PACIFIC NW SOFTWARE QUALITY CONFERENCE

Fall Conference – Portland, OR - October 2005

# UPGRADING YOUR TOOLBOX:

## Adapting and Adopting Tools & Practices

**October 10-12, 2005**
**Oregon Convention Center – Portland OR**

**PRESENTATION BY**

**Joe Towns and Cordell Vail**
Washington School Information Processing Cooperative

# Stress, Load, Volume, Performance, Benchmark and Baseline Testing at a price you can afford!

WSIPC™

*Technology for Educational Communities*

## THE AUTHORS

**Joe Towns brings to the presentation a developer's perspective. He has eighteen years experience as a software developer, systems engineer, and supervisor.**
**jtowns@wsipc.org – www.wsipc.org**

**Cordell Vail brings to the presentation a test engineer's perspective. He is a Certified Software Test Engineer and Certified School Business Specialist with nine years experience in manual and automated testing. Cordell has made several presentations on "Improving Testing Processes" at both local and national conferences.**
**cvail@wsipc.org - www.vcaa.com**

## INTRODUCTION

This presentation has two purposes. The first purpose is to give you a knowledge base to help you learn how to select the correct testing tool for your testing environment. The second purpose of this presentation is to help you actually find an inexpensive testing tool that will do the job for your specific situation. This presentation is not intended to teach you how to do the testing.

## OVERVIEW

At the Washington School Information Processing Cooperative (WSIPC) we are in the process of deploying a web based, teacher desktop application in a complex multi-component, integrated Progress database environment. This product had never been released into production. We needed to know what the capacity of our servers would be in relation to handling the load. We wanted to know what the user response time would be if a great number of teachers all logged in at the same time. We also wanted to know what effect that would have on the web server.

We had experience in selecting an automated testing tool, and had dealt with the leading testing tool vendors in the past to select that testing tool,  however, we had not done any performance or load testing in that environment before. The concern we had was that we did not have a tested and proven testing tool that would allow us to test this application or web server for things like user response time, maximum server load, or database locking.  We were not even sure what else we would need to test to make sure the application would stand up to the tremendous potential load that could be placed on it the first day.

Any time you are using an application over the Internet there is a possibility of 1000s of users logging in and hitting the server at the same time.  This would almost certainly be the scenario in our case.  We were introducing the application to large school districts comprised of numerous schools with scores of teachers in each.  Each of these teachers, as well as administrators, could potentially log on and take attendance or run administrative reports via the web during the first day the application was available.

When the management asked us to do this testing, we did not have a testing environment set up to perform this load testing.  We were starting from square one.

Our requirement then was to be able to test load-balanced web servers to see how many school teachers could be logged on at the same time, all doing different tasks.  In some of our school districts there could be as many as 2,000 teachers logged on at the same time, submitting requests to the servers.  We needed to be able to tell the school districts and data centers approximately how many servers they would need available, at any one time at the district level, to handle their load, and still have a user response time of less than two seconds.

## KNOW WHERE TO START

Our experience told us that we could not find the right testing tool unless we knew the end goal of our testing.  We knew that we would have to understand the testing environment. We knew we would have to organize a testing team that understood the terms of the testing. And we knew we would not know if the testing results were accurate unless we had a baseline for test results comparison.  So our quest to find a testing tool started by organizing the testing process.

## BEGINNING WITH THE END IN MIND

In order to be able to find the correct testing tool, our organization had to know the full scope of the task at hand.  It has been our experience that one of the critical mistakes that many organizations make is to begin testing without a test plan.  There is a common joke among testers that goes like this.  The manager says, “Here, start testing this and I will go work on a test plan while you are testing”.  Sadly, that is all too true and often the test plan is never even created. Stephen Covey gave good advice that applies to all testing, “Begin with the end in mind”.

Among other things, we knew that a test plan would help everyone on the testing team to know what the objective was, what was to be tested, what was not to be tested, what resources were available, and the test results to be reported. Here is an outline of the simple test plan we created:

- Scope
- Resources
- Timeline
- Risk Analysis
- Deliverables
- Glossary

We wanted the test plan to be as simple as possible. The kind of test plan selected, of course, depends on the complexity of the project and especially the degree of risk associated with the project if it fails. We have provided, to the attendees of our presentation, a CD with a copy of the final test plan we used in this project. The CD also has other example test plan templates of differing complexities, for you to use as a resource in your test planning as you start the task of finding the right testing tool for your situation. The test plan could be a simple check list, a basic test plan for short projects, a full range test plan for projects that will last several weeks, or a very complex and comprehensive test plan for enterprise testing projects that will last several months, have high risk and involve multiple departments within your organization.

## UNIFORMITY

It is very common to find testing groups using some of the common testing terms as synonyms. This can be very confusing in an organization if you have the managers thinking that Load Testing is finding out how many users can be on the system at one time and the testers are calling that type of testing, Performance Testing. Therefore we propose that, at the outset, you include as a part of your test plan, a glossary of terms so everyone in your organization is defining the testing terms in the same way. It is not so important what definition you give to the testing terms as it is that everyone in your organization define and understand the terms in the same way.

This is an important consideration in finding the right test tool so you do not buy a testing tool that the vendor says will do one thing and then find out it actually does something else. We point this out because we actually had that experience. After working with a vendor who said the tool did “LOAD TESTING” we discovered that our definition of “LOAD TESTING” was different than the vendor’s and the tool did not do what we wanted to do at all.

Due to our experience, we suggest that you define at least the following list of terms listed below. These are not proposed as the industry standard definitions. They are just examples of how your glossary might look in your test plan. If you do not agree with these definitions, then change them to definitions that your testing team and management all agree upon. When you talk with vendors make sure that they define these terms the same way you do.

**Stress Testing:** Tests the server. Peak volume over a short span of time.

**Load Testing:** Tests the database. Largest load the database can handle at one time.

**Volume Testing:** Tests the server and the database at the same time - heavy volumes of data over time (combining Stress Testing and Load Testing over time).

**Performance Testing:** Tests user response and processing time.

**Benchmark Testing:** Compares your testing standards to the same testing standards in other similar organizations in the industry.

**Baseline Testing:** Setting testing standards to be used as a starting point for subsequent comparisons within your own organization.

Naturally there are many other terms that can, and may, need to be defined in your environment. The important point here is to make sure that the glossary in your test plan keeps all of the members of the testing team on the same vocabulary page so you can use the uniformity to help you in finding the right testing tool for the job.

## UNDERSTAND OR IDENTIFY THE PURPOSE OR TARGET OF THE TEST

Again, the purpose of this presentation is not to teach you how to do the testing but to help you in knowing how to select the right testing tool for the job at hand. Every testing situation will be different. There will be different reasons why you are doing your testing and different things you will monitor based on those variables. Some of the things that you may consider testing are:

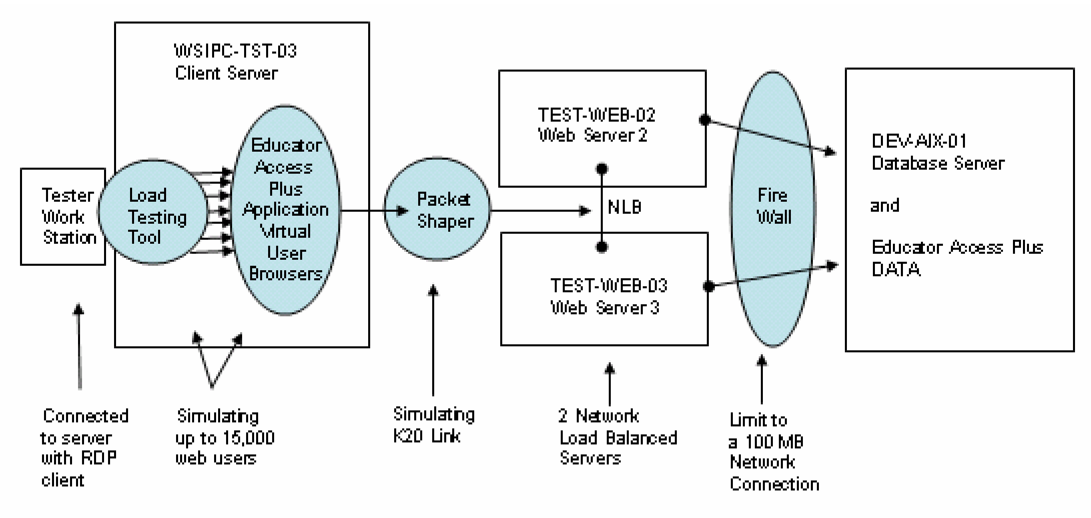
**Bandwidth**
**Concurrent users**
**Multiple platforms**
**Multiple browsers**
**Users per server**
**Multithreading**
**Disk capacity**
**Faults**
**Memory**
**User response**

## KNOW THE TESTING ENVIRONMENT

You will also need to consider the environment necessary to execute the testing. The test plan preparation will help you to determine this. This information is critical in finding the right testing tool, because when you look for an inexpensive tool, you will be able to determine whether it has enough horsepower to test in your environment and give reliable results. Therefore, you need to consider adding to the test plan things like:

**Number of client workstations**
**Database availability**
**Application and web servers**
**Test tools to monitor results**
**Connectivity to servers**
**Production copy of application**
**Bandwidth and LAN**
**Reporting requirements and capabilities**

You may want to create some kind of flowchart to outline what your test environment is helping you to know and what testing tool or tools you will need to do the testing. Here is an example of a flowchart of the testing environment we set up at WSIPC to do this testing:


WSIPC-TST-03
Client Server
Tester Work Station
Load Testing Tool
Educator Access Plus Application Virtual User Browsers
Packet Shaper
TEST-WEB-02
Web Server 2
NLB
TEST-WEB-03
Web Server 3
Fire Wall
DEV-AIX-01
Database Server
and
Educator Access Plus DATA
Connected to server with RDP client
Simulating up to 15,000 web users
Simulating K20 Link
2 Network Load Balanced Servers
Limit to a 100 MB Network Connection

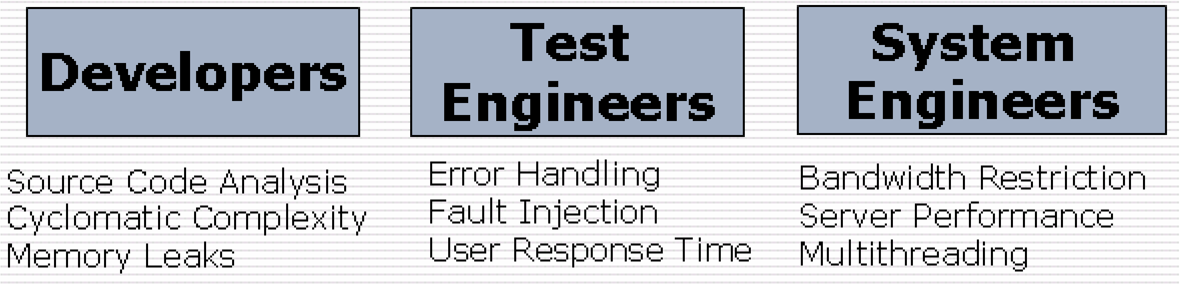
(Notice in this diagram we called the testing tool a "LOAD TESTING TOOL" and in this case we used WAPT with 2000 virtual users)

## ONLY BUY WHAT YOU NEED

To find an economical tool you must know what your testing needs are. Each tool is very different. You are not going to pay $200 for a tool and then get a tool that will do what an $85,000 tool will do. So you must consider what your need is in testing and then find a tool that will fill that need.

## CONSIDER TEAM COMPOSITION

The most effective testing team will be composed of individuals who will be able to address the scope, needs, risks, execution and monitoring of the test plan.



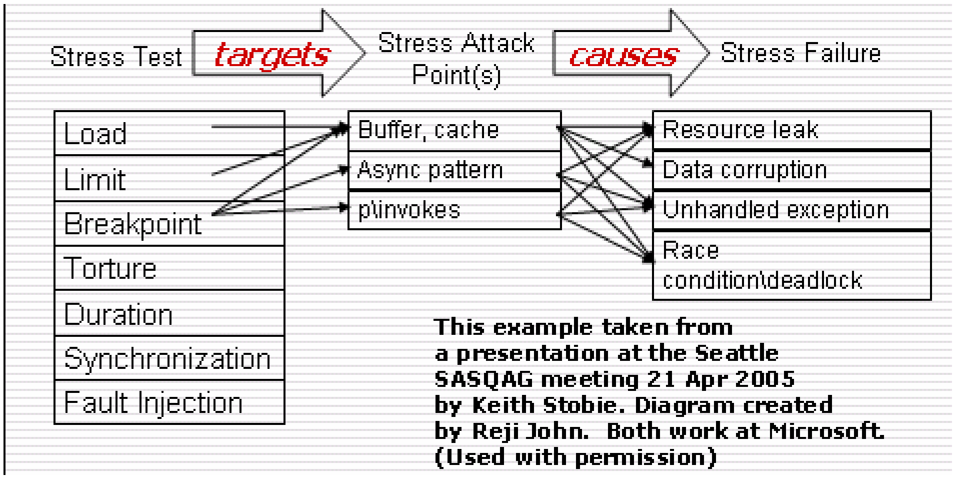
| Developers | Test Engineers | System Engineers |
|---|---|---|
| Source Code Analysis | Error Handling | Bandwidth Restriction |
| Cyclomatic Complexity | Fault Injection | Server Performance |
| Memory Leaks | User Response Time | Multithreading |

The members of your testing team will be able to help determine which tests need to be done and therefore the testing tool or tools that will be required to accomplish the testing in an acceptable manner.

## KNOW HOW YOU ARE GOING TO TEST IT

Having picked a specific target of testing, for example bandwidth, user response time, and server performance, you will be in a position to look for a tool that will perform those tasks. You do not need a tool that will do all things for all testing tasks. There are tools on the market that will do that. Normally, they cost thousands of dollars. We wanted to find inexpensive tools that would perform only the tasks that we need done in our testing. Again, you may want to create a diagram to illustrate what your testing methodology will be.

Here is an example of a testing strategy used by Microsoft where they determined the types of testing that they would do and then what the testing tool would test.



**This example taken from a presentation at the Seattle SASQAG meeting 21 Apr 2005 by Keith Stobie. Diagram created by Reji John. Both work at Microsoft. (Used with permission)**

**TRUSTING THE TEST RESULTS**

One of the main issues we have discovered in this type of testing is to know if the test results are valid when you have no live production data for comparison.  The key element in the validity of test results is having something to compare your test results to.  If you have a new web application and it has not been released to production, with a new testing tool, how will you know that your test results are realistic?  This is especially true if you are using a new, inexpensive tool, for which there is little support and no known test results available from other testers.

Unless you have been around open pit mining, you likely would have no idea how big this equipment is.  Testing tool results can be very similar to this example.  It is very easy to use some of these inexpensive tools, but because it is not so easy to know if your test results are valid, you need to have something to compare the test results to.

NOW HOW BIG IS IT?

**POSSIBLE SOLUTIONS**

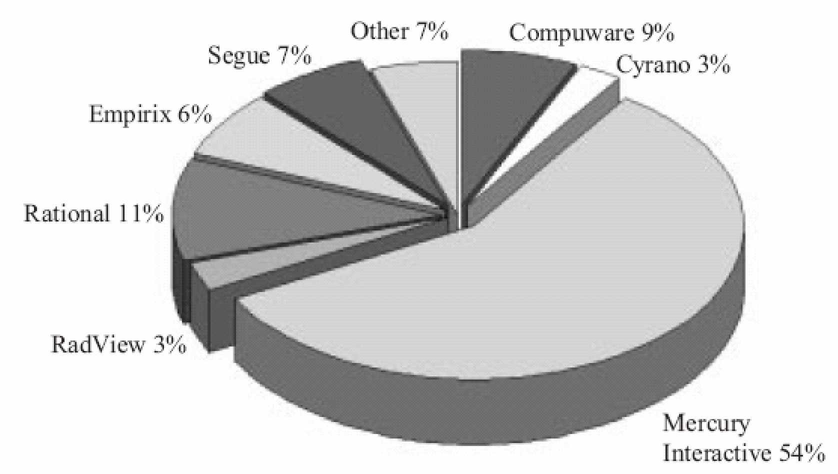
One possible solution in using a new testing tool would be to test it against an application where you already are able to measure the test results by monitoring the servers or with test results from previous testing. This test results comparison could come from your own baseline testing on a previous or known application where you already know what the results should be. You could also compare your results to a bench mark test from some other company doing similar tests on similar applications with the same tool.

Sometimes when you have a new application and a new tool, you will not be able to validate the test results until the application is actually released into production. Then, if your test results match the actual production results, you have established a baseline with that tool for further testing of other applications in the future.

Another possible solution, if you are using a new tool on a new application, would be to test the new application with more than one of these inexpensive tools. Next compare the results. That is one of the main points of this presentation. With inexpensive tools, using more than one tool is a feasible thing to do.

## CAN WE TRUST MARKET SHARE?

In finding the actual tool, it was logical to first go to the marketing people and find out what others were doing to solve similar situations in testing. We found that 93% of the market share for this kind of testing was controlled by seven large vendors.



**Source:** http://www.mercury.com/us/pdf/company/newport_load2000.pdf

## AFFORDABLE TESTING TOOLS

The first thing that became apparent to us was that the prices being charged to the 93% of the market share being controlled by the seven large venders was very high. That was more than we wanted to pay, thousands of dollars more. Therefore, we decided to take a look at that other 7% of the market to see what kind of tools that represented and what kind of pricing would be offered.

## TESTING TOOL COMPARISON LINKS

To do the tool evaluation we first went to many sources on the Internet including vendor web pages to gather information about every tool available on the market. You will find a listing of all of those vendors on the presentation CD handout that you will receive in the class. There are over 200 vendors in that list.

Here are some of those sources we used to find those vendors and make our testing tool comparison list:
http://www.vcaa.com/testengineer/links.htm
http://www.testingfaqs.org/t-load.html
http://hammerhead.sourceforge.net/
http://opensourcetesting.org/performance.php
http://www.grove.co.uk/Tool_Information/Choosing_Tools.html
http://www.softwareqatest.com/qatweb1.html#LOAD
http://www.sqa-test.com/toolpage.html
http://www.webservices.org/index.php/ws/content/view/full/102
http://opensourcetesting.org/performance.php
http://sourceforge.net/projects/dieseltest/

Using those sources, we found Internet links to all of the seven vendors that represent the 93% of the market share (each of these vendors charge more than $5,000 and some over $100,000):

**www-306.ibm.com/software/awdtools/tester/performance/index.html**
**www.segue.com/products/load-stress-performance-testing/index.asp**
**www.mercury.com/us/products/performance-center/loadrunner/**
**www.radview.com/products/WebLOAD.asp**
**www.quotium.com/qpro_overview_load_testing.html**
**www.empirix.com/default.asp?action=article&ID=418**
**www.compuware.com/products/qacenter/performance.htm**

We also found links to many smaller vendors that were more reasonably priced.  Here are the ones that we evaluated and will demo in the presentation:

| | |
|---|---|
| **TestMaker – PushToTest:**<br>**http://www.pushtotest.com/Downloads/** | **FREE** |
| **WAST – Microsoft:**<br>**http://www.microsoft.com/downloads/details.aspx?FamilyID=E2C0585A-062A-439E-A67D-75A89AA36495&displaylang=en** | **FREE** |
| **LoadTester – AppPerfect:**<br>**http://www.appperfect.com/products/devsuite/lt.html** | **FREE** |
| **TestMaker – PushToTest :**<br>**http://www.pushtotest.com/Downloads/** | **FREE** |
| **Site Tester 1 – Pilot:**<br>**http://www.pilotltd.com/eng/index.html** | **$29** |
| **Portent Supreme – Loadtesting.com:**<br>**www.loadtesting.com** | **$279** |
| **WAPT - Logasoft:**<br>**http://www.loadtestingtool.com** | **$299** |
| **Webserver Stress Tool 7 – Paessler:**<br>**www.paessler.com** | **$625** |
| **HOLODECK - SISE:**<br>**http://www.sisecure.com/holodeck/learn.shtml** | **$1,500** |

**NOTE: Holodeck is a fault injection tool - not a normal virtual user testing tool**

## CONTINUED RESEARCH

We would like to ask you to join us in this testing tool evaluation project. We have included on the CD handout several hundred testing tool web pages that we located. We have not evaluated all of them personally. We have just provided you with our research results hoping to save you the time of having to spend the same weeks and weeks of work that we spent looking for these testing tool links. That way we hope you will be able to make your own evaluations of the ones that we did not evaluate.

In addition to this conference documentation, all of our testing results are also posted on our testing web page at http://www.vcaa.com/testengineer/toolcomparison.htm . We will continue to post our testing results there as we do further testing of this type.

If you have tested a testing tool in this category of testing and would be willing to send us your evaluation to post on the web page it would be welcomed. The more evaluated tools we have, the easier it will be to find an inexpensive tool that fits our needs. All evaluations received will be posted to our web page with proper credits given to you and the vendor. Also, if you would like to have contact information such as your email address, web page or other contact information posted there with your evaluation for networking purposes, we will do that as well.

Joe Towns jtowns@wsipc.org - Cordell Vail cvail@wsipc.org http://www.vcaa.com