# A Distance Learning Laboratory for Engineering Education

**Clinton D. Knight, Stephen P. DeWeerth**

**Georgia Institute of Technology**

## Introduction

The World Wide Web (WWW) got its start as a publishing medium and soon evolved into a large worldwide repository of heterogeneous material[1]. Educators quickly realized the potential of the Web for "distance learning," as hypertext material and multimedia segments could be delivered across computer networks via a simple, user-friendly interface. And while hypertext certainly promotes efficient learning and caters to diverse learning styles, WWW-mediated access to static information is only the beginning. The ultimate use of the Web will be as a software delivery mechanism[1]. Now that the Java language and development tools are maturing, truly interactive WWW applications are becoming feasible[2].

Distance learning is one of the most promising applications of the Internet, and many universities are creating new correspondence courses with WWW delivery in mind. Certainly the Web is a phenomenal avenue for navigating efficiently through course material. The distance learning model falls short, however, in a critical component of any electrical engineering curriculum: instructional laboratories. This need can be partly filled by electronics simulation projects, but there is no substitute for real measurement data and hands-on bench experience.

We present a system that provides access to and control of systems across the WWW. Specifically, our system delivers remote access to electronic test equipment and is currently being used at Georgia Tech in graduate electronic circuits classes. Applications of remote circuit testing abound in instructional and research laboratories, and the concepts developed are useful for more generic applications.

## Overview

The model for the remote testing system is a device under test (DUT) connected to a set of GPIB-controlled test instruments via a switching matrix. A remote user can extract data from the DUT by configuring the matrix and stimulus instruments and then querying response instruments, all from a remote location. The graphical capabilities of Web browsers allow the system to offer data plotting for quick analysis and further testing.

Our first remote testing system required the user to submit a script via a WWW page, which was subsequently interpreted by the server and used to control GPIB-connected instruments. This "batch-mode" system worked well for large classes, but debugging a test program was often difficult without visual feedback from the instruments. Using the same hardware configuration and the Java programming language, we have created a fully interactive remote testing system that operates across the WWW. The system recreates the “look and feel” of the laboratory bench as measurement data is fed back to the user in real time.

The topology of the interactive system is shown in Figure 1. A remote user directs a Java-enabled browser to load the remote testing applet. The applet opens a network connection back to a repeater daemon on the WWW server which, after handling user authentication and scheduling, forwards the connection to a workstation in the testing lab. Connected via GPIB to the workstation is a suite of test instruments, tailored in this case to evaluating custom analog integrated circuits. A daemon running on the GPIB host accepts the transmission from the WWW server, completing the two-way link back to the user applet.

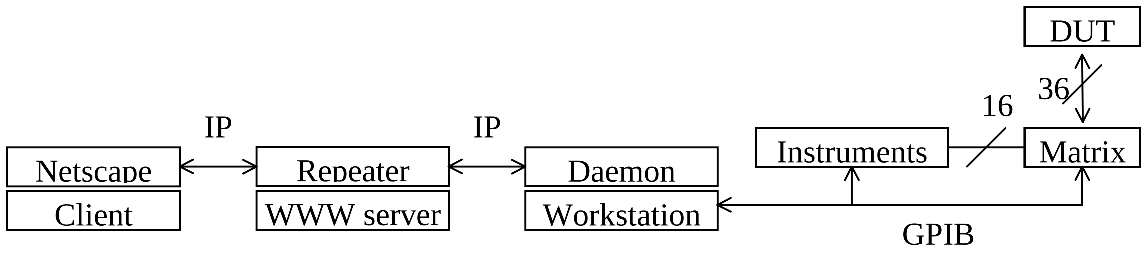


*Figure 1. System diagram. (IP=internet protocol, GPIB=general purpose interface bus)*

Once network connections are established, the user may begin configuring the testing suite for a specific experiment. Connections between instruments and the DUT are accomplished via a switching matrix. Measurement data is fed back to the user in a continuous stream. The applet supports both manual measurements and looping constructs and can display data graphically, enabling the user to generate and interpret test results quickly. The state of the system may be saved and restored later so that a specific test may be easily repeated.

## System Design Issues

The principal goal of this work is to create a “distance laboratory” by providing access to electronic test equipment to a remote user. The remote lab concept is not new[3,4,5], but previous implementations have suffered various limitations. Several problems must be overcome in designing a remote lab[3]: availability of  widely used operating systems and programming languages, laboratory scheduling and availability, safety and fail-safe design, and development of control algorithms for remote use.

The first question, that of operating systems and languages, is answered by implementing the system using the WWW. The Web provides platform independence, a ready-made hypertext environment, and a gentle learning curve for new users, and the underlying Internet promises universal access. The user interface is written as a Java applet, thereby allowing persistent network connections within the context of the otherwise connectionless WWW.

The scheduling problem is also easily solved using the WWW. Via a Web page, users can reserve blocks of time in the lab. If not previously reserved, the lab is available on a first-come, first-served basis. The system maintains open network connections to all users waiting to use the lab equipment and updates each user with the queue status continuously.

Safety issues are addressed in various ways. Voltage and current limits are built into the stimulus instrument libraries, but they may be disabled by privileged users. This prevents accidental or malicious destruction of the DUT while still supporting special needs. A global disconnect feature enables a panic response in case a test goes awry. The interface prevents connecting more than one stimulus to the same node, preventing harmful contention between instruments.

We are developing algorithms for remote control of equipment. A key question, especially in a varying-bandwidth environment like the Internet, is synchronization. The state of each lab instrument must always be correctly represented at the remote site, except as prevented by network latency. It is inefficient to continuously transmit the entire control parameter set for each instrument, so a simple acknowledgment scheme is used. When user logs into the lab via the testing applet, all the instruments and the user interface controls are set to a known initial state. Each command or command group sent to a specific instrument is followed by an acknowledge request. An instrument is only considered synchronized when all of its pending requests have been answered by the lab server. In the user interface, a red/green sync indicator apprises the user of the status of each instrument, and a global sync indicator shows the status of the entire testing system. On the campus network, a typical round-trip sync delay for a simple instrument configuration command is less than one second.

## User Interface

In designing a remote circuit testing system, the most important question to be answered is the operation of the interface itself. The goal of remote testing is to put the user “in the control room,” with full and immediate access to all pertinent system parameters. It should be configurable to a user’s tastes, from presenting only a minimal set of control objects to showing the entire system state at once. To this end, some inspiration has come from published user interface design guidelines from the supervisory control and data acquisition (SCADA) systems used by utility and manufacturing companies to control large processes[6,7]. The user interface should be graphical, as the systems under control as well as their interconnectivity can be represented pictorially (Figures 2,3). The interface should use a windowing paradigm to maximize flexibility. The interactive remote testing system displays the front panel controls of each instrument in its own optionally-visible window, allowing the user to see only the instruments being used. By segregating the controls of each instrument, the look and feel of the laboratory is preserved. The lab veteran will feel less cut off from the instruments while the testing novice will gain valuable testing skills .

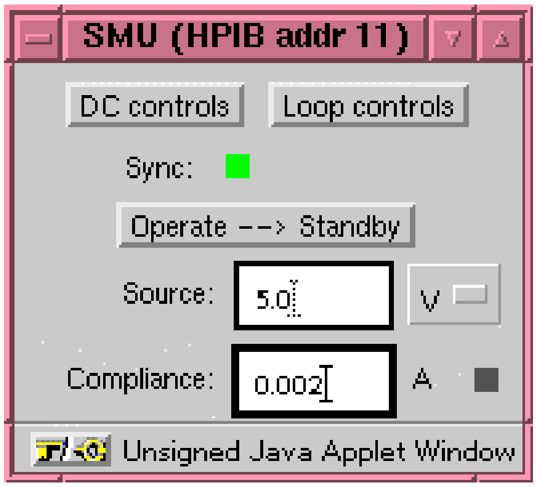
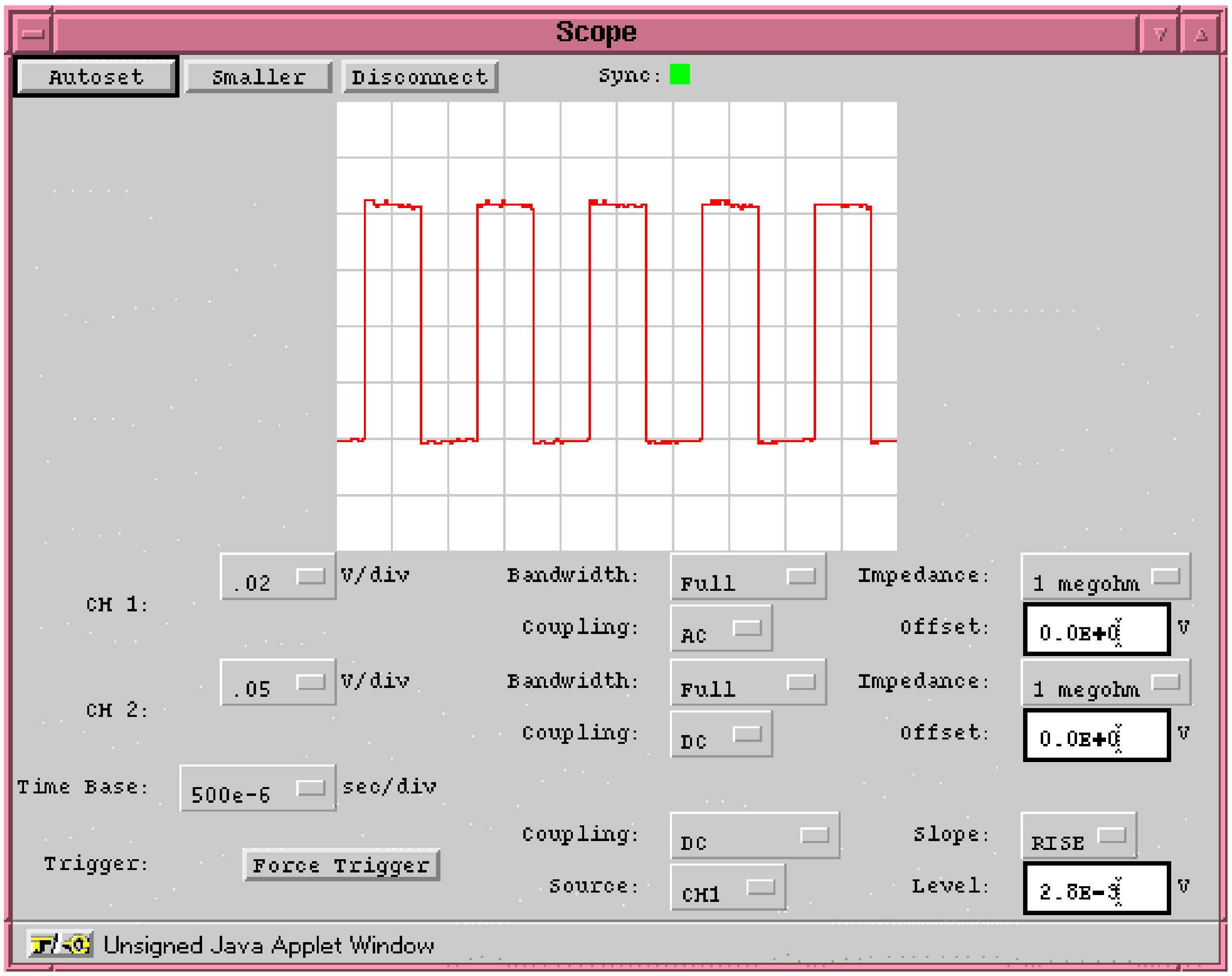


*Figure 2. Control window for a source / measure unit.*

SCADA systems usually provide a “world map,” allowing the user an overview of the entire system at a glance. Figure 4 shows the equivalent in the remote testing system. A list of nodes is presented, along with an iconic representation of the instruments and DUT pins associated with each node. Instruments with single stimulus or response values update their status in real time in their respective icons. A mouse click in any list icon toggles the visibility of the associated instrument control window, providing the window decluttering feature found in SCADA systems. The status of the switching matrix is shown below the node list, and a global disconnect button provides a panic stop feature.

A typical SCADA system can display measurement results both textually and graphically, and the remote testing system is no different. The user may select which measurement sources should be logged, and the resulting data arrays can be listed or plotted in separate windows. The data may also be saved in various ways. Applets currently prohibit access to a client’s file system for security reasons, but data may be saved on the WWW server in hypertext format for simple retrieval or it may be sent to the user via email.



CH 1: .02 V/div Bandwidth: Full Impedance: 1 megohm
Coupling: AC Offset: 0.0E+0 V

CH 2: .05 V/div Bandwidth: Full Impedance: 1 megohm
Coupling: DC Offset: 0.0E+0 V

Time Base: 500e-6 sec/div

Trigger: Coupling: DC Slope: RISE
Source: CH1 Level: 2.8E-3 V

*Figure 3. Control window for a digitizing oscilloscope. Given a fast Java virtual machine, the remote testing system can update the scope display at 6-8 frames per second over a local area network.*

Of course, the system must operate in real time. As long as response instruments are connected, the remote testing applet polls the instruments and queries those with measurement readings available. The system can update single readings from all connected devices several times per second, easily matching the measurement speed of the instruments. If used across a local area network, the applet can even query and update two simultaneous traces from a digitizing oscilloscope at several frames per second, suitable for most testing needs. The system presently requires a maximum bandwidth of 32 kbits/sec; ninety percent of that is needed to service two scope traces. With zero or one scope channels active, bandwidth needs drop significantly. As in manufacturing disciplines, remote testing is a "pull" system[8]; all test data transmissions occur in response to queries from the applet, preventing data overruns in slower Java implementations and allowing system performance to degrade gracefully on low-bandwidth channels.

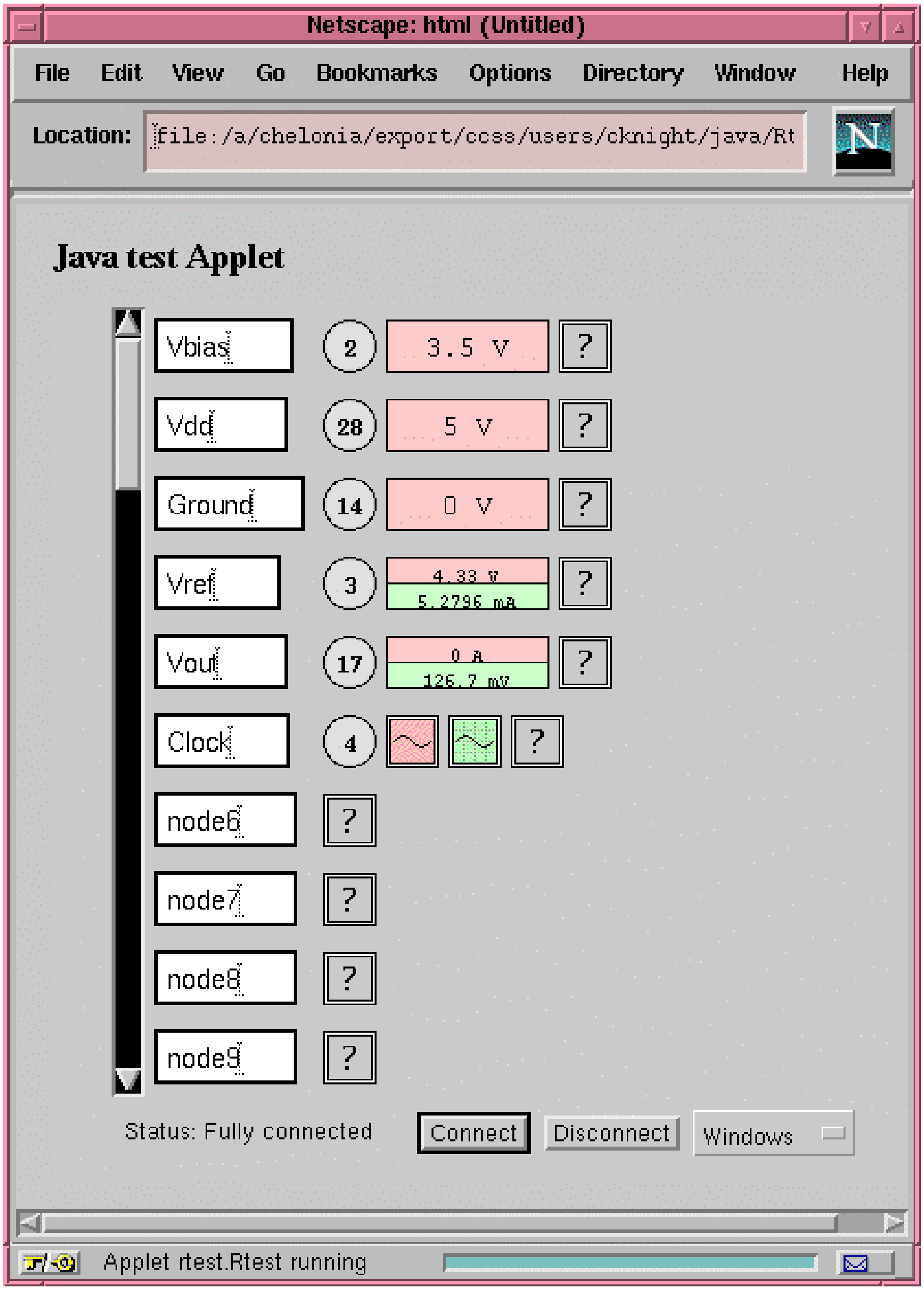


*Figure 4. The node list of the remote testing applet describes connections between instruments and device under test.*

## Conclusions

In this paper we have described an interactive remote circuit testing system accessible via the WWW. The Internet and the Web offer many innovative avenues to learning, but such applications need not be limited to accessing static material. The testing system presented here establishes a live, interactive link between a remote user and laboratory apparatus. The system is useful in circuits classes, where lab assignments correlate with lecture material, as well as in research labs, where multiple users must share test equipment. In either case, remote electronics testing adds a new weapon to the arsenal of distance learning.

The remote testing system embodies an open lab, available at all times without supervision. In initial trials the interactive testing system easily accommodated both a graduate circuits class of eight students working individually and several others doing original work in an analog VLSI research lab; servicing a group of 40 or more with a single testing station is feasible. Future system improvements include scripting, to improve sharing efficiency; multiple instrument suites, for serving larger classes; and a chip jukebox, to better support users doing custom IC design.

[1] Erkes, J.W., et. al. "Implementing Shared Manufacturing Services on the World Wide Web." *Communications of the ACM*. v39, #2, pp. 34-45.

[2] Arnold, K. and Gosling, J. The Java Programming Language. Addison-Wesley: Reading, MA. 1996.

[3] Aburdene, et. al. "A Proposal for a Remotely Shared Control Systems Laboratory." *Proceedings of the 1991 Frontiers in Education Conference*. pp. 589-592.

[4] Aktan, B., et. al. "Distance Learning Applied to Control Engineering Laboratories." IEEE Transactions on Education. v37, #3, pp. 320-326.

[5] Cambiotti, F., et. al. "Multimedia Training and Remote Operating Laboratory: Innovative Solutions for Instrumentation and Electronic Measurements Courses." *Proceedings, Computer Aided Learning and Instruction in Science and Engineering, Third International Conference, 1996*. pp. 245-251.

[6] Ghoshal, K. and Douglas, L. "GUI Display Guidelines Drive Winning SCADA Projects." *IEEE Computer Applications in Power*. v7, #2, pp. 39-42.

[7] D'Amour, P. and Block, W. "Modern User Interface Revolutionizes Supervisory Systems. *IEEE Computer Applications in Power*. v7, #1, pp. 34-39.

[8] Vollmann, T.E., et. al. Manufacturing Planning and Control Systems. Irwin: Chicago. 1992.

CLINTON D. KNIGHT is holds BSEE and MSEE degrees from the Georgia Institute of Technology and is currently pursuing a PhD there. He is exploring methods to query and control physical systems via the Internet and World Wide Web.

STEPHEN P. DEWEERTH received the M.S. degree in Computer Science and the Ph.D. in Computation and Neural Systems from the California Institute of Technology in 1987 and 1991, respectively. He is presently an

Assistant Professor of Electrical and Computer Engineering at the Georgia Institute of Technology.  His primary research is in the design of analog and mixed-signal VLSI circuits for modeling neurobiological systems.