Importance of Stress Testing Web Survey Applications

Bryan Davis
Westat
Sr. System Analyst

Overview

- What is stress testing and why it is important
- When it is appropriate to consider stress testing
- High level overview of the stress testing process
- Remediation Strategies
- Lessons Learned

What is stress testing

Stress testing is a type of performance test focused on determining an application's robustness as well as measuring the availability, reliability and scalability of your server infrastructure under peak load conditions.

Why is stress testing important

Stress testing helps to identify:

- The maximum number of concurrent users your survey application can support
- Bottlenecks that might exist within your survey application that may interfere with or impede optimal performance
- Identify load capacity limitations of your web and database servers

Why is stress testing important

- Applications often perform well with a small number of users but often do not scale with larger numbers of users

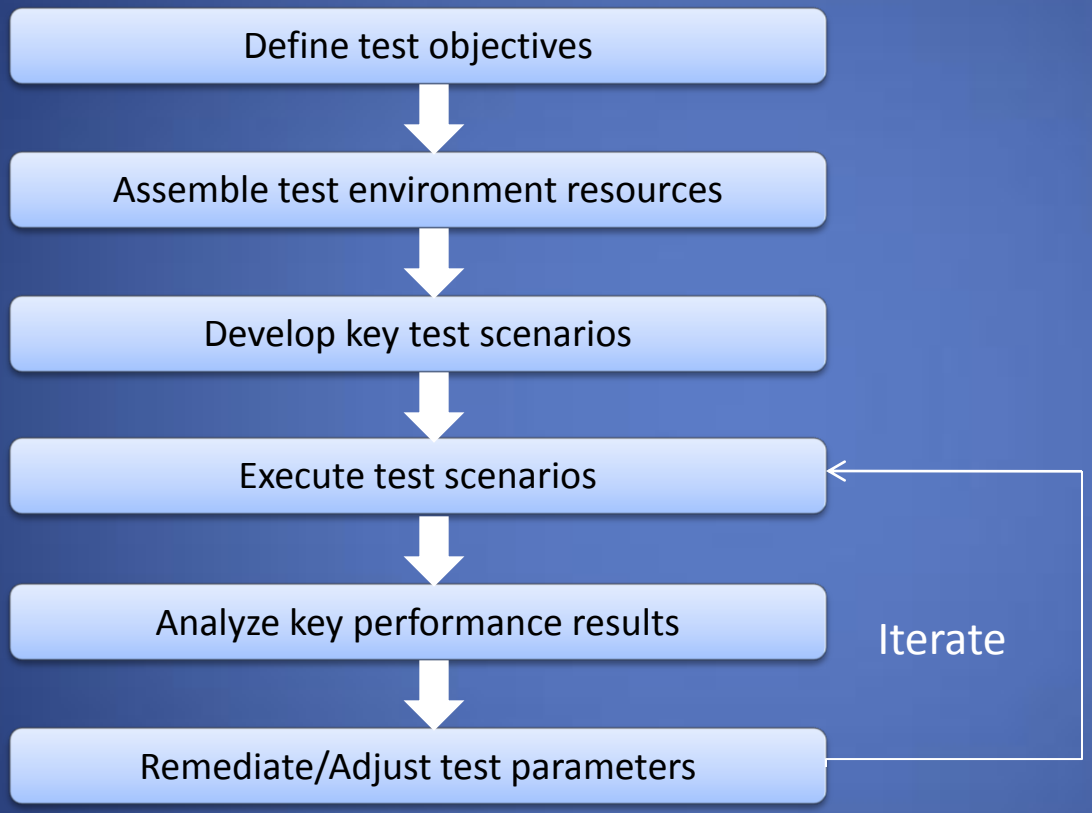
  - Inability to connect to the survey
  - Dropped user sessions
  - Slow response times
  - Unexpected error messages
  - Lost or corrupt data

All of these affect user perception and overall satisfaction with your survey and ultimately your response rates

When is it appropriate to consider stress testing

- Very large sample population
- Long and/or complex surveys (e.g. with complex skip patterns, data validation, edit checks)
- Compressed data collection period
- High profile mission critical survey
- Utilizing a new software platform, COTS or custom code, new hardware platform

High level overview of the stress testing process



Define your performance goals

For survey applications examples include:

- Identify the target number of users you want the system to support under normal and extreme conditions
- Define acceptable page response times
- Identify any hidden bugs, memory leaks in the application
- Identify resource thresholds before system performance begins to degrade or fail
- Determine if the current system architecture (i.e. network bandwidth, server capacity etc.) is adequate to meet your performance goals and if not what additional resources are needed

Assemble test environment resources

- Identify and assemble IT Staff, (Systems Analysts, Developers, Network administrators/ engineers that will participate in and support the tests

- Acquire, and configure hardware resources such as web and database servers, workstations, laptops that will be used to conduct/simulate the stress test

- Decide on which stress testing tools to use to test the survey application for performance, reliability and scalability:

  - Web application stress tool (WAS)
  - Web capacity analysis tool (WCAT)
  - Visual Studio Team System 2010 Ultimate Edition
  - Web Application Performance Test (WAPT) -- Softlogica

- Server monitoring tools such as Windows Performance Monitor and SQL Server Profiler should also be used to monitor resource allocation and utilization

- Allocate sufficient time to acquire and configure test environment

Identify and create key test scenarios

- To get the most out of the stress test, create test scenarios that matter most to the overall success of the application

- Identify test scenarios based on how critical they are to the overall application performance (i.e. those operations that are most likely to impact performance)

- These might include:
  - Users logging into the application (both valid and invalid login attempts)
  - Simulating users taking various navigations paths through the survey instrument
  - Disk intensive input/output (I/O) operations such as reading and writing to the database
  - Any other areas of your application identified as potential bottlenecks to verify that how well the application handles extreme conditions

Execute test scenarios

- Configure and record test scripts – using scenarios defined in previous step
  - Pick the number of simultaneous users (max user load)
  - Randomize the “think time” of each test scenario so that every test is not identical
  - Specify the length of time to run the test
- Start the test with a small number of users and incrementally add users
- Monitor tests and server CPU/memory, database utilization etc. using server monitoring tools

Analyze Results

- Key performance metrics to review include:
  - Max User Load
  - Number of completes
  - Number of failed requests
  - Page Response Time
  - Processor utilization
  - Memory utilization vs. memory availability
  - Disk (I/O) utilization
  - Network bandwidth utilization
  - Database read /write metrics
    - Transactions/sec
    - Transactions succeeded/failed

Remediate

- Fix errors or issues resulting from tests;
- Adjust parameters as necessary to reach performance thresholds;
- Repeat process as necessary

Common Remediation Strategies

Survey application

- Perform functional tests on your survey application to ensure that it functions and behaves as expected
  - Skip patterns work, data validation checks work; data are captured in your database etc.
- Stress test / Performance tune application
  - Ensure code is optimized and that there are no hidden bugs, issues such memory leaks;
  - RedGate Memory Profiler is a good tool to use for this
- Consider adding more than one survey question per page to minimize server hits;
- Consider saving data every 2nd or third page as opposed to every page to minimize writes to database
- Add indexes to optimize database reads; consider flatter database structure; avoid complex queries to read/write data to optimize performance

Common Remediation Strategies

Server side

- Consider using dedicated resources in lieu of shared server resources
- Scale up -- heavy up with more Ram and CPU processors
- Scale out – network load balance on multiple servers
  - Distributes the server load and also provides redundancy in the event of a server failure

Survey administration / operations

- Consider staggering notifications to study participants

Lessons Learned

- Stress testing is not easy but that's not to say it's not necessary or to discourage you from doing it
  - It takes time to plan, prepare, tune/optimize the application and infrastructure
  - Requires knowledge, experience to plan, devise tests plans / scenarios, configure software and interpret data produced from stress test
  - Consider hiring consultants if in-house expertise is not available or non-existent
- Stress testing is not free
  - There is a business and cost decision that has to be made to ensure performance objectives are met
  - 3 elements to consider
    - Labor costs associated with stress testing
    - Infrastructure capacity cost
      - May require additional hardware to simulate load on the server
    - Time
      - Must build into development schedule / allocate sufficient time to perform tests
      - Stress testing is typically an iterative process to achieve performance goals

Email Sent, Surveys Completed