QUICK LOOK

- Accessible to technical and non-technical staff
- Useful reports, charts, and testing features

A Look at Testing Web Applications with eValid

by Robert Sabourin

Last summer I set up a testing lab for a major e-commerce Web-based application. I had to ensure that the lab could be used to test-drive software under development in realistic conditions, and I wanted to make sure that the testing facilities could be used at all phases of development. We needed tools that were practical, had short learning curves, and could improve productivity.

We chose **eValid** from Software Research, Inc., as a tool for use in functional, performance, and load testing of the application. The product did the job for us at a very reasonable price, and we were able to find some very important bugs well ahead of our target delivery dates.

General Description

The eValid tool is a test-enabled Web browser built to share several dlls and resource files with Microsoft Internet Explorer. It uses the same Web cache, history, cookies, plug-ins, and other objects as the installed version of Internet Explorer. (A Netscape version is currently being developed.) This is a natural tool to exercise and test any Web-based application; if you can surf, then you can test with eValid. Its ease is due in part to its familiar browser interface (see Figure 1). As you can see, eValid's testing features are included *in* the browser itself.

With eValid you can create a more realistic simulation than you can with other tools that generate http requests without composing and rendering the results. It can be used to perform many types of Web application testing. I focused on using eValid to perform functional, performance, and load testing. (The product Web site, **www.soft.com/eValid/**, offers examples of the other types of testing you can do with eValid.)

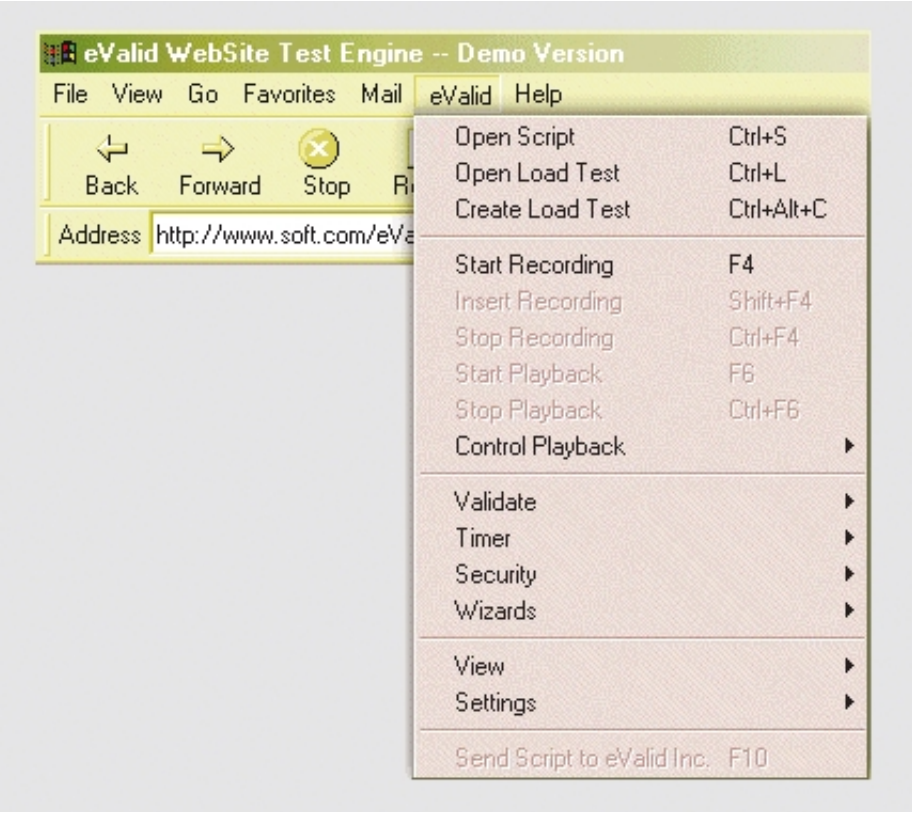

FIGURE 1 An eValid pulldown menu allows control of all test-related functions from the browser tool bar

Functional testing is accomplished with eValid testing scripts (also called *evs* scripts). Scripts simulate user sessions, and allow test validation of any or all links, text, images, or other objects encountered in the Web application.

Performance testing is accomplished automatically by eValid whenever an *evs* script is executed. The tool keeps track of timing information, logging the time to download entire pages and all individual components. You can view performance results in graphical or spreadsheet

formats, and use multiple playbacks to study how script execution times vary.

Load testing is accomplished with special load testing *(evl)* scripts. A load testing script allows the tester to run multiple concurrent eValid test scripts, each in a different browser window. Each line of a load test script spawns an eValid session that repeatedly runs an *evs* script.

Functional Tests

The functionality of any Web-based or e-commerce application can be validated with eValid. Functional tests are created with eValid test scripts, which simulate all user interactions with the Web application being tested. For example, Figure 2 is an eValid script that finds a book on amazon.com.

Creating eValid Scripts

During last summer's project our team used business analysts (domain experts) to define typical usage scenarios. Test scripts were then created by these same analysts—people with little to no computer programming experience, but with a rich understanding of the users' needs. While most other scripting tools I have previously used require programming experience, with eValid these non-technical staff could develop scripts with minimal coaching.

Creating a functional test script is a straightforward procedure:

- Identify the function to be tested
- With eValid navigate to the page you want to start testing
- Select eValid/Start Recording (F4)*(As you interact with the Web application, eValid captures the different user events and interactions)*
- Use eValid as a Web browser and perform the function being tested
- Indicate verification points with the appropriate eValid/Validate/Selected menu option *(Validation points are used to confirm*

```
ProjectID "Project"
GroupID "Group"
TestID "Amazon.com-find"
LogID "AUTO"
#
# Start test session at amazon home page
#
InitLink "http://www.amazon.com/exec/obidos/subst/home/home.html/107-1318440-9319710"
Wait 5000
#
# Search for book "seven habits"
#
InputValue 104 "TEXT" "field-keywords" "seven habits" "" ""
#
# click on GO icon - to start search
#
InputImageClick 105 "http://g-images.amazon.com/images/G/01/v9/search-browse/go-button-gateway.gif" "" ""
Wait 4830
#
# select the Steve Covey classic
#
FollowLink 195 "The 7 Habits of Highly Effective People : Restoring the Character Ethic"
"http://www.amazon.com/exec/obidos/ASIN/0671708635/o/qid=983309794/sr=8-
1/ref=aps_sr_b_1_1/107-1318440-9319710" ""
Wait 5000
#
# add it to the shopping cart icon
#
InputImageClick 338 "http://g-images.amazon.com/images/G/01/detail/buybox/add-to-shopping-cart-
blue.gif" "" ""
Wait 5000
#
# back to books - for more shopping!
#
FollowLink 136 "" "http://www.amazon.com" ""
```

FIGURE 2 Sample eValid script

that links, text, images, screen regions, and other objects are loaded correctly)

- Select eValid/Stop Recording (CTRL + F4)
- The eValid script editor is used to adjust the recording, and provides detailed, context-sensitive help

Scripts can include tester-specified verification points that confirm the occurrence of text, images, and other objects. The eValid tool records functional test results in a detailed log file that indicates the pass or fail status of the events—and also includes timing information associated with each line of the script. You can view the log file as a series of charts, tables, or as a spreadsheet with Microsoft Excel. The spreadsheet format makes it quite easy to perform further analysis of test results.

Performance Data

We can construct different *evs* scripts to measure the performance of various usage scenarios. For each line in the script, eValid will log detailed timing information.

The tool's performance data is provided in spreadsheet and graphical format. Timing resolutions in eValid are to the millisecond, and its performance charts make it easy to identify bottlenecks (which pages take the longest to load and which components of a page are chewing up the most time).

Performance data is displayed in a bar graph format. Each bar in the chart represents the amount of time it takes to execute a line of the script. The line numbers are indicated below the bars, and time is given in milliseconds. You can also view a chart showing timing information at this finer level of detail including the time to download and render elements of a page.

Load Testing

Performing load testing with eValid is a three-step process:

1. Create functional test scripts to simulate typical usage scenarios
2. Create load tests to execute multiple concurrent instances of each scenario
3. Execute load tests while monitoring system operation and measuring performance

Typical usage scenarios are scripted as described under the functional testing section. Scripts may be parameterized so that certain data, such as user names and passwords, may be varied each time a script is executed. Script parameters are defined as simple name value pairs in each line of an eValid load script.

The following is an example of a line from an eValid load test *(evl)* script.

```
eValid "Ami001.evs" "ID002" "50"
"$USR=Bil $PWD=Xy1 $WIDE=200
$HIGH=400 $XPOS=201 $YPOS=1"
"-pm 1.0"
```

This *evl* script line instructs eValid to launch a session identified as ID002. The *evs* script **Ami001.evs** is executed in its own window "50" times. Script parameters include a user name **($USR)**, password **($PWD)**, window width **($WIDE)**, height **($HIGH)**, and position **($XPOS, $YPOS)**. A similar *evl* script line is included for each parallel session launched.

Load test results in eValid are presented in a bar chart. For each *evs* script used in the load test, the chart shows the slowest elapsed time, the fastest elapsed time, and the average of the (in this case) fifty runs. All times are in milliseconds.

As with performance testing, all load test log data can also be viewed as an Excel spreadsheet.

In our lab we generally used one PC to run performance measurements with *evs* scripts while other PCs were running load test scripts. We ran eValid load tests from standard Windows-based PCs, and were able to simulate over fifty concurrent sessions on each PC in the lab (with PIII processors and 128 mb RAM on Windows NT 4). Additional PCs were used to extend our range beyond fifty concurrent sessions. (The eValid Web site indicates that the maximum capacity can exceed 300 sessions on suitably configured systems.)

Pricing

We found eValid to be a very affordable testing tool; it would be within the reach of most budgets to have several licenses available to developers and testers during all phases of development. Detailed pricing information, including quantity and volume discounts, is available on the product Web site.

Fully functional versions of eValid may be downloaded from the vendor's site for evaluation prior to purchase.

Checking It Out

Software Research, Inc.'s, Web site offers several examples, presentations, white papers, tutorials, and examples about how to use eValid in many different testing situations. These are an excellent starting point to any Web testing project—they walk you through exactly what to do, in a clear, concise, step-by-step manner. I think eValid is a very practical, cost-effective tool, and I would recommend it to Web application developers and testers who need to perform functional, load, and performance testing. STQE

Robert Sabourin (Rsabourin@amibug.com) *is President of AmiBug.Com, Inc.* (www.amibug.com), *specialists in management consulting, teaching, and professional development in software engineering, testing, and software quality assurance. In addition to his magazine articles and lectures, Robert authored the children's book* I Am a Bug, *a primer on what SQA moms and dads really do at work all day.*

STQE magazine is produced by STQE Publishing, a division of Software Quality Engineering.