ETALIQ
intelligent quality

Test Automation:
Solve Problems Quickly

A Case Study
by Etaliq Inc.

innovation in automation

www.etaliq.com

Etaliq Inc.
4B–2548 Sheffield Road
Ottawa, ON K1B 3V7

Phone: (613) 241–1385
Fax: (613) 241–1523
http://www.etaliq.com

Challenge

Executive Summary

Who • Carrier class network equipment provider • Over 10,000 engineers
Challenge • Place the DUT under heavy stress • Allow other tests in parallel
Solution • ETA Client and Server Install • 6 person days development time
Result • Solution delivered • Fast, reliable, customizable • ROI well beyond expectations

Stress and Performance measurement using today's state of the art automation tools is difficult; if not, impossible. In order to stress a carrier-class product using today's tools, you must combine a set of single-function scripts into a stress-inducing engine. Each individual script will login to the existing device one or more times and perform a set of actions to consume CPU and memory resources.

Ideally, these single-function scripts can be controlled by a master script, which will control the amount of resources consumed, and periodically measure processor and memory use on the device under test (DUT), including its master and individual line card components. The Internet Architecture Board's Stress and Performance Working Groupis defining just such an objective aimed at defining an RFC. While differences exist between the Stress and Performance Working Group requirements and the ones defined herein, the objectives remain the same.

In general, the challenge is to "put a DUT *into a stress condition and measure its performance, while running various functional tests."*

This test utility is primarily meant as an Assist tool for manual testers while doing stress and performance testing. Also, using this utility during normal automated regression and stress testing, is defined as a stretch goal.

It is pertinent to note here that the methods used to impose stress on each node or device is product-specific.

Detailed Requirements

Etaliq's and the client's Lead Test Engineers cooperated to define the following requirements:

- The tool or utility must connect to the DUT with ten or more simultaneous sessions.
- There are several different categories of session that are to be usable:
  - Session type "Operator" will be used for non-disruptive operator actions only, where no changes to existing operational state or status are permitted.
  - Session type "Configuration" will perform configuration changes to existing components which may or may not cause disruption to the affected sessions.
  - Session type "Availability" will induce various types of failure and recovery scenarios on the DUT.
  - Session type "measurement" will continually monitor CPU and memory usage statistics and watch the system log for unexpected alarm conditions.
- All session will be run from a state machine where the Operator, Configuration and Availability sessions are to be kept as busy as possible, and the Measurement sessions are to be scheduled.
- Each of the sessions will have a **`SEND`** scheduled to the DUT, exclusive of waiting for a complete response.
- Upon return to each of these sessions, the response will be gathered and it will be determined whether a full response is received.
- Upon receipt of a complete response, the state machine is advised that the session is again ready for another **`SEND`**.
- Each of the session categories, and in some cases the sessions themselves, must be configurable:

  - The tester must be able to specify which components of the DUT are eligible for disruptive tests, and which are not.
  - The tester must also be able to configure the series of operational commands that will be used by all or each individual Operator session.
  - The various types of Configuration and Availability mechanisms must also be selectable.
  - The tester should be able to selectively include or exclude interface delete/add, line card reset, disable/enable, and route flapping or the like.
- Various other tool configuration options should also be included, like frequency of measurement, total session count, target CPU utilization, target memory consumption, etc.

Ideally, the tool should provide an interactive capability where control over various sessions can be imposed. For example, testers should be able to disable and enable disruptive sessions, either completely or for specific components. As another example, the tester might request a dump of the most recent measurement statistics for a specific component. In general, a series of usability functions should be included in the tool to enable either manual testers or other test cases to control its features.

Solution

They already had an existing ETA Client/Server installation. A Senior Etaliq Automation Engineer worked with an on-site Network Engineer to create such a tool. The tool exceeded all expectations. It was completely implemented in just six days using ETA's standard multi-session support and **`SEND`** with no-wait/resume-wait features. Its' features met or exceeded all requirements, including the stretch goals, where other ETA

test cases operated in parallel with the stress tool in an automated environment.

Results

The utility was run against several edge and core products with much success, driving main and card-based processors to extreme stress and resulted in the discovery of several previously unknown product defects.

The resulting utility was only 1,000 lines of highly-structured ETA code, operating as a state machine and controlling the operation of ten or more sessions. It was fast, reliable and customizable.

It required only a Junior Automation Engineer to maintain and customize. That same engineer, after only two hours of training, was able to add new sessions and utilities without the aid of Etaliq's Senior Automation Engineer.