

PERFORMANCE SERVICES – CASE STUDY

MISSION

360Logica performance services can help you understand the meaning of 'acceptable performance' thresholds in the context of your business. The performance characteristics of the software may be a key business differentiator. However, do you know what these thresholds are. Can you derive metrics to determine that the software meets your thresholds. Can you cost-effectively measure your chosen performance metrics. If you rely on third parties to implement the complete solution, what performance quality gates can you deploy to retain some degree of control. Even at the earliest stages, 360Logica can help you formulate strategies to meet the challenge of delivering high performance software systems.

BUSINESS IMPLICATION

Once a new system is in live operation, performance continues to become a critical factor. We provide a bespoke solution using our remote web load testing infrastructure, enabling you to select your performance test and quickly verify changes prior to implementation. We also recognize that identifying problems only provides part of the solution. 360Logica in-depth experience in technology development and testing ensures we can always identify the source of the problem and recommend a remedial course of action.

No matter what stage you are at in the lifecycle of your IT systems, 360Logica can provide performance services that optimize your systems' performance.

Performance Testing Objectives

The objective of a performance test is to demonstrate that the system meets requirements for transaction throughput and response times simultaneously.

The main deliverables from such a test, prior to execution, are automated test scripts and an infrastructure to be used to execute automated tests for extended periods. This infrastructure is an asset and an expensive one too, so it pays to make as much use of this infrastructure as possible. Fortunately, this infrastructure is a test bed, which can be re-used for other tests with broader objectives. A comprehensive test strategy would define a test infrastructure to enable all these objectives be met.

The performance testing goals are:

- End-to-end transaction response time measurements.
- Measure Application Server components performance under various loads.
- Measure database components performance under various loads.
- Monitor system resources under various loads.
- Measure the network delay between the server and clients.

APPLICATION OVERVIEW

The Application had search engine which retrieves data from the web service and stores in the application database. The application architecture comprises of web server handling the request from the end users, passing to the different layers of application through application server. The application server there after interacts with web service or the database as per the data requirement.

Application Server : JBoss server

Web Server : Tomcat

Database : My SQL

Search Engine: The key component of the application is the search engine which is using the JBoss cache and Caching of data done in database. The user request is handled and the initial attempt is made to search the data from the database cache, if not available in database the request is routed to the web service from where the data is fetched and displayed to the end user and also updates the database.

Pre-Requisites for Performance Testing

We can identify four pre-requisites for a performance test. Not all of these need be in place prior to planning or preparing the test (although this might be helpful), but rather, the list defines what is required before a test can be executed.

Stable system

A test team attempting to construct a performance test of a system whose software is of poor quality is unlikely to be successful. If the software crashes regularly, it will probably not withstand the relatively minor stress of repeated use. Testers will not be able to record scripts in the first instance, or may not be able to execute a test for a reasonable length of time before the software, middleware or operating systems crash.

Realistic test environment

The test environment should ideally be the production environment or a close simulation and be dedicated to the performance test team for the duration of the test. Often this is not possible. However, for the results of the test to be realistic, the test environment should be comparable to the actual production environment. Even with an environment which is somewhat different from the production environment, it should still be possible to interpret the results obtained using a model of the system to predict, with some confidence, the behavior of the target environment. A test environment which bears no similarity to the actual production environment may be useful for finding obscure errors in the code, but is, however, useless for a performance test.

Performance testing toolkit

The execution of a performance test must be, by its nature, completely automated. However, there are requirements for tools throughout the test process. Test tools are considered in more detail later, but the five main tool requirements for our 'Performance Testing Toolkit' are summarized here:

- Test database creation/maintenance

- Load generation tools

- Resource monitoring

- Results analysis and reporting.

Performance Requirements

Performance requirements normally comprise three components:

- Response time requirements
- Transaction volumes detailed in 'Load Profiles'
- Database volumes

TEST ENVIROMENT

The TEST ENVIRONMENT COMPRISES of the stable Application deployed on the servers. The server set up is same as that of the Production environment i.e. Web server and Application server on the same server class machine with Database server on the separate server machine.

The Performance testing exercise was conducted using Mercury Interactive LoadRunner 8.0. The LoadRunner (LR) Probes were deployed on the Servers and the data collected through LR Controller Console.

Application Env. setting:-

JBoss Session Timeout: **15 min**
JBoss Max Thread: **200**
JVM Heap Size: Initial - **3GB** Maximum - **4 GB**
Max DB Connections: **200**

APPROACH

Analyzing the situation, team decided to conduct the tests in a phase wise manner

Phase 1— Deriving Test Scenarios based on the requirements and scripting

In this phase, 360Loagica team derived test scenarios and user workflows by interacting with different stakeholders in the performance exercise and gleaning server logs to understand the existing usage patterns. The derived user models acted a starting point for scripting and determined the scripts structure for maximum reusability.

Phase 2—Benchmarking tests conducted

Test Scenarios were prepared in LoadRunner Controller by combining the user models derived in first phase with the scripts. The scenarios were then executed to collect the data points depending on the performance testing objectives. Also the scripts were executed a pre configured number of times to average out any inconsistencies because of occasional spikes in network/Server performance.

The performance data for benchmarking tests were presented to the client and determined.

Phase 3— Data Collection, Analysis and Bottleneck Identification

The data from different probes on the server was collected and analyzed by our Technology experts at the backend and possible bottlenecks identified.

Various server stats are also monitored: -

1. CPU Utilization of servers
2. Database Connection usage
3. JVM Usage
4. Threads Usage

EXECUTION

First Test Run

The first test run indicated that the application could handle only 100 concurrent users. By analyzing the server logs, it was identified that the inability to scale was due to insufficient pool queued length of web server and worker processes.

Second Test Run

Increasing the pool queued length to 10000 and worker processes to 100 on the web server increased the number of concurrent users to 240. However going above the user-load of 240 users the average response time unexpectedly increase which do not meet the client expectation.

On further analyzing the database monitors logs; the problem was traced to generation of dead lock conditions on database.

Thread dump is being taken at regular interval to analyze the proper state – Running/Idle of the various threads.

SUGGESTION FOR BOTTLENECK FINDINGS

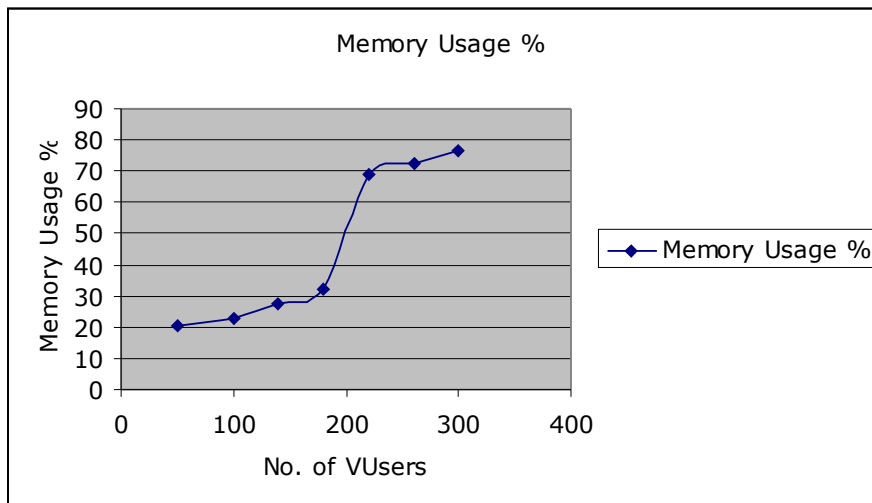
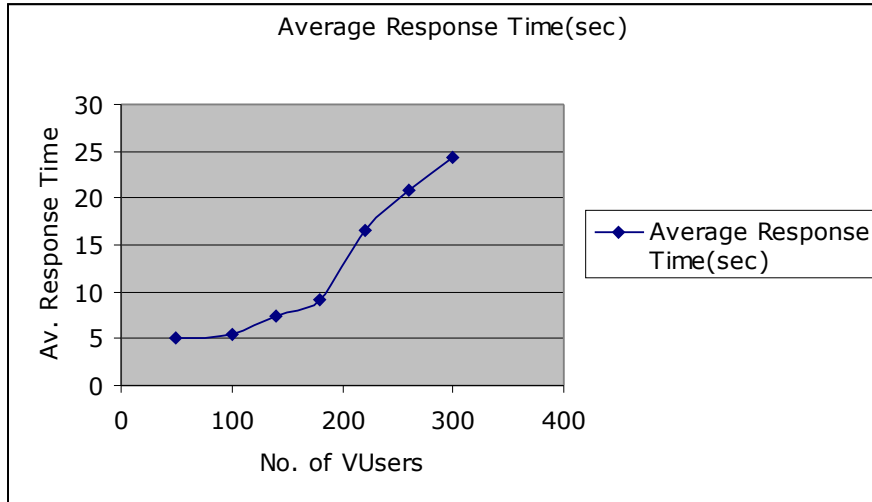
Web server Configuration changes:

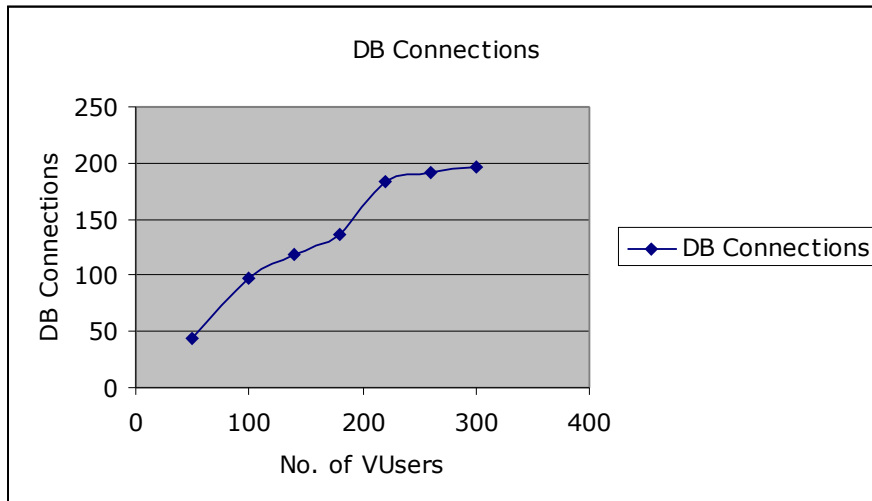
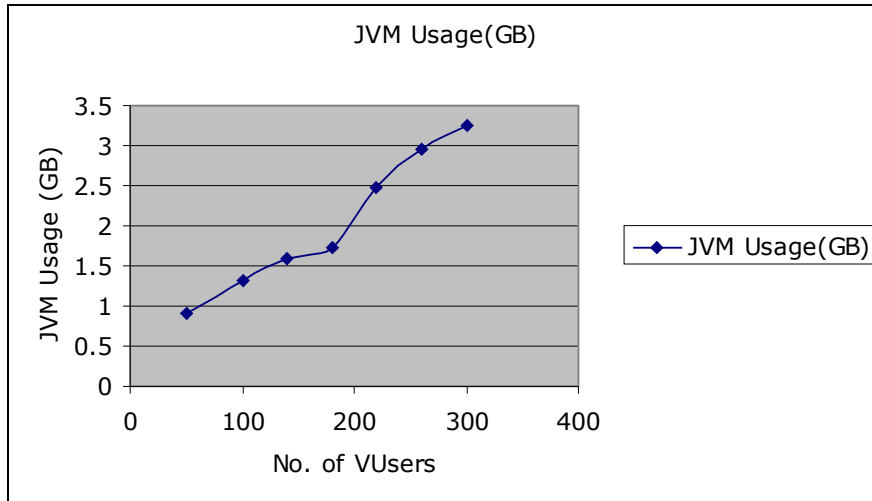
Suggested configuration changes on web server e.g. increasing the pool queued length to 10000 and worker processes to 100 to name a few.

Database design changes:

These included suggestions to execute queries without Locks where the data conflict had the least possibility. Using stored procedures instead of firing direct queries to database. Changing the indexing structure for the optimization of search queries.

RESULTS





Benefits:

Meeting Performance objectives for the application helped our client project their solution to hardware and ultimately edge out other competitors based on performance

-

Scalability projections given by the 360Logica team helped client gear up for expected user load before hand preventing them last minute headaches and saving customer from all important data loss during peak load on the application.