If you havent done this already,

Ensure Java is installed (>= 1.6):

- Run 'java –version' at command line to confirm

Have jmeter downloaded:

- jmeter.org

Firefox:

- mozilla.org

Firefox Addons:

- FoxyProxy Basic
- Firebug

Performance Testing using Apache JMeter

Hans Dushanthakumar

hans@thoughtworks.com

Leonor Salazar

themexican@thoughtworks.com

ThoughtWorks®

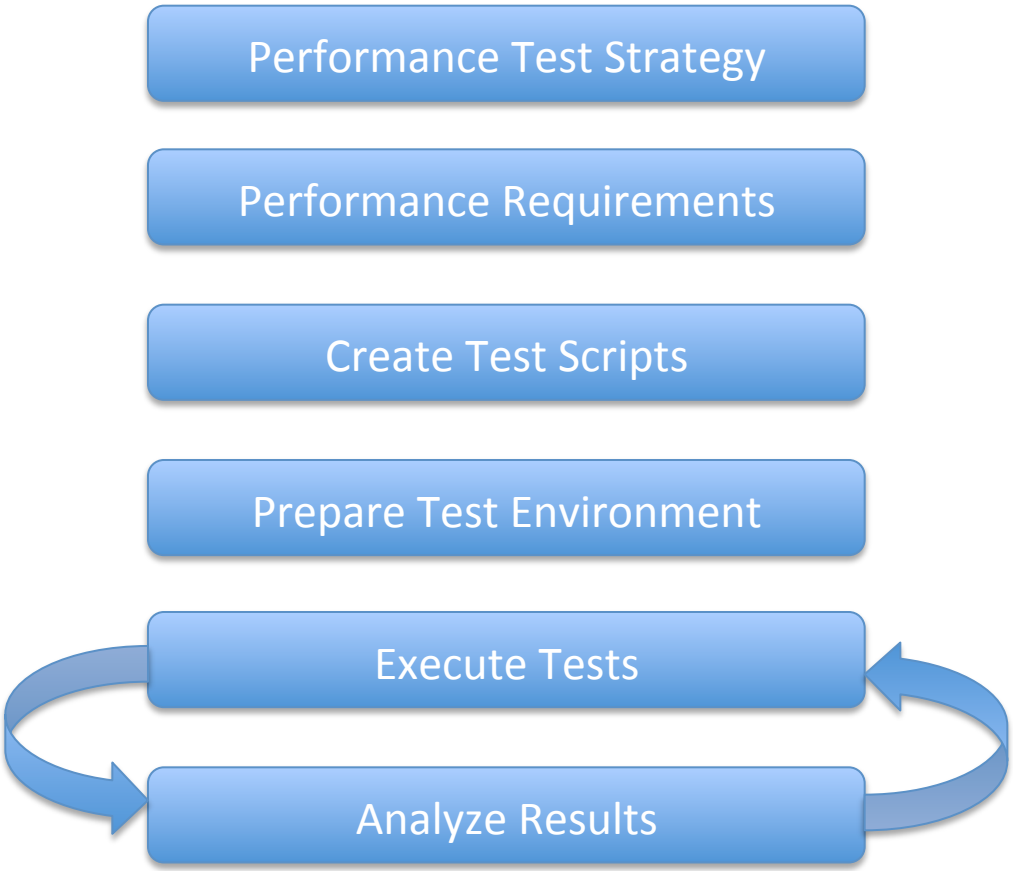
Agenda

- **Jmeter Basics**
  - Record & Playback

- **Dinner**

- **Jmeter Advanced**
  - Parameterizing, Assertions
  - Debugging
  - Analysing results

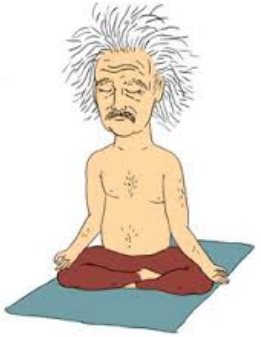
Performance Test Workflow



Some definitions

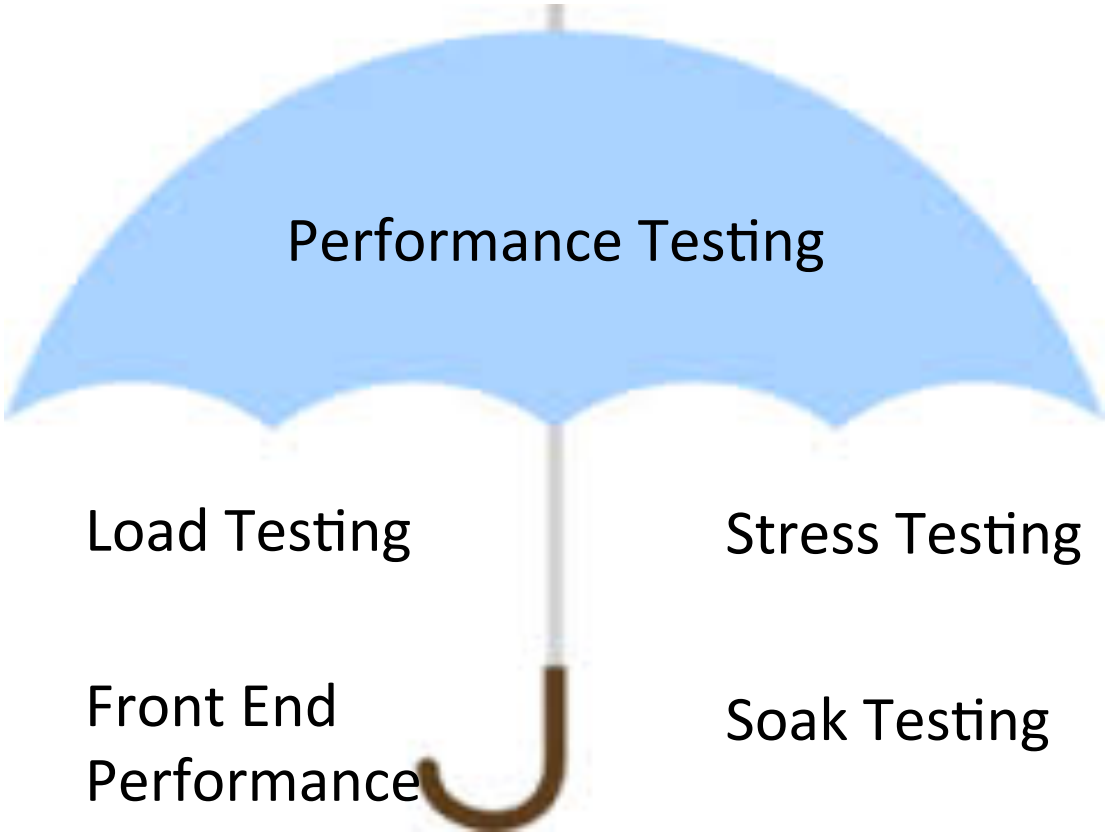
- Load Test
  - Can the app handle expected load?

- Soak Test
  - Is app stable handling load for a long time?

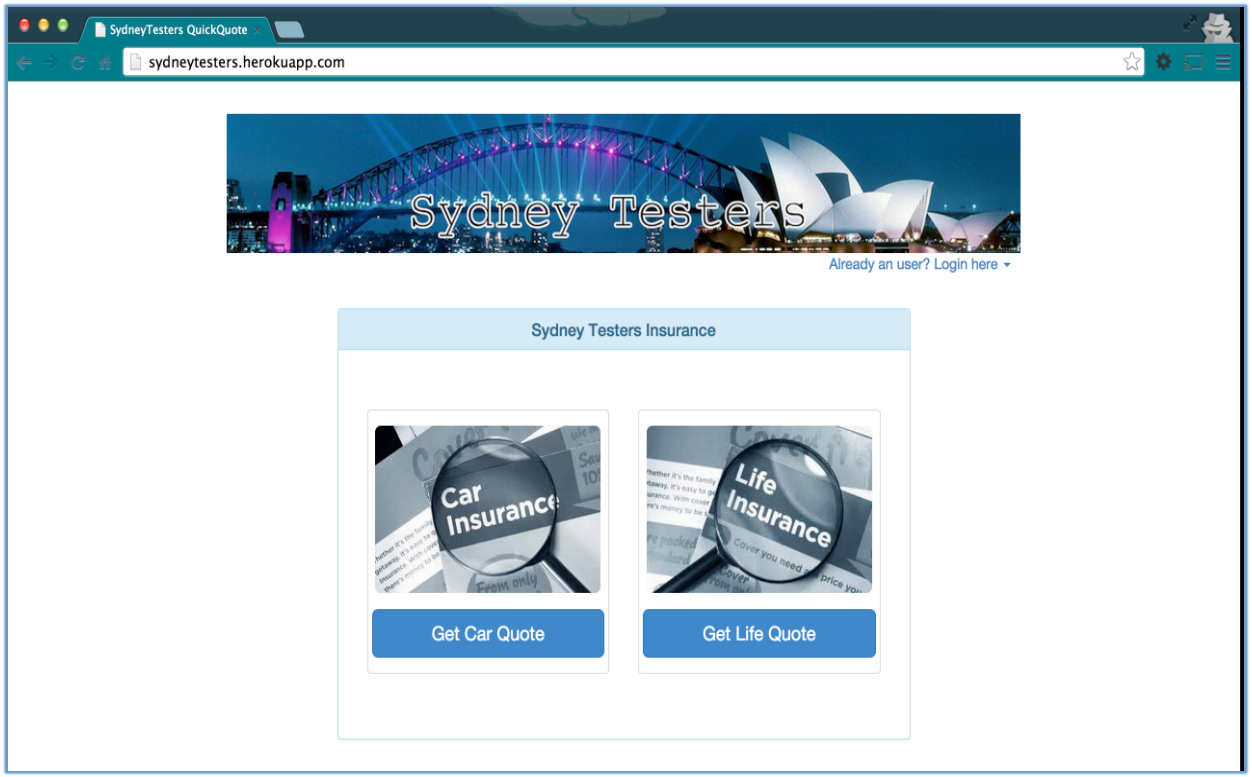


- Stress Test
  - At what point does the app break?

Some definitions

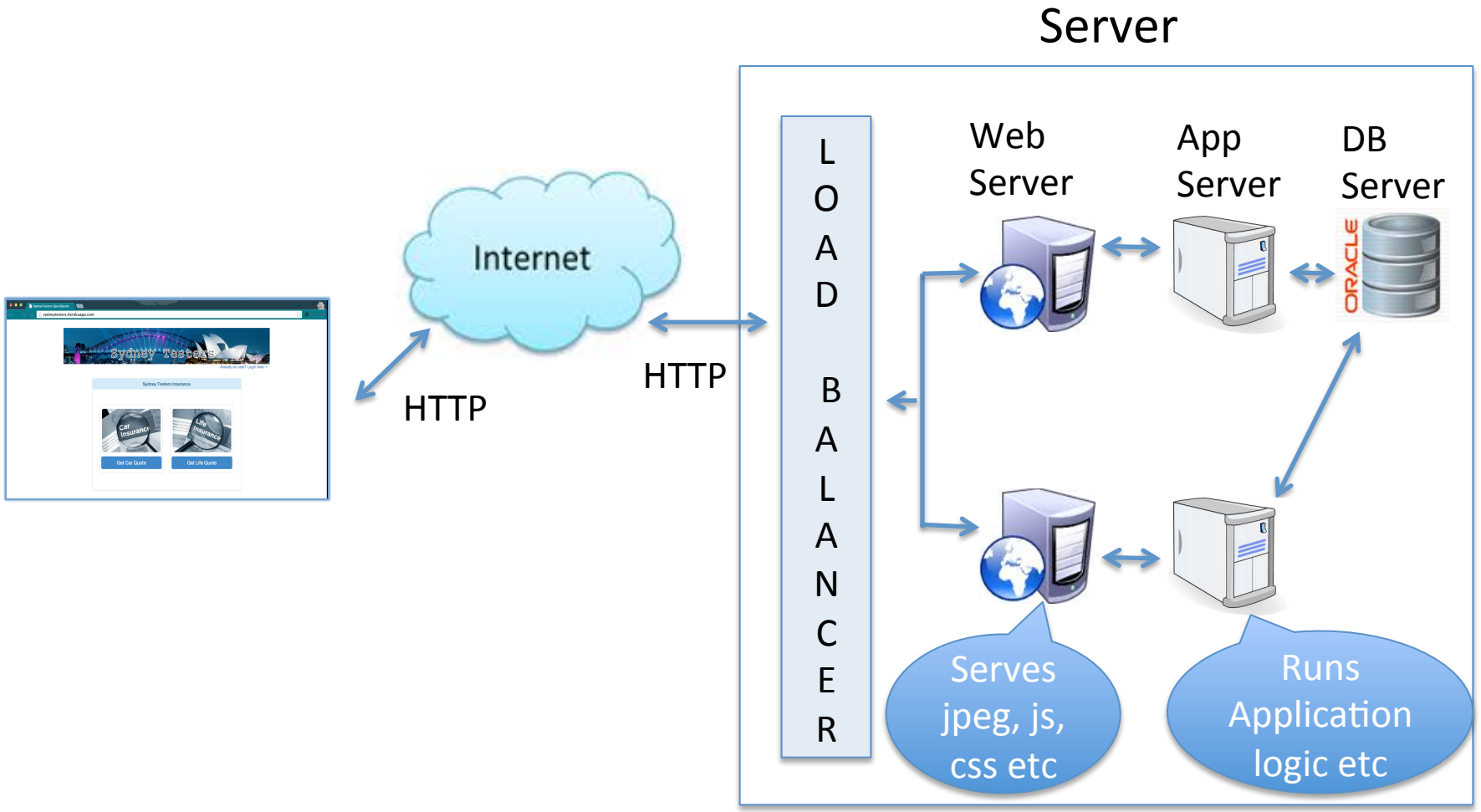


App Under Test

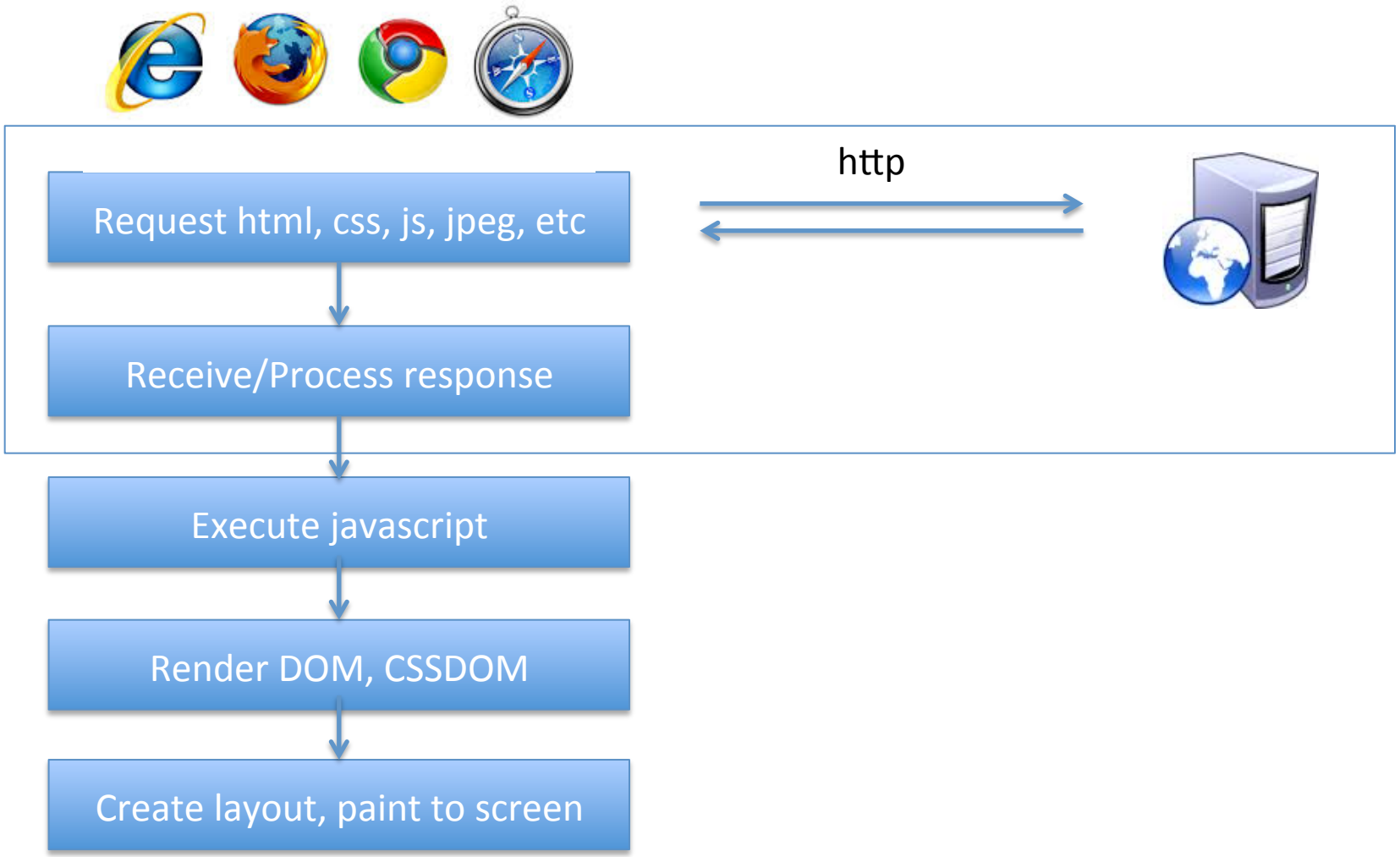


http://sydneytesters.herokuapp.com/

Anatomy of a Web Request



How Loadtest tools work



Performance Requirements

Throughput

Load Requirements	
Concurrent Users (Average)	20
Concurrent Users (Peak)	50
% of users performing a quote	90%
% of users buying a policy	10%

Performance Requirements

Response times

Transaction	Acceptable Time
View Home Page	< 2s
Perform a Insurance Quote	< 30s
Buy an Insurance Policy	< 30s

Start up JMeter

Unzip jmeter zip file

cd apache-jmeter-2.11

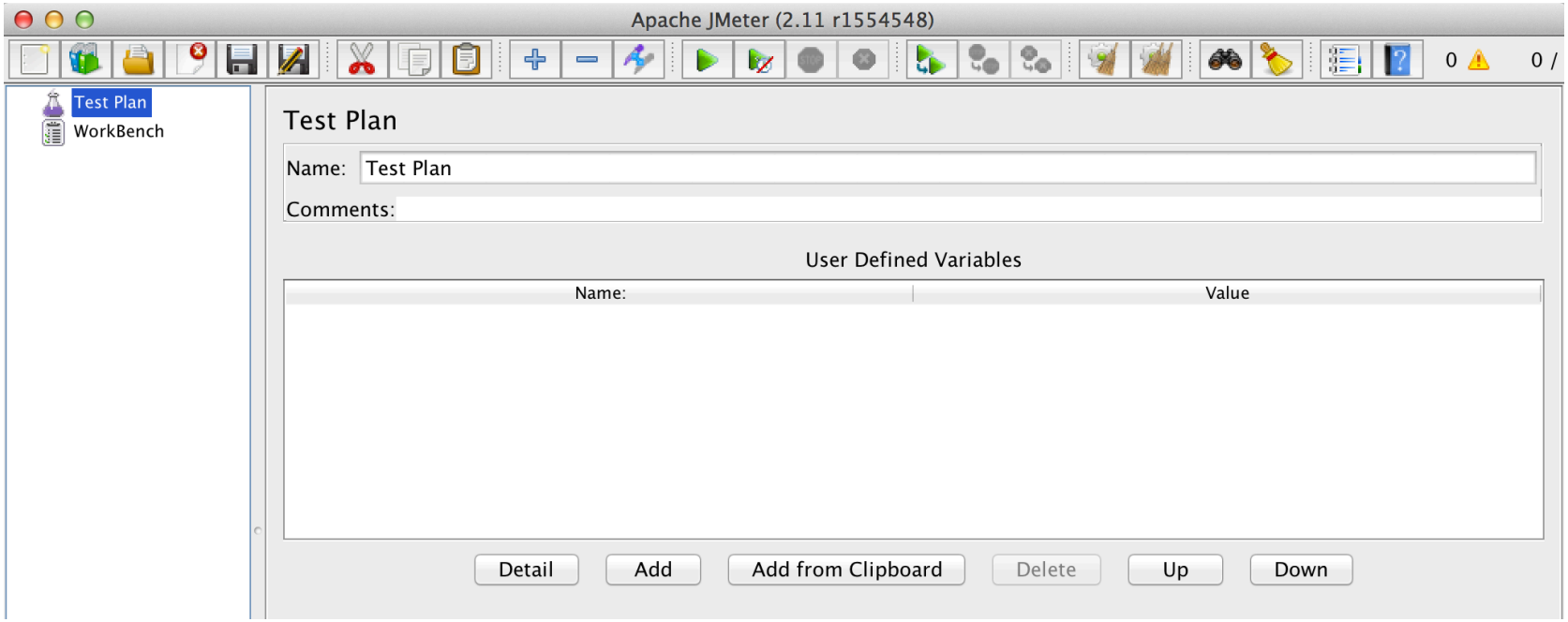
cd bin

chmod +x jmeter.sh

./jmeter.sh (Linux/OSX)

jmeter.bat (Windows)

Jmeter GUI


Apache JMeter (2.11 r1554548)
0
0 /
Test Plan
WorkBench
Test Plan
Name:
Test Plan
Comments:
User Defined Variables
Name:
Value
Detail
Add
Add from Clipboard
Delete
Up
Down

Components & Terminology

- Test Plan

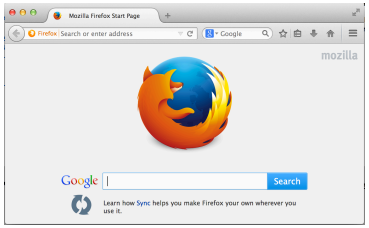
- Workbench

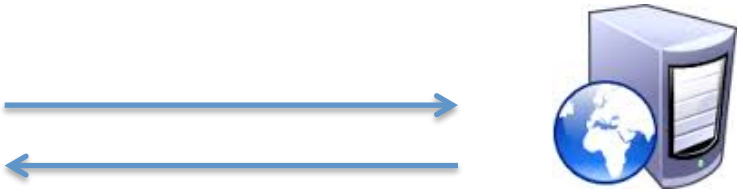
Components & Terminology

- Thread

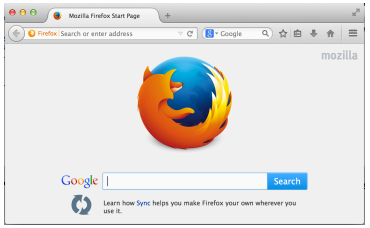
- Thread Group

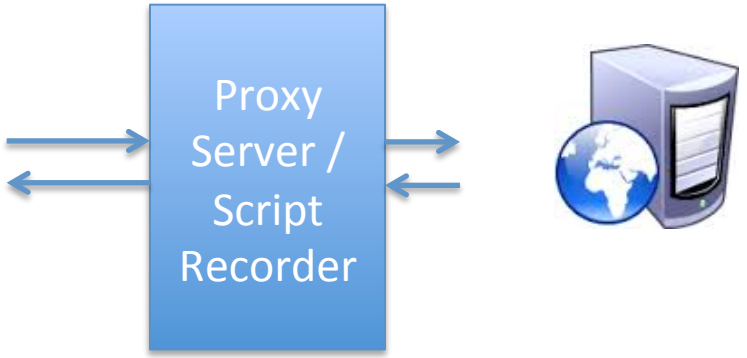
Recording web requests





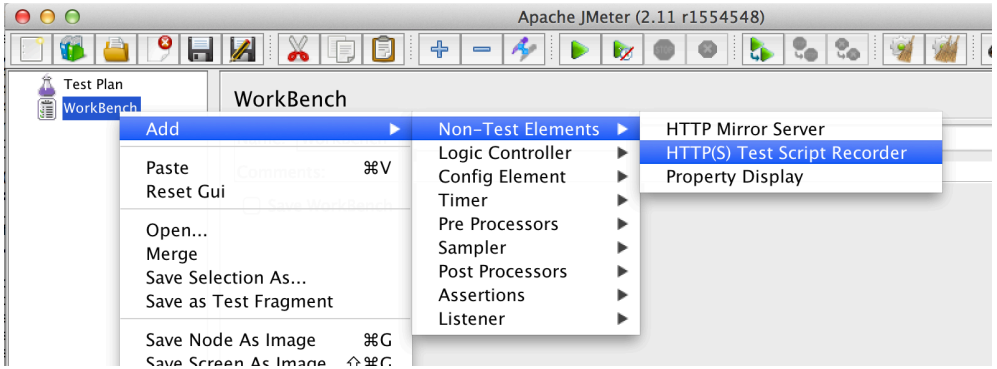
Recording web requests



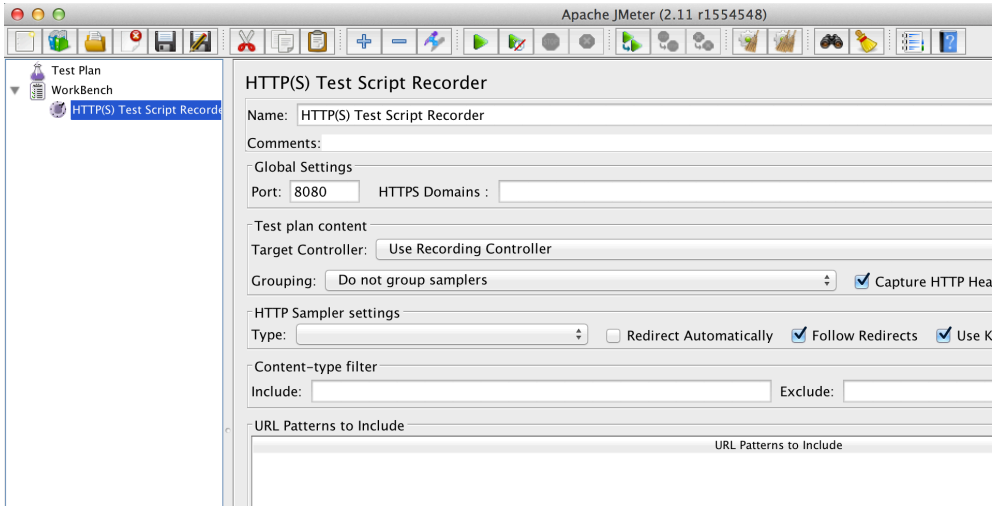


Recording web requests

Step 1: Start Jmeter
Test Script Recorder



Port: **8080**

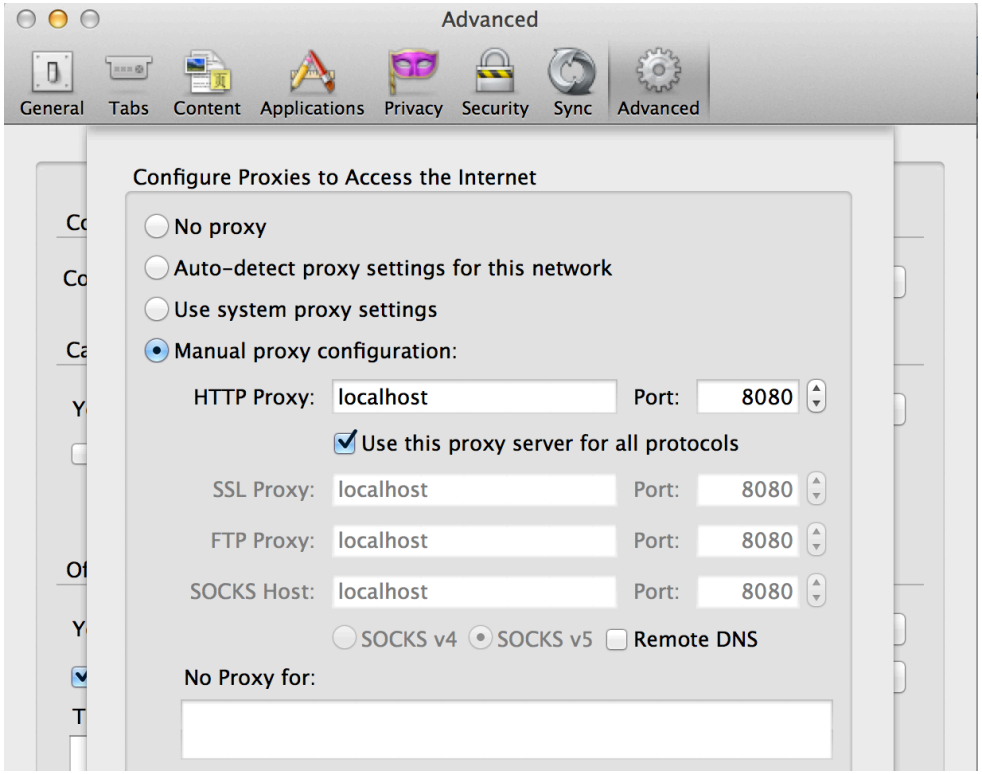


Recording web requests

Step 2: Configure Browser to use Jmeter Proxy Server

HTTP Proxy: **localhost**

Port: **8080**



Preferences -> Advanced -> Network -> Settings

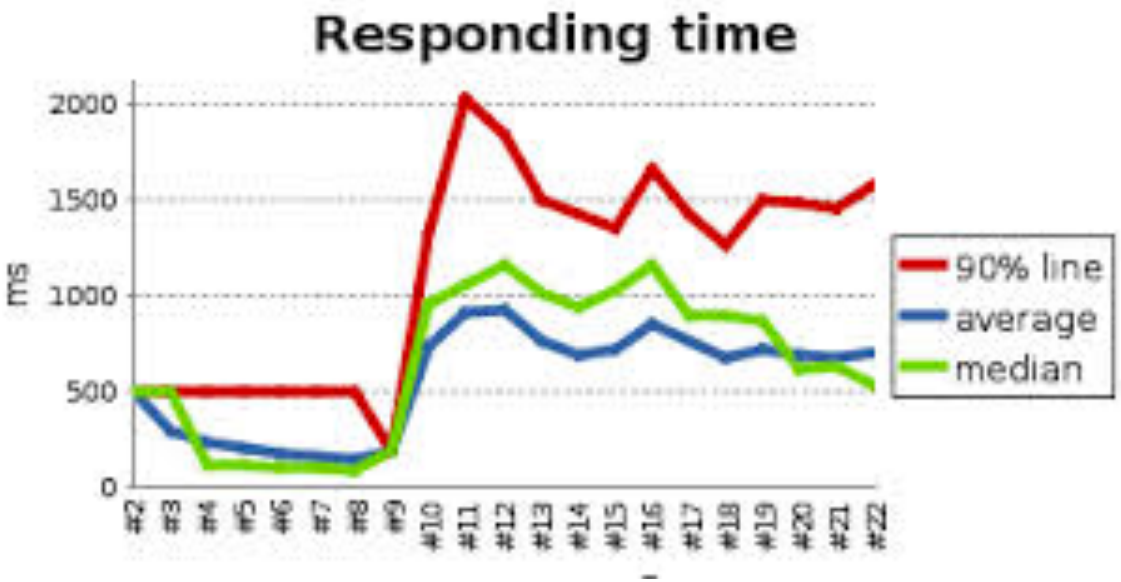
Components & Terminology

- Samplers

Recording Filters

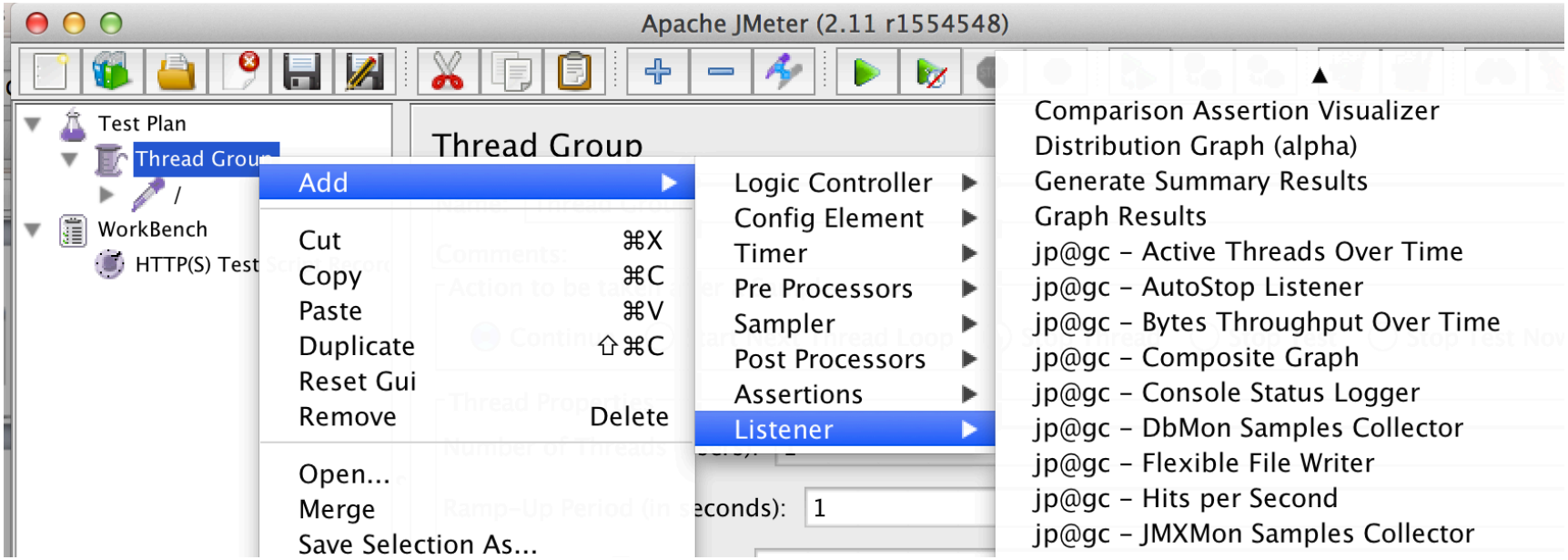
Components & Terminology

- Listeners

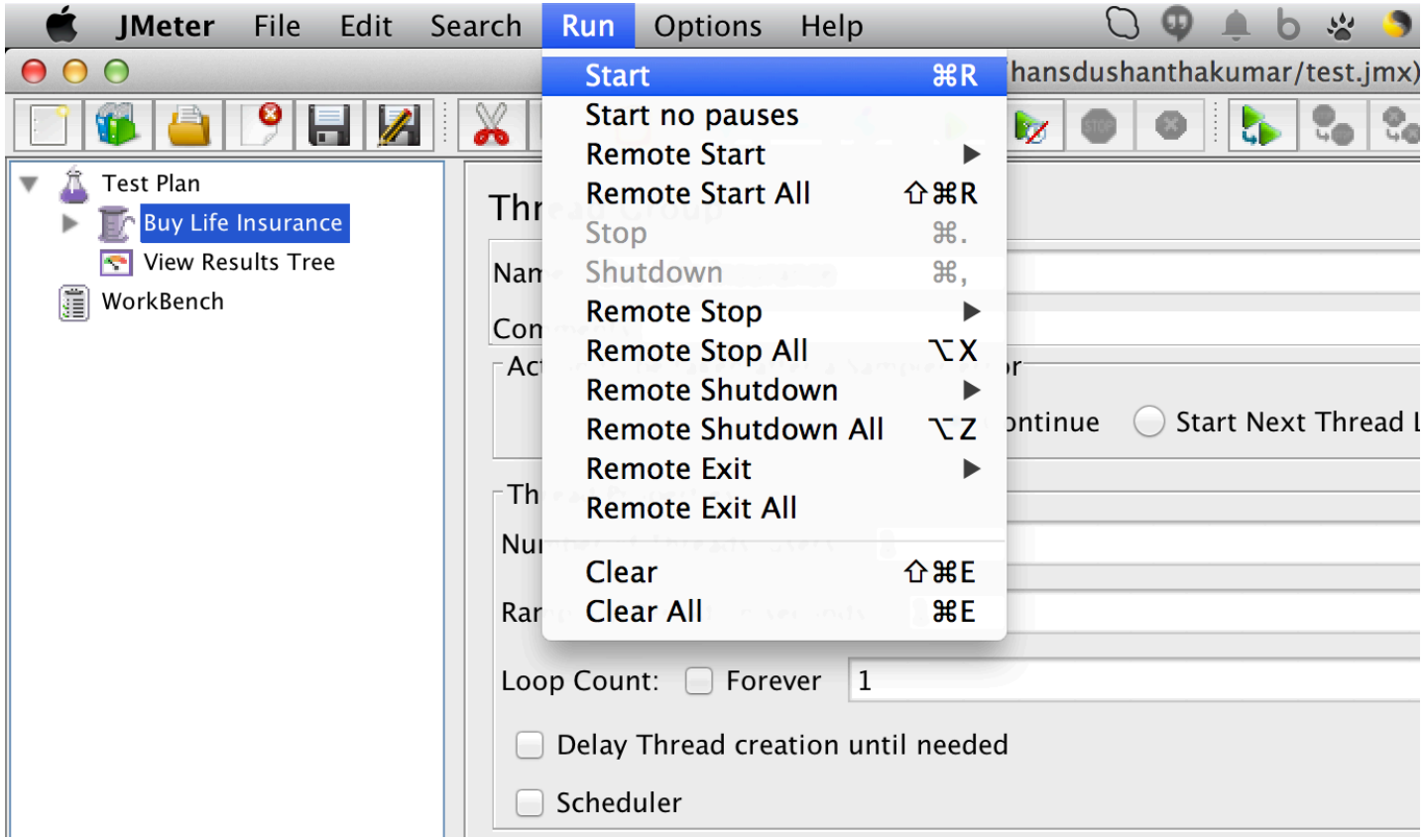


Seeing Requests & Responses

Add -> Listeners -> View Results Tree



Playback

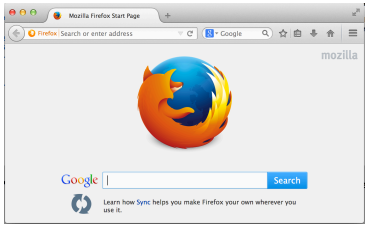

JMeter
File
Edit
Search
Run
Options
Help
Start ⌘R
Start no pauses
Remote Start
Remote Start All ⇧⌘R
Stop ⌘.
Shutdown ⌘,
Remote Stop
Remote Stop All ⌥X
Remote Shutdown
Remote Shutdown All ⌥Z
Remote Exit
Remote Exit All
Clear ⇧⌘E
Clear All ⌘E
Test Plan
Buy Life Insurance
View Results Tree
WorkBench
Loop Count:
Forever
1
Delay Thread creation until needed
Scheduler

HTTP Basics

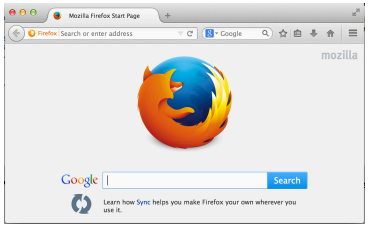
- HTTP Request
  - Verb: GET / POST etc
  - Request Header
  - Request Body

- HTTP Reponse
  - Header
    - Response Code (200, 303, 500, 404 etc)
    - Cookie etc
  - Body

Cookies



Set-Cookie: key=value



Cookie: key=value

Add -> Config Element -> HTTP Cookie Manager

Assertions

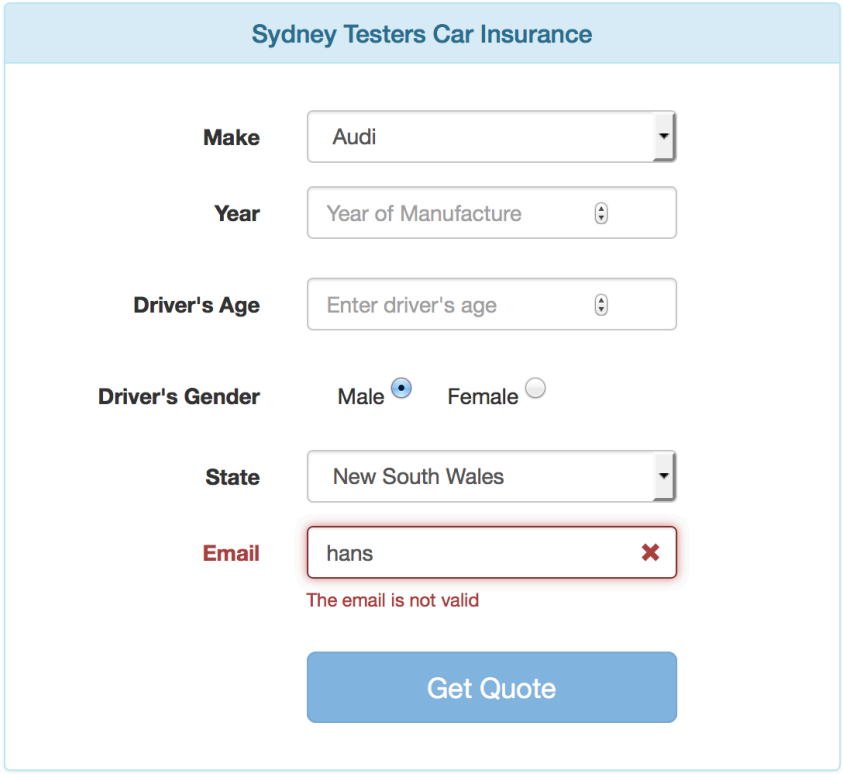
Add -> Assertions -> **Response Assertion**

Patterns to test: **Sydney Testers Insurance**

Think Time

Add -> Timer -> **Gaussian Random Timer**
Deviation: **1000**
Constant Delay Offset: eg: **5000**

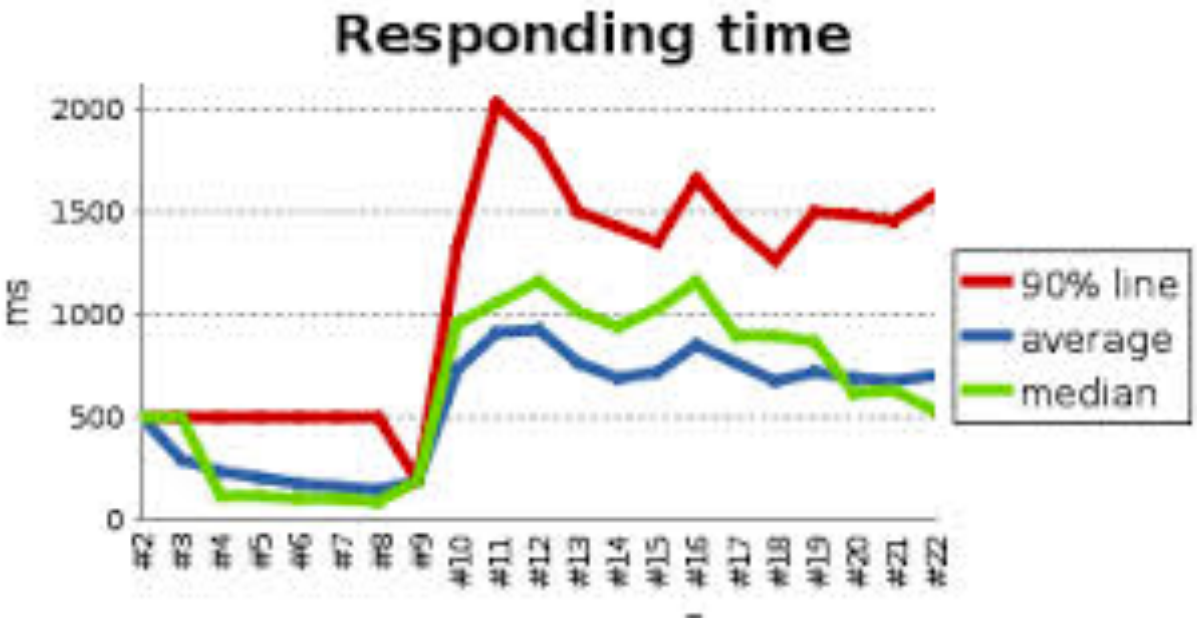
AJAX requests



http://..../checkemail

{"valid":false}

Monitoring: Listeners



1. **View Results Tree**
2. **Aggregate Report**

Configuring Loadtest

- Concurrent Users

ThreadGroup: Number of threads: **20**

Configuring Loadtest

- Ramp Up

ThreadGroup: Ramp-up Period: **20**

Putting it all together: Running a Loadtest

Analysing Results

Are Results valid?

- Errors?
- Throughput?
- Realistic Response times?
- Server Logs

Analysing Results

Useful Metrics:

- Response Times (Avg, 90th Percentile)
- Throughput achieved
- CPU (VM)
- Memory (VM)

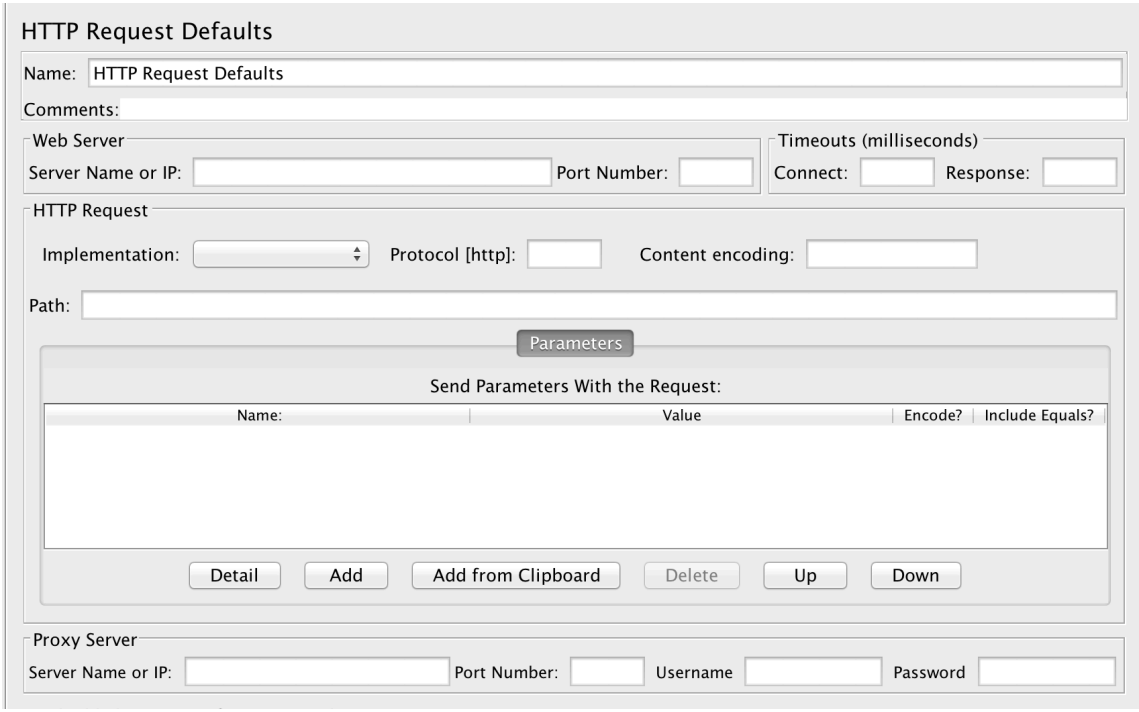
Refactoring

Problems with the script:

- Hard-coded URLs
- Same test data used always
- Metrics are not business-level

HTTP Request Defaults

Config Element -> HTTP Request Defaults

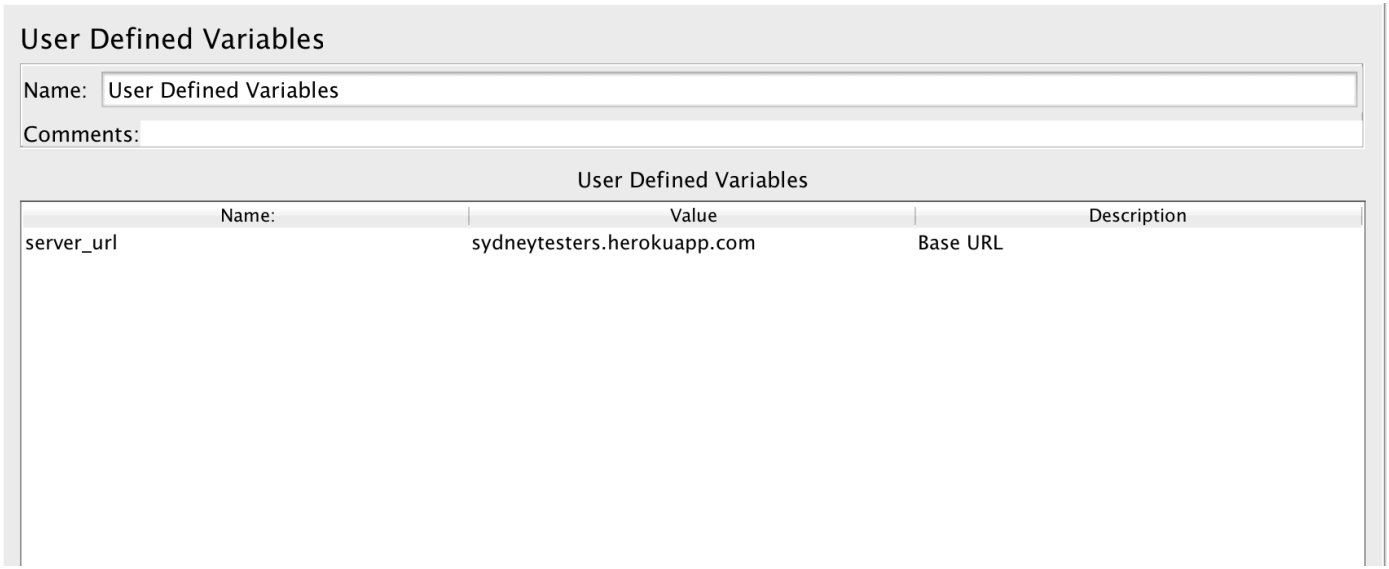


Server Name: **sydneytesters.herokuapp.com**

Port: **80** Protocol: **http**

User Defined Parameters

Config Element -> User Defined Variables



User Defined Variables:

SERVER_URL : **sydneytesters.herokuapp.com**

${SERVER_URL}

Test Data

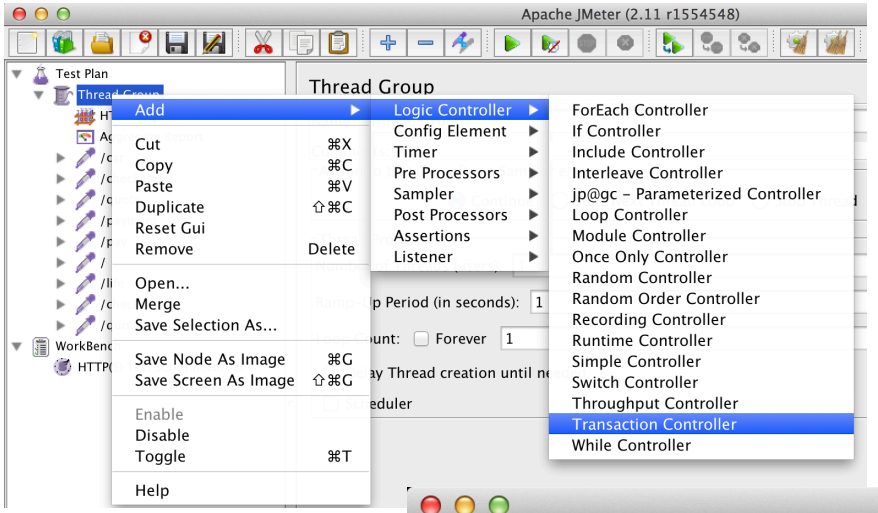
Config Element -> CSV Data Set Config

Filename: **testdata.csv**
Variable Names: **AGE, GENDER**

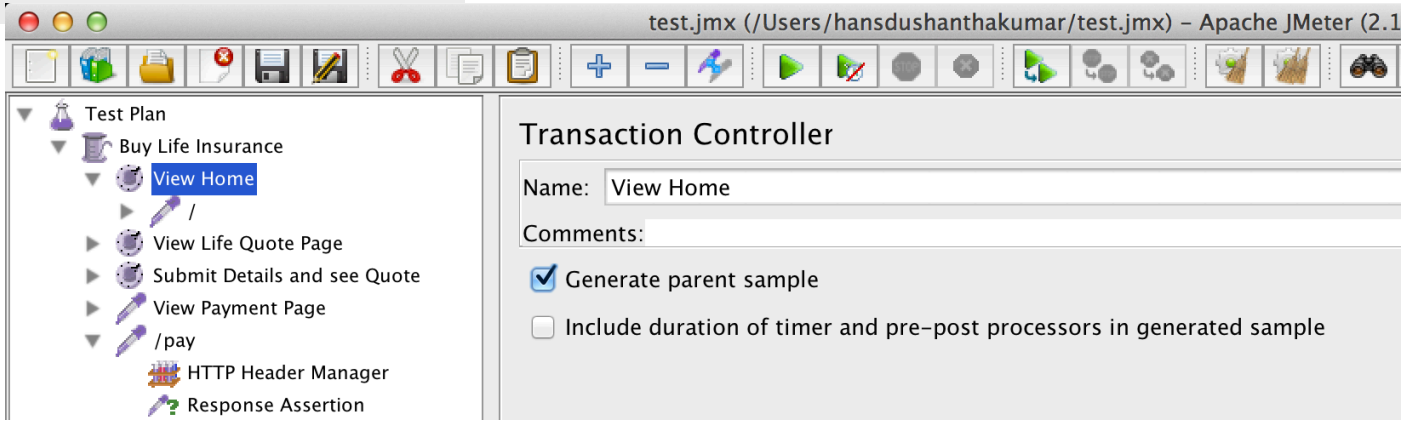
Grouping requests into Business Transactions:

View Home Page	**GET "/"**
View Life Quote Page	**GET "/life"**
Get Life Quote	**GET "/checkemail"** **POST "/quote"**
Purchase Insurance	**GET "/payment"** **POST "/pay"**

Grouping requests into Business Transactions



Logic Controller -> Transaction Controller

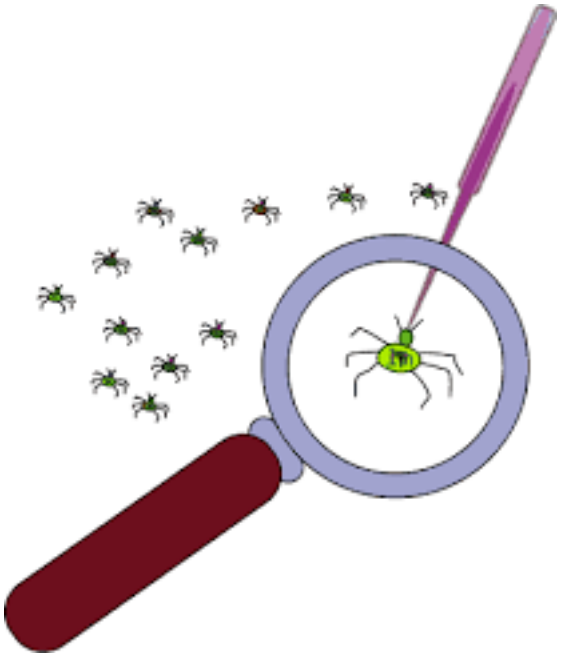


Beanshell scripting

www.beanshell.org

Debugging

- Results Tree Listener
- Debug sampler

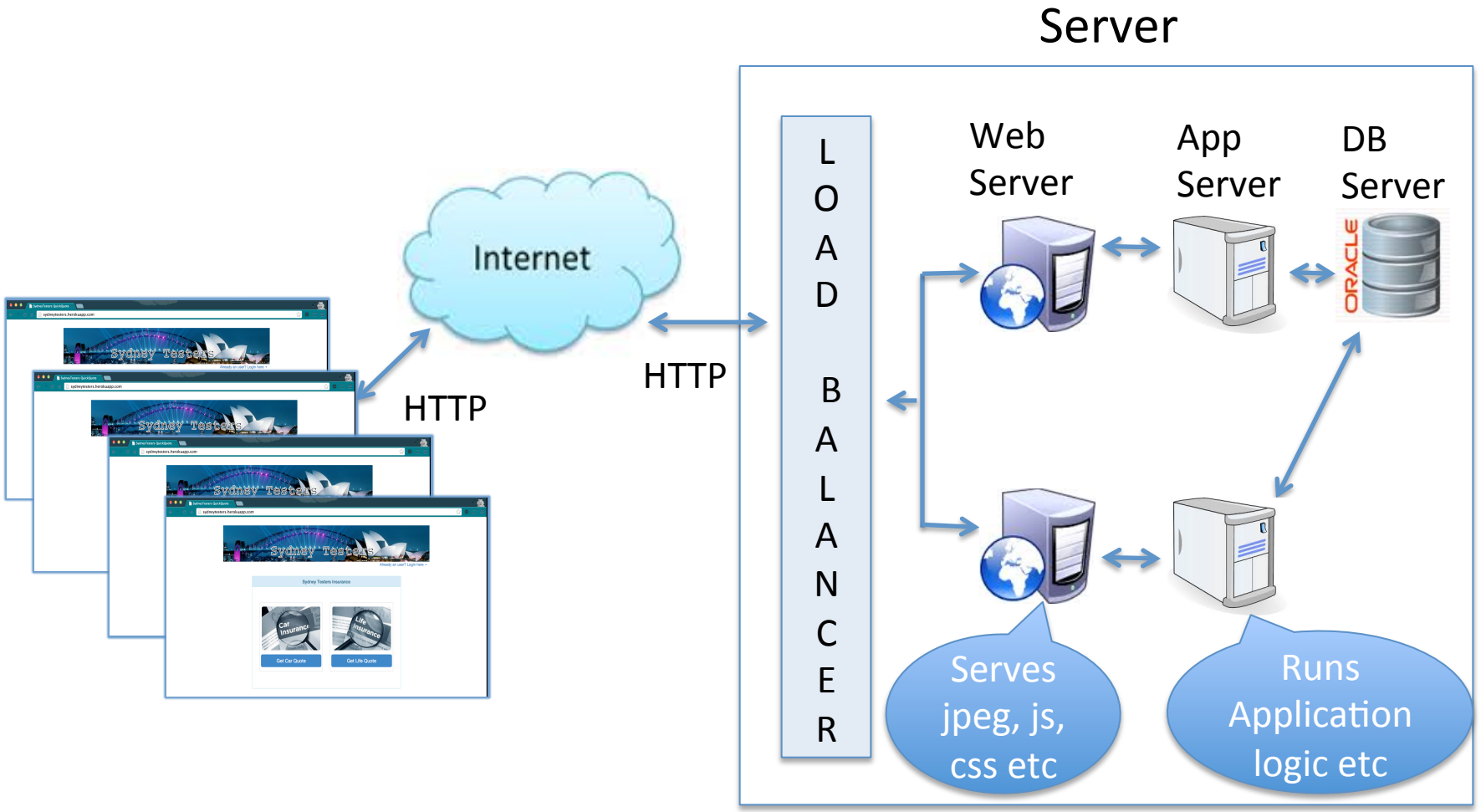


Simulating different browsers/ devices

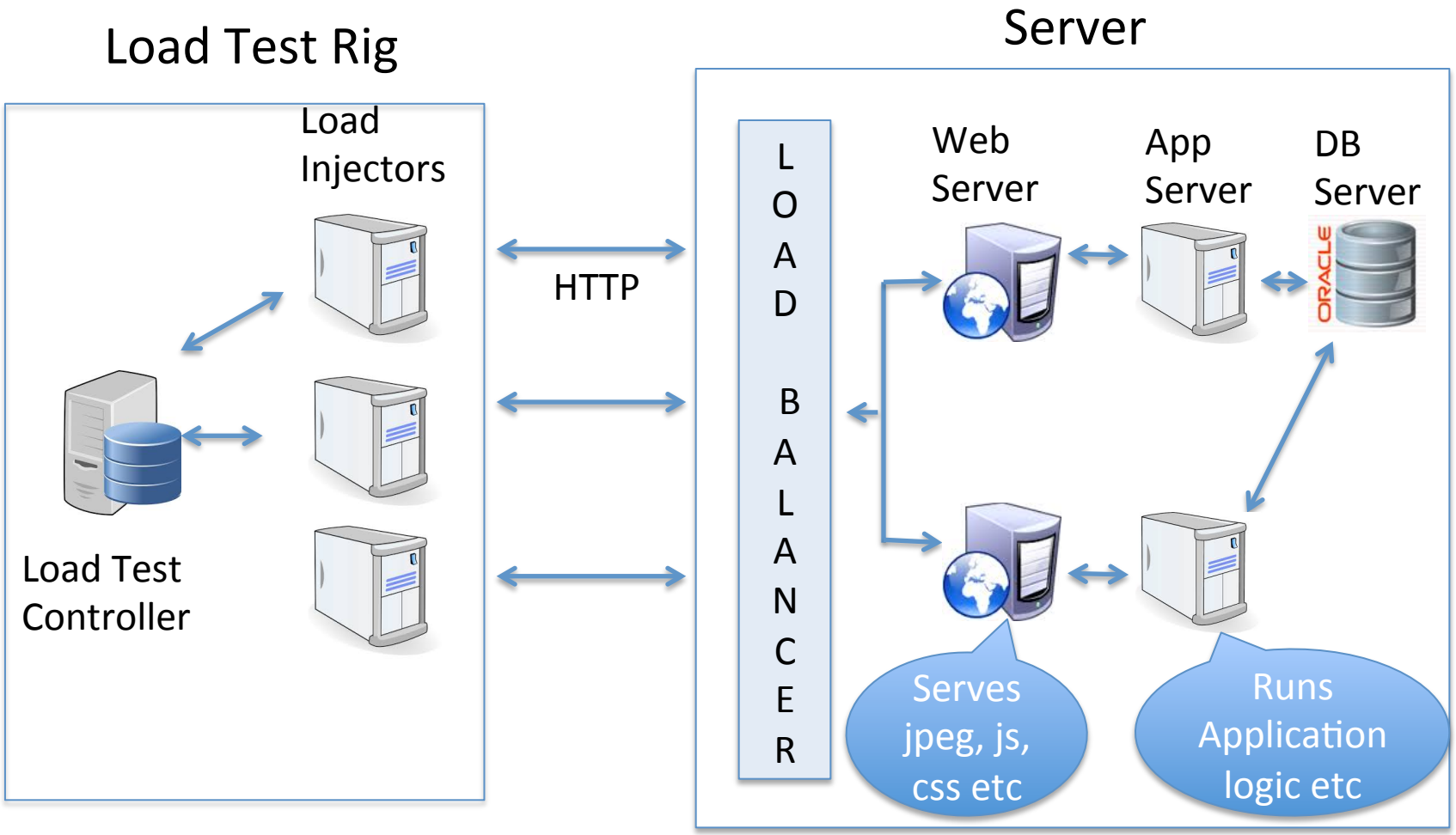
Recap

- Record a jmeter script
- Customise it
- Run load test
- Monitor results

Anatomy of a Web Request



Load Test Environment



Certifying an Environment

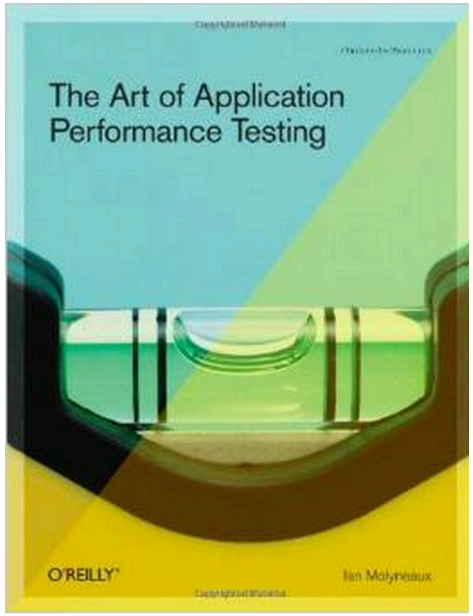
- Repeatable results
- Stable
- Similar results as prod
- Low CPU/Memory usage on load generators
- Ensure results are valid etc

Limitations of JMeter

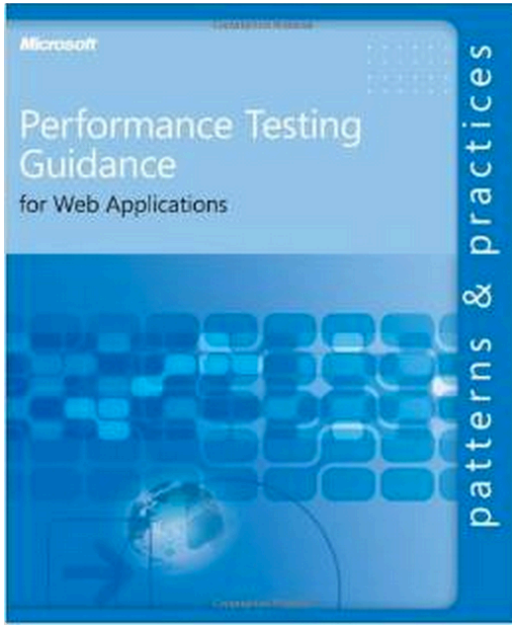
- Monitoring & Analysing results
- Version Control
- Front End perf
- Resource consumption
- Managing load generation agents

Other Load Test tools

Further Reading

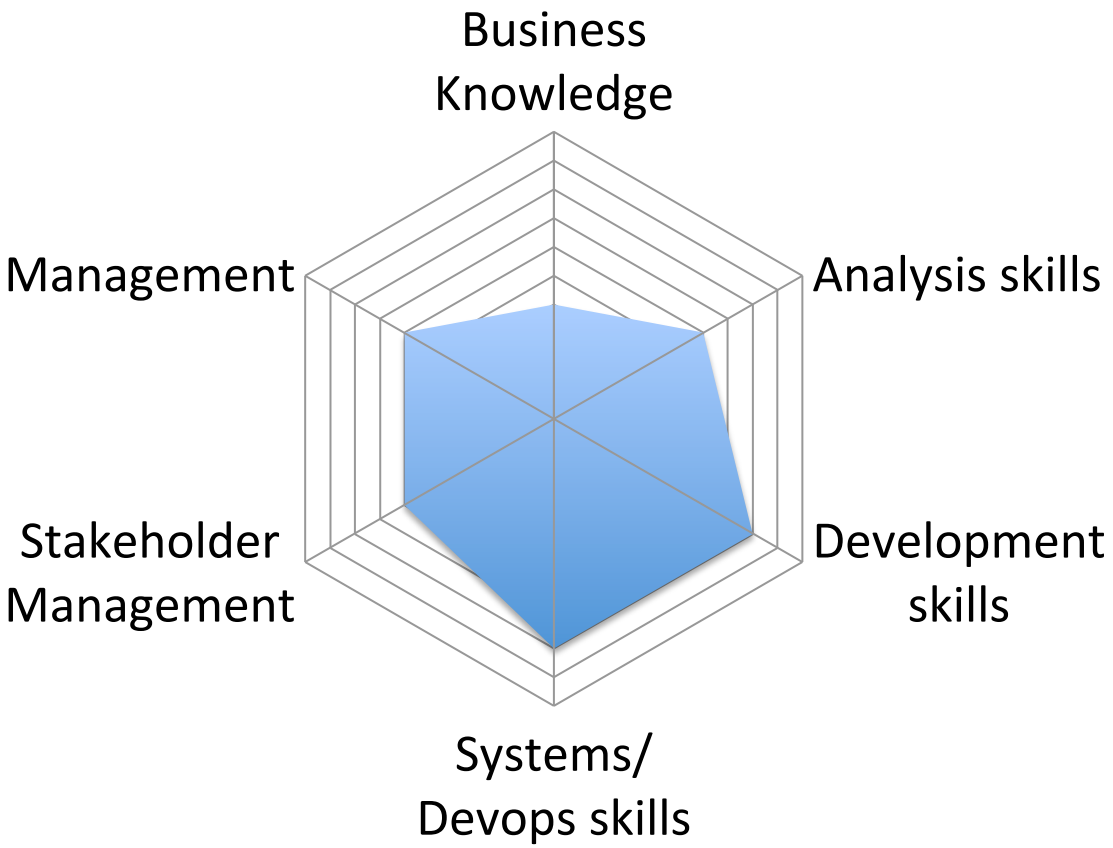


Ian Molyneaux



Microsoft

Profile of a Performance Tester



Questions & Comments?