Interactive Distance Learning over Intranets

KURT MALY, HUSSEIN ABDEL-WAHAB, C. MICHAEL OVERSTREET, J. CHRISTIAN WILD, AJAY GUPTA, ALAA YOUSSEF, EMILIA STOICA, AND EHAB S. AL-SHAER

Old Dominion University

Many distance learning systems claim to be interactive, but few can offer two-way video, on-the-fly interaction, and application sharing. The authors describe the Interactive Remote Instruction system, which links sites over a high-speed intranet.

Distance learning — education without a central classroom — has helped busy people obtain college credits or complete training they might otherwise not have done. Methods of distance learning range from simple correspondence courses and broadcast TV with reverse audio to specialized video conferencing tools, such as Proshare or Flashback, and Web-based courses.

The current version of Old Dominion University's Teletechnet system, for example, uses broadcast satellite technology with terrestrial audio feedback from students and e-mail to connect the main campus in Norfolk, Virginia, to up to 23 community colleges throughout the state, as well as selected industrial and government sites. More than 3,000 students are enrolled in Teletechnet.

However, limitations in the technologies supporting Teletechnet and similar systems become critically apparent as the demand for them continues to rise. To address these limitations, our research group built the *Interactive Remote Instruction* system, which allows students at

geographically dispersed satellite campuses and community colleges to take a class "together." Access from home PCs through a Windows NT port is planned but not yet available.

IRI improves on Teletechnet technology in four areas:

- *Video resolution*. The limited resolution of Teletechnet's TV images restricts the quality of information that can be presented. IRI offers images with a resolution of 1152 x 900 pixels.
- *Asymmetrical video presence.* Instructors cannot view the students at remote Teletechnet sites nor can students see students at other sites when they speak. With IRI, each course participant's (student's and instructor's) workstation is a window to a virtual classroom. All class participants see the speakers on their screens, so everyone has the same experience.
- *Interaction*. Teletechnet students have limited opportunities to interact both in and out of class. In class, IRI helps each student prepare for assignments and take notes and aids collaboration in group projects.
- *Instructor support*. TV-based classes typically have technicians and camera people managing the connection but little user support is provided. IRI helps instructors prepare a lesson plan off-line, which they can then use to guide class presentations. The class management helps orchestrate and manage an interactive classroom. , The instructor can selectively call on students or check the status of their workstations.
- *Computer simulations.* TV-based systems provide no way to do a hands-on computer simulation of, say, a chemical experiment. With IRI, students can ask questions or demonstrate proficiency by directly manipulating such a simulation.

In this article we describe IRI and the lessons learned deploying it. We first deployed IRI to teach a fall 1995 graduate course in software metrics. We then evaluated it in terms of logistics, reliability, performance, and usability, performing off-line experiments to try out new features and to develop protocols that would improve IRI use. We subsequently reengineered IRI into an open architecture with a published specification as version 1.0, which became available just recently (http://www.cs.odu.edu/ tele/iri). We used version 1.0 to teach a junior-level software engineering course in the fall 1996 semester.

INTERACTIVITY CHALLENGE

The days of the "talking head" approach to video instruction are ending. Digital technologies that support interactive, multimedia virtual classrooms are delivering education much more efficiently and effectively to students in range of situations. Any student who can has links to an intranet can participate in this type of education.

The main problem with the talking head technology is that it did little more than present information that students might easily digest on their own. Only when students put information into a different or larger context, such as interaction in a classroom situation, do they fully develop their creativity and critical thinking skills. In fact, students using TV-based systems complain about the lack of interaction with a faculty mentor, often dropping out because of their frustration at not being able to get rapid feedback to questions. Faculty also cite lack of interaction as the top drawback to televised instruction.[1]

Thus, interactivity is a primary goal of distance learning systems—but it also poses challenges. An energetic instructor might use slides, overheads, paper handouts, photographs, graphics, video, and audio — each of which requires separate tools to deliver. Distance learning systems must make these resources available at each workstation and at the instructor's fingertips, allowing instructors to tailor course content dynamically to the students' needs and learning styles and to select appropriate intervention strategies on the basis of what the system reveals as it monitors students' interactions.

For TV-based systems such interactivity is both complicated and costly. Custom hardware and software are required before students at one site can see students at another. TV-based systems with two-way video require either a high-bandwidth multicast network or if satellite transmission is used, uplink stations at each receiver site. In contrast, computer-based systems can send compressed audio, video, and data streams, as well as two-way video, with microphones and speakers supporting the audio connection—all relatively simply and inexpensively.

VIRTUAL CLASSROOM

Our work in IRI has concentrated on providing a virtual classroom for geographically dispersed students to share a traditional environment. To guarantee a certain level of performance, IRI is implemented on an *intranet*, a network in which access is controlled so as to guarantee system performance.

Before class

The virtual classroom environment supports a variety of roles and session types. The user invokes IRI from a startup screen, as shown in Figure 1, which allows him to define a role and a session type. Supported roles are

♦ *Instructor*. The person in charge of a class, normally the person who teaches, handles grading, and manages a class during lecture time. The instructor controls the class-management policy (the

freedom of individual students to influence the class, the functions and resources available to students, and so on).

- *Student.* Anyone registered to take the class**.**
- A*dministrator*. The person who configures the hardware, registers students, and so on.

Teacher, student, and administrator are *static* roles (persisting throughout the class). There are also two possible *dynamic* roles—presenter and tool controller—which may change during class.

The *presenter* is typically the instructor but can be any class member. At a specific time, the presenter uses one or more tools, some possibly running in collaborative mode, to lead a discussion or present information to the class. By "collaborative," we mean that all students in the class can see the output. The *tool controller* is any person who controls the application the presenter has selected. This collaborative approach is important because it allows everyone in the class to see the same tool actions and results simultaneously.

The startup screen asks the user to specify the session type and in most cases the date the class will be given. Session types include

- *Lecture.* Lecture sessions can be class or class with recording. Selecting "class" starts an interactive virtual classroom on the set of workstations defined by the administrator for that class. Selecting "class with recording" records the class session for later viewing.
- *Review*. In this session type, students who miss a class or wish to review class material can replay the class session, which IRI had automatically recorded in the "class with recording" mode.
- *Lesson planning*. In this session type, a presenter identifies resources or tools (such as application software or files) that will be used in an upcoming class session. This simplifies class start-up time and because needed resources are known beforehand lets software and files be available at each site.
- *Resource addition*. In this session type, the user (instructor or student) identifies resources needed for a class, such as homework, quizzes, surveys, grades, assignments, projects, programs, and syllabi. IRI copies this information from the user's files into the appropriate folders maintained for the course material. The folders become part of *WebBook*, an IRI resource available to students with Internet access from any location (home, for example) and a browser like Netscape 2.0 or higher. Thus, students can access class material outside of scheduled class times through the intranet. WebBook contains instructor's notes and handouts, as well as the notes the student has taken during class. It also contains information about the student taking the class.

Returning to Figure 1, the instructor has activated a resource addition session to add previously prepared Powerpoint slides . The presentation list **identifies resources which have been identified as part of this session which we explain in more detail later**.

During class

Figure 2 shows the computer screen during class mode. (The images in this figure were taken with older video cards. We plan to use cards with better resolution in the future as less expensive cards become available. The large video screen is an NTSC quality image with a high frame rate, which originates at a presenter's workstation**.** If the presenter's workstation is equipped with alternative video sources, the presenter can switch the source for this image to any NTSC video so that VCRs, TVs, and laptops can be used for presentations. Typically only the instructor's workstation would be equipped with an NTSC video switch.

Group discussion

Figure 2 also illustrates IRI's support of group discussions. On the right side are video images of students participating in a discussion. Students can elect to join the discussion by selecting one of the video images, or the instructor can call on them by selecting the Call Student function (bottom function bar), which displays the class roll, and choosing their names. IRI can display up to four videos on all workstations: the presenter's, up to two students', and a site image the instructor selects ("ODU" bottom right). At start-up, the instructor is in the presenter's window (large video image) and no students are on-screen.

Audio is hands-free (no buttons must be pressed to initiate or discontinue a conversation), although students can turn off audio.

In the center of the screen in Figure 2 is a list of class participants, their locations, and the status of their workstations in the virtual classroom. This allows the instructor to call on individual students and tells the presenter that individual workstations are functioning and thus that students are still part of the class. Workstation status is also useful information in troubleshooting network problems, since the instructor can sometimes rectify a problem by restarting the session.

Tool sharing

The IRI environment provides for three types of tools. (We describe the IRI tools in more detail later.)

- *Common X Windows.* These include tools such as Netscape, acroread, Matlab, gdb debugger, and xterm, which were not written to be shared.

- *IRI multiuser*. These tools were written to exploit the shared networking environment of the IRI architecture.
- *IRI single-user*. These tools run only on one user's workstation.

The first two classes are *shared* tools. When a shared tool is in use, the presenter's video image shifts to a smaller window in the top right corner (maly), as shown in Figure 3. In this figure, Netscape (top window) and the IRI slide tool (bottom window) share the main work-area. Along the top left side of the screen is the lesson plan, which lists the tools the presenter will use this class and notes on the objectives of the lesson (taken from the lesson planning session, as described earlier). The presenter can invoke tools as needed from a menu or IRI can preload them before the class session (on the basis of input from the lesson planning session.)

Because IRI incorporates XTV, an X Windows tool-sharing engine,[2,3] any student can operate any tool the instructor uses. A student can take control of the tool by selecting "Tool" (second token from the left along the top shared-resource bar in Figure 3). IRI displays the name of the tool controller on all workstations ("maly" to the bottom right of the token). It then broadcasts all changes to that tool's state to all other workstations.

Shared functions vs. private views and functions

IRI users must understand the difference between what is being shared and what is private. All public resources are identified in a resource bar along the top of the work area. The private functions, which the workstation user controls, are at the bottom of the screen. In addition, IRI identifies the controller of each shared resource for everyone to see. It also keeps a registry of all shared windows with their position and size on the presenter's screen. The presenter can use a synchronization button to force everyone's screen to coincide with her own or let a student synchronize his screen to the generic shared image.

Teleawareness

Users should know the consequences of an operation, particularly if it affects everyone on the intranet. Suppose the presenter wants to launch Netscape, which may take up to a minute to appear on everybody's screen. IRI alerts her to that overhead and asks for confirmation that the operation should continue. It also periodically estimates the time to launch common tools and uses the results to predict the cost. Thus, every time a new tool is launched or a slide is changed in the slide tool, a slide bar indicating the percentage of viewer screens in sync with this action is shown.

Students can create personal multimedia notebooks to type notes during class sessions or to capture part of the screen in the notebook (snapping). Students can annotate the snapped image with freehand drawings, typed or written comments, audio, or hyperlinked text.

The student's notebook need not be open to snap an image, which means that the material being presented can be in full view at all times. IRI periodically time-stamps and saves the notes into the student's private subdirectory. The student can access these notes through the WebBook. At present, WebBook security is limited; only browser password protection is provided to control access to personal information (grades, notebooks). This will change as more effective and convenient security mechanisms are developed for browser/net access. On the Unix side, IRI provides traditional Unix protection (through file permission access).

Design objectives satisfied

IRI's virtual classroom environment not only emulates a traditional classroom, but meets the challenge of making traditional classroom resources easy to access. It has satisfied many of our design objectives:

♦ Be able to run any application (Matlab, debugger, Netscape) in a collaborative mode. Each student can control the application and bring information from various media to the screen.

♦ Enable students to learn more material and learn more easily. With WebBook and class with recording and review sessions, students have access to details of the class without being there. They can absorb not only the lecture, but all the interaction they missed.

♦ Support the complete teaching-learning process, including lecture preparation, lesson planning, presentations, quizzes, exams, grading, homework, studying, projects, instant student and presenter feedback).

♦ Support different teaching styles, including traditional synchronous class, small groups, self-guided, and asynchronous participation.

♦ Support students with different levels of computer literacy. We wanted students with even just a little computer knowledge to get something out of the class and eventually to exploit advanced features as they become more comfortable with IRI.

♦ Give presenters a feedback mechanism that gives the status of the system and the class.

SOFTWARE ARCHITECTURE

Figure 4 shows the software architecture of IRI. Ovals indicate processes, and rectangles contain related processes. Color indicates the degree of survivability : magenta indicates that failure of this process would cause a student to loose some functionality, blue indicates that a group of functions

would be affected , orange means that the workstation would no longer respond to class activities, and red indicates that the entire session would fail.

As the figure shows IRI processes in the same host communicate using Unix sockets,[4] which in the same-host scenario are usually twice as fast as TCP/IP. Session control and class information server are not necessarily on the same host, so we use TCP for their communication.

Class information server

The *class information server* maintains information about all IRI classes. Information is either static (on courses, instructors, and students) or dynamic (on classes in progress, telling which students are on-line). Students registered for a class can join a session at any time. The server assigns a unique identifier for each active class and for each attending student. These IDs are the basis for creating multicast group addresses and for identifying the sources and destinations of messages exchanged among IRI processes.

Session control

A separate *session control* process is created whenever a student joins an ongoing class. Session control immediately contacts the class information server using a TCP connection, sends information about the student to the server, and receives back its assigned ID and information on other attending students. The TCP connection is closed after the information is exchanged. In Figure 4, the dotted line connecting the server and session control represents the temporary nature of this connection. Session control then creates and manages all other processes using Unix socket connections (solid lines). This careful control of communication channels is necessary both for reliable operation and scalability.

The last task of the session control is to establish the graphical user interface, creating and managing the main window and smaller video windows. IRI's GUI is written in Motif/Xlib. Users access all resources and applications through the main window, while each tool (such as the personal notebook) creates and manages its own interface.

After initial contact with the class information server the session control processes cooperate among themselves to control and manage all IRI resources (audio, video processes, for example) in a distributed fashion **using UDP (User Datagram Protocol) /IP multicast communication**.

Tool sharing engine

The *tool sharing engine* in Figure 4 is the XTV X Windows tool described earlier. IRI allows X tool sharing by intercepting and altering the traffic between the X clients (applications) and the X

servers. At any moment, only one user (instructor or student) can interact and provide input to the application. IRI multiplexes output to all participants and displays it locally though the X server on that workstation.

Single-user tools

Single-user tools are available at each host and run independently. One example is the personal notebook we described earlier, which can run as a stand-alone tool when students edit notes outside class. Another example is the *class management tool illustrated in Figure 2*, a single-user tool that displays information about the class to the instructor, such as current attendees, their locations and status of the processes on their machines.

Multiuser tools

Multiuser tools are *collaborative-aware*; that is, they know the identities of all participating users and can distinguish and accept input from all of them. Examples are the global pointer and the slides and survey tools. The *pointer* tool lets the presenter point to any place on any window to direct viewer attention.

The *slide* tool lets the presenter display slides on all workstations. Although the presenter can use any X tool, we have found that an easy-to-use, efficient slide presentation tool is also needed. The presenter typically identifies the class material to be presented with the slide tool during lesson planning. Then, before class begins, the tool downloads all materials to the site file servers, thus reducing network-induced delays during class. Slides can be randomly or sequentially accessed. The slide tool also supports annotation and the use of a pointer to focus attention. *Annotation* lets the user type additional text or highlight parts of a slide using a freehand drawing mode in selectable colors. In Figure 3, the presenter has annotated the slide by circling CS 350. The pointer, controlled by the presenter (maly), is pointing to the URL for the class.

The *survey* tool helps the presenter get feedback from the students. The presenter can use the survey tool at strategic points to decide the next course of action. When invoked, it tabulates student responses anonymously, and at the presenter's discretion, can display them at each workstation.

Reliable Multicast Protocol server

The *Reliable Multicast Protocol server* uses a protocol jointly developed by the University of California at Berkeley and West Virginia University.[5] The server creates a process to handle the communication needs for each multicast group. As Figure 4 shows, the session control, tool-sharing engine, and multiuser tools use the server. Every session control for an ongoing class joins the

reliable multicast group RMPSGrpX, where *X* is a unique number derived from the class ID allocated by the class information server and the IP address of the workstation where the class server is running. The Reliable Multicast Protocol delivers messages sent by any member of RMPSGrpX to all group members.

Audio Processes

Audio processes handle voice data. One audio channel is reserved for the instructor's use at any time, while students share the remaining audio channels. The number of student audio channels per class depends on the number of students in that class. There should be enough channels to allow any student to speak at any time. For the fall class we limited the number of simultaneous speakers to four, since the class had 22 students. When a student wants to speak, the *send* process sends the captured audio data to a free channel. IRI uses audio silence detection to allocate/deallocate the tokens for the audio channels.

The *receive/mix* process receives audio data from all audio channels and uses our audio mixing algorithm to play the audio data. At each site, one workstation sends the mixed audio to the site speaker. To avoid feedback, audio is not played in the origination room.

Video processes

Video processes capture and display the presenter's and students' images as well as the image of the site the instructor selected. Again, one video channel is reserved for the presenter , but (currently) the remaining channels are divided among the students and the classroom site videos. When a student wants to show her picture, IRI allocates the token of any available video channel and the *send* process sends the captured images to that channel. The send process captures and sends the image of the presenter at 30 frames per second (fps) and of the students at 10 fps.

The *receive instructor* process displays a presenter's image in the large window if no shared application is running (as in Figure 2); otherwise it displays it in the small window in the top right corner (as in Figure 3). A *receive student i* process receives a student image from the video channel and displays it on the screen in a small window such as those in Figures 2 and 3.

Each workstation at each site participating in the class captures an image of the student at five frames per second and sends it to the site video channel. The *receive Site i* process receives a site image from the site video channel and displays it in a window (bottom right in Figures 2 and 3).

IRI version 1.0 has two student video channels and one site video channel. Like audio, the number of video channels, the competition for screen area, the limitations of what humans can comprehend at a time, and network bandwidth limitations dictate the number of useful channels. In

future versions, IRI should be able to monitor network conditions and adjust the frame rate of individual video streams in response to network loads.

IP multicast

IP multicast is used to transmit audio and video data. IRI maps each continuous media channel to a unique IP multicast group. In choosing IP multicast group addresses and port numbers, it uses a scheme developed by Sassan PejhanPejan,[6,7] in which the IP address of the host machine and the class ID combine.

To send data to a particular channel, the send process (in both the audio and video processes) sends UDP messages to the corresponding IP multicast group. To receive data from any particular channel the *receive* process joins the corresponding IP multicast group using the Internet Group Management Protocol.[8] IRI uses raw IP multicast to send audio and video streams as UDP messages.

Design objectives satisfied

The IRI software meets several of our design objectives.

Performance and scalability:

Our main objective here was to be able to scale to 100 students at 10 sites with acceptable performance.

We conducted benchmarking experiments to evaluate the performance and scalability of the Reliable Multicast Protocol server.[5] We focused on this component because without it we would not be able to use IRI in an educational setting with O(100) students per class. The results support our approach and make us believe we can reach the goal of operating a large-scale virtual classroom. For example, using our current setup, IRI version 1.0, with the throughput of multicasting 4 Mbytes of data from one workstation in Norfolk to five workstations in Virginia Beach (about 20 miles away) is 6.81 Mbits per second. An FTP transfer of the same data between the two sites is 7.14 Mbps, making throughput with the Reliable Multicast Protocol server about five percent lower. However, the aggregate throughput of the server is obviously much higher (5×6.81 Mbps) than that with an FTP transfer because the Reliable Multicast Protocol server lets users transfer data to multiple destinations simultaneously.

The throughput drops to 6.21 Mbps (9 percent lower) when we add five receiving workstations and to about 5.41 Mbps (21 percent lower) when 23 workstations at both Norfolk and Virginia Beach are participating and receiving 2 Mbytes of data from a single source.

Three factors contribute to this performance level. First, we used multicast communication to eliminate unnecessary network traffic and processing overhead caused by unicast or broadcast

communications and to avoid limitations imposed by point-to-point communication protocols such as TCP (the maximum number of simultaneous TCP connections is 64, for example). Second, we used a distributed rather than centralized approach in session management to avoid performance bottlenecks and a single-point of failure. Finally, we used the Reliable Multicast Protocol server, which has a token-ring configuration to distribute the load and the buffering requirements imposed by the retransmission operation in reliable multicast communication.

Clearly, if we extrapolate these figures to 10 sites and 100 students, our target ceiling for each class, we would have performance and scalability bottlenecks. However, we are working with GlobalCast Communications, Inc. (http://www.gcast.com) to improve the performance of the Reliable Multicast Protocol to handle 1,000 sites. We expect these improvements to meet our objectives for future versions of IRI.

Response time

We wanted a system response time of one second or less for tool operations such as flipping a slide, sending a survey, or calling a student.

We used several techniques to reach this goal. Users can initialize the slide tool, for example, to preload intended slides to remote machines before class begins. Thus slide flipping does not exceed one second. Also, the slide and survey tools use caching to save previously used documents (such as slides) in memory. This allows users to reload documents in less than one second.

For off-the-shelf X tools such as Xcalc and Netscape, response time is higher because of the overhead from the tool sharing engine processing, but increases only slightly as the number of participants increases. For example, Xcalc takes about seven, eight, nine, and 11 seconds to start for two, four, eight, and 20 participants, respectively. Netscape requires about 50, 53, 55, and 58 seconds for two, four, eight, and 20 participants, respectively.

Reliability

The objective here was to have a system which would be more robust should parts of the the system fail. That is, we wanted IRI to continue to function even with some loss of functionality, if at all possible.

IRI meets this object in several ways. The multiprocess architecture protects key IRI components, such as session control and the Reliable Multicast Protocol server, from crashing if a tool fails. Session control automatically restarts a crashed or failed process when the session control receives a SigPipe signal (telling it a failure has occurred).

HARDWARE ARCHITECTURE

The hardware architecture is in essence the intranet and its links, which provide an effective and efficient communication mechanism for all class participants. The wide-area 10-Mbps bandwidth per class is guaranteed through ATM links and dedicated links provided by Cox Fibernet, a cable TV service provider. For the local site (college classroom with up to 15 workstations), switched Ethernet provides 10 Mbps to each workstation.

Site layout

Figure 5 shows the layout at a local site. Each student desktop consists of a workstation equipped with a microphone, speaker, video camera and video board (large dot). In addition, a single student workstation will function as the audio server for the whole site, sending all audio from other sites to its speaker.

The instructor's desktop is more elaborate. The workstation has a microphone, a speaker, and a stylus, which the instructor can use as a pen to annotate figures more easily. . It also has two video boards instead of one. The first board connects to a class camera, which provides a site overview. The second board connects to an NTSC signal switch, which allows the instructor to switch between a video camera displaying his image, a VCR, and (for future use) a scan converter that will enable displays on a home-based PC.

The Ethernet carries the instructor and student videos, a site image, instructor's and students' audio, and data traffic for tool management to the router, which then sends these data to remote sites.. On a student workstation, the video board receives and compresses the student's image captured by the local camera. Currently we use CellB compression for the student and class videos, and mu-law encoding for all audio streams.

To avoid additional specialized audio hardware, each site has only one speaker system (fed by the audio server for that room) which plays any nonlocal audio. That is, everybody at a particular site hears a speaker in that room nonelectronically, but all other sites use the speaker system. All students and the instructor have their own microphone which can filter out noise from the room.

Cost

The current cost of a student workstation is less than $8,000, including the video board, assuming a regular educational discount. The instructor's workstation (which can also be the server for a site network) is about $15,000. An average site will handle up to 12 student workstations. At Old Dominion University we currently use Sun Sparc 5 workstations for both the students and instructor. The Sparc 5 has a 24-bit frame buffer to simultaneously handle complex tools (such as

Netscape) and video streams. IRI runs in 24 bit mode, but for faster performance displays video streams in 8-bit mode. The operating system is Solaris 2.5 running the Common Desktop Environment as the window manager.

Design objectives satisfied:

When the system is in full production mode, we will meet our objective for cost-effectiveness, which is to make the system less expensive or comparable in cost to traditional TV distance education. At present, we use IRI for only one class per semester. The rest of the time is shared between IRI development and use as a general CS lab. In full production mode, we expect at least 10 classes per semester. At this level of use, the equipment, amortized over four years, is about $200 per student. Subtracting the remaining 80 hours a week the equipment is available as a general lab, the per-student cost to use IRI equipment is only $40. Further, the university is currently installing an ATM network to support Teletechnet and expects a significant cost reduction from the old satellite-based system. Hence, sufficient bandwidth will be available to support IRI-based courses at a communication cost comparable to that for TV courses.

LESSONS LEARNED

For the fall 1995 semester, we used IRI to teach a graduate computer science course in software metrics with 12 students (enrollment was limited by available hardware and IRI's experimental use). We subsequently evaluated reliability, performance, and the user interface for both students and instructors. We used the results of our evaluation to revise IRI significantly. We used IRI again for the fall 1996 semester to teach a required junior-level software engineering course with 22 students (enrollment again limited by available workstations). We used this opportunity to experiment with different approaches to teaching and learning in a real-time network environment.

At each stage, particularly in the fall 1995 application, we learned lessons that allowed us to evolve a set of design principles and expectations. We also learned some important lessons about user acceptance.

Fall 1995 application

Most of our lessons came from the fall 1995 application. Specifically, we encountered four problem areas, all of which we resolved in the fall 1996 application.

Attention focusing

Several developers and researchers observing the fall 1995 class found that students' attention tended to wander. Because the workstation screen can contain much information, the presenters

must have mechanisms for focusing listeners' attention to relevant areas on the screen. We first attempted to use the notebook as a public annotation tool. However, we soon found that this was too cumbersome. It involved bringing up the tool with the material to be annotated, bringing up the notebook, snapping an image of the material into the notebook, making the notebook public, and finally annotating the copied image.

We also found that we needed a global pointer to indicate the part of the screen the presenter wanted the viewer to focus on.

Presentation synchrony

Because of delays from networking and computing bottlenecks, material displayed at the presenter's workstation may appear at student workstations at different times (varying by a few seconds). That is, the *presentation synchrony* may be off. If the presenter begins talking about this material before it appears at all workstations, students without the image may be confused. In addition, a process on a student workstation may have crashed without the presenter knowing.

Moreover, we found that issues of presentation synchrony are not just technical. Students could change their screen layouts (as when note taking) and hide the information the presenter wanted to present.

We recognized that autonomous control of individual workspace is important to a student's acceptance of IRI, but we had to achieve presentation synchrony. Thus, we had to strike a balance between student initiative and having a synchronized common viewpoint controlled by the presenter. We did so by allowing students to rearrange their screens to suit them, but also allowing the presenter to bring all audience screens into an arrangement similar to his, with a single mouse click . Each student can also have the same view as the presenter with a single click.

Response time

In the fall 1995 application, we discovered problems with the amount of time taken to respond to a request because of both the protocol used and the system architecture.

The protocol we used at that time required several steps to get the presenter's attention through an audio channel. The student had to first click on an attention button and then wait for the presenter to recognize that the student had asked for attention, open up a call-management tool to find which student(s) had a question, and select the student from the class role. Only then could the student speak. Moreover, the presenter was often unaware for some time that an attention button had been activated.

This violated our objective of one-click operations. In fact, we found this protocol so inhibiting that we changed it midsemester so that students could open the audio channel without waiting for presenter permission. We first had them click on an audio button before speaking, but they often forgot, so finally decided on voice-activated audio channels.

Another problem was the system's response time. Because we used postscript for class lecture material, even the simple task of going to the next page in a presentation often required several seconds. This dead time was awkward for both the students and the presenter. The time to launch a new application, such as Netscape, was also unreasonably long, which tended to discourage its use. This was one motivation for including a lesson planning component; now tools such as Netscape can be started automatically before class begins.

User acceptance

Student response has been positive. Students commented that relative to conventional TV classes, the material presented was much more readable. We have had more demand for IRI classes than we can accommodate. Several students have expressed a desire to transfer from conventional classes to the IRI class.

We found several interesting facets of user acceptance.

Individual control

Some degree of user acceptance is necessarily based on individual preference, which tells us that IRI must remain flexible to be appreciated. An example from our 1996 software engineering class illustrates this point. It raises many questions about both student comfort and the benefits of using video in education.

In an IRI class, the instructor can turn on any student's video without warning—similar to calling on a student in a regular class. Students were generally intimidated by this feature. Several students asserted that turning on their video image felt less like answering a question from a desk and more like being singled out to make a presentation in front of the class.

We discussed giving students final control of their video through a software option, but the instructor preferred to use verbal clues to warn individuals that they would be on screen, saying something like "Mary, we'd like to see your homework on question 5 and I'm going to turn your camera on now. Ready?" Mary could then respond verbally that she is or is not ready. If she wanted, she could also cover the camera lens to avoid being seen.

If this is the right solution to student discomfort with video, it illustrates that solutions lie not necessarily in letting the technology (and programmers) control decisions, but in finding appropriate

procedures for using the technology's capabilities. IRI can easily allow the instructor to decide what students can access during class and can allow students final control over their video. But many fundamental decisions like this should be left open, not mandated by IRI.

Collaborative tool manipulation

The ability to collaboratively manipulate shared tools adds a new dimension to the classroom experience. In the software metrics class, for example, IRI's tool-sharing engine enabled a discussion about lines of code. For homework, each student wrote a simple program to count the lines of code in any Pascal program. During class, students ran their programs against a sample program. The class then discussed differences in the answers and showed their solutions for group analysis and discussion. They also proposed new solutions, and some students changed their code and explored the effects of changes during class. This kind of spontaneous collaboration would have been impossible in a traditional classroom, where resources are limited.

Teacher preparation

After teaching two classes with IRI, we are in a position to comment on preparation time. We caution, however, that the same person taught both classes, so generalizations are questionable.

In a conventional classroom, both preparation time and how faculty choose to present material vary greatly. Some use chalk and a blackboard with a few printed handouts. Others use PowerPoint or its equivalent, along with other tools, providing printed copies to students before class. For faculty operating in the latter mode, preparation time would not be fundamentally different. During the software engineering course, we used the same slides (prepared in Powerpoint) in both the IRI and conventional classes, so preparation time was in largely identical for this material.

In a conventional class, a instructor often uses a blackboard to respond dynamically to student questions or other "on-the-fly" discussions. In the IRI class, the instructor experimented with several presentation tools to achieve the same end—sometimes a drawing tool (part of the slide tool) and sometimes a simple text editor (emacs). We found that we also took slightly more time preparing surveys and identifying other uses of IRI's capabilities. For our fall 1996 course, for example, we prepared survey questions as part of a Powerpoint slide presentation, reminding us to activate the survey tool at the appropriate point.

Thus, although some extra time may be needed for IRI, it can be attributed to inexperience and exploring the most effective way to support "on-the-fly" discussions. After several uses, a instructor should be able to establish a standard repertoire of techniques to exploit IRI capabilities. The University also has an instructional technology center to help instructors s optimize their use of

distance education. Additional effort will probably be required, but the presentation of some technical material should be simpler because the instructors can easily display samples of code and code review checklists, for example, on each workstation. Presenting the same material in a conventional classroom would require more preparation and it would probably be in summary form, which might be less effective.

System reliability

As we pointed out earlier, the system is not likely to be used if no one believes it will work—hence reliability affects user acceptance. However, we cannot yet draw any conclusions about how users viewed the system because the users themselves have different priorities. For example, when a file server crashed just before a class and we couldn't reboot, the typical user reaction was a mixture of despair (our part) and euphoria (the students' part). This division isn't likely to change because to many students a missed class will always represent an unexpected holiday and instructor's will always be concerned about covering all the material effectively in the allotted time. Reliability will have a different meaning to the two groups since they often may not have the same objectives. Needs of both groups must be considered in evaluating the reliability of IRI.

At present both instructors and students see IRI as an experimental system and thus are more apt to tolerate its failures.

Our observation is that an IRI classroom tends to be more unreliable than a conventional classroom, which is canceled only for rare events like severe weather. However, the unexpected (and uncontrollable) reasons like weather, but the difference in reliability is not significant and we expect it to decline as the system matures.

CONCLUSION

IRI offers unique capabilities for education. Whether it helps students learn more material more easily we don't yet know for certain. However, anecdotal evidence and the results of student surveys and observations by outside instructors indicate that IRI could greatly improve the distance learning experience. The flexibility permitted by the collaboration technology really does foster an interaction that is difficult to achieve with other distance learning systems. Giving both instructors and students the ability to respond dynamically to class events (questions or ideas) and to enter into on-the-fly discussions is what distinguishes IRI from other approaches to distance education that use canned material (such as Web-based or correspondence courses). Bringing people together allows unexpected things to occur; we believe this is a key part of education.

Enhancements

We will continue to refine IRI though use in more classes and by more diverse groups of instructors and students. On the basis of user feedback so far, we are improving the record/playback feature to better synchronize the various video, audio, and Xtraffic streams during playback. The new implementation of playback will allow fast forward and backward.

We are also planning a cross-platform implementation that will let PC users running NT to participate in the virtual classroom with other members using Unix and to share Windows applications as well as X applications.

Another major effort is under way to implement agents for handling clients with different levels of capabilities and resources in the same session. This is essential to our target of extending IRI not just to sites with local LANs but to home sites, where facilities would typically consist of a multimedia PC connected by an ISDN line or a cable modem to the virtual classroom.

Exploratory efforts

We continue to explore questions on how best to exploit this technology's potential for both students and instructors. In addition to refining IRI functionality as the system is used, we must develop procedures and approaches for using the existing functions more effectively.

On the development research side we are exploring support facilities, including support for group projects to emulate a typical lab with several groups working at the same time and automated help to provide off-line and on-line tutoring. We will also seek new paradigms for learning. We are just beginning to see how IRI's ability to monitor and record user interactions can help support these new paradigms. Future work will be directed at user interface models for shared/private learning environments and the use of collaboration technology to provide human and programmed student assistance.

Application extensions

Finally, we are looking at new application areas. The most significant is industrial training. The three steps necessary for a successful, cost-effective training program are a high performance intranet with multimedia workstations, IRI software, and instructors who can prepare courses that properly use the technology. While IRI obviously cannot provide the hands-on experiences needed for some training, when appropriate, it can significantly reduce both travel time and costs for those being trained. Because IRI, in addition to being a multimedia system, is workstation-based, once the facilities and networks are available, they can support such things as on-line consulting to repair hardware in the field. Specialists from multiple sites can form a unique team to resolve the problem, while people at additional locations can observe the process for their own training.

Moreover, because these sessions can be recorded, they can be available for future review by individuals at any time and for incorporation in formal training classes.

Industrial training is just one of many applications that can benefit from truly interactive distance learning. As multimedia technologies improve, we expect to see more IRI-like systems in a variety of applications.

REFERENCES

1. T. Clark, "Attitudes of Higher Education Faculty Toward Distance Education: A National Survey," *American J. Distance Education*, Vol. 7, No. 2, pp. 19-33.
2. H. Abdel-Wahab and M. Feit, "XTV: A Framework for Sharing X Window Clients in Remote Synchronous Collaboration," *Proc. TriComm '91: Communications for Distributed Applications & Systems*, IEEE Press, New York, 1991, pp. 159-167.
3. H. Abdel-Wahab and K. Jeffay, "Issues, Problems and Solutions in Sharing X Clients on Multiple Displays, *J. Internetworking Research and Experience*, Mar. 1994, pp. 1-15.
4. W.Stevens, *TCP/IP Illustrated, Vol. 3: TCP for Transactions, HTTP, NNTP and the UNIX Domain Protocols*, Addison-Wesley, Reading, Mass., 1996.
5. B. Whetten, T. Montgomery, and S. Kaplan, "A High Performance Totally Ordered Multicast Protocol," *Theory and Practice in Distributed Systems*, 1994.
6. S. Pejhan, A.Eleftheriadis, and D. Anastassiou, "Distributed Multicast Address Management in the Global Internet," *IEEE J. Selected Areas in Communications*, Oct. 1995, pp. 1445-145x.
7. H. Abdel-Wahab, "Working Group Report on Distance Learning: Enabling Technologies," *Proc. IEEE*, IEEE Press, New York, 1996, pp. **XX-XX**.
8. S. Deering and D. Cheriton, "Multicast Routing in Internetworks and Extended LANs," *ACM Trans. Comp. Systems*, May 1990, pp. 85-110.

Figure 1: IRI startup screen. The user, who is the instructor role is adding slides through a resource addition session.
Figure 2: Screen with no shared tools. In this mode the presenter's image is in the main window.
Figure 3: Screen with shared tools. The presenter's image has moved to the upper right corner, while the main workspace contains a window running the slide tool and a window running Netscape.
Figure 4: IRI software architecture. Most processes connect using Unix sockets instead of TCP communication.
Figure 5: Layout of a typical site with instructor and student desktops. The dots represent video boards, one for the student and two for the instructor.

IRI: UNIQUE SUPPORT FOR THE CLASSROOM ENVIRONMENT

Many products support distance learning, each with specific capabilities and target audiences. We believe, however, that no commercially available system combines these features into a total environment specific to educational needs. As the main article describes, IRI offers a fully interactive (audio, video, and tool sharing) system that can support 100 simultaneous participants. It also supports the entire learning process, aiding class preparation (for both students and instructors),

providing asynchronous views outside class (through the WebBook) and facilitating group collaboration for studying, projects and discussions.

Many systems provide digital video-conferencing facilities. In this respect IRI is not unique. Some provide video- and audio-conferencing together with an electronic whiteboard. For example, Xing Corp.'s StreamWorks (http://www.xingtech.com) provides full motion video and CD-quality audio. However, it does not offer any planning or staging capabilities, as does IRI**.** Cornell University's CU-SeeMe (http://cu-seeme.cornell.edu) allows one-to-many or one-to-one live audio conversations, but with a modest number of on-screen video windows presented simultaneously and a whiteboard only for sharing data, not applications.

Other systems focus on the World Wide Web to support asynchronous learning activities. The Internet University (http://www.caso.com/iu.html) arranges online study resources by Internet delivery and is just one example of many universities that put individual courses or even entire programs on the web. The material can be anything from supplements to lectures given outside the Web to completely self-contained, self-paced courses. However, most of these Web-based courses lack the instantaneous interactivity needed to make distance learning meaningful.

Some systems allow for synchronous collaboration through the use of shared applications. The Share system[1] provides a set of programs for sharing audio, video, text, and graphics. However, it operates over ISDN lines, supports sessions with relatively few participants, and is designed for general purpose groups. Intel's Proshare video conferencing system (http://www-us-east.intel.com/comm-net/proshare) provides audio, video, and application-sharing mechanisms, as well as a private and public notebook for each participant but currently video is only 320-by-240 or 160-by-120. In addition, neither were developed with the special needs of the classroom in mind so the idea of automatically prestaging files and tools through lesson planning does not exist.

Some systems support conferencing on a very large scale. For example, the Open University's KMi stadium (http://kmi.open.ac.uk/stadium) is an experiment in very large scale telepresence at both live events and on-demand replays. It is intended to support a variety of telepresence events and includes several ways for attendees to make their views known, including voting buttons and enhanced audio effects such as applause and laughter. It provides a classroom view and a full-screen slide show and allows a student to replay a previously-recorded session. However, it does not support application sharing.

Thus, as these examples show, while existing systems support various domains in unique ways, IRI is the only system we know of that supports the specific needs of a classroom environment.

REFERENCES

1. G. Toye et al., "Share: A Methodology and Environment for Collaborative Product Development," Int'l J. Intelligent and Cooperative Information Systems, June 1994, pp. 29-53.

Kurt Maly is a Kaufman professor and chair of computer science at Old Dominion University. His research interests include modeling and simulation, very high performance network protocols, reliability, interactive multimedia remote instruction, Internet resource access, and software maintenance. Maly received a PhD in computer science from the Courant Institute of

Mathematical Sciences, New York University. He is a member of the IEEE Computer Society and the ACM.

Hussein Abdel-Wahab is a professor of computer science at Old Dominion University, an adjunct professor of computer science at the University of North Carolina at Chapel Hill, and a faculty member at the Information Technology Lab of the National Institute of Standards and Technology. He was the principal investigator in the design and implementation of XTV. His main research interests are collaborative desktop multimedia conferencing systems and real-time distributed information sharing. Abdel-Wahab received a PhD in computer communications from the University of Waterloo. He is a senior member of IEEE Computer Society and a member of the ACM.

C. Michael Overstreet is an associate professor of computer science at Old Dominion University. His research interests include remote instruction technologies, model specification and analysis, distributed simulation, high-performance networking, and static code analysis to support software maintenance. Overstreet received a PhD in computer science from Virginia Tech
He is a member of the IEEE Computer Society and ACM and was the last chair of the ACM's Special Interest Group in Simulation (SIGSIM).

J. Christian Wild is an associate professor of computer science at Old Dominion University His research interests include software engineering, knowledge-based systems, distributed computing, distance learning systems and software reliability. Wild received a PhD in Compute Science ? from Rutgers University. He is a member of **ACM, the IEEE Computer Society, and the American Association for Artificial Intelligence.**?

Ajay K. Gupta is the director of computer resources and a member of the computer science faculty at Old Dominion University. His research interests include network performance issues, security in local and wide area networks, and computer-supported collaborative work in a networked environment. He received an MS in computer science for Old Dominion University. **Alaa Youssef** is a Ph.D. candidate in computer science at Old Dominion University. His research interests include quality of service support for distributed multimedia systems and computer-supported collaborative work. Youssef received an MSc in **computer science** from Alexandria University.

Emilia Stoica is a PhD candidate and research assistant in computer science at Old Dominion University. Her research interests include synchronization of heterogeneous streams in multimedia systems, multimedia architectures and shared computer-supported workspaces. Stoica received an MSc in computer science from the Politechnical Institute of Bucharest.

Ehab S. Al-Shaer is a PhD candidate and research assistant in computer science at Old Dominion University. His research interests include efficient monitoring for large-scale distributed systems and adaptive transport multicast protocols. Al-Shaer received an MSc in computer science from Northeastern University.