Dealing with Pre Examination Stress

Robert Sabourin, P. Eng.

AmiBug.Com, Inc.

rsabourin@amibug.com

www.amibug.com

Why this paper?

This paper is written to share some stress testing approaches based on real experiences.

This article is written for software development and testing professionals. Readers may want to apply some of the approaches discussed to their own projects.

Basis for experience

This paper is based on work done in 2002 and 2003 in collaboration with the INC (name changed for confidentiality) in Canada. The work done involved analysis, project organization, testing, experiment definitions, test set up, test executions, test result summaries. Some of the results, based on tests run in April 2003, were compared with actual performance measures made on the live system. Results under simulation compared favorable with results measured during live testing. The major performance measure used for the basis of this comparison was percentage CPU usage as a function of number of concurrent users of the system.

Background and Past Problems at INC

INC performs computer based certification exams in Canada. INC runs all exams from a central server in Ottawa. Students only take an exam at allotted times in one of seven centers across the country.

INC had experienced serious live failures during the spring 2002 examination period. INC servers failed dramatically under normal load early in the day when students were beginning to take the exams. Thanks to the heroic efforts of the INC network support staff a new server was set up and put in operation in record time that allowed for the completion of the examination period. Students only experienced a couple of hours of down time. This situation was considered a serious problem by INC executive. AmiBug.Com (*Robert Sabourin*) was contracted to set up and implement stress testing well in advance of future examinations so as to ensure the exam software would operate correctly under the target load required. INC knows exactly how many students will be taking the exams and from which locations and specific dates. Stress testing experiments were required to realistically simulate the expected target loads.

In addition to simulating the expected load required during the live examination period, the R&D team at INC wanted to have a test environment which allowed them to conduct a series of "what if" experiments. A "what if" experiment would look at how system performance varied with different software design, software architecture, hardware, database or network configurations. Developers would be able to use the test set up to perform performance measures under varied load to determine the impact of different design alternatives – without having to wait for the INC system to go into full production.

The three measures INC were primarily interested in were:

1. Typical Transaction Time to generate and render an exam section.
2. Memory consumption on the INC servers under load.
3. Processor usage on the INC servers under load.

Approaches to Stress Testing

Stress testing can be defined as ensuring a system operates correctly in a harshly constrained environment. Stress testing at the INC was limited to operating the EXAM software under varied amount of load from 0 to 1000 concurrent sessions.

Project Description

Identify requirements for performing various stress tests required by INC.

Implement stress testing lab licensing suitable software making use as much as possible of equipment and software already available to the INC.

Run a well defined series of "what if" experiments to determine impact of different design and technology approaches used in Exam software.

Run simulations of INC exam based on a realistic model of the expected load for all section of the EXAM.

Run a series of tests to determine the performance impact of different network topologies and configurations as specified by the IT and NETWORK Management team.

Use stress testing environment to isolate a memory leak which has been suspected but not isolated by the INC R&D team.

Timetable

Stress testing for June 2003 exam period took place in April 2003.

The project involved about three weeks testing effort over a six week period.

Software Architecture

INC exam software is developed using a three tiered web architecture.

Presentation Layer

Client Interface is implemented as a dynamically generated series of JAVASCRIPTS which run in a browser; current released versions of Internet Explorer or Netscape are supported. The user interface involves a series of questions with multiple choice answers. Each time a series of questions is generated they are taken from a database of possible exam questions. Each question has a different number of possible responses and also may include multimedia attachments representing pictures, video clips or other documents. The order of responses is randomly decided at the time the exam section is generated. The exam is progressive, the difficulty and number of subsequent questions depends of which answers have already been given to previous questions. The design of the INC exam user interface is especially focused on eliminating cheating. Students can choose either French or English versions of questions which can be generated. The client JAVASCRIPT application communicates to the Business logic every 15 seconds (tunable) to synchronize the current state of answers and progress in questions of the current exam section on the server. This frequent information exchange is achieved using cookies and allows students to resume exam on another workstation should some sort of hardware or software failure occur. During development and testing actual exam questions could not be used.

Application Layer

The Application Layer is implemented as a series of Perl programs running on a Microsoft Windows 2000 server. Microsoft IIS was used to service HTTP requests. The Perl used was a recent version of ActiveState perl. The bulk of the application is dedicated to building the exam section and sending it to the host. The amount of code is on the order of about 1000 lines of Perl.

Data Layer

The Data Layer includes two distinct sections, Question Repository and Response Data.

The Question Repository is implemented using a Microsoft SQL 2000 database which can be run on the same or a different server than the Application Layer. During most testing the database resided on the same server as IIS and all Perl application logic. During an EXAM session the SQL 2000 database is read but never written to.

Response data is kept in a series of small (under 10k) text files in the Windows 2000 file system. Standard Perl file IO is used to create and modify this files for all candidates.

Hardware Architecture

Intel based dual processor servers with 4 gigabytes of RAM were used to host the Windows 2000 servers. The examinations are run in 7 different centers across Canada.

During the EXAM a dedicated high speed network connects each individual center to the INC servers which are physically in Ottawa Canada.

Stress Testing

Several HP notebook computers configured with wireless network interfaces were made available to the stress testing project. These notebook computers are available to INC staff for different projects and to invited guests for use during meetings at the INC.

A stress testing lab was set up at the INC to facilitate this project. Load testing was accomplished using the SRI tool eValid which when suitably configured simulates sever dozens of virtual EXAM users on the HP notebook computers available to the lab. eValid is a browser based testing tool designed to perform functional and load testing of web based applications. Although other tools offer such functionality eValid was the most cost effective tool identified at the time. (commercial load testing tools from Rational, Mercury and Radview were considered). eValid licensing allowed unlimited load generation over a 15 day period for under US$2000 after initial licensing fees on the order of US$5000. Other solutions investigated offered similar functionality for about US$45000 and more. It should be noted that the features of the tool used for these tests were limited to measuring timing of EXAM question generation transactions (from the viewpoint of the client) and generating load by executing different virtual EXAM sessions on load generating systems.

Tools

Load Generator

Load was generated via scripts simulating examination sessions using the eValid tool. Some rented high power PCs (fast CPU and 1Gigabyte Ram) were rented for the testing period each of which were able to simulate approximatley100 virtual EXAM sessions (each running in a separate window).

Transaction Simulator

A workstation was reserved for the purpose of simulating typical EXAM transactions in a continuous loop as load was generated on other computers. eValid was used to run scripts a number of times keeping a detailed log of all http events. Logged data used included all load times and times to accomplish transactions.

System Console

In order to monitor performance of the servers during load generation the Microsoft performance monitor was run. Several different values were sampled including available memory and processor usage.

IIS was configured to generate detailed logs of all http events.

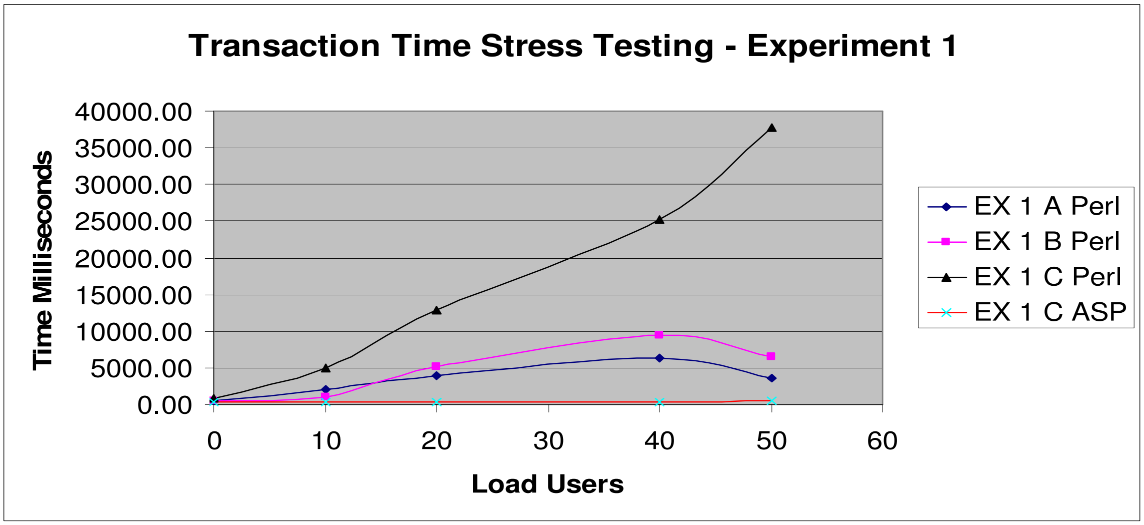
What If Analysis

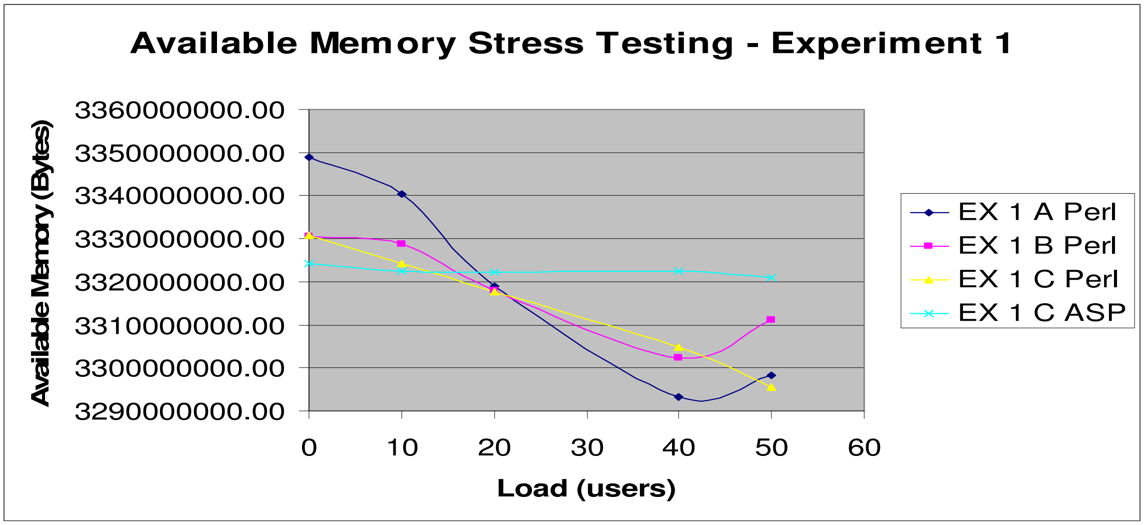
Three major What If experiments were done with the stress testing set up to study different design alternatives for future INC exam development. None of the experiments were designed to influence the upcoming June 2003 exam period.

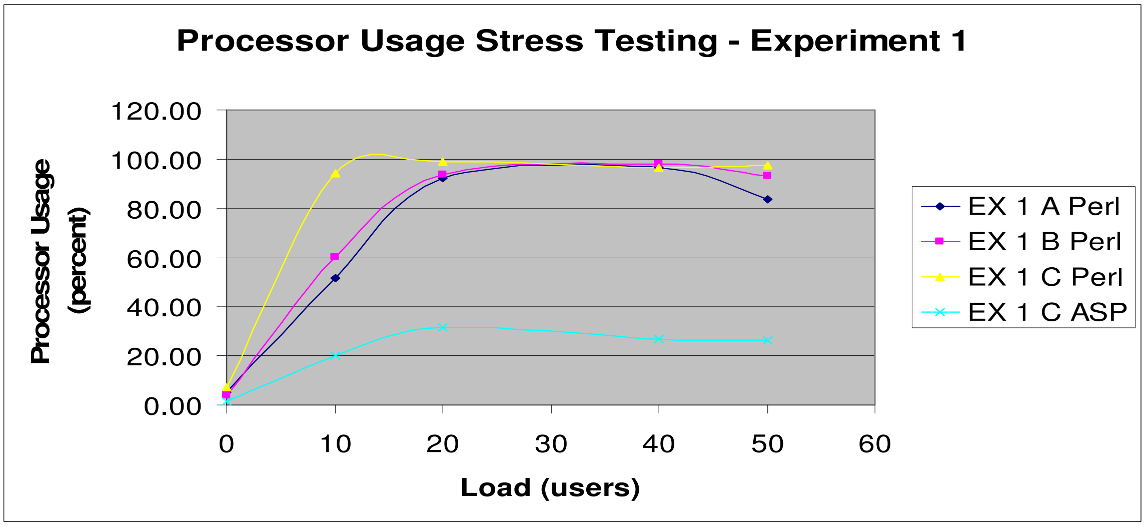
An example of the experiment was related to how Perl code should access SQL 2000 data. Three different Perl approaches and one ASP approach were coded by the development team.

Experiment	*Server Side Scripting*	*Database Access Mechanism*
1 aa	Perl	DBI with ODBC (status quo)
1 bb	Perl	Win32 ODBC
1 cc	Perl	OLE/ADO
1 c	ASP	OLE/ADO

The following three graphs represent the result. It is interesting to note that the 1 C ASP alternatives had the fastest transaction time and lowest processor usage. Each point of data was an average of five different test samples.





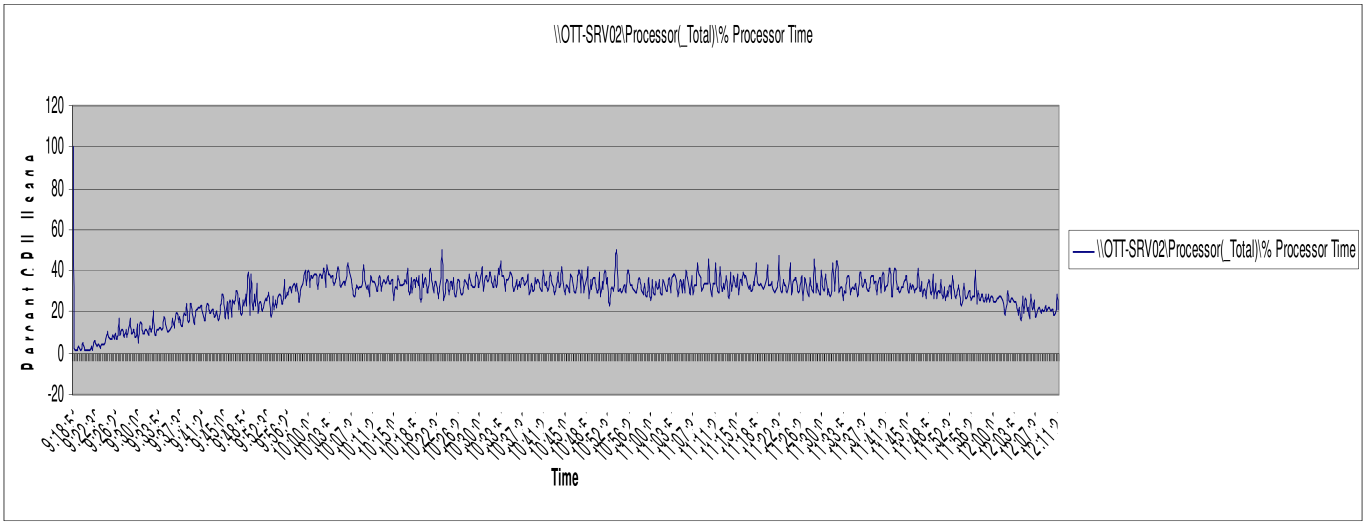


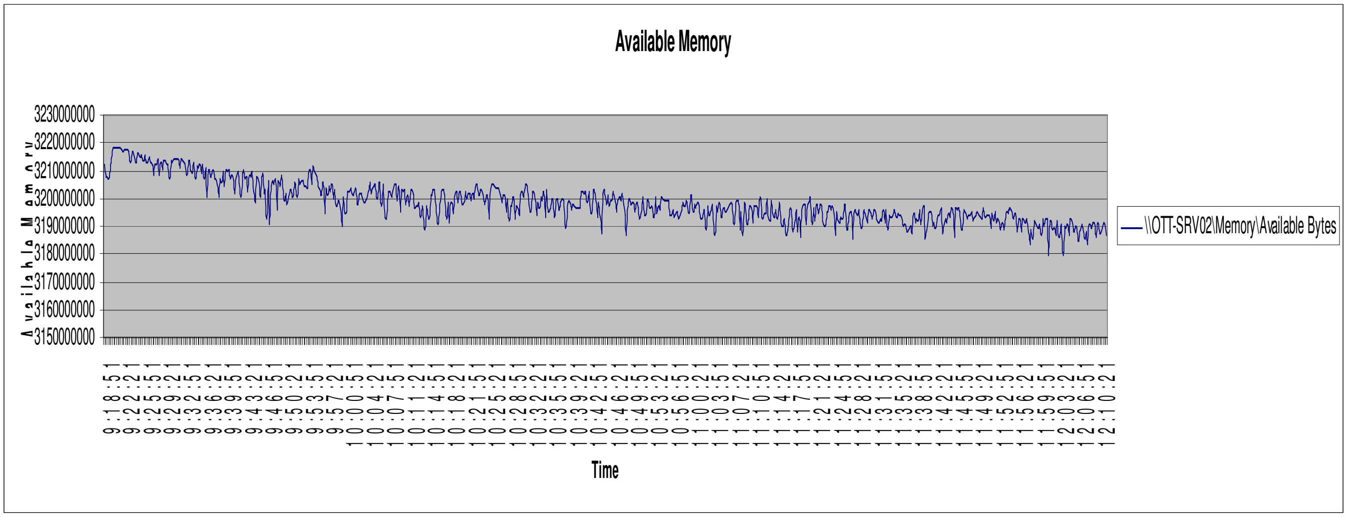
Real time simulation

A realistic load simulation test was run in order to observe how the INC Exam server system behaved under a load similar to that anticipated in live operation.

The *realistic load simulation* involved simulating 740 concurrent sessions.

While tests were running the CPU and memory usage was monitored by testing staff.





It was observed that during the simulation processor usage never exceeded 50% and averaged about 30%. Memory usage however was problematic and a distinct memory leak was observed. In fact the memory leak consumed an average of 186 megabytes of memory per day. (this is low compared with the available ram of 4 gigabytes but was still a large enough amount to merit further investigations).

The INC was quite concerned about the cause of the memory leak. Was the problem in INC code, the ActiveState Perl environment, the SQL 2000 database server or some other component of the system?

The stress testing lab was used to isolate this important bug.

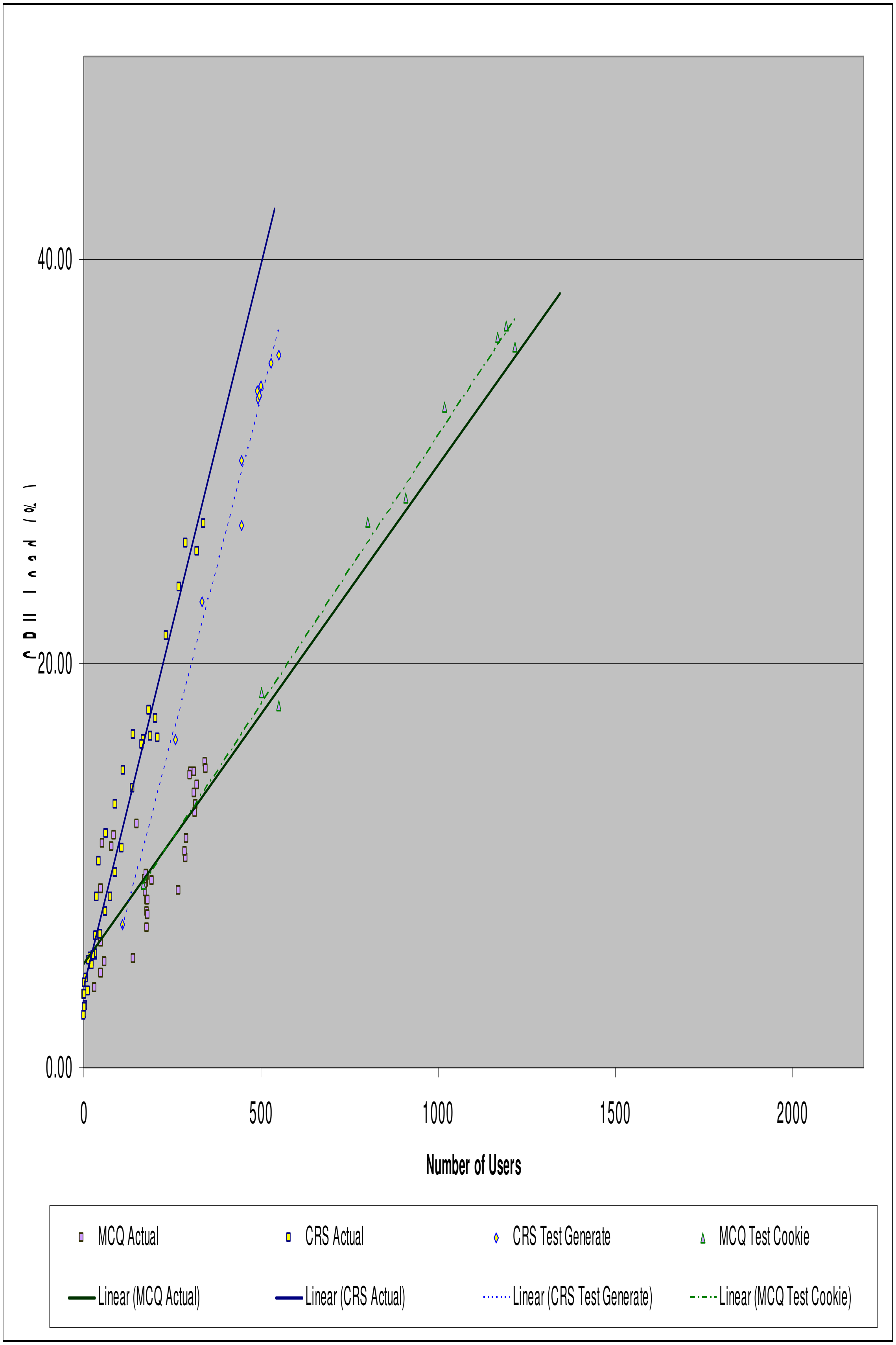
The source of the memory leak was identified by using a series of repeated overnight stress experiments with different performance monitory settings used. It was discovered that Microsoft SQL Server was the culprit – further investigation indicated that the SQL

2000 memory leak we identified was well know and updates and work arounds to the problem were readily available.

Results during live operation

During live operation performance and IIS logs were collected so that actual system behavior could be correlated to stress testing results to determine if our lab results were a reasonable predictor of actual behavior.

Results of the comparison are favorable and indicate that cpu usage in the lab simulation correlates well with actual cpu usage. This indicates that lab simulation of load exercises the server in a manner consistent with actual system usage. Further study may be required to quantify the relationship.


CPU Load (%)
40.00
20.00
0.00
0
500
1000
1500
2000
Number of Users
MCQ Actual
CRS Actual
CRS Test Generate
MCQ Test Cookie
Linear (MCQ Actual)
Linear (CRS Actual)
Linear (CRS Test Generate)
Linear (MCQ Test Cookie)

Conclusions

Experimental Development Approach Works

Stress testing by varying load and monitoring system performance enables development teams to perform what if experiments to help decide, well in advance, which technology or design solution to use. Stress testing can be used as part of the experimental development approach.

Flexible Licensing for Load Generation Tools Makes it Feasible

The use of eValid for load generation enabled the INC to set up a very inexpensive stress testing lab. eValid allows consumers to purchase unlimited load for a short period of time for under US$2000. Clients should design the stress experiments in advance, and develop all monitoring and load generation scripts in advance of the actual testing. eValid unlimited load licenses were only required during the actual stress testing. Normal eValid licenses were used for developing the various scripts used in the project.

Load Testing Tools allow for effective Bug Isolation

The INC project demonstrated that Load testing tools can be effective at helping to isolate nasty performance or load related bugs. The SQL 2000 memory leak was identified and isolated through a series of systematically executes stress testing experiments.

Actual results from Performance Monitor during live examination

Actual results of performance during live EXAMs showed that the simulation of load during the stress testing project was realistic and matched closely the actual operational behaviour of the system.

About Robert Sabourin

Robert Sabourin, P.Eng. has more than 20 years management experience leading teams of software development professionals to consistently deliver projects on-time, on-quality and on-budget.

As a respected member of the software engineering community, Robert has trained and mentored literally hundreds of top professionals in the field.

Robert is an Adjunct Professor of Software Engineering at McGill University who often speaks to conferences around the world on software engineering, Software Quality Assurance, testing and management issues.