeight)) ;
            Top := 0 ; end
        ELSE BEGIN
            Width := cTotalWidth ;
            Height := cTotalHeight div 2 ;
            Top := round((cTotalWidth div 2 - Left) *
                (cTotalHeight / cTotalWidth)) ;
            Left := 0 ;
        END; end
    ELSE BEGIN
        oldX := Left ;
        vOffset := 0 ;
        IF (Top < 150) AND (Left < 125) THEN vOffset := -25 ;
        IF (Top > 260) AND (Left < 125) THEN vOffset := 50 ;
        IF (Top > 260) AND (Left > 300) THEN vOffset := -25 ;
        Left := Top + vOffset;
        Top := cTotalWidth - oldX - width ;
        IF Left > (cTotalWidth - width) THEN Left := cTotalWidth - width ;
        IF Left < 0 THEN Left := 0 ;
        IF Top > (cTotalHeight - Height) THEN
            Top := cTotalHeight - height ;
        IF Top < 0 THEN Top := 0 ;
    END ;
end;

function TRoomObj.PackAsRoomString : string ;
var aString      : string ;
    aShortString : string ;

begin
    IF length(Hint) > 5 THEN BEGIN
        aString := copy(Hint,1,5) + ';' ; end
    ELSE BEGIN
        aString := Hint + ';' ;
    END ;

```

```

str(Left,aShortString) ;
aString := aString + aShortString + ';' ;
str(Top,aShortString) ;
aString := aString + aShortString + ';' ;
str(Width,aShortString) ;
aString := aString + aShortString + ';' ;
str(Height,aShortString) ;
aString := aString + aShortString + ';' ;
str(PercentPos:4:2,aShortString) ;
aString := aString + aShortString + ';' ;
CASE Wall OF
    wLeft  : aString := aString + '0;' ;
    wBack  : aString := aString + '1;' ;
    wRight : aString := aString + '2;' ;
    wFront : aString := aString + '3;' ;
    wNo    : aString := aString + '4;' ;
END;
PackAsRoomString := aString ;
end;

end.

```

RDU File Format

RDU room images are stored in regular text files with the default extension .RDD. Each object is stored on a line containing seven pieces of information separated by semicolons.

Example:

Pictu;111;24;44;90;0.14;1;

Field	Description
Name	Up to the first 5 characters of the object name
Left	The X coordinate of the object
Top	The Y coordinate of the object
Width	The width in pixels
Height	The height in pixels
PercentPos	For wall objects, the percent position of the object's Left field from the left side of the wall (How far along the wall is it.). For other objects this is 0. This value is calculated anytime the object is dragged in the room on the MouseUp event.
Wall Type	An integer which indicates the wall the object was on when saved. Values are: Left = 0, Back = 1, Right = 2, Front = 3 or No = 4