# Introduction to Performance Engineering and Performance Center

Northeastern University
INFO6255

# Different QA Jobs

1. **Manual Tester** – All tests are through the GUI or the application interface.
2. **Automation Tester** – Writes scripts for Automation testing using one of many tools in the market, e.g. Selenium, HP UFT, or other scripting languages.
   - Also may use different languages to write scripts to test the interconnectivity of an application, such as (VB Scripting, Python, and etc.)
3. **Data/Data Base Tester** – All testing is done through writing and executing SQL statements, Stored Procedures or running various scripts for testing.
4. **Performance & Load Tester** – Writing Performance, Load & Stress testing scripts using one of many tools that are in the market. E.g. HP Performance Center, JMeter, and etc.
5. **Security/Cybersecurity Tester** – Performs application security testing, or Cybersecurity testing.

# Performance Engineering

performance engineering解决的问题

Can I deploy my application to other regions?

How much hardware do I need?

Why is my application slow sometimes?

*Why does my application run intermittently?*

*How many more users can I support?*

**Can my application support X # of users and maintain a X seconds in response time?**

If I patch the Operating System, am I risking performance?

My application is fast…but I have to reboot it periodically!

***What is the variance of my application's performance?***

# Performance Engineering

How can we answer all those

questions?

# What is Performance Engineering?

"**Performance engineering** within systems engineering, encompasses the set of **roles, skills, activities, practices**, **tools**, **and deliverables** applied at every phase of the SDLC which ensures that a solution will be designed, implemented, and operationally supported to meet the non-functional performance requirements defined for the solution."

Source: Wikipedia

# Performance Engineering

性能测试的目的

**Questions**

What do you want to learn from your system?

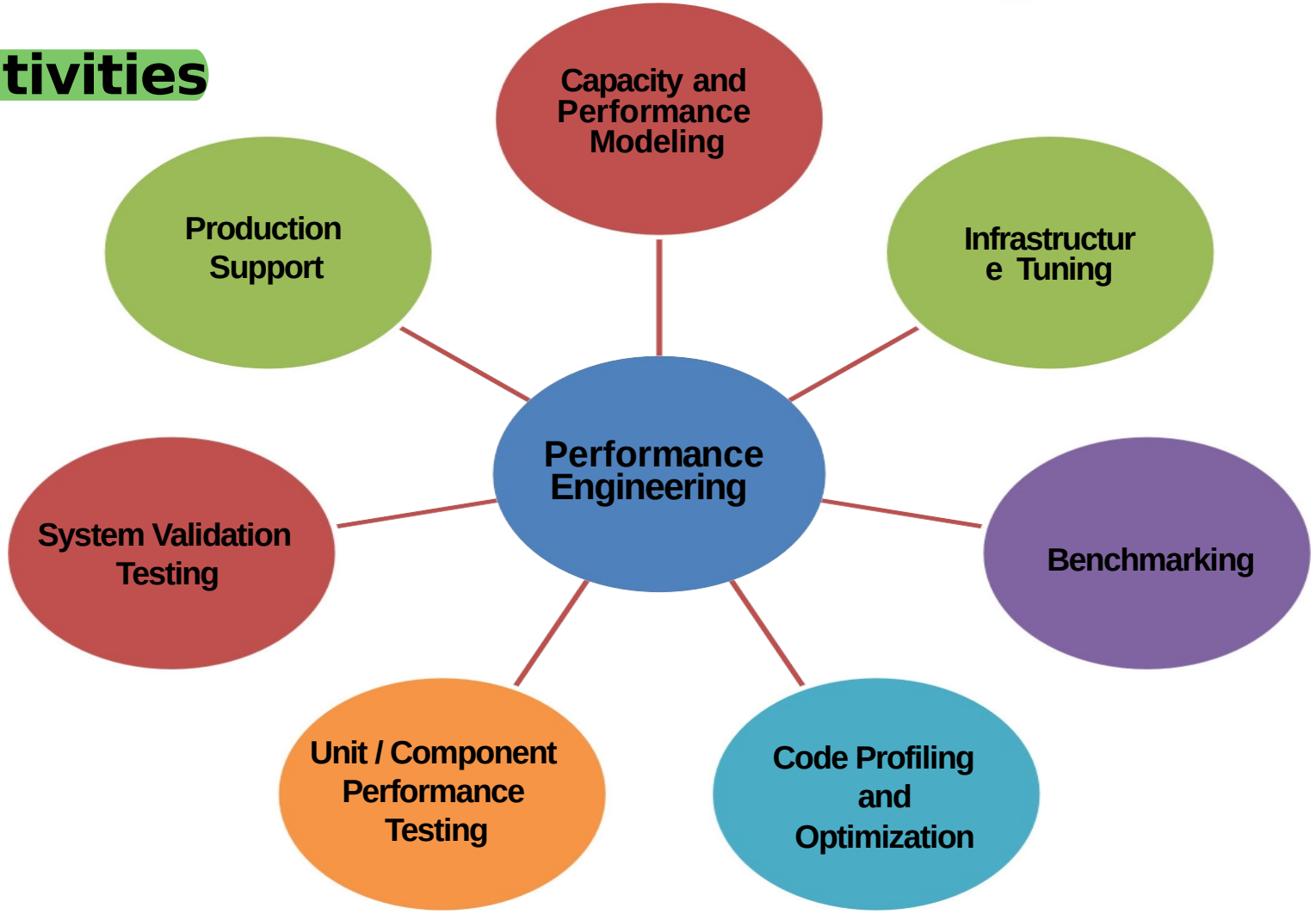
- How fast is it?
- How does it scale?
- Where does it break?

What do you want to prove about your system?

- **Response** times.
- The amount of **Throughput.**
- Can my system **Scale Up** to a specified number of users?

# Performance Engineering

**Activities**



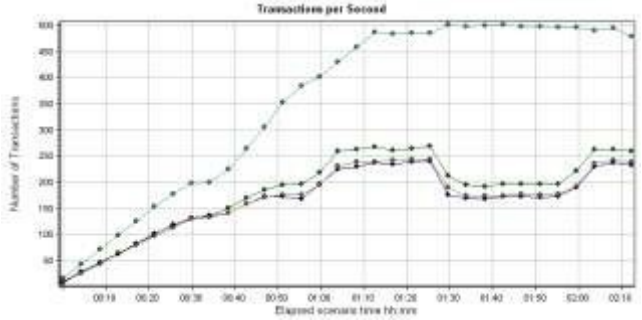
# Performance Engineering

## KPIs

**Key Performance Indicators**



- **Responsiveness**
- **Throughput**
- **Stability**
- **Scalability**



# Why Performance Testing?

## Case Studies

- **Lloyds Bank's ATMs** unable to do transactions for 3 hrs.
- **RBS paid $175** million to its customer for IT failures apart from fines.
- **HealthCare.gov** couldn't handle the traffic when rolled out.
- **Facebook** unavailable for 15 mins because of infrastructure configuration issues.
- LinkedIn, Apple iTunes Store, Mac App Store, and App Store cannot go down.

# Why Performance Testing?

## More Recent Software Performance Failure

**Barclays Bank:** Early this year, **customers of major banks like Barclays, Natwest and Santander could not access their mobile apps for several hours**. The sites were experiencing major traffic as it was payday, causing the system to crash.

人们会发泄自己的不满，让企业陷入困境。
lost users
Lost revenue

**Dissatisfied customers took to social media like Twitter to vent their grievances, further putting the banks in a bad light**.

# System Performance Facts and Stats:

- **73%** of mobile internet users say that they've encountered a website that was **too slow** to load.
- **51%** say that they've encountered a website that **crashed, froze, or received an error**.
- **38%** of say that they've encountered a website that **wasn't available**.
- **47%** of consumers expect a web page to load in **2 seconds or less**.
- **40%** of users **abandon** a website that takes more than **3 seconds** to load.

**If an e-commerce site is making $100,000 per day, a 1 second page delay could potentially cost you $2.5 million in lost sales every year.**

# Performance Engineering

Types of Tests

- **Performance testing -** Determines or **validates the speed, scalability, and/or stability** characteristics of the system or application under test. It is intended to determine the
  - **Responsiveness**
  - **Throughput**
  - **Reliability**
  - **Scalability**

  of a system under a given workload

- **Load Testing** is asserting how the architecture performs under load with a view to monitoring the response times for key transactions. **(Anticipated volumes)**

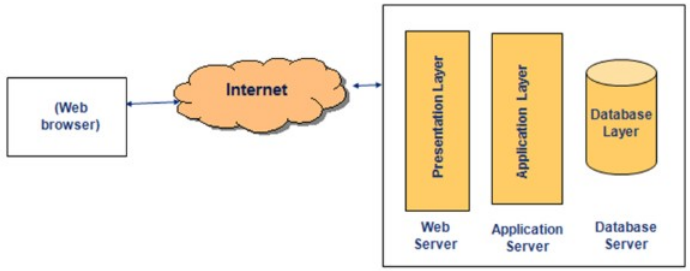
# Performance Engineering

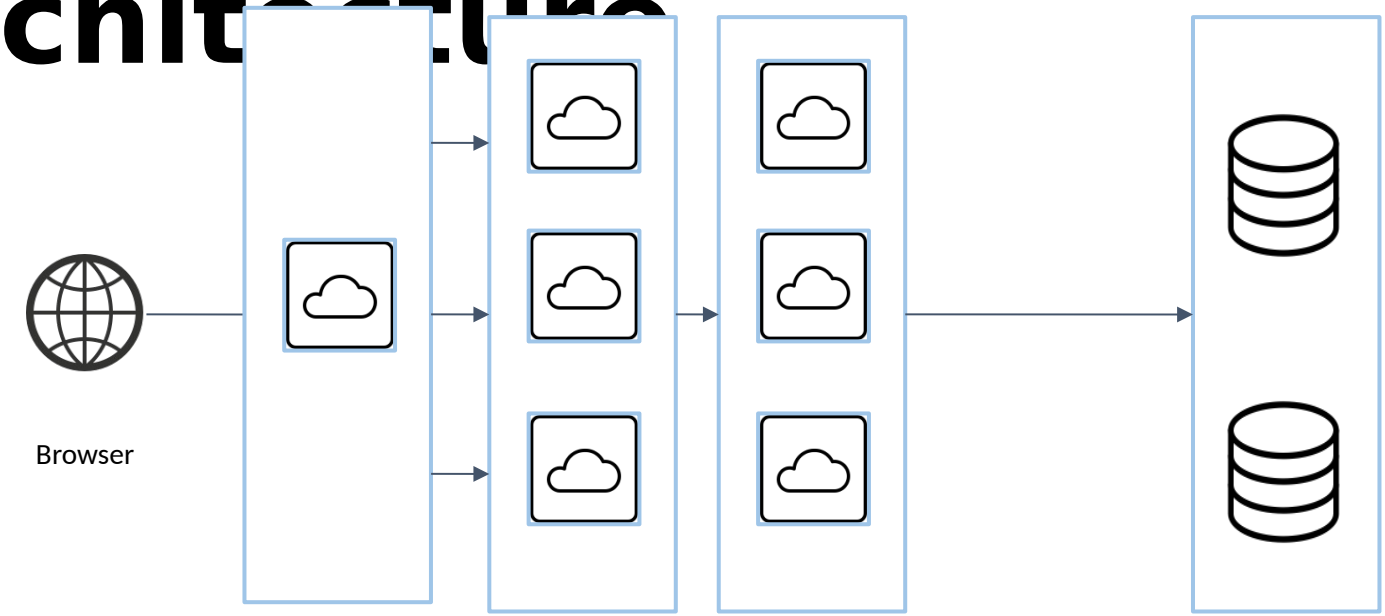
## Types of Tests…

- **Stress Testing** is asserting what the upper bounds are for the scalability of the architecture, understanding how it reacts when **stressed**. (Beyond Anticipated volumes).

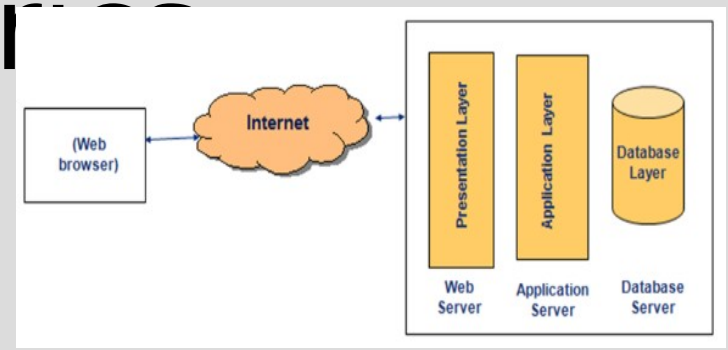
- **Capacity Testing** is to determine how many users and/or transactions a given system will support and still meet the performance goals
  - **Capacity** Testing measures the products' maximum **capacity.**

# Typical Web Application Architecture



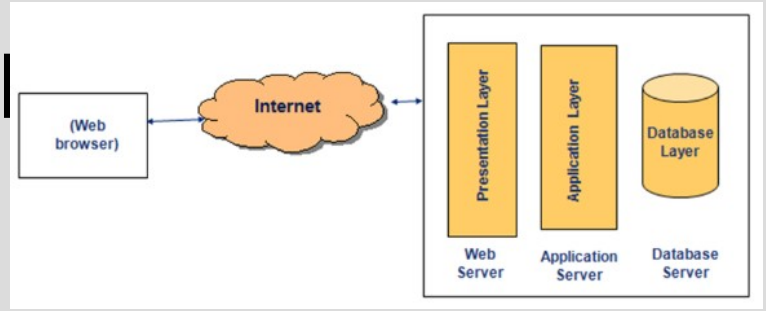



# Web Server Metrics



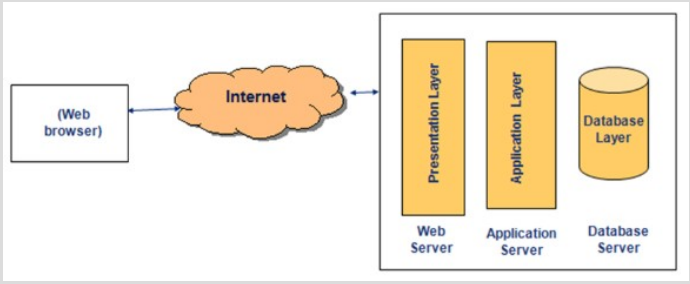

- **Busy and Idle Threads**
  - Worker threads per web server
  - Number of web servers
  - Long running **threads** because of application performance hotspots
- **Throughput**
  - Number of **transactions / minute**
  - When do we need to scale out and add more web servers?
- **Bandwidth Requirements**
  - Network bottlenecks
  - Average page size

# App Server Metr




- **Load Distribution**
  - Transactions are handled by each JVM engine
  - Load balancing
  - Number of **Application Servers** needed to handle the load
- **CPU Hotspots**
  - CPU requirements
  - Parts of programming causing **high CPU** requirement
  - CPU power

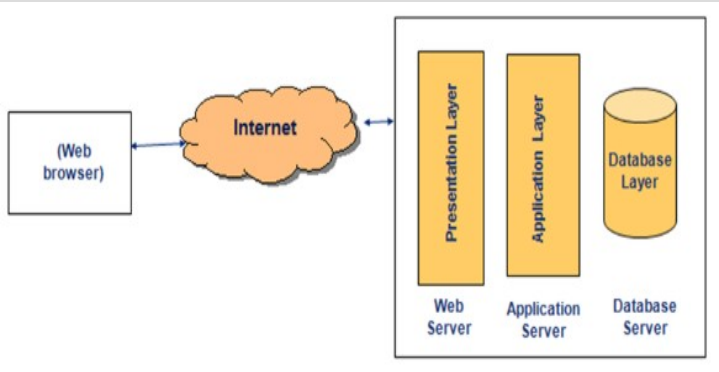
# More App Server Metrics...




- **Worker Threads**
  - Number of **worker threads** configured
  - Threads blocked by other modules
- **Memory**
  - **Bad memory** patterns
  - Memory leaks
  - Impact of **Garbage Collection** on CPU and Transaction Throughput

# Host Health Metrics

- **CPU, Memory, Disk, I/O**
  - Physical or virtual resources hea
  - Log files
  - Network interfaces
- **Key Processes**
  - Processes run on the host
  - Resource hungry processes
  - Management of processes



# 1. Assess Production Readiness

- Predict or estimate the performance characteristics of an application in production

要考虑两个问题

  - These predictions are also valuable to the stakeholders who make decisions about:
    - Is the application is ready for release or capable of handling future growth
    - it requires a performance improvement/hardware upgrade prior to release?

- **It Provides data for:**
  - Indicating the likelihood of **user dissatisfaction** with the performance characteristics of the system.
  - To aid in the prediction of **revenue losses or damaged brand credibility** due to scalability or stability issues.

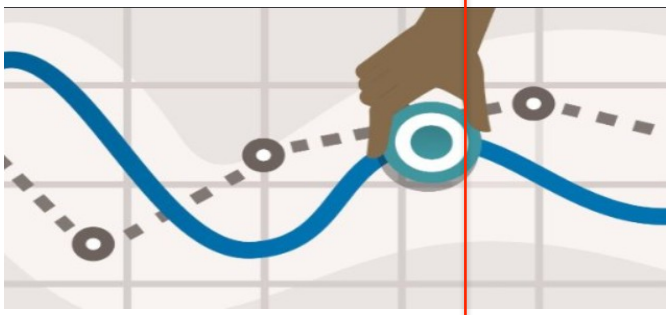
# 2. Assess Infrastructure Requirement

- Evaluating the adequacy of current capacity.
- Determining the acceptability of stability.
- Determining the capacity of the **application's infrastructure**, as well as determining the future resources required to deliver acceptable application performance.
- Comparing **different system configurations** to determine which works best for both the application and the business.
- Verifying that the application exhibits the **desired performance characteristics,** within budgeted resource utilization constraints.

# What is a Baseline?

**A baseline is a set of tests that capture the Performance Metrics for the purpose of evaluating the effectiveness of subsequent performance-improving changes to a system or an application.**

- It can set the standard for **comparison** to track future optimization or regression.
- It can help identify changes in performance.
- Its metrics are articulated by using a broad set of Key Performance Indicators, including
  - **Response time**
  - **Processor Capacity**
  - **Mem Usage**
  - **Disk Capacity**
  - **Network Bandwidth**



# What is Benchmarking?

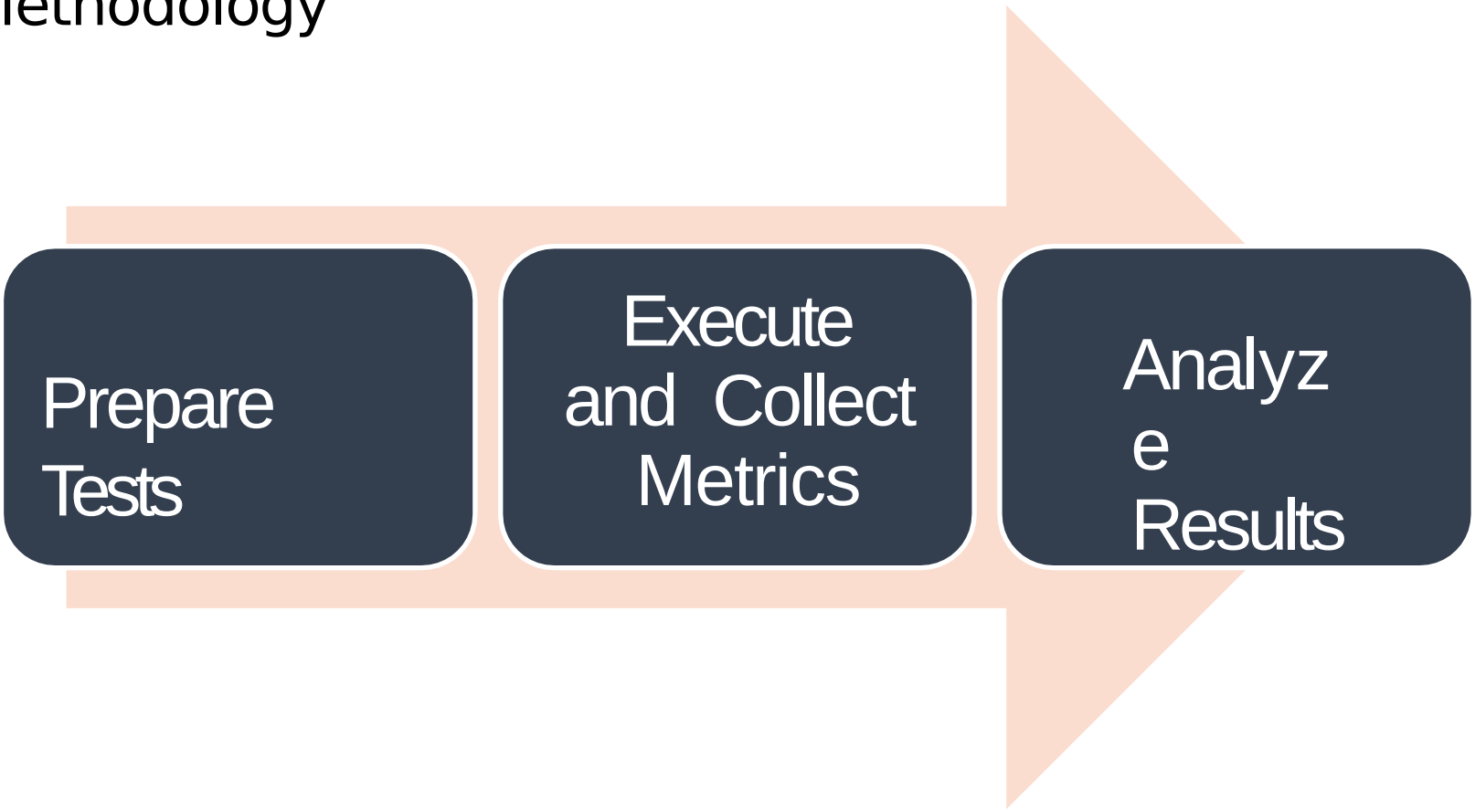
**Benchmarking is the process of comparing your system's performance against a baseline that you have created internally or against an industry standard endorsed by some other organization**

- Benchmark is achieved by working with Industry Specifications or by porting an existing implementation to meet the standards

- Many useful metrics can be shared or communicated:
  - Load Time
  - # of transactions processed
  - Web Pages accessed
  - Processor Usage
  - Memory Usage
  - Search Times

# Performance Engineering

Methodology



# Performance Engineering

## Tools

为什么使用工具

- You will need **tools** to provide a meaningful performance assessment.

- Even a simple **stopwatch** test will require a stopwatch.

- The more complex your tests get, the better tools you will need to support them.

# Performance Engineering

## Tools

- **Open Source**
  - JMeter, Grinder, openSTA, loadUI
- **Commercial**
  - Silk Performer, VSTS, LoadRunner, Performance Center

# Micro Focus Performance Center

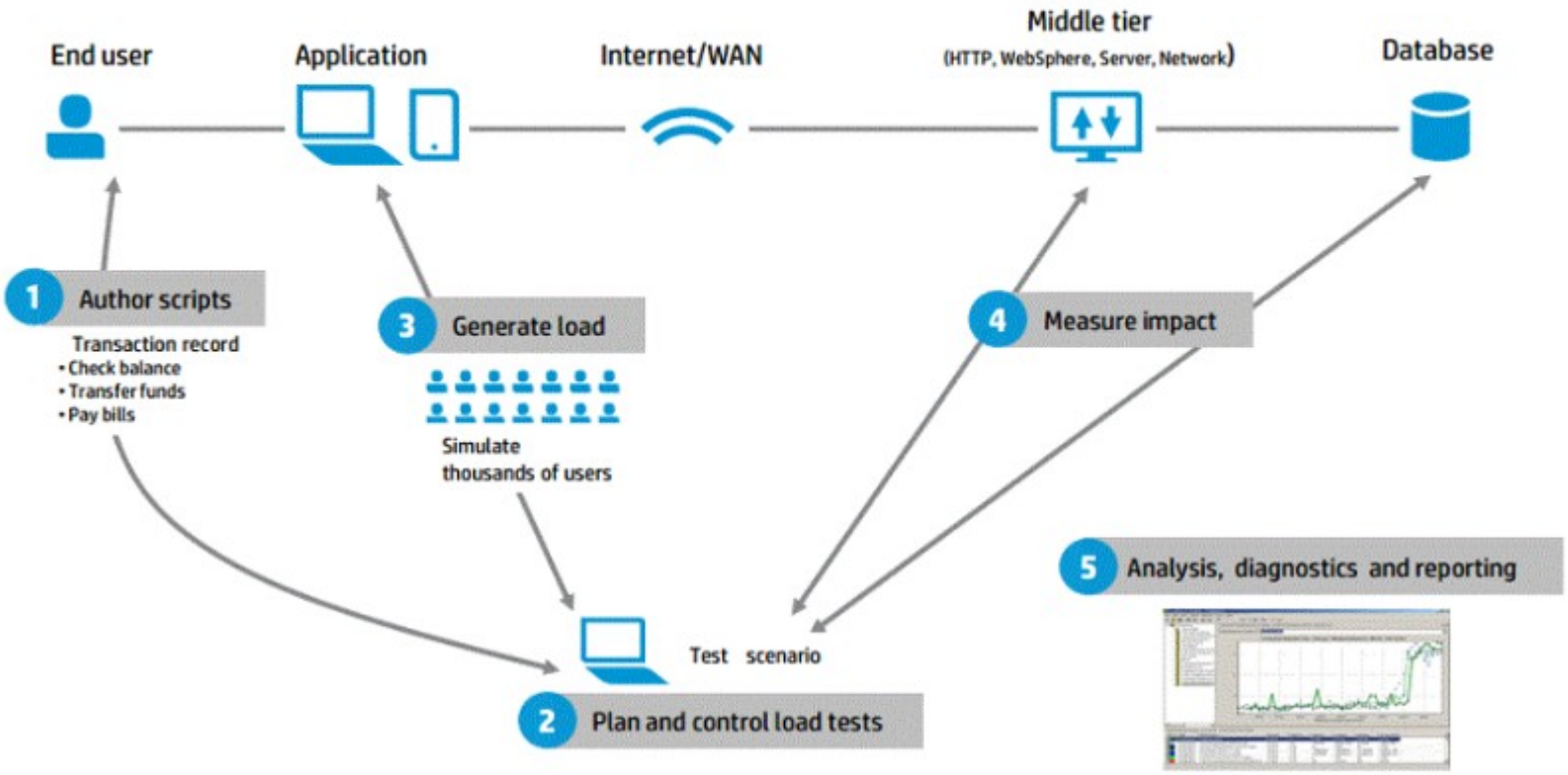
"**MicroFocus Performance Center** is a suite of **integrated performance testing** solutions that can emulate hundreds or thousands of concurrent users to apply production workloads to virtually any environment; **identify potential performance bottlenecks**; and help **diagnose and fix the root cause of the problems**."


ource: Performance Center Brochure

# Performance Center

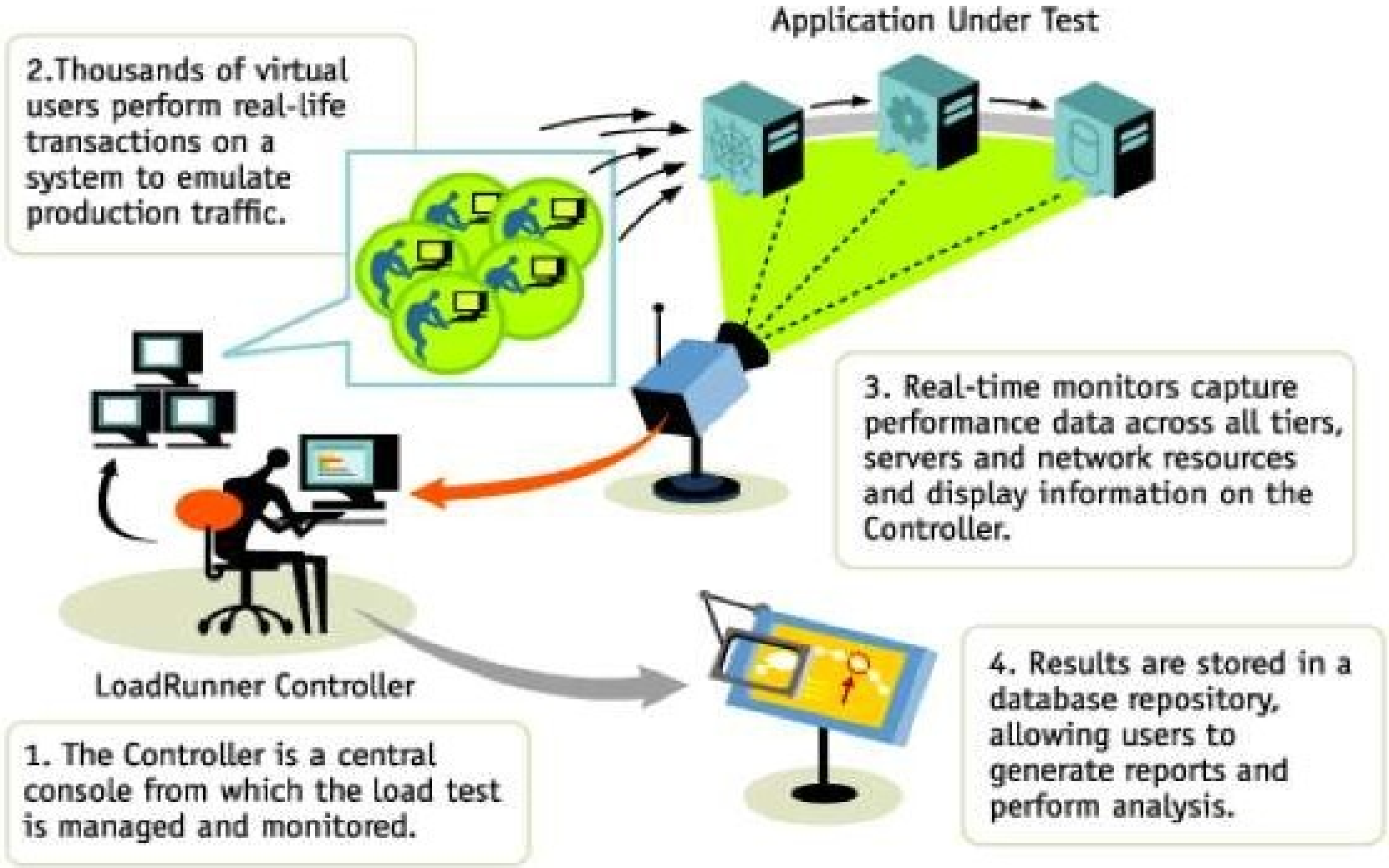
- **LoadRunner**
  - Standalone set of tools to prepare, execute and analyze performance tests
  - Supports a wide variety of protocols
  - Can collect metrics from a wide variety of systems
- **Performance Center**
  - Centralized test environment
  - Web interface
  - Leverages capabilities of LoadRunner

# How Performance Test Runs



# Performance

## How LoadRunner Works



# Performance Center - Protocols

- **Protocols**
  - HTTP/HTML, WebService, FTP, RMI, LDAP, Citrix, MQ, Ajax ...
  - SQL Server, Oracle ...
  - Oracle EBS, SAP, Siebel, PeopleSoft ...
- **Monitors**
  - Windows, Unix ...
  - WebLogic, IIS, Apache, SiteScope ...
- **Scripting Language**
  - C, Java

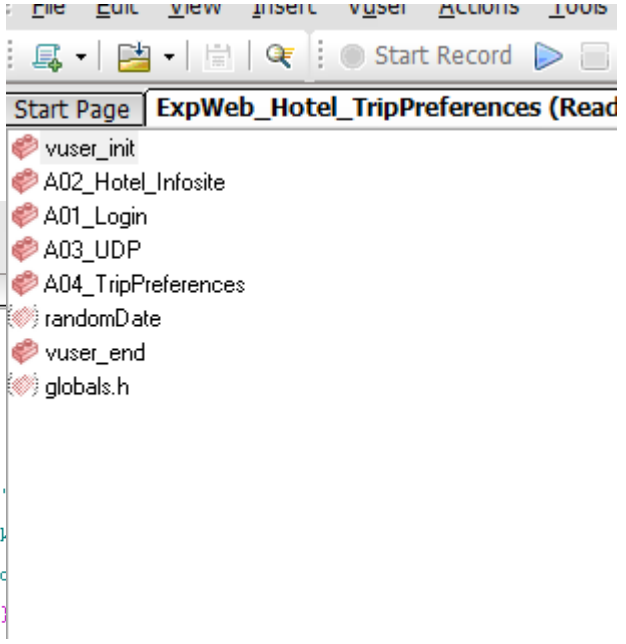
# Performance Center

- **Centralized repository** 其中存的是
  - Scripts, test scenarios, results and reports
  - Versioning (new on PC 11)
- **Dedicated Resource Pools** 其中存的是
  - Load generators and controllers
- **User Management**
  - Integrate with AD and LDAP for Authentication
- **Organized by domain and project**
  - Usage limits can be applied per project

# Performance Center

- **Has a Web interface**
  - Create test scenarios
  - Setup system probes
  - Execute and monitor tests
  - View reports
- **Desktop clients**
  - Connect to Performance Center server
  - Script development (VuGen)
  - Analysis

# VuGe
# n




dow Help

Tasks Script Tree

) - Web (HTTP/HTML)

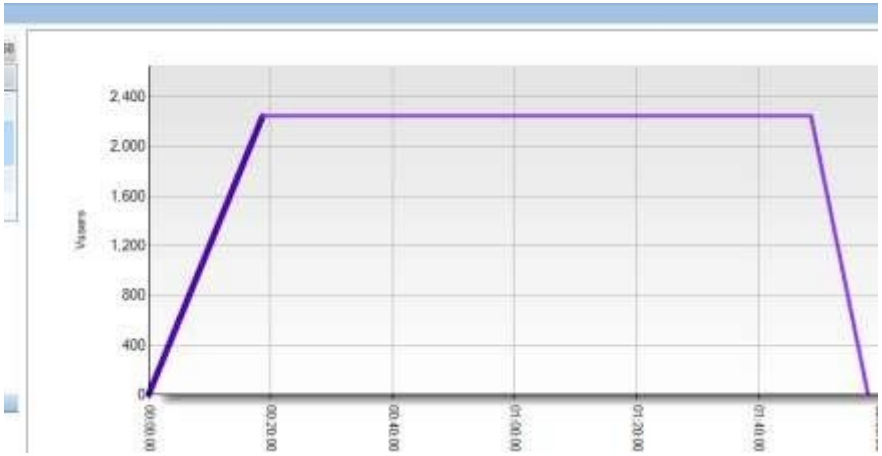
```
                        "Notfound=warning",
                        LAST);

    web_reg_find("Text/IC=Welcome",LAST);

    web_reg_find("Text/IC=Hotel Details",LAST);

    web_reg_find("Text/IC=No rooms available", "SaveCount=no_rooms", LAST );

    web_reg_find("Text/IC=We are unable to search", "SaveCount=room_search_fail"

    web_reg_find("Text/IC=This lodging cannot be booked", "SaveCount=cannot_book

    web_reg_find("Text/IC=This lodging does not allow", "SaveCount=does_not_allo

    lr_start_transaction(lr_eval_string("Hotel-Infosite-Page-Dated_{ReleaseName}

    web_url("Hotel-Information",
        "URL=http://{WebIP}/{HotelURL}.h{HotelID}.Hotel-Information?chkin={startDate}&chkout={endDat
        "Resource=0",
        "RecContentType=text/html",
        "Referer=",
        "Snapshot=t1.inf",
        "Mode=HTML",
        LAST);

    lr_end_transaction(lr_eval_string("Hotel-Infosite-Page-Dated_{ReleaseName}"), LR_AUTO);

    lr_think_time(5);

    no_rooms = atoi(lr_eval_string("{no_rooms}"));

    room_search_fail = atoi(lr_eval_string("{room_search_fail}"));

    cannot_book = atoi(lr_eval_string("{cannot_book}"));

    does_not_allow = atoi(lr_eval_string("{does_not_allow}"));

    if (strcmp(lr_eval_string("{piid}"),"")==0) {
        //lr_output_message("empty!");
```

# Web Interface

Global scheduler

New

| Action | Properties |
|---|---|
| Initialize | Initialize each Vuser just before it runs |
| Start Vusers | Start All Vusers gradually<br>10 Vusers every 00:00:05 (HH:MM:SS) |
| Duration | Run for 01:30:00 (HH:MM:SS) |
| Stop Vusers | Stop all Vusers: 20 every 00:00:05 (HH:MM:SS) |

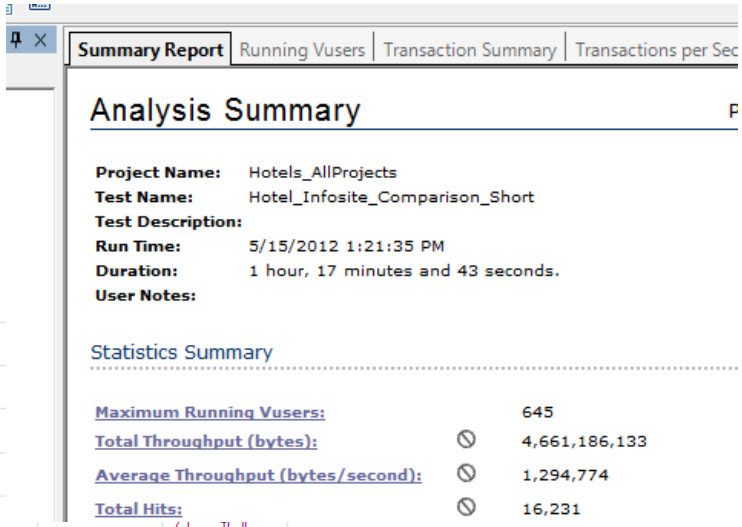


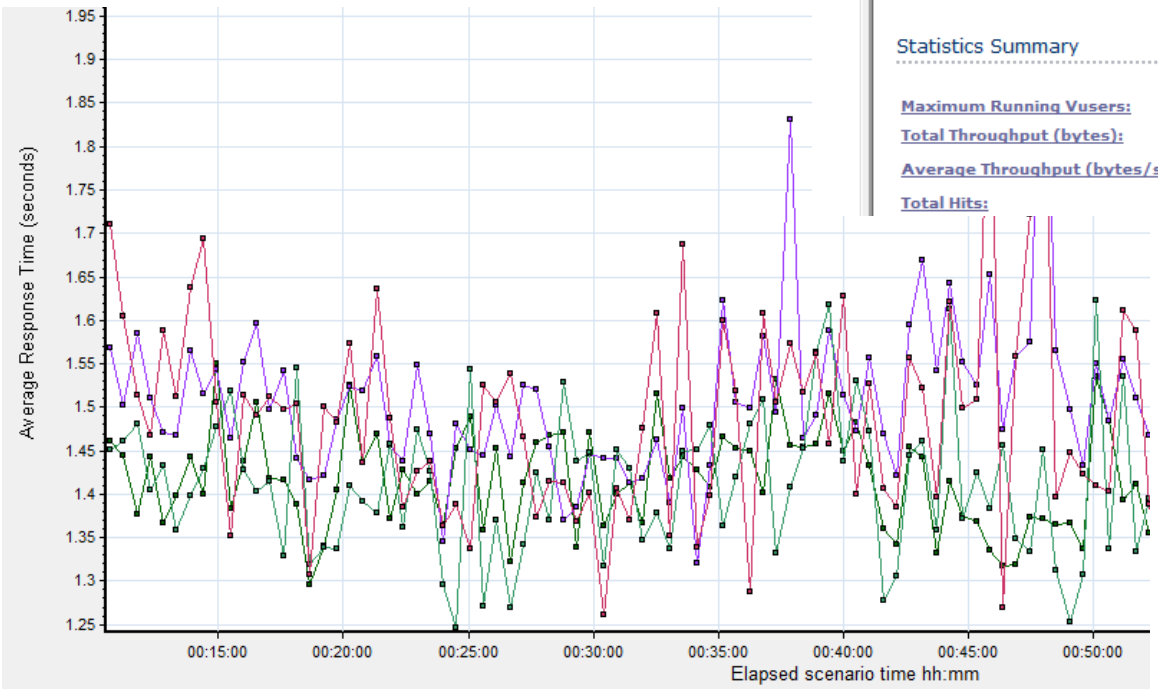

Groups

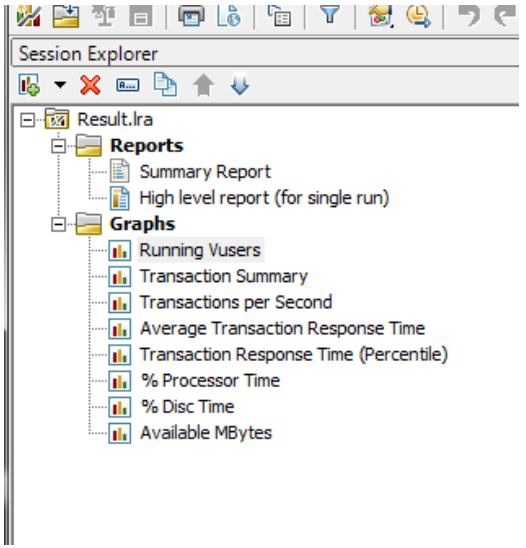
Select Scripts | Total Vusers: 2250 | LG Distribution: Assign all to each group

| Vusers-100% | Group Name | Script Name |
|---|---|---|
| 7 | expweb_flights_search | ExpWeb_Flights_S |
| 7 | expweb_home | ExpWeb_Home |
| 3 | expweb_home_search | ExpWeb_Home_Se |
| 30 | expweb_hotel_information | ExpWeb_Hotel_Info |
| 4 | expweb_hotel_information_urgencyajaxcall | ExpWeb_Hotel_Info |
| 3 | expweb_hotel_launch_page | ExpWeb_Hotel_Lau |
| 2 | expweb_hotel_trippreferences | ExpWeb_Hotel_Trip |
| 2 | expweb_isworking | ExpWeb_isWorking |
| 12 | expweb_navigation_feed_page | ExpWeb_Navigation |
| 5 | expweb_package_shopping | ExpWeb_Package_S |

# Analysis




Legend

| Col | Sce | Measurement | Minimum | Average | Maximum | Std. Deviation |
|---|---|---|---|---|---|---|
| ☑ | 1 | Hotel-Infosite-Page-Dated_trunk-trunk.ci.462289_ | 0.717 | 1.427 | 3.152 | 0.299 |
| ☑ | 1 | Hotel-Infosite-Page-Dated_trunk-trunk.ci.462289_ | 0.715 | 1.51 | 4.258 | 0.348 |
| ☑ | 1 | Hotel-Infosite-Page-Dateless_trunk-trunk.ci.46228 | 0.73 | 1.426 | 2.864 | 0.3 |

# Advantages of Performance Center

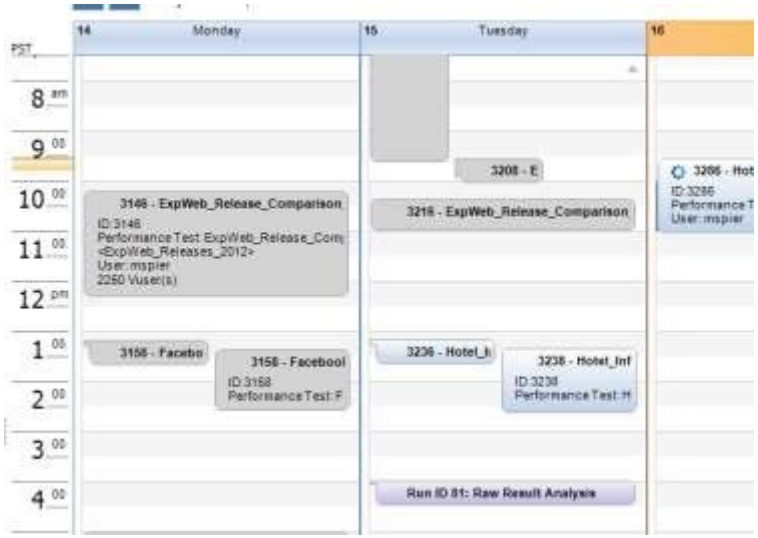
1. Environments

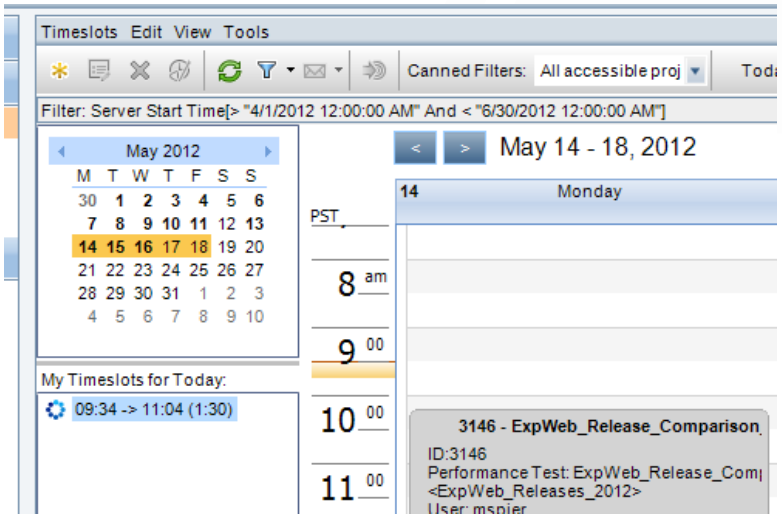
- Multiple organizations inside the same company
- Geographically dispersed teams (Global development)
- Supports wide variety of technologies
- Supports Agile development
- Continuous delivery (Weekly releases)
- Pressure to reduce costs

# Efficient Use of Resources

## How?

- Sharing resources




***Timeslot reservation enables resources to be shared without creating an environment management nightmare.***

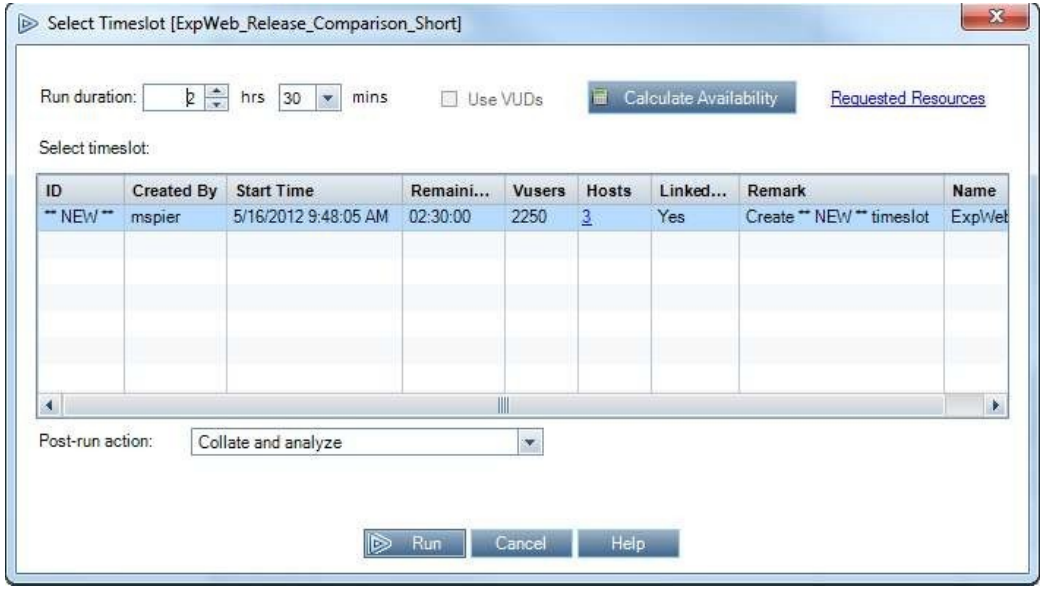
***Resource pools and per project limits enable a fair share of resources to be distributed per team.***

# Efficient Use of Resources

## How?

- Test scheduling

***The ability of scheduling tests to execute in the future enable teams to use the precious resources even when no one is around***



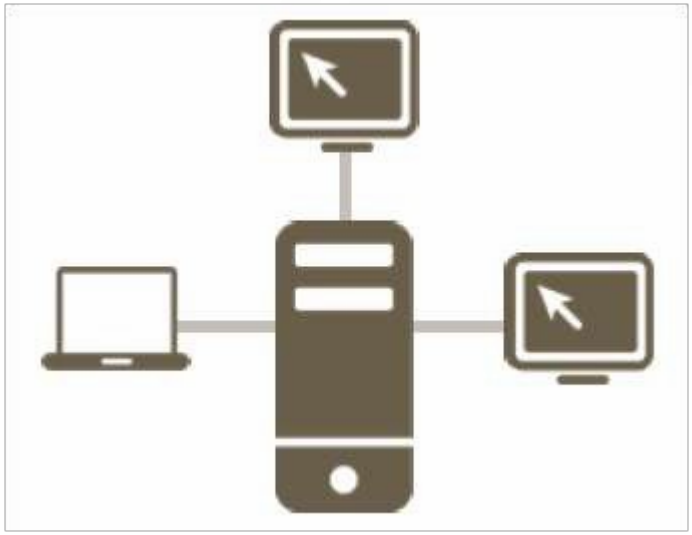

*24/7 test coverage*

## Efficient Use of Resources

## How?

- Leveraging software licenses

*License costs would be higher if each team had their own set of controller and virtual user licenses*



*There is*

## Distributed Testing Capabilities

Why?

- Limited number of performance engineers
- Increasing number of requests for early and frequent performance tests
- Test earlier in the process
- Execute small and more specialized tests

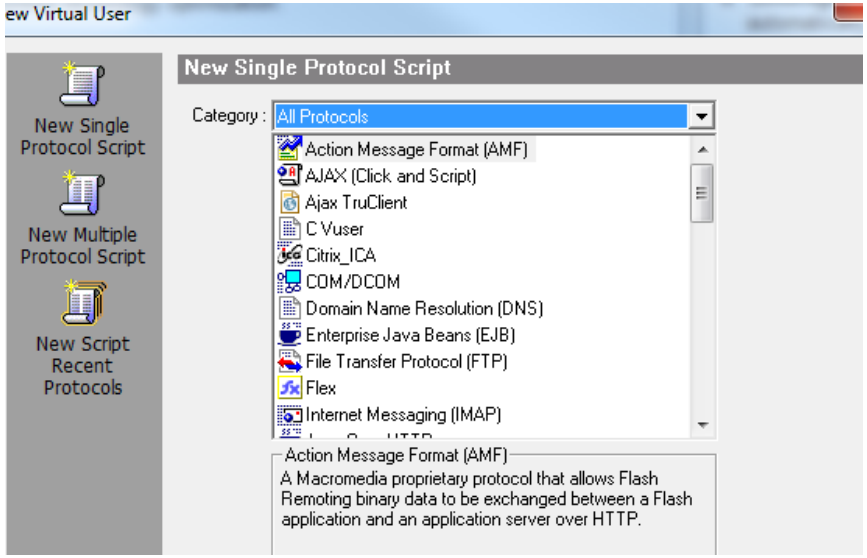
# Distributed Testing Capabilities

How?

- Enabling non-performance engineering teams to participate

***Support for a wide variety of protocols enable specialized tests such as database specific, web services, etc***

***A simple web interface enables non- performance engineering teams to execute and analyze simple tests***

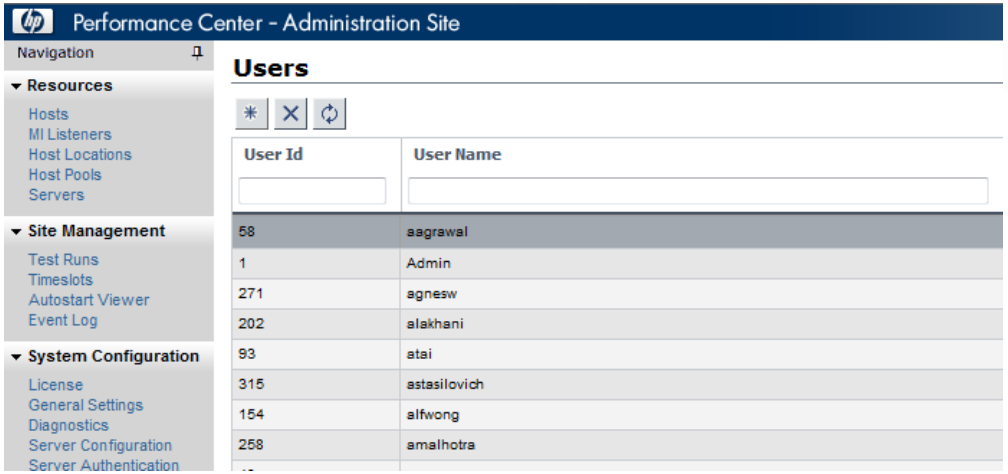


## Administration?

## How?

- **Single load test tool**

***Central administration makes it easy to add new users, create new projects and set up new resource pools.***



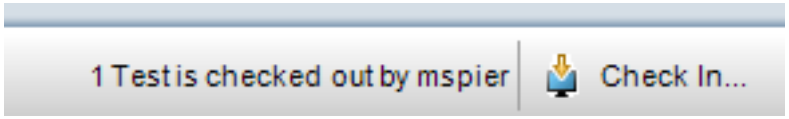

***Integration with LDAP/AD avoids the hassle of remembering another User Id & Password.***

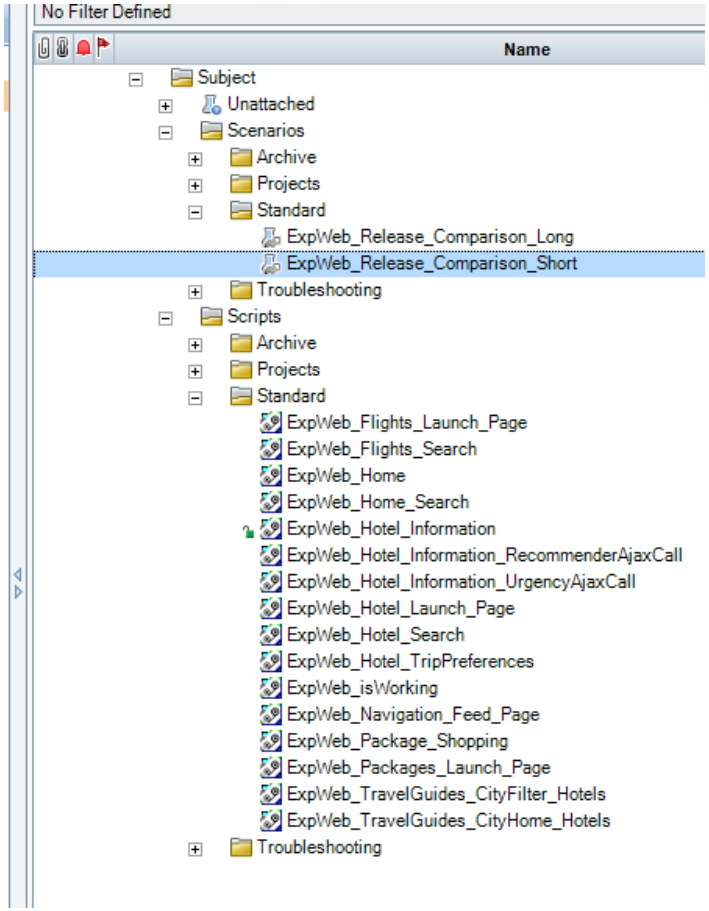
***Less than ½ FTE is necessary to support the tool.***

# •Re-Use Test Scripts

- How?
- A Central Repository for Test Artifacts

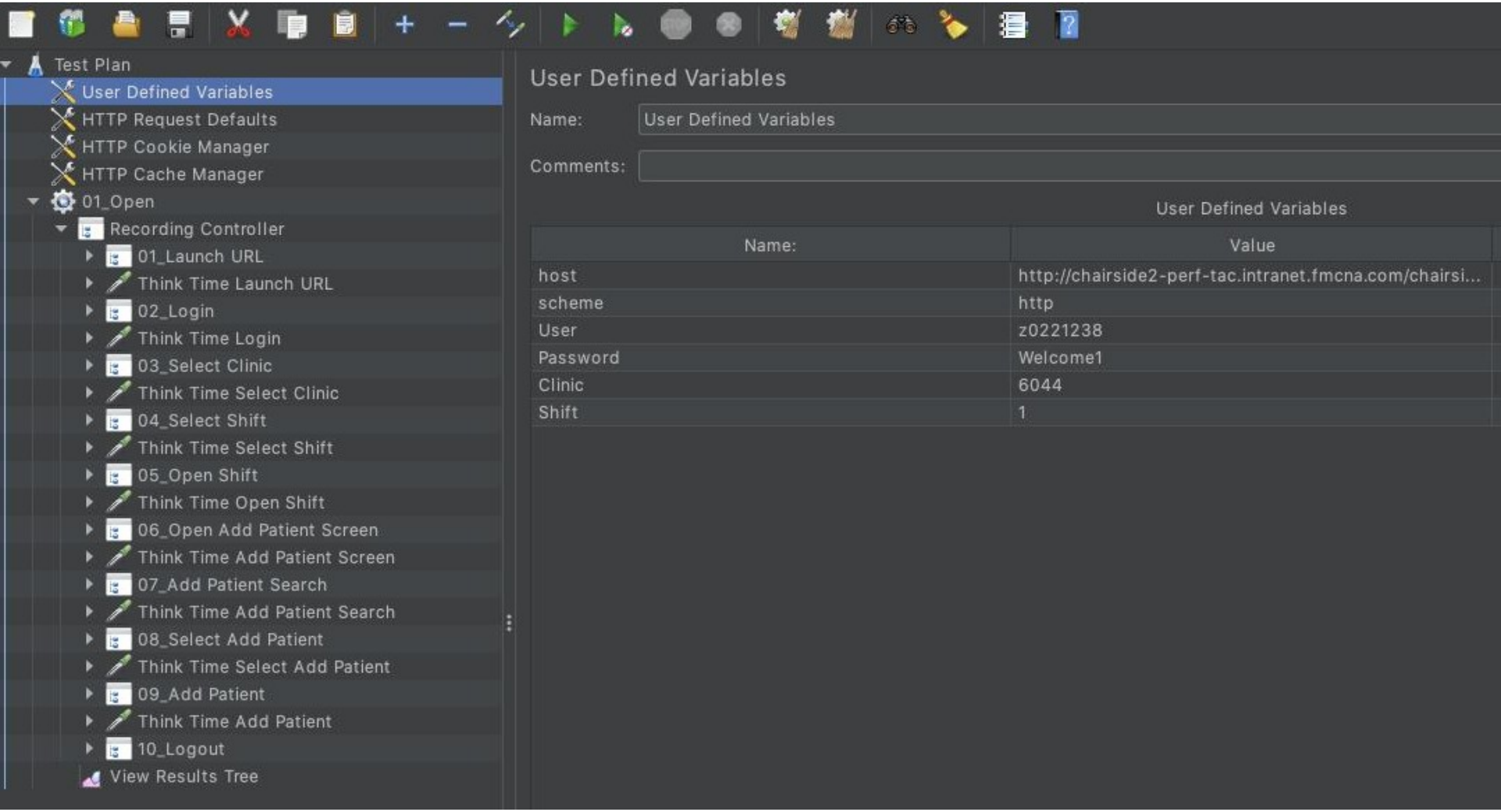
*A central repository for scripts and scenarios with version control makes it easy to find and reuse test artifacts*




***A single location for results and reports helps keeping track and comparing executions***

# Creating Scripts in Jmeter



# Typical Load Testing results

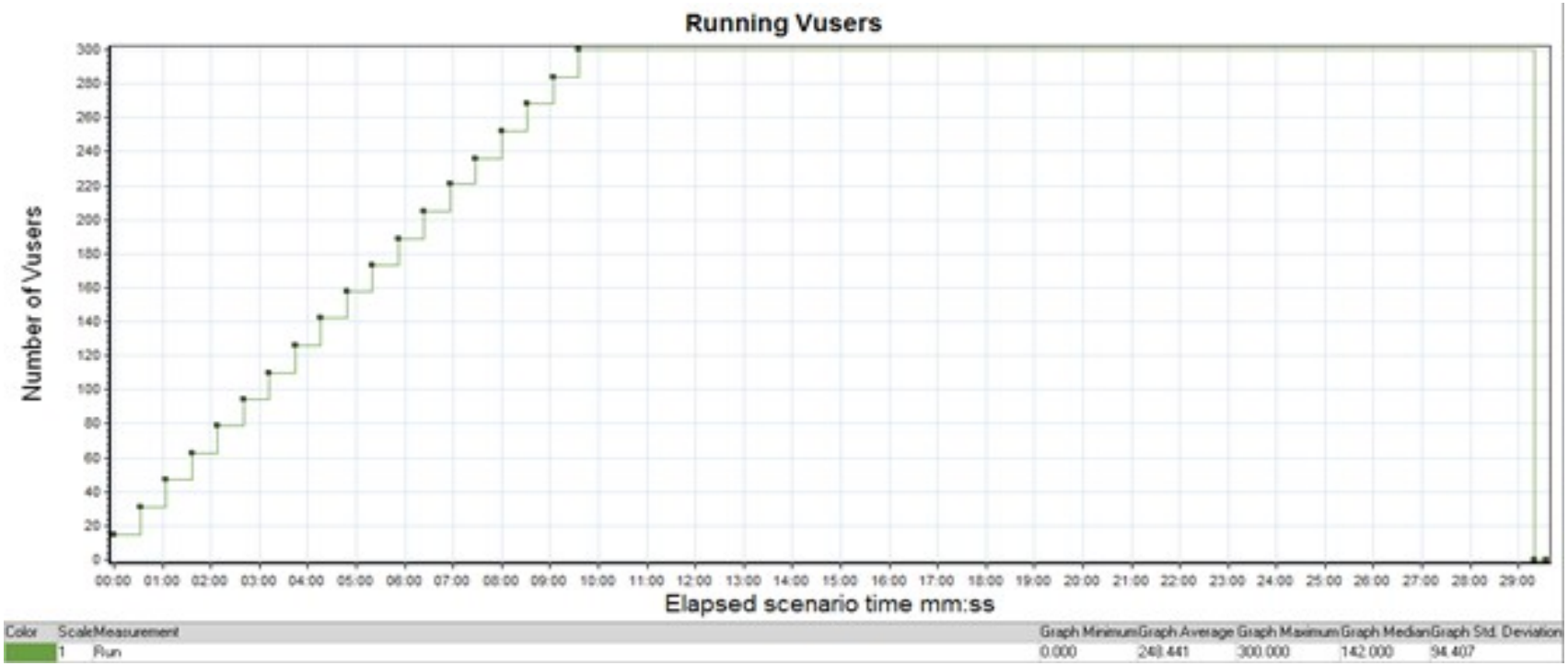
. **Summary**:

| | |
|---|---|
| Test Date | 07/23/2018 |
| Test Start Time | 10:17:12 AM |
| Test End Time | 10:46:45 AM |
| Test Duration | 29 Minutes and 33 Seconds |
| Maximum Running Vusers: | 300 |
| Total Vusers | 300 |
| Total Throughput (bytes): | 2,496,746,560 |
| Total Hits: | 54,984 |
| Total Number of Transactions: | 54,983 |

# Concurrency

Concurrency means multiple computations are happening at the same time.

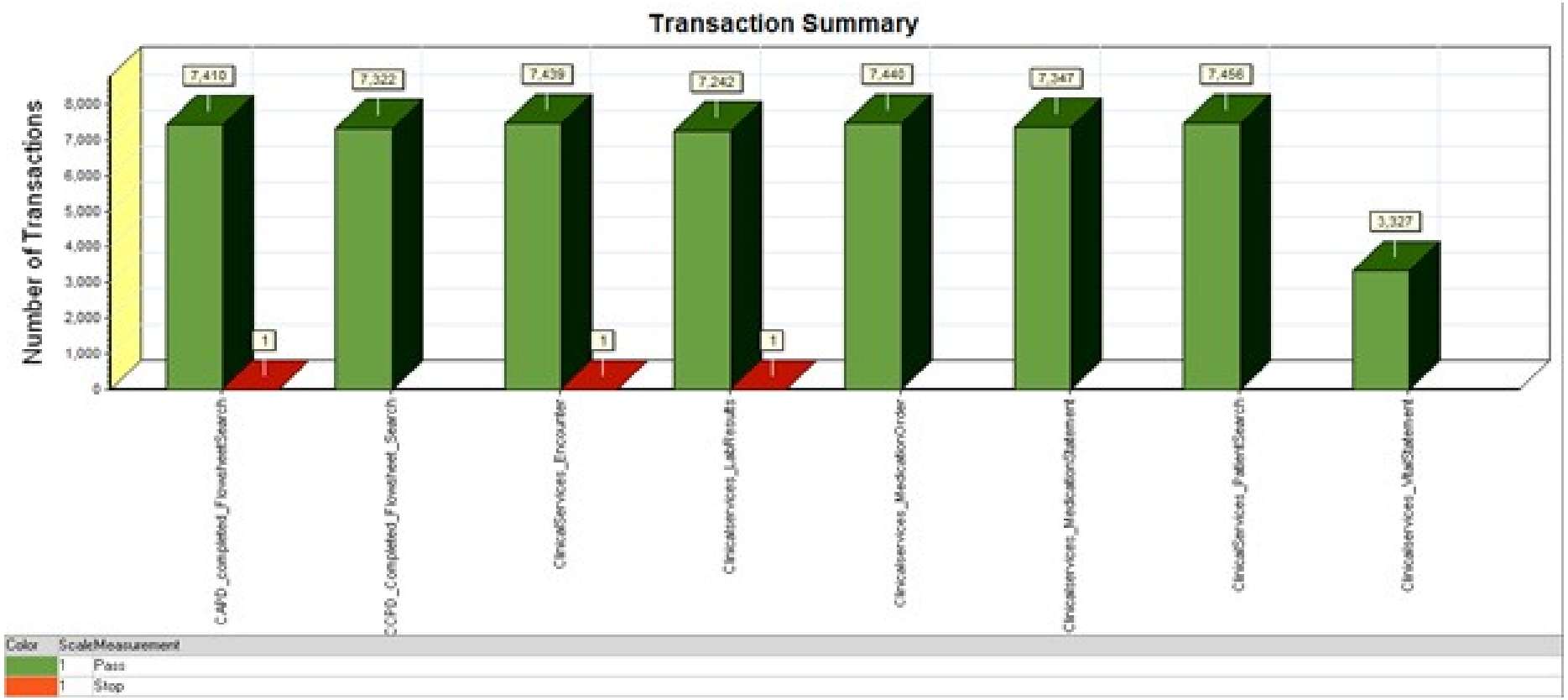
- **Concurrency:** in this test we achieved the maximum concurrency of 300 users.



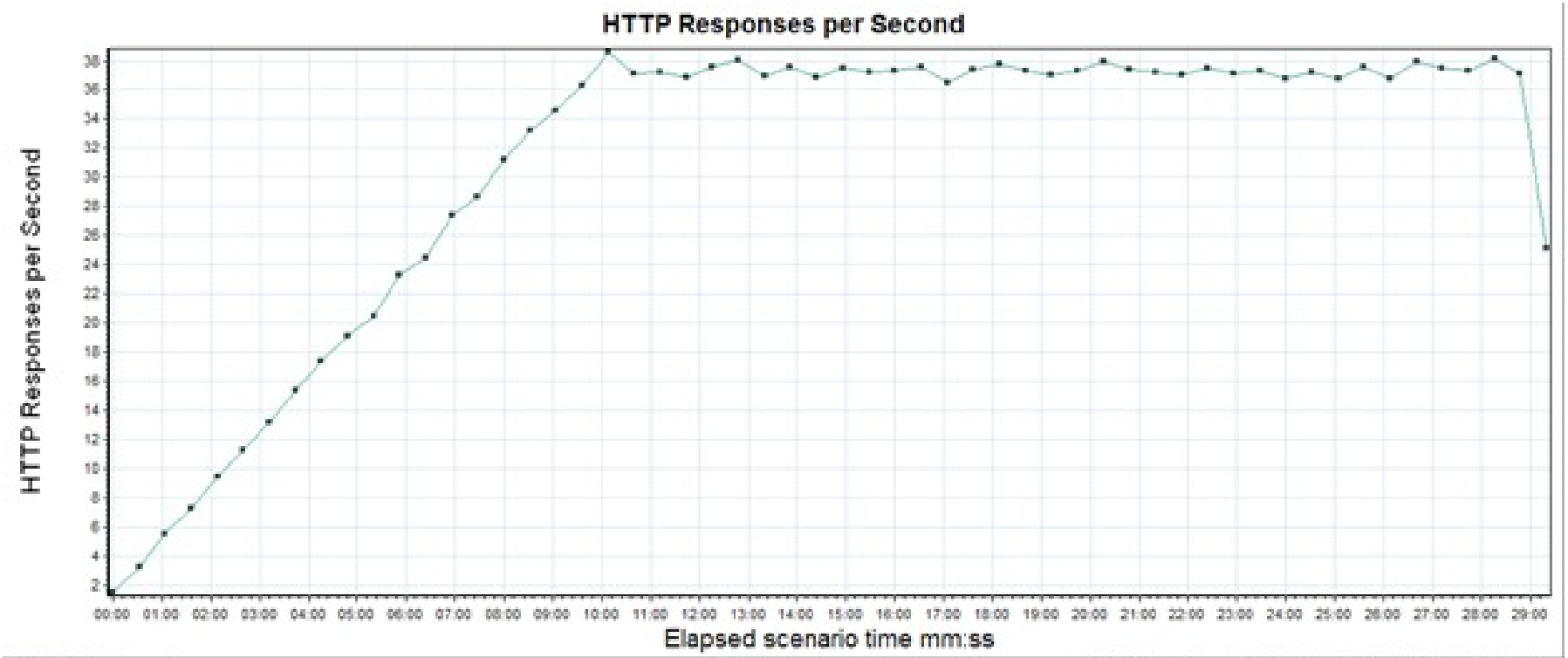
# Transaction Summary

| | 7/23/2018<br>10:17:12 AM-10:46:45 AM<br>29 min and 33 Sec<br>300 Users | | | | |
|---|---|---|---|---|---|
| Transaction Name | Avg | Std. Dev | 90 Per | Pass | Fail |
| CAPD_completed_FlowsheetSearch | 0.084 | 0.038 | 0.11 | 7,410 | 0 |
| CCPD_Completed_Flowsheet_Search | 0.079 | 0.071 | 0.1 | 7,322 | 0 |
| ClinicalServices_Encounter | 0.028 | 0.014 | 0.037 | 7,439 | 0 |
| Clinicalservices_LabResults | 0.18 | 0.1 | 0.244 | 7,242 | 0 |
| Clinicalservices_MedicationOrder | 0.019 | 0.013 | 0.029 | 7,440 | 0 |
| Clinicalservices_MedicationStatement | 0.126 | 0.079 | 0.187 | 7,347 | 0 |
| ClinicalServices_PatientSearch | 0.017 | 0.094 | 0.011 | 7,456 | 0 |
| Clinicalservices_VitalStatement | 0.124 | 0.098 | 0.18 | 3,327 | 0 |

# Transaction Summary



# HTTP Response



# Top 15 BEST Performance Testing Tools

- WebLOAD
- LoadUI NG Pro
- SmartMeter.io
- LoadView
- Apache JMeter
- LoadRunner (Performance Center)
- Appvance
- NeoLoad
- LoadComplete
- Loadster
- LoadImpact
- Rational Performance Tester
- Testing Anywhere
- OpenSTA
- QEngine (ManageEngine)
- Loadstorm
- CloudTest
- Httperf
- WAPT

https://www.softwaretestinghelp.com/performance-testing-tools-load-testing-tools/

# Rational Performance Tester

- Rational Performance Tester(RPT) is a performance and load testing tool developed by IBM Corporation:
    - No coding involved
    - Scheduled and event based testing
    - Real time reporting for immediate performance problem identification
    - Run with large multi users tests
    - Accurate user profile workloads
    - Automated test data variation
    - Automatic identification of dynamic server responses
    - Rendered HTML view of web pages visited during test recordings
    - Environment and Platform support
    - Entrust security protocol support
    - Java code insertion for customization

https://www.guru99.com/performance-testing-tools.html

# References:


https://msdn.microsoft.com/en-us/library/bb924356.aspx

http://www.gallop.net/blog/2-classic-cases-where-performance-testing-failures-plague-large-organisations/

http://www.techrepublic.com/article/outages-on-facebook-linkedin-paypal-and-other-sites-might-point-to-bgp-failures/

https://blogs.akamai.com/2015/06/performance-matters-more-than-ever.html

https://blog.kissmetrics.com/loading-time/

# References:

https://www.dynatrace.com/blog/key-performance-metrics-load-tests-beyond-response-time-part/

# Software Monitoring

- Software Monitoring Types:
  - Application Monitoring
  - Network Monitoring
  - Performance Monitoring
  - And more…(Performance Monitoring, DB Monitoring and etc..)

# What is Application Monitoring?

- **Application monitoring** is a process that ensures that a software **application** processes and performs in an expected manner and scope.
  - Also known as Application Performance Monitoring (APM)

- An '**Alert**' is sent to the particular distribution groups upon detecting any unexpected errors or abnormalities.

# What is Application Monitoring?

- This technique routinely **identifies, measures and evaluates** the performance of an **application** and provides the means to isolate and rectify any abnormalities.

- **Dedicated teams** monitor and receive alerts based on criteria set in the monitoring tool.

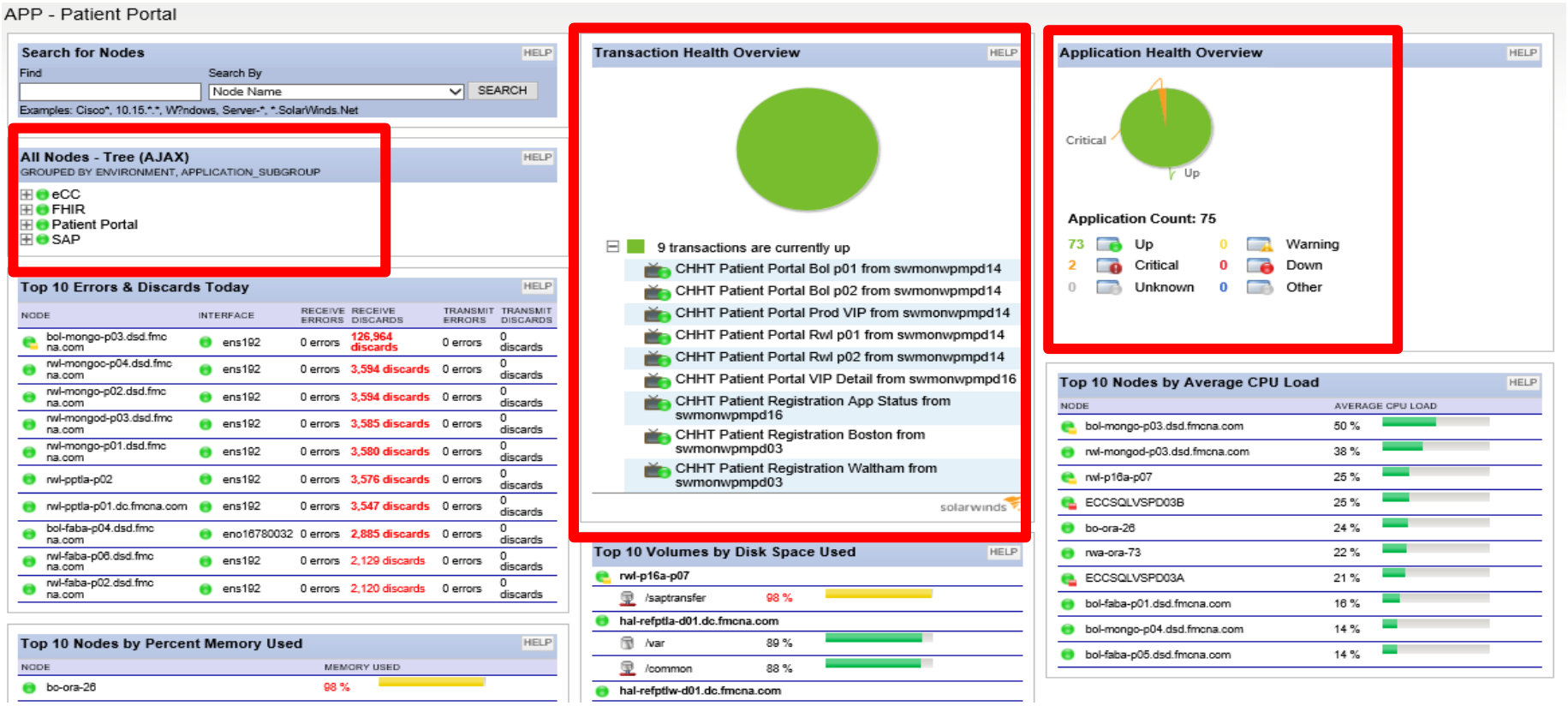
Based on the alerts, processes are followed to address the issue and restore the services.

# Application Monitoring Tools

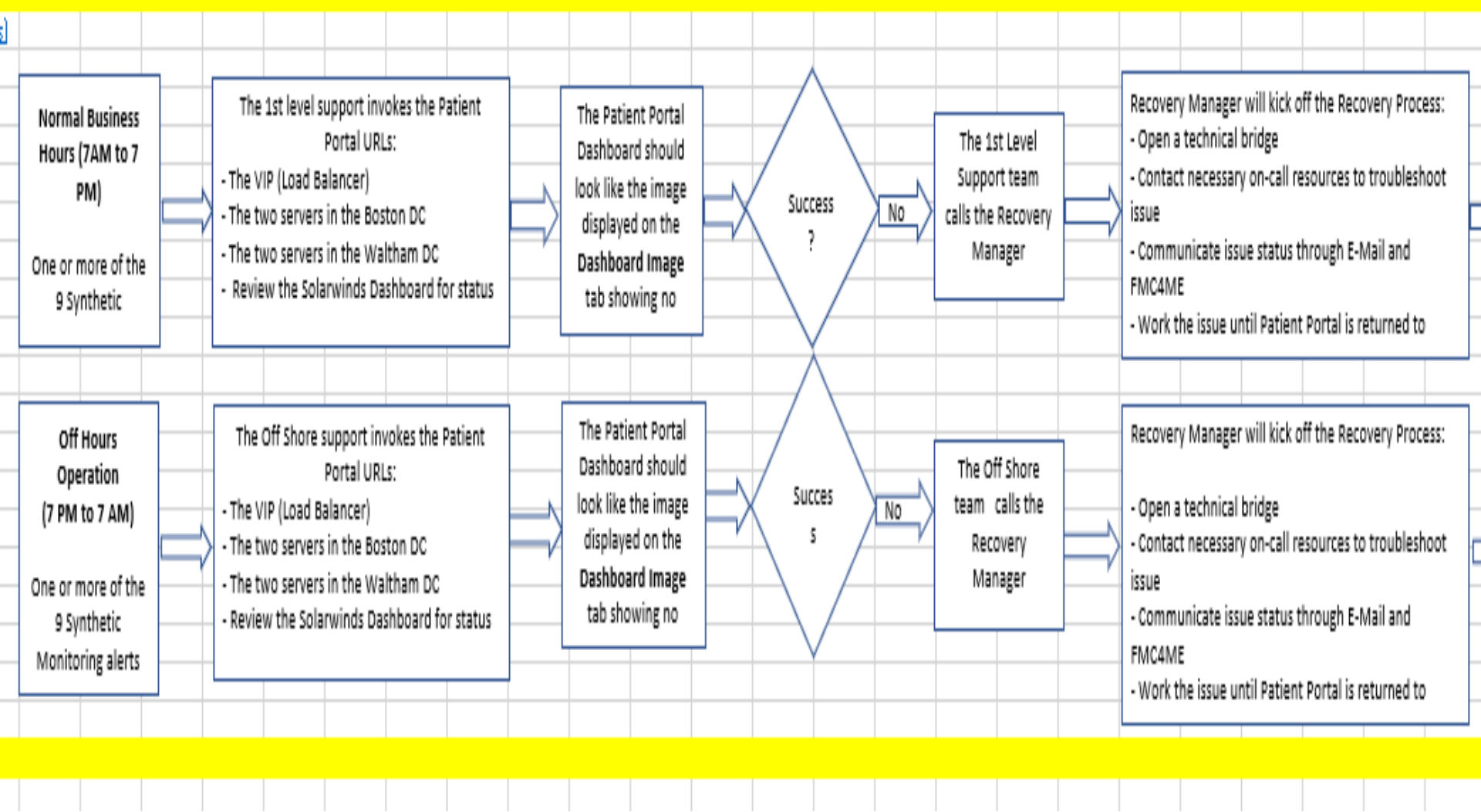
- App Dynamics
- Splunk
- Amazon CloudWatch
- Solarwinds
- New Relic APM
- Coradiant
- Riverbed SteelCentral
- Dell Foglight
- IDERA Precise
- Nastel


https://www.imobdevtech.com/Blog/list-of-top-10-application-performance-monitoring-tools/

# Solarwinds Dashboard



# Application Monitoring Process Flow



# Questions?