Software Testing Genius

Performance and Load Testing

Software Testing Genius

Part 1

Performance & Load Testing Basics

>>>>>>>>>>>>>>>>>>>>>>>>> www.softwaretestinggenius.com <<<<<<<<<<<<<<<<<<<<<<<<

Software Testing Genius

Performance & Load Testing Basics

- ✓ Introduction to Performance Testing
- ✓ Difference between Performance, Load and Stress Testing
- ✓ Why Performance Testing?
- ✓ When is it required?
- ✓ What should be tested?
- ✓ Performance Testing Process
- ✓ Load Test configuration for a web system
- ✓ Practice Questions

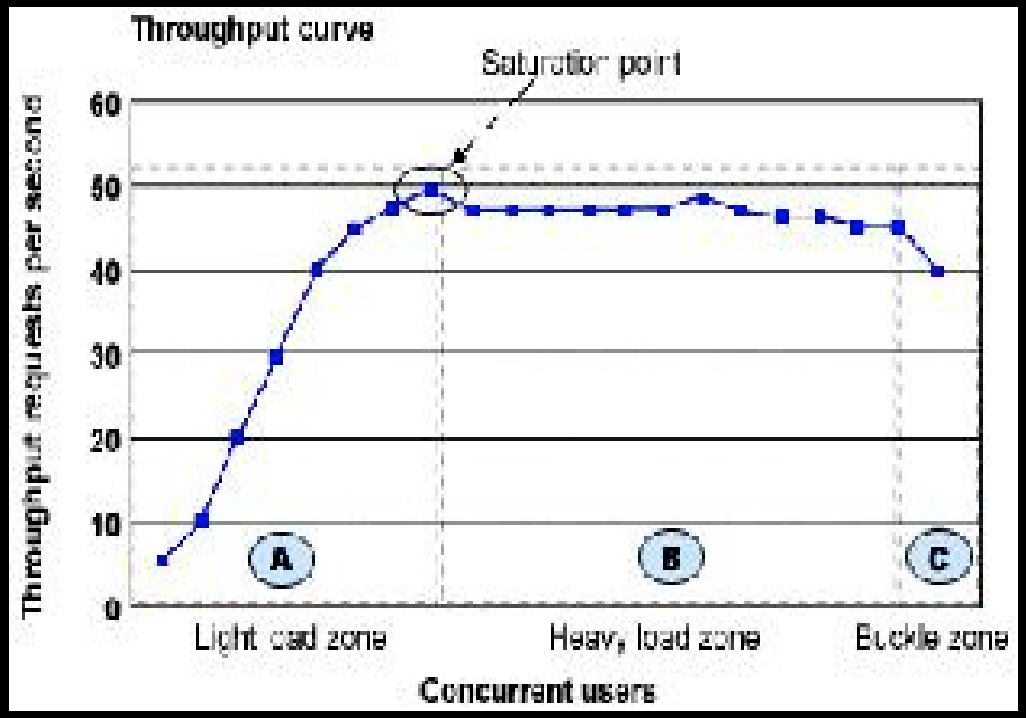
Introduction to Performance Testing

- Performance testing is the process of determining the speed or effectiveness of a computer, network, software program or device.

- Before going into the details, we should understand the factors that governs Performance testing:

  - ✓ Throughput
  - ✓ Response Time
  - ✓ Tuning
  - ✓ Benchmarking

Throughput



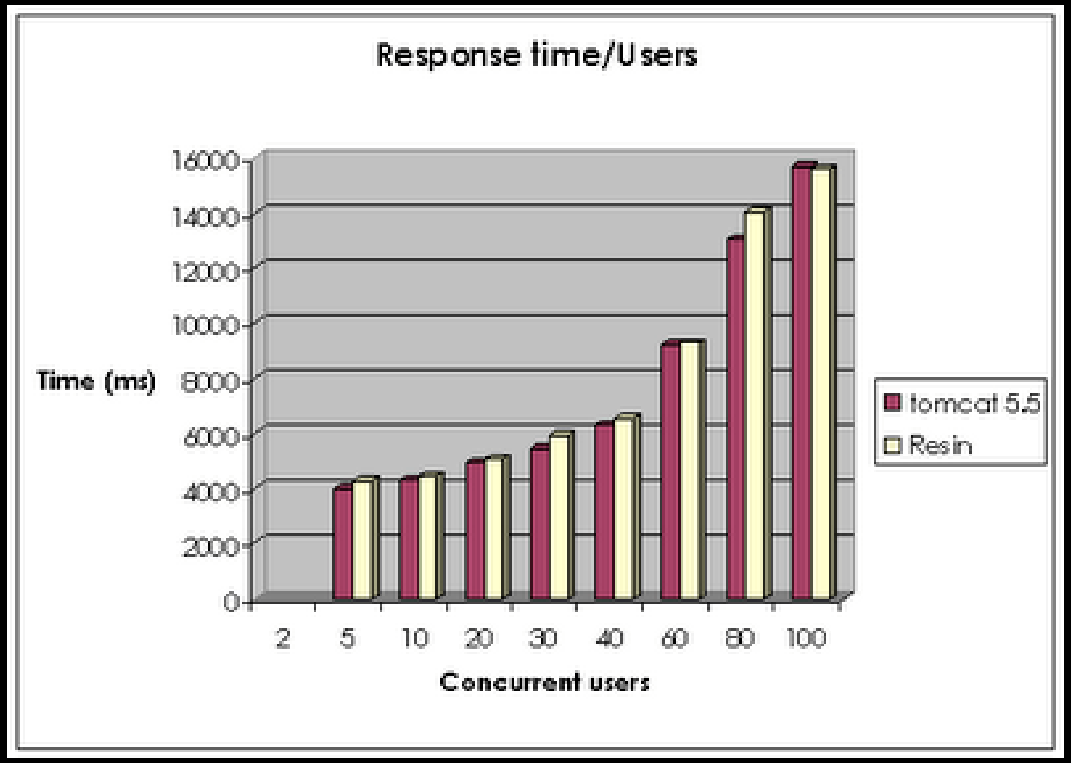
- Capability of a product to handle multiple transactions in a give period.
- Throughput represents the number of requests/business transactions processed by the product in a specified time duration.
- As the number of concurrent users increase, the throughput increases almost linearly with the number of requests. As there is very little congestion within the Application Server system queues.

Throughput

- In the heavy load zone or Section B, as the concurrent client load increases, throughput remains relatively constant.

- In Section C (the buckle zone) one or more of the system components have become exhausted and throughput starts to degrade. For example, the system might enter the buckle zone when the network connections at the Web server exhaust the limits of the network adapter or if the requests exceed operating system limits for file handles.

Response Time

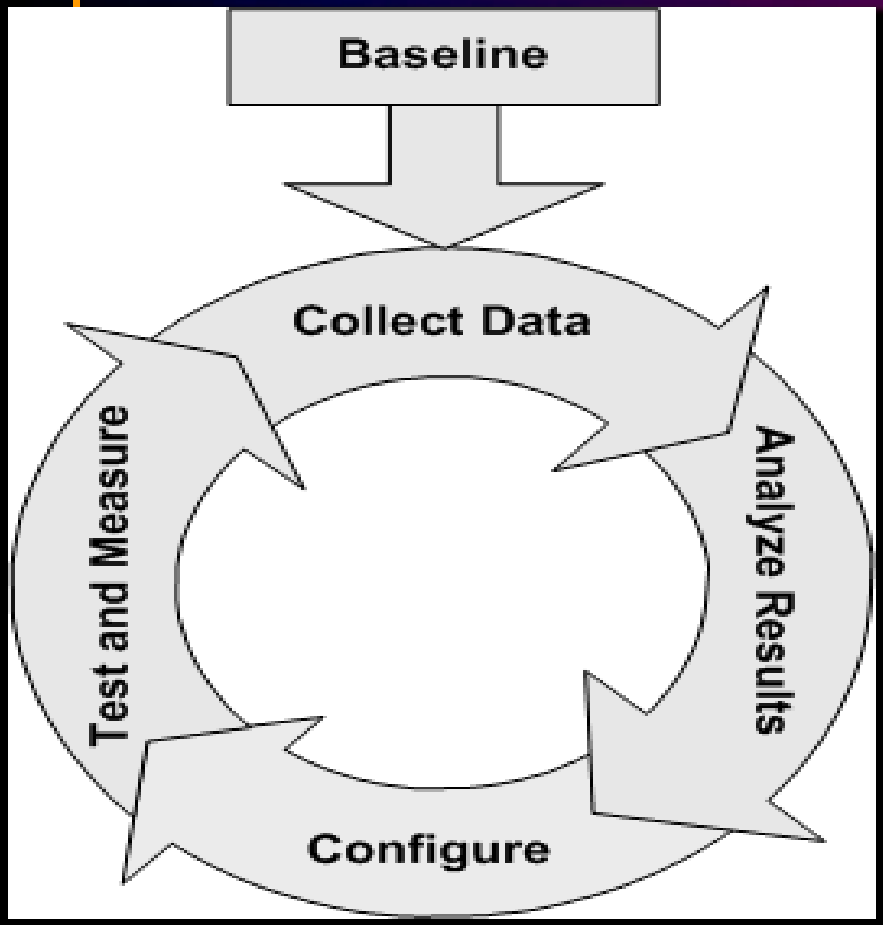


It is equally important to find out how much time each of the transactions took to complete.

Response time is defined as the delay between the point of request and the first response from the product.

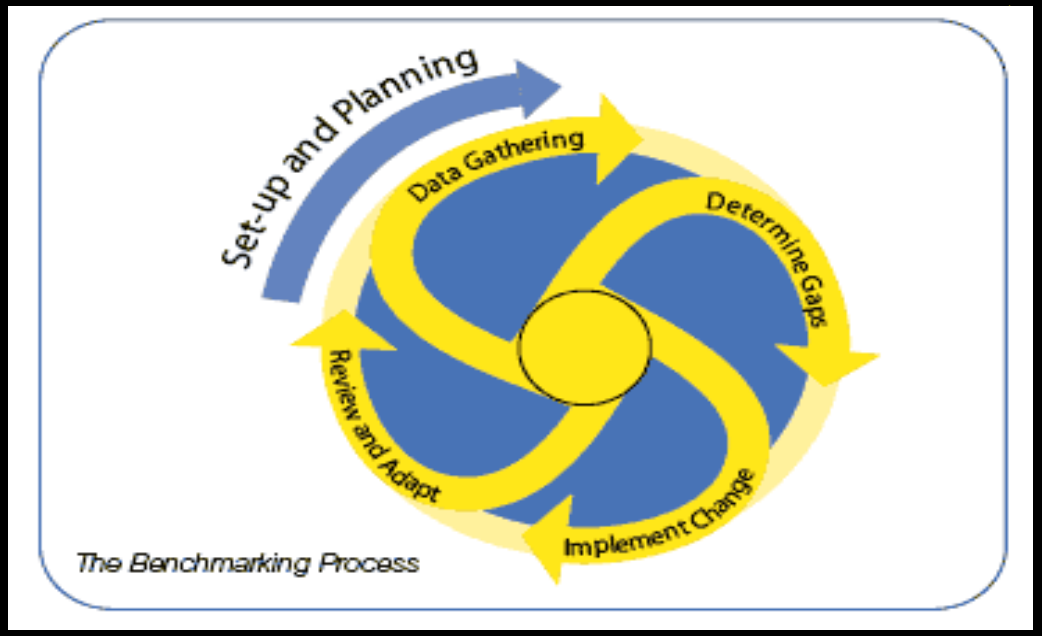
The response time increases proportionally to the user load.

Tuning



- Tuning is the procedure by which product performance is enhanced by setting different values to the parameters of the product, operating system and other components.
- Tuning improves the product performance without having to touch the source code of the product.

Benchmarking



A very well-improved performance of a product makes no business sense if that performance does not match up to the competitive products.

A careful analysis is needed to chalk out the list of transactions to be compared across products so that an apple-apple comparison becomes possible.

Performance Testing- Definition

- The testing to evaluate the response time (speed), throughput and utilization of system to execute its required functions in comparison with different versions of the same product or a different competitive product is called Performance Testing.
- Performance testing is done to derive benchmark numbers for the system.
- Heavy load is not applied to the system
- Tuning is performed until the system under test achieves the expected levels of performance.

Difference between Performance, Load and Stress Testing

Load Testing

- Process of exercising the system under test by feeding it the largest tasks it can operate with.
- Constantly increasing the load on the system via automated tools to simulate real time scenario with virtual users.

Examples:

- Testing a word processor by editing a very large document.
- For Web Application load is defined in terms of concurrent users or HTTP connections.

Difference between Performance, Load and Stress Testing

Stress Testing

- Trying to break the system under test by overwhelming its resources or by taking resources away from it.
- Purpose is to make sure that the system fails and recovers gracefully.

Example:

- Double the baseline number for concurrent users/HTTP connections.
- Randomly shut down and restart ports on the network switches/routers that connects servers.

Why Performance Testing?

- **The week of Feb 6, 2000:**
  Hackers delivered over 1-Billion transactions concurrently to each one of these sites.
  - Yahoo
  - eBay
  - Buy.com
  - Amazon
  - eTrade

How did you think they performed ?

Why Performance Testing

- Identifies problems early on before they become costly to resolve.
- Reduces development cycles.
- Produces better quality, more scalable code.
- Prevents revenue and credibility loss due to poor Web site performance.
- Enables intelligent planning for future expansion.
- To ensure that the system meets performance expectations such as response time, throughput etc. under given levels of load.
- Expose bugs that do not surface in cursory testing, such as memory management bugs, memory leaks, buffer overflows, etc.

When is it required?

Design Phase:

Pages containing lots of images and multimedia for reasonable wait times. Heavy loads are less important than knowing which types of content cause slowdowns.

Development Phase:

To check results of individual pages and processes, looking for breaking points, unnecessary code and bottlenecks.

Deployment Phase:

To identify the minimum hardware and software requirements for the application.

What should be tested?

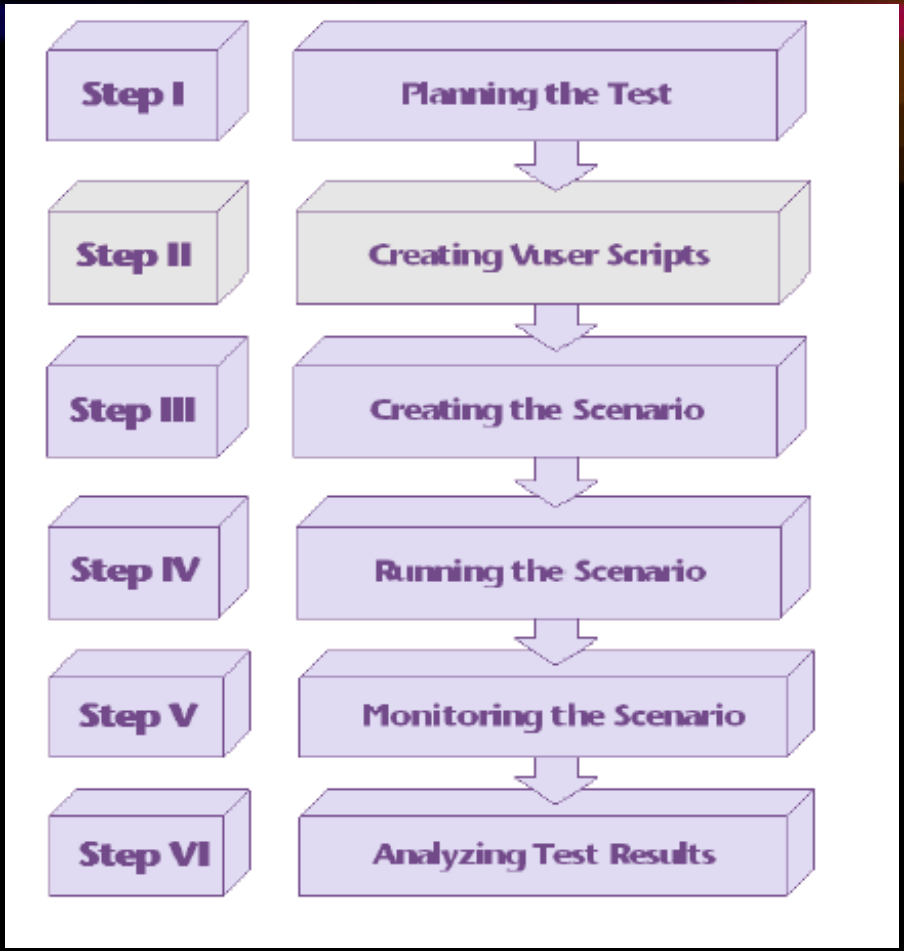
- **High frequency transactions:** The most frequently used transactions have the potential to impact the performance of all of the other transactions if they are not efficient.

- **Mission Critical transactions:** The more important transactions that facilitate the core objectives of the system should be included, as failure under load of these transactions has, by definition, the greatest impact.

- **Read Transactions:** At least one READ ONLY transaction should be included, so that performance of such transactions can be differentiated from other more complex transactions.

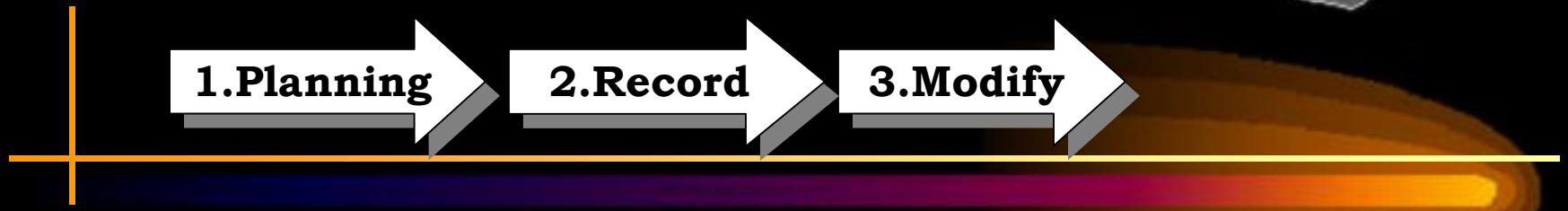
- **Update Transactions:** At least one update transaction should be included so that performance of such transactions can be differentiated from other transactions.

Performance Testing Process



1.Planning

- **Determine the performance testing objectives**
- **Describe the application to test using a application model**
  - Describe the Hardware environment
  - Create a Benchmark (Agenda) to be recorded in Phase 2.
    - Define what tasks each user will perform
    - Define (or estimate) the percentage of users per task.



Record

- Record the defined testing activities that will be used as a foundation for your load test scripts.
- One activity per task or multiple activities depending on user task definition

Modify

- Modify load test scripts defined by recorder to reflect more realistic Load test simulations.
- Defining the project, users
- Randomize parameters (Data, times, environment)
- Randomize user activities that occur during the load test

4. Execute

Virtual Users (VUs):

Start: 5

Incremented by: 5

Maximum: 200

Think Time: 5 sec

Test Goals

Max Response Time <= 20 Sec

Test Script:

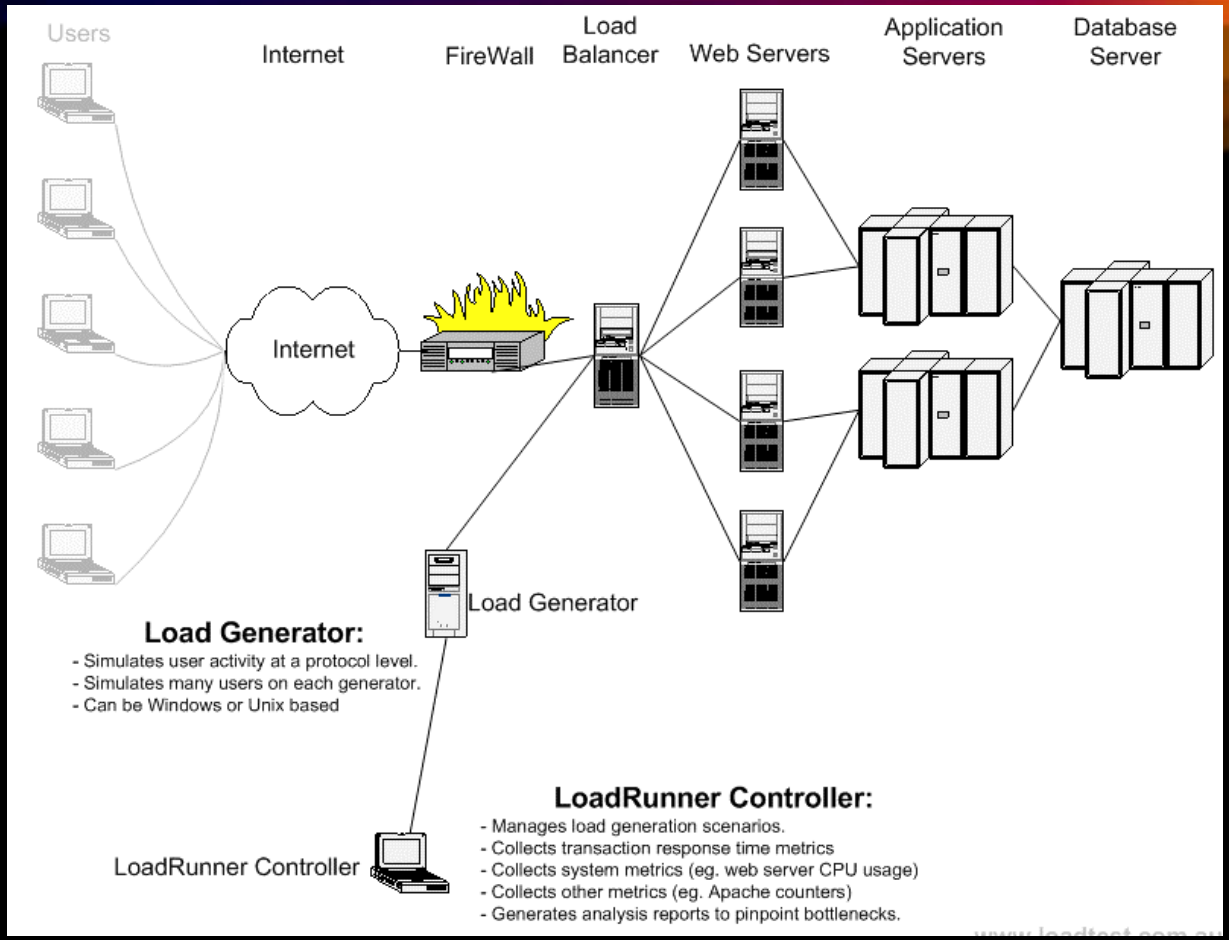
One typical user from login through completion.

5.Monitor → 6. Analyze

- **Monitoring the scenario:** We monitor scenario execution using the various online runtime monitors.

- **Analysing test results:** During scenario execution, the tool records the performance of the application under different loads. We use the graphs and reports to analyse the application's performance.

Load Test configuration for a web system



Questions to Review your Skills

- What are the factors that governs Performance Testing?
- How are Throughput and Response time related with user load?
- How do we decide whether the application passed or failed the load test?
- What do you mean by Capacity, Stability and Scalability of an application.
- What is the difference between Performance, Load and Stress testing?
- What is Longevity, endurance, spike and Volume Testing?
- At what point in SDLC, performance testing is required?
- How to identify the transactions in a complete application for load testing?
- Define the 6 steps involved in Performance Testing Process?
- Explain the Load Test configuration of a web system and what is the role of Load Generators in it?

Software Testing Genius

Part 2

Load Test Planning

>>>>>>>>>>>>>>>>>>>>>>>>> www.softwaretestinggenius.com <<<<<<<<<<<<<<<<<<<<<<<<

Load Test Planning

- ✓ Why Planning
- ✓ Analysing the Application
- ✓ Defining Testing Objectives
- ✓ Gathering Requirements
- ✓ Load Test Checklist
- ✓ Practice Questions

Why Planning

- As in any type of system testing, a well-defined test plan is the first essential step to successful testing.

Planning load testing helps to:

- **Build test scenarios that accurately emulate your working environment:** Load testing means testing the application under typical working conditions, and checking for system performance, reliability, capacity, and so forth.
- **Understand which resources are required for testing:** Application testing requires hardware, software, and human resources. Before beginning testing, we should know which resources are available and decide how to use them effectively.
- **Define success criteria in measurable terms:** Focused testing goals and test criteria ensure successful testing. For example, it's not enough to define vague objectives like "Check server response time under heavy load." A more focused success criterion would be "Check that 50 customers can check their account balance simultaneously & that server response time will not exceed 1- minute"

Why Planning

- **Load test planning is a three-step process:**
  - ✓ **Analyzing the Application**
    - Analysis ensures that the testing environment we create using LoadRunner will accurately reflect the environment and configuration of the application under test.
  - ✓ **Defining Testing Objectives**
    - Before testing, we should define exactly what we want to accomplish.
  - ✓ **Gathering Requirements**
    - All the requirements and resources should be evaluated and collected beforehand to avoid any last minute hurdles.

Analyzing the Application

- Load testing does not require as much knowledge of the application as functional testing does.
- Load tester should have some operational knowledge of the application to be tested.
- Load tester should have the idea on how the application is actually used in production to make an informed estimate.
- Load tester must know the application architecture (Client Server, Local Deployment, Live URL), Platform and Database used.

Defining Testing Objectives

- Determining and recording performance testing objectives involves communicating with the team to establish and update these objectives as the project advances through milestones

- **Performance, Load or Stress testing:** Type and scope of testing should be clear as each type of testing has different requirements.

- **Goal Setting:** General load testing objectives should be defined.

Defining Testing Objectives

- **Common Objectives:**
  - ✓ Measuring end-user response time
  - ✓ Defining optimal hardware configuration
  - ✓ Checking reliability
  - ✓ Assist the development team in determining the performance characteristics for various configuration options
  - ✓ Ensure that the new production hardware is no slower than the previous release
  - ✓ Provide input data for scalability and capacity-planning efforts
  - ✓ Determine if the application is ready for deployment to production
  - ✓ Detect bottlenecks to be tuned

Defining Testing Objectives

Stating Objectives in Measurable Terms:

- Once you decide on your general load testing objectives, you should identify more focused goals by stating your objectives in measurable terms.
- To provide a baseline for evaluation, determine exactly what constitutes acceptable and unacceptable test results.
- For example:
  - **General Objective:**
    - Product Evaluation: choose hardware for the Web server.
  - **Focused Objective:**
    - Product Evaluation: run the same group of 300 virtual users on two different servers, HP and NEC. When all 300 users simultaneously browse the pages of your Web application, determine which hardware gives a better response time.

Gathering Requirements

Users: Identify all the types of people and processes that can put load on the application or system.

- ✓ Defining the types of primary end users of the application or system such as purchasers, claims processors, and sales reps
- ✓ Add other types of users such as system administrators, managers, and report readers who use the application or system but are not the primary users.
- ✓ Add types of non-human users such as batch processes, system backups, bulk data loads and anything else that may add load or consume system resources.

Transactions: For each type of user we identified in the previous step, identify the tasks that the user performs.

Production Environment:

- ✓ Performance and capacity of an application is significantly affected by the hardware and software components on which it executes.

Gathering Requirements

Production Environment:

- ✓ Speed, capacity, IP address and name, version numbers and other significant information.

Test Environment:

- ✓ Should be similar to the production environment as is possible to be able to get meaningful performance results.
- ✓ It is important that the databases be set up with the same amount of data in the same proportions as the production environment as that can substantially affect the performance.

Scenarios:

- ✓ Select the use cases to include
- ✓ Determine how many instances of each use case will run concurrently
- ✓ Determine how often the use cases will execute per hour
- ✓ Select the test environment

Gathering Requirements

Load test Tool:

- ✓ Ability to parameterize data.
- ✓ Ability to capture dynamic data and use on subsequent requests.
- ✓ Application infrastructure monitoring.
- ✓ Support for the application's protocols

Load test Lab must include the following:

- ✓ Test Servers.
- ✓ Databases.
- ✓ Network elements, operating systems and clients and server hardware.

Load Test Check List

Planning

- ✓ Objective goals defined
- ✓ Test plan written and reviewed

Staff Skills

- ✓ Experience in load testing
- ✓ Application knowledge
- ✓ Systems knowledge
- ✓ Communication and people skills

Support Staff

- ✓ Key staff identified and allocated

Load Test Lab

- ✓ Test servers allocated
- ✓ Databases populated
- ✓ Load test tools allocated

Questions to Review your Skills

- Why planning is required before starting load test?
- What are the three steps involved in load test planning?
- What information should be collected about the application to be load tested?
- What are the common testing objectives?
- State the following testing objective in measurable term: “Ensure that the new production hardware is not slower than the previous release”
- Why is the knowledge of Production Environment necessary before load test?
- What are the factors that need to be considered for creating a scenario?
- How to choose a load test tool?
- What are the requirements to setup a load test lab?
- What are main points in a Load test checklist?

Software Testing Genius

Part 3

Load Testing Tools

>>>>>>>>>>>>>>>>>>>>>>>>> www.softwaretestinggenius.com <<<<<<<<<<<<<<<<<<<<<<<<

Load Testing Tools

- ✓ Manual testing Limitations
- ✓ Benefits of Automation
- ✓ Tools used for Performance Testing
- ✓ Practice Questions

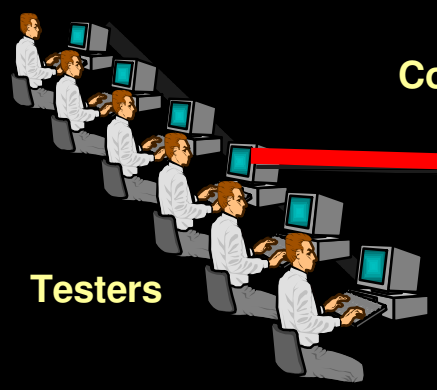
Manual Testing Limitations

Do you have the testing resources?
- Testing personnel
- Client machines

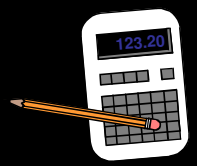
How do you coordinate and synchronize users?

How do you collect and analyze results?

How do you achieve test repeatability?



Analysis?



Load Generation

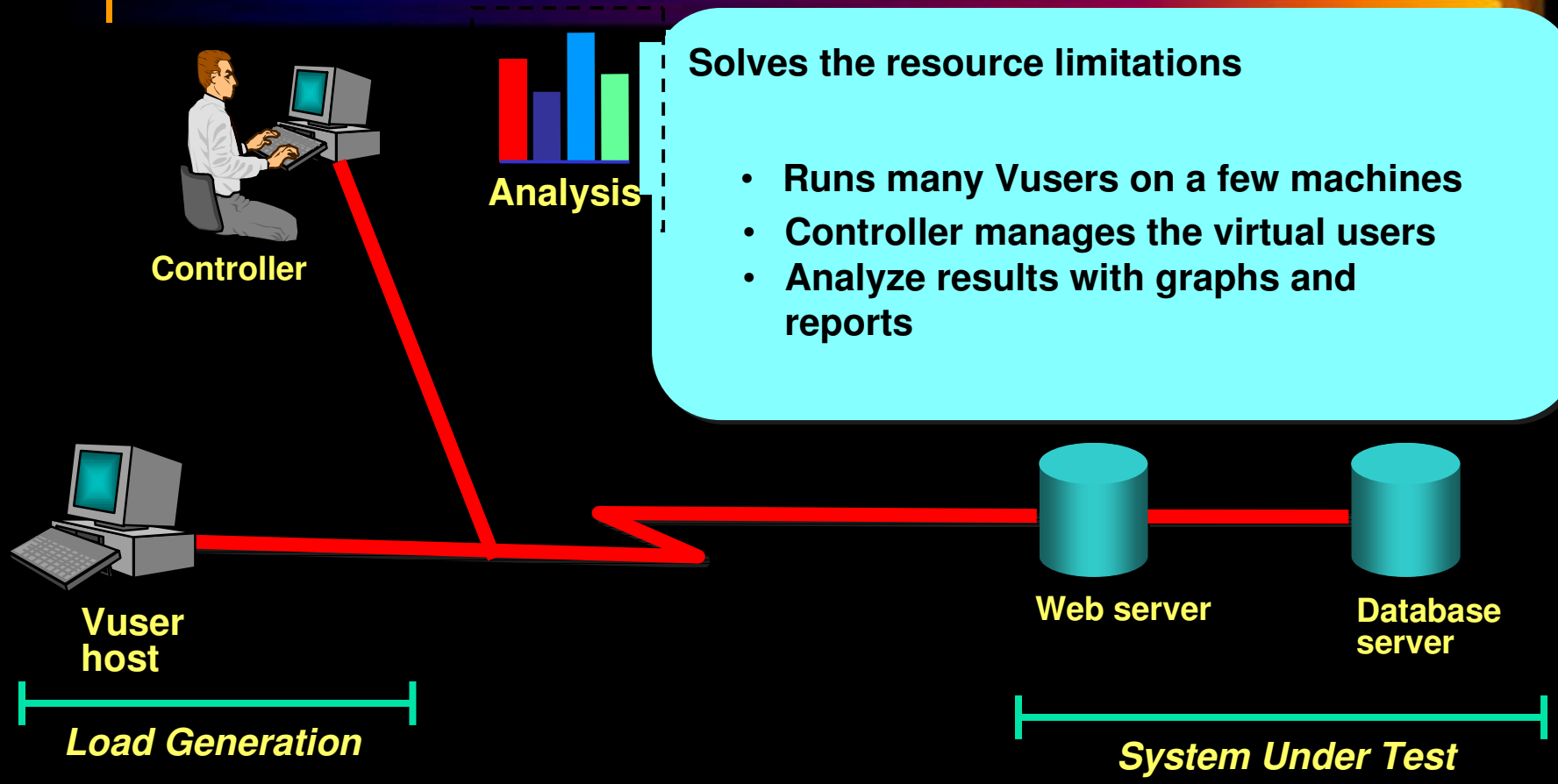
System Under Test

Manual Testing Limitations

Manual Testing Limitations

- ✓ Expensive, requiring large amounts of both personnel and machinery.
- ✓ Complicated, especially co-ordinating and synchronising multiple testers
- ✓ Involves a high degree of organization, especially to record and analyse results meaningfully
- ✓ Repeatability of the manual tests is limited

Benefits of Automation



Solves the resource limitations

- Runs many Vusers on a few machines
- Controller manages the virtual users
- Analyze results with graphs and reports

Benefits of Automation

Using Automated Tools

- ✓ Reduces personnel requirements by replacing human users with virtual users or Vusers. These Vusers emulate the behaviour of real users
- ✓ Because numerous Vusers can run on a single computer, the tool reduces the amount of hardware required for testing.
- ✓ Monitors the application performance online, enabling you to fine-tune your system during test execution.
- ✓ It automatically records the performance of the application during a test. You can choose from a wide variety of graphs and reports to view the performance data.
- ✓ Because the tests are fully automated, you can easily repeat them as often as you need.

Tools used for Performance Testing

Open Source

- OpenSTA
- Diesel Test
- TestMaker
- Grinder
- LoadSim
- Jmeter
- Rubis

Commercial

- LoadRunner
- Silk Performer
- Qengine
- Empirix e-Load

OpenSTA

- Developed in C++
- HTTP Load Test Application

Advantages:

- Open Source Software
- A user-friendly graphical interface
- The script capture from the browser
- The monitoring functionality

Drawbacks:

- Only designed for Windows
- Only for HTTP

DieselTest

- Software designed in Delphi5
- For systems under NT Environment
- For HTTP/HTTPS applications

Advantages:

- Open Source
- The quality of the chart
- Simple and fast to use
- The logging functionality

Drawbacks:

- The manual edition of the tests is badly designed
- The ambiguity of certain results
- Distributed tests are impossible
- Specific technology environment (Delphi, NT)

TestMaker

- Developed in Java
- For HTTP, HTTPS, SOAP, XML-RPC, Mails (SMTP, POP3 and IMAP) applications

Advantages:

- The possibility to build any kind of test agent
- The power of Java programming with some Python simplifications
- Open source

Drawbacks:

- Familiarity with the Jython scripting language, Java language and to write it from scratch
- The monitoring tools are very basic, since it is limited to the response analysis
- Must pay for distributed testing

Grinder

- Generic framework for load testing any kind of target systems, with scenario in Jython
- Developed in Java

Advantages:

- Open Source
- You can test everything with scripts in Jython

Drawbacks:

- Deployment for distributed test
- Poor results and graphical interface

LoadSim

- LoadSim is an open source software developed in Java, which is designed for HTTP distributed load testing

Advantages:

- Open Source
- Generation of script
- Each client have a different configuration (user, script…)

Drawbacks:

- No graphical interface
- Poor results
- No graphical representation of result
- No monitoring

Jmeter

- 100% Java desktop application
- For Web and FTP, Java, SOAP/XML-RPC, JDBC applications

Advantages:

- Open Source
- The distributed testing
- Various target systems
- Extensibility: Pluggable samplers allow unlimited testing capabilities

Drawbacks:

- Chart representation quite confuse
- Terminology not very clear
- Necessary to start remote machine one by one
- Remote machines must be declared in a property file before starting application

Rubis

- Provided with some load-test tool (designed for Rubis, but some parts of code could be re-used) and a monitoring system.
- Developed in Java.

Advantages:

- Open Source
- Monitoring capabilities
- Charts representations and automatic generation of HTML report

Drawbacks:

- Specific to Unix environment and Rubis application

Empirix eLoad

- Accurate testing of the response times and scalability of web applications and web services
- Recording in VBscript

Advantages:

- Can simulate hundreds and thousands of concurrent users
- Monitoring capabilities and Charts representation
- Reasonable Price

Drawbacks:

- Complex User Interface
- Limitations in recording of complex scenarios

Questions to Review your Skills

- What are the limitations of manual load testing?
- Why tools are used for automating load test?
- List 5 Open Source and 5 Commercial load test tools.
- What are the disadvantages of LoadRunner?
- Explain the following Load Test tools: Silk Performer, Qengine.
- Give a detailed comparison between Empirix E-load and LoadRunner.
- Which other tools are commonly used for load testing?

Software Testing Genius

Thank You