

# The World Wide Web Leads a Revolution in ATE Programming Environments

Peter Hansen  
Assembly Test Division  
Teradyne, Inc.  
Boston, MA 02118

Phone (617) 422-3309 Fax (617-422-3440) E-Mail: peter.hansen@teradyne.com

**Abstract** – New software technologies, including the World Wide Web, may seem far removed from the tasks facing test program set (TPS) developers, but they promise to revolutionize the way TPS data is organized, presented, and used. This paper will describe how an integrated TPS development and execution environment can capitalize on these new technologies to improve test programming efficiency.

cost integrated development environments on stable 32-bit operating systems, and new presentation methodologies pioneered for the World Wide Web. ATE system developers can now leverage these capabilities to provide end users with far greater functionality in a much shorter timeframe than has ever been possible.

## I. INTRODUCTION

Until recently, it has been difficult to implement a robust graphical programming environment for ATE. The tools available to developers of ATE system software were crude, and the computing platforms were slow and expensive. Over the past five years, computing has changed radically. New software developments include the Windows 95/Windows NT 4.0 user interface, low-

## II. WINDOWS-BASED USER INTERFACES

The nearly universal acceptance of the Windows 95/Windows NT 4.0 user interface has made numerous user interface techniques commonplace.

For example, the Windows Explorer uses two powerful controls for perusing the file system (Fig. 1). On the left, the *tree control* is used to display the folder hierarchy. On the right, the *list control* shows information about the files in the folder selected from the tree.

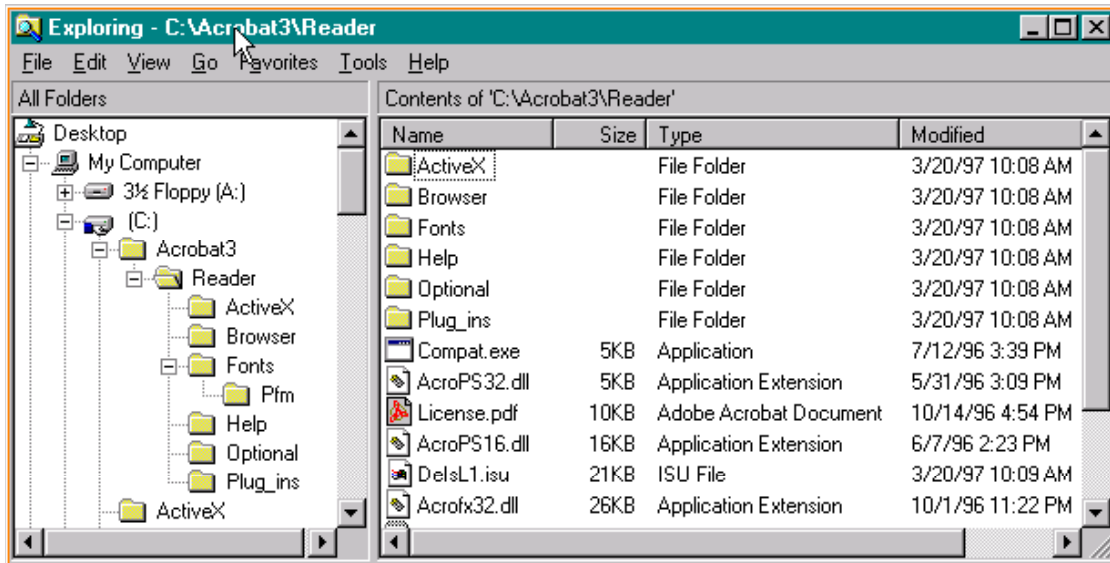


Fig. 1. Tree Control (left) and List Control (right) in the Windows Explorer.

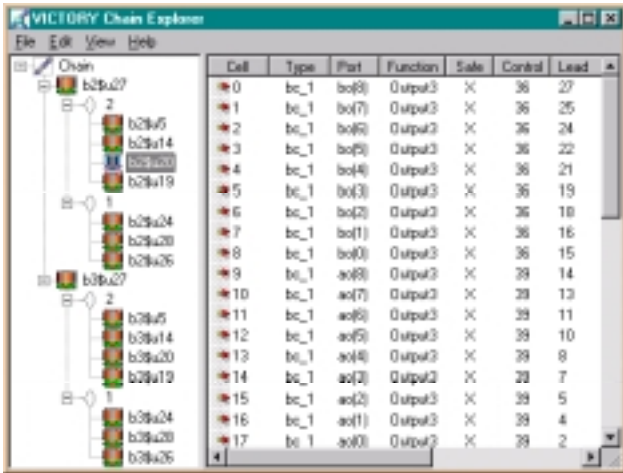


Fig. 2. Boundary-scan Chain Explorer.

A test environment can use these same controls to display critical information that is used for TPS development and debugging. The tree control can display hierarchical elements, such as devices within a circuit or steps within a development or execution sequence. The list control can display information related to the selected tree element. Fig. 2 shows the boundary-scan Chain Explorer, supplied with Teradyne’s VICTORY™ software, that uses the tree control (on the left) to show devices in a hierarchical assembly [1]. Any one device can be selected from the tree using the mouse or keyboard. The list control (on the right) displays the cells of the boundary register within the selected device. Users familiar with the Windows Explorer quickly become comfortable with the Chain Explorer.

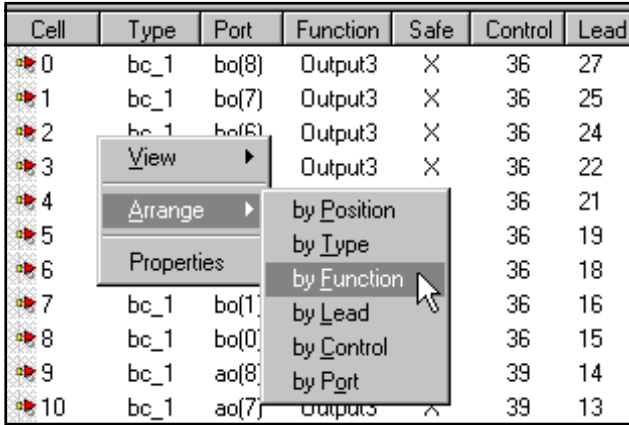


Fig. 3. Cascading Popup Menu.

Another example of a common user interface technique is the popup menu that is created by clicking on the right-mouse button while the cursor is on a selected object. With the Chain Explorer, a test developer can arrange the detailed rows by any category by using the list view context menu (Fig. 3). Popup menus are much easier to use than conventional application-wide menus because the selections are specific to the object under the cursor. In Fig. 3, the cursor is not on any specific cell in the list, so the menu applies to all the cells. The rows the list controls can be arranged in order of any of the various categories listed in the right-most cascading popup menu. If the cursor were over a specific cell, the menu choices would be specific to that individual cell object.

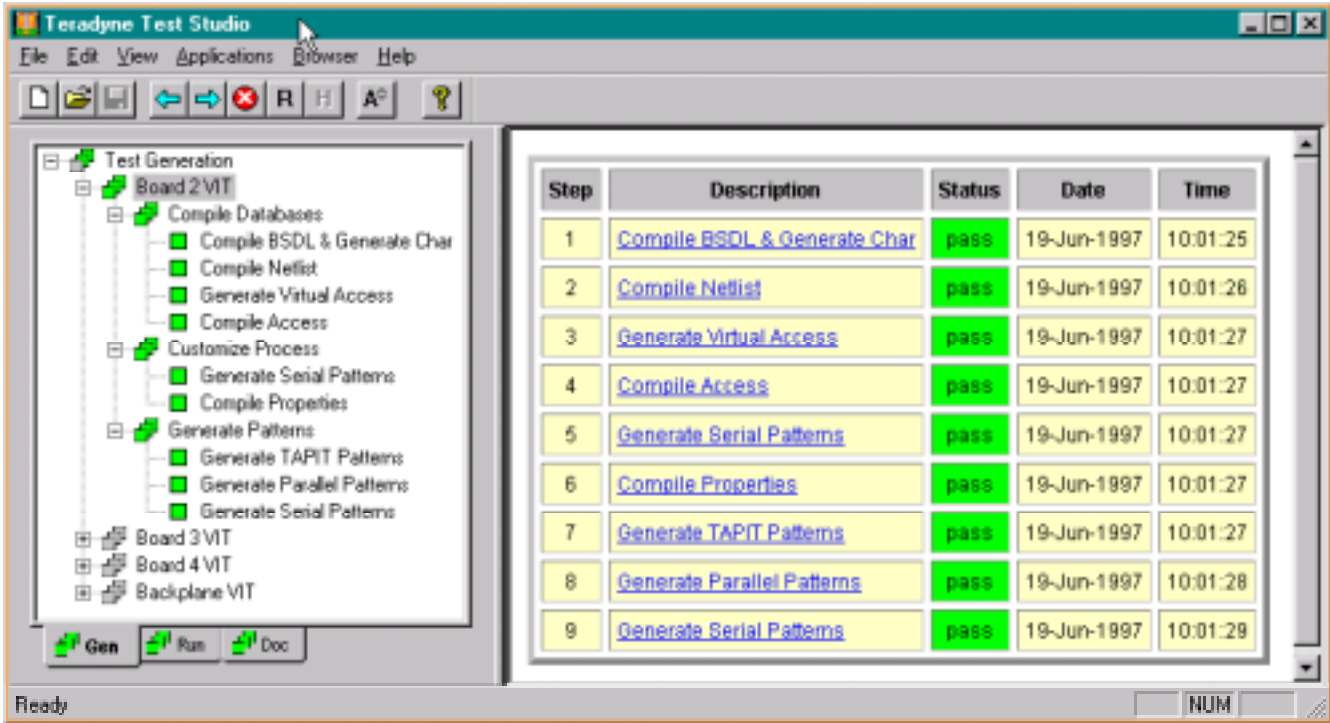
The previous examples demonstrate the application of three of the most common user interface elements found in many popular Windows programs to tester software. Many other controls are applicable as well, such as multiple programmable toolbars, status bars, and property sheets (also known as tabbed dialog boxes). The use of these familiar user interface controls is critical to decreasing the cost of learning a new tool and, more important, making users as productive as they can be [2].

### III. WEB PRESENTATION TECHNIQUES

While modern user interface techniques described above will provide great benefit to test software, they will most likely be overshadowed by innovations developed for the World Wide Web. There are Web-based user interface concepts that are powerful by themselves, in the absence of a network. Even more powerful are the possibilities for extending these concepts to tester access and program generation over an intranet or the Internet.

In order to take full advantage of Web technology, tester software requires the resources provided by a Web browser. Modern component technology facilitates the integration of a browser and other software components such as tree controls, toolbars, and specialized test components [3]. This component technology can also be used to facilitate integrating test-related standards advocated by ABBET, IEEE Std 1226.6-1996 [4].

Fig. 4 on the next page shows an example of a Web browser interface embedded in software for a complex test-generation sequence.



Test Studio is a trademark of Teradyne, Inc.

Fig. 4. Embedded Web Browser for Boundary-Scan Test.

One of the many reasons for the success of the Web, with technical and non-technical users alike, is the ability to “drill down” into more detailed data or to quickly move on to new topics. In applying this Web model to the TPS development environment, the test application could build files in the Hypertext Markup Language (HTML) format to link pages representing the TPS projects that exist or are under construction [5].

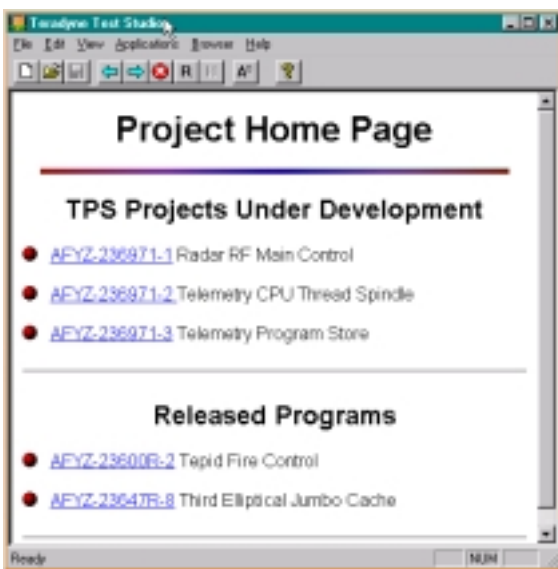
Fig. 5. Sample Home Page for a TPS Project.

Fig. 5 shows a sample Home Page for a TPS project. Drilling down into an individual TPS, you would see pages representing the test development steps and the resulting execution steps.

In addition to offering easy accessibility to test development and execution steps, Web techniques are ideal for distributing product documentation. The table of contents can be displayed by the tree control, while the pages of documentation, expressed in HTML, appear in the browser window. Since HTML authoring tools are becoming commonplace, a user could easily add TPS or site-specific pages. The correlation of test development, execution, and documentation pages could then be readily cross-linked in such a unified environment.

#### Multimedia Presentations

The Web’s facility for presenting graphical and multimedia content offers significant benefits for test developers. With the Web, there is a simple, cost-effective way to include diagrams in specifications, pictures of the UUT (Fig. 6) in operator instructions, or waveform displays in manual debug procedures. The hardware and software required to capture these graphics becomes less expensive and more powerful every day.



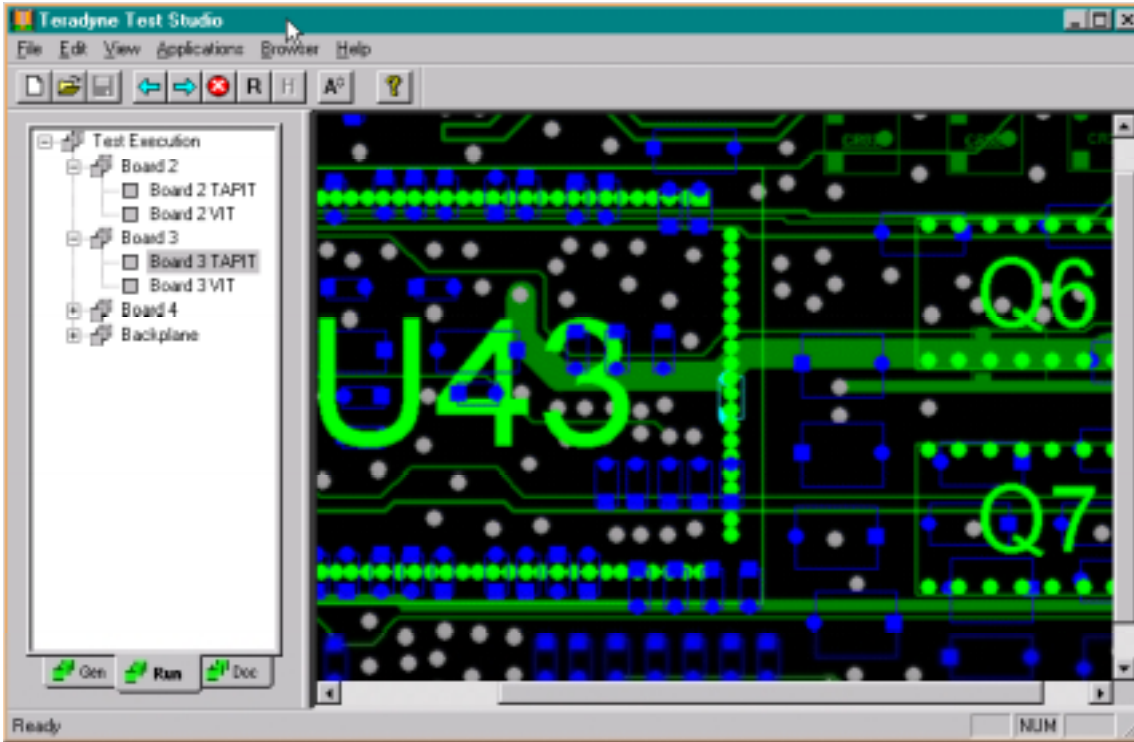


Fig. 6. Web Browser Displaying a Board-Under-Test.

Multimedia techniques such as sound and full-motion video will certainly gain popularity via Web technology. These can be used to further improve the user interface to test equipment by providing video-based learning and audible context-sensitive help.

#### *Active, Smart Web Pages*

There are many more innovations in Web technology that will dramatically influence the way we interact with testers and generation software. One important concept is "active" Web pages, where customized HTML is created on demand so that a page may never look the same twice. The customization may reflect the current status of a project or the results of a test sequence, created in response to a particular user's set of preferences. The current status of a development project is shown in the browser window in Fig. 4. The amount of detail on the page may be a function of the user classification, with all details shown for the TPS developer, but only brief status for non-technical personnel.

Another significant innovation is scripting and small programs known as *applets* that are loaded by the browser along with the normal text and graphics. Scripting and applets are used to create "smart" Web pages that can directly interact with the user. Traditional user interface elements such as dialog boxes can be present on a page as an applet, where it is only visible when a user clicks on the appropriate hyperlink.

Scripting and applets can be used to create adaptive forms that instantly reconfigure themselves based on user preferences and the question-and-answer session history. An enormous advantage of smart pages over conventional hard-coded applications is the degree of customization that can be accomplished by end users, without any assistance from the software vendor. Just as computer users can customize their desktops today, test developers will be able to use smart-Web-page technology to reconfigure the look-and-feel, and even the functionality, of their programming environments. Languages such as Java, Java Script and Visual Basic, and tools such as image and HTML editors, will typically be used for application-specific content creation and customization, leaving languages such as C and C++ for the core vendor-supplied product.

#### IV. CONCLUSION

This paper has explored how user interface elements and Web technology can improve test software, even in the absence of a network. Perhaps the greatest benefit of using Web-based tester software will come when the client and server functions associated with test are fully separated and made Web-accessible. Test generation software developed to execute under the control of a Web server could be accessed from any computer in a network. This is analogous to the commonplace scenario where a user, from a browser client, asks a server for information contained in a database. In the same way, ATE system software could be under the control of a Web server so that tests could be executed, or test results accessed, anywhere in the network from a Web browser client. Ultimately, test generation and execution could be carried out in a totally distributed Web-like environment, where test programming resources are effortlessly shared to achieve the highest possible productivity.

#### REFERENCES

- [1] P. Hansen, "Hierarchical Test Generation: Taking Boundary Scan to the Next Level," *Proceedings of IEEE Autotestcon, 1996*.
- [2] *Windows® Interface Guidelines for Software Design*, Microsoft Corp., Redmond, WA 1995
- [3] P. Stern, "High-Performance Component Software Changes the Rules for Configuring ATE," *Proceedings of IEEE Autotestcon, 1997*.
- [4] *Guide for Understanding of the "A Broad-Based Environment for Test (ABBET)™" Standard*, IEEE Std 1226.6-1996, IEEE, New York, 1996.
- [5] W3C (World Wide Web Consortium), HyperText Markup Language (HTML) Home Page, <http://www.w3.org/MarkUp>.