REACHING THE HOLY GRAIL of EFFECTIVE APPLICATION PERFORMANCE TESTING and ANALYSIS

CQAA April 2014
Speaker Program

Chicago, IL

April 23, 2014

© copyright www.sweeneypr.com

Introductions

Vic Soder:

- Director with Deloitte Consulting LLP's Systems Integration practice; Seattle, WA
- BS, Mathematical Sciences, from Stanford University

Liben Workneh:

- Senior Consultant with Deloitte Consulting LLP's Systems Integration practice; Chicago, IL
- BS in Computer Science, MS in Telecommunications Systems, and MBA, from University of Iowa

Fred Dawood:

- Senior Consultant with Deloitte Consulting LLP's Systems Integration practice; Chicago, IL
- Bachelor of Science in Computer Information Systems, from DePaul University

All specializing in application performance, system sizing, and capacity planning advisory services

Deloitte Consulting LLP:

- We offer our consulting services to clients in three major service areas – Technology, Strategy and Operations, and Human Capital
- Over 17,200 technology consulting professionals in the U.S. and India
- Deloitte Consulting serves well over half of the Global Fortune 500, most of the 50 U.S. states, and has a significant presence within the U.S. Federal Government

Technology / Systems Integration Practice / Application Performance Center of Expertise:

- Deloitte Consulting's Technology practice helps clients manage the critical business of information technology; over 4,000 professionals in our Systems Integration practice in the U.S. & India
- Application Performance COE established over 8 years ago, and now has approximately 95 participants

REACHING THE HOLY GRAIL of EFFECTIVE APPLICATION PERFORMANCE TESTING and ANALYSIS

Introduction & Topics

- Why are we doing Performance Testing?
- Is there a Silver Bullet?
- Is Performance Testing with New Tools and Techniques Getting Easier?
- Application Workload Understanding with Effective Monitoring
- Application Workload Variations
- Application Workload Control
- Effective Workload Monitoring
- Performance Testing Toolsets and Challenges
- Common Performance Test Planning Pitfalls
- New Challenges and their impact on Quality Assurance
- Summary & Achieving Complete Alignment of Capabilities
- Q&A

WHY ARE WE DOING PERFORMANCE TESTING?

Performance Testing and Analysis Needs to be aligned with the Business Mission of the Application, and needs to be driven by Service Level Agreements for the Application

Quality Assurance and Performance Testing:

- ➢ Focusing on non-functional requirements
- ➢ Preventing performance defects in solutions and products
- ➢ Striving to be both compliant with SLAs AND cost-effective

The objective is not always to make the application run as fast as possible;

It involves reaching a business negotiation and determining a middle ground:

What is truly required by the end users

vs.

What the application and technology can deliver

WHY DO WE DO STRESS TESTING?

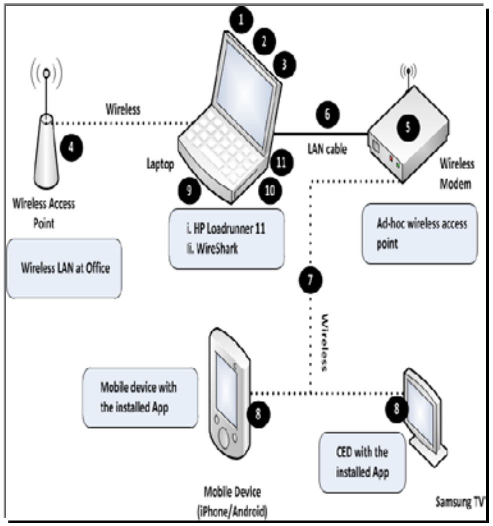
- ➢ To see which resources get saturated first
- ➢ To gain an understanding of the upper limits of the application and supporting infrastructure

IS THERE A SINGLE SILVER BULLET?

Unfortunately, NO:

We need to keep our hands on the pulse of multiple factors in our application and computing environments:

1) Transaction volumes
2) Application code
3) Application database configuration
4) User training and behavior
5) Server and storage infrastructure
6) Interfaces
7) Network
8) Third Parties and Vendors
9) Testing and Monitoring Tools
10) Alignment of schedules



- ➢ In addition, the ever-increasing number of application access channels force us to look at more parameters and complexities

- ➢ ***The <u>multiple</u> silver bullet<u>s</u> can be anywhere!***

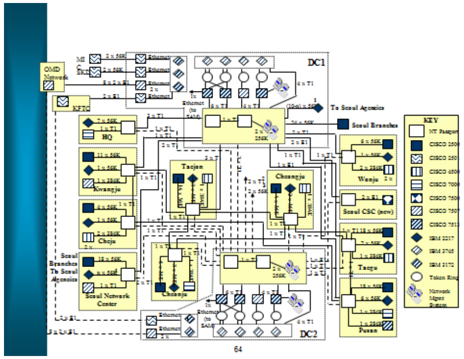
IS PERFORMANCE TESTING WITH NEW TOOLS AND TECHNIQUES GETTING EASIER?

Not Necessarily:

Our systems environments are far more complex than they were 10-15 years ago:

- More layers
- More Third Party components
- More "X as a Service"
- Greater Expectations:
  - "Do more with less"
  - "Hardware is faster; Why aren't applications faster?"
  - Legacy application always had response time < 2 seconds;

    "Why not for this new system?"



There are more and more good tools, but still many challenges.

To be further discussed later in section on performance testing tools

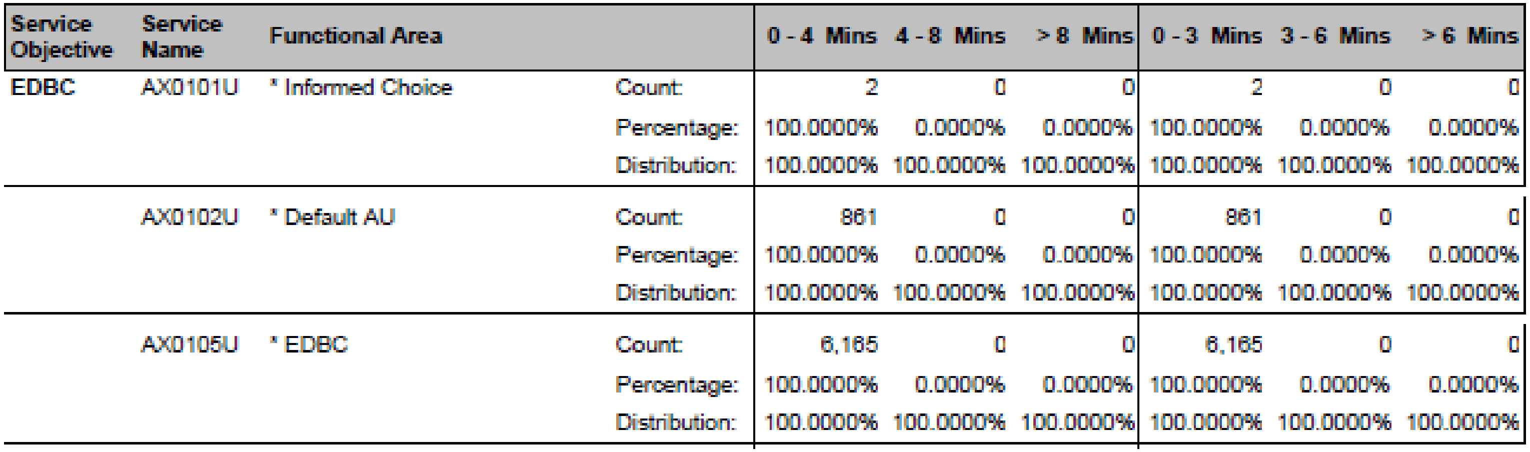
APPLICATION WORKLOAD UNDERSTANDING, WITH EFFECTIVE MONITORING

Performance Testing Needs to be aligned with our business workloads, and must be supported by Effective Monitoring:

Alignment with Business Workloads:

- We must report application activity from a business workload perspective.
- A CIO or Business Manager rarely cares that Apache response time averaged 3 seconds;
- They want to know how long orders take, how long critical call center inquiry transactions take, etc.

EXAMPLE - (mismatch between reporting and true SLA):



Service Objective	Service Name	Functional Area		0 - 4 Mins	4 - 8 Mins	> 8 Mins	0 - 3 Mins	3 - 6 Mins	> 6 Mins
EDBC	AX0101U	* Informed Choice	Count:	2	0	0	2	0	0
			Percentage:	100.0000%	0.0000%	0.0000%	100.0000%	0.0000%	0.0000%
			Distribution:	100.0000%	100.0000%	100.0000%	100.0000%	100.0000%	100.0000%
	AX0102U	* Default AU	Count:	861	0	0	861	0	0
			Percentage:	100.0000%	0.0000%	0.0000%	100.0000%	0.0000%	0.0000%
			Distribution:	100.0000%	100.0000%	100.0000%	100.0000%	100.0000%	100.0000%
	AX0105U	* EDBC	Count:	6,165	0	0	6,165	0	0
			Percentage:	100.0000%	0.0000%	0.0000%	100.0000%	0.0000%	0.0000%
			Distribution:	100.0000%	100.0000%	100.0000%	100.0000%	100.0000%	100.0000%

APPLICATION WORKLOAD UNDERSTANDING, WITH EFFECTIVE MONITORING

Performance Testing Needs to be aligned with our business workloads, and must be supported by Effective Monitoring:

Effective Monitoring:

If you can’t measure it, you can’t report on it!

This is a very complex endeavor:

- Very large numbers (e.g., transactions)
- Very small numbers (e.g., service times)
- Tons of data (e.g., millions or billions of database rows)

EXAMPLE:

- A batch process that must do 50,000,000 operations per night:
- Assume 1 millisecond per operation.
- How long will it take (if it is serial / single-threaded)?

50,000,000 X (.001) = 50,000 seconds = 13.9 hours

APPLICATION WORKLOADS: WORKLOAD VARIATIONS AND COMBINATIONS

Who Understands the Peak Day in the Life of the Application?

- ➢ Understanding and forecasting how the business will use the application is critical.
- ➢ This is an important aspect of planning for performance under peak conditions.
- ➢ Where can performance test planning problems start? Ask the following questions at the beginning of performance testing and test scripting:
- ➢ Who is assessing how the application will be used, and who is telling the performance test scripter how the application will be used?
  - An external contractor?
  - An internal developer?
  - A business end user?

+ how to overcome these issues?

APPLICATION WORKLOADS: WORKLOAD VARIATIONS AND COMBINATIONS

Understanding the Peak Day in the Life of the Application:

Examples:

Seemingly simple assumptions can have a major impact on system sizing and performance. For example, assumptions about peak workload conditions can make or break a system performance plan:

- Clocking Out: Do shop floor workers always submit their final task within 5 minutes of clocking out, or within 15 minutes of clocking out?
- Daily, weekly, month-end business cycles: Does all month end processing need to be completed by the 1st, 2nd, or 3rd? Does it start before the 1st?
- Tax-filing peaks: Do 90% of the returns get submitted in the final 2 weeks or during the final 4 weeks?

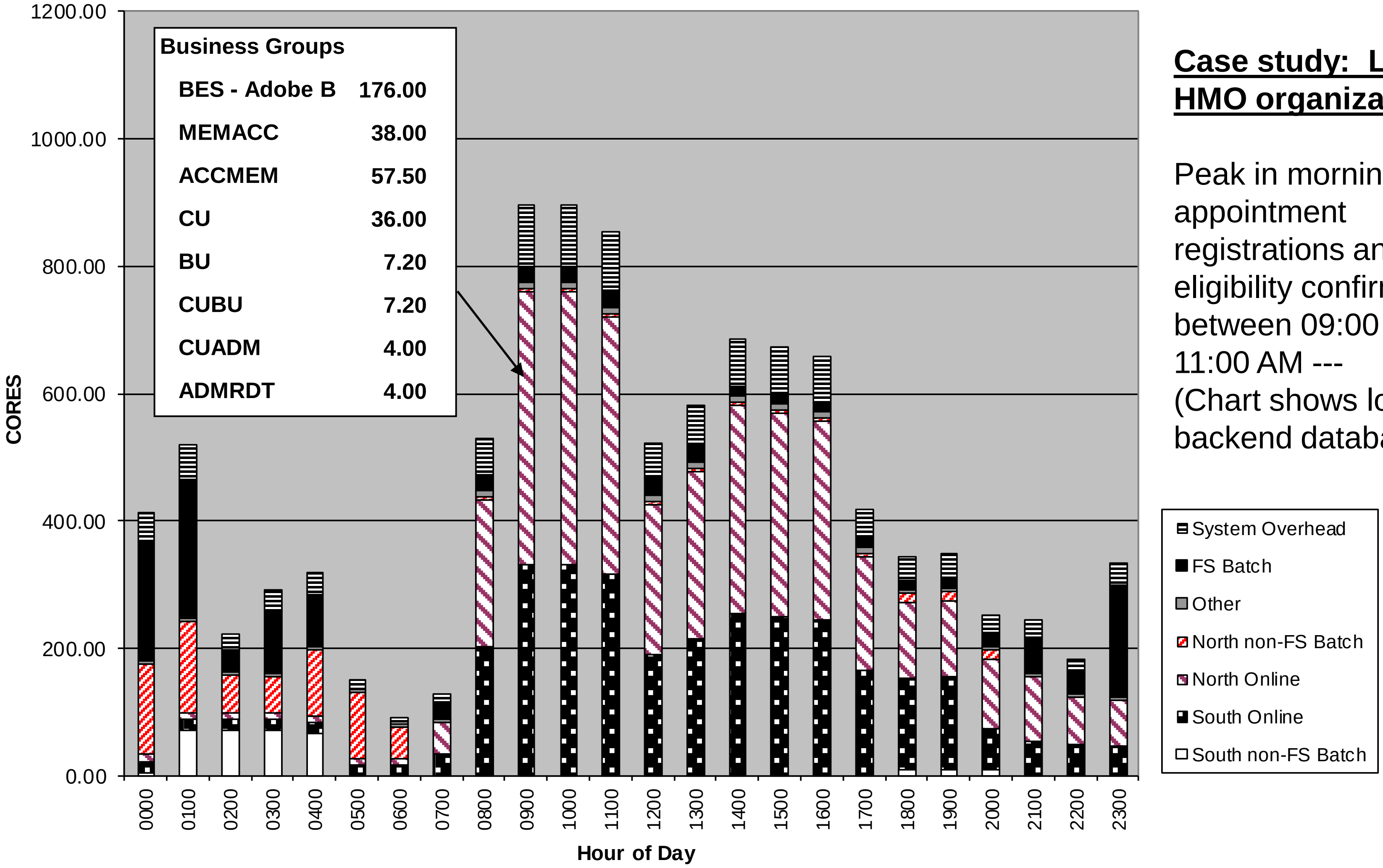
➔ How many transactions per second need to be sustained during a peak hour?

Once peaks are identified, another straightforward best practice is:

Break the total workload into a larger number of sub-workloads of similar magnitude.

APPLICATION WORKLOADS: UNDERSTAND YOUR PEAKS AND VALLEYS, AND BREAK YOUR WORKLOADS DOWN INTO MANAGEABLE COMPONENTS

Capacity Forecast: MEDIUM



Business Groups	
BES - Adobe B	176.00
MEMACC	38.00
ACCMEM	57.50
CU	36.00
BU	7.20
CUBU	7.20
CUADM	4.00
ADMRDT	4.00

Case study: Large HMO organization:

Peak in morning appointment registrations and eligibility confirmations between 09:00 and 11:00 AM ---
(Chart shows loading on backend database)

APPLICATION WORKLOADS: CONTROLLING THE WORKLOAD: Key Experiences and Observations

User Training: We shouldn't need to control everything with infrastructure.

End user behavior, if uncontrolled, can cause performance problems for nearly any production application system (for custom or commercial packages)

Poorly qualified searches: Searches for "ALL", or searches for huge subsets (such as all last names beginning with S in a 10,000,000 name database) will usually degrade performance and consume significant resources. If no edits or controls in the application ➔ it must be controlled via user training

Inefficient requests for ad-hoc reports: Such as reports that return thousands of pages or millions of rows. If no edits or controls in the application ➔ it must be controlled via user training.

Security Overheads: Are there known tradeoffs that can be made w/r to security functionality and performance?

Remediation and Resolution Examples:

- **More effective controls and edits**
- **More effective user training**
- **Understanding impacts of security bells and whistles**

APPLICATION WORKLOAD UNDERSTANDING, WITH EFFECTIVE MONITORING

Performance Testing and Analysis Needs to be aligned with our business workloads, and must be supported by Effective Monitoring:

- Strive to report by business workload, and not by technical workload
- Reports should relate to SLA targets
- Monitoring for performance test vs. monitoring for production
  - Measuring production response times, non-intrusively, and with low overhead

Performance Reporting Requirement Drivers

- SLA compliance
- Performance Issue Identification and Analysis
- Workload trending and Capacity Planning

Important side note: MONITORING TOOLS -- STRENGTH BY AREA

One tool is usually not enough. Some tools have wider application than others

Infrastructure

Middleware

Application

ONE EXAMPLE:

Tools	Network	Storage, CPU, OS	App server Web server	SOA / Web services	MQ	Database Server	Batch subsystem	Application
CA WILY	◔	◑	●	●	◑	◔	◔	◔
CA CEM	◑	○	◔	◔	○	○	○	◑
Splunk	◑	◑	◑	◑	◔	◔	◑	●
NetQos	●	○	○	○	○	○	○	○
Hobbit	◔	●	○	○	○	○	○	○
I3	○	○	○	○	○	●	◔	○
Qpasa	○	○	○	○	●	○	○	◔
OpCon	○	○	○	○	○	○	●	◔
Custom	◔	◔	◔	◔	○	◔	◔	●

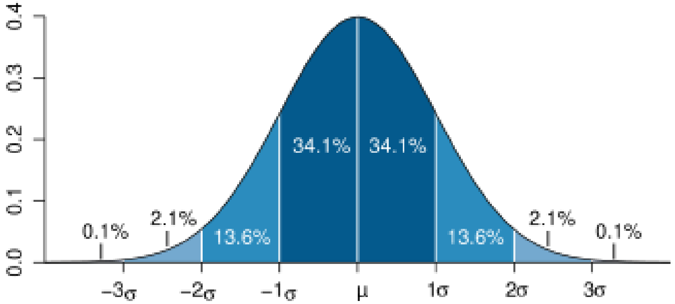
PERFORMANCE MONITORING & REPORTING REQUIREMENTS

Help your organization or client set objective, measurable and economical SLA’s

SLA Guidelines

- SLAs must be based upon Business needs
- Educate client - Stricter SLAs cost more
  - Implementation cost
  - And/or risk mitigation cost

- Online & Web Services: Response time requirements
  - Highlight dependency on hardware utilization to support required throughput.
    - Guaranteed or Accept Response Time SLAs ONLY when below a max CPU/memory utilization
  - Avoid absolute limits by individual transactions. Consider:
    - Statistical distribution and standard deviations (e.g. 95% of transactions under 1 second)
    - Group transactions by business function/complexity of processing for SLA timeslots.



PERFORMANCE MONITORING & REPORTING REQUIREMENTS

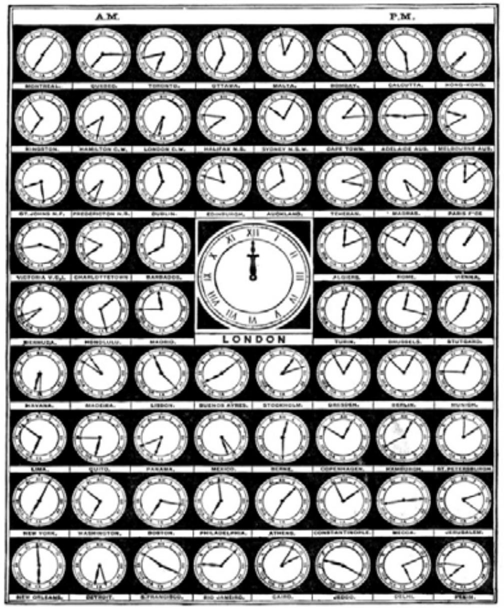
Batch requirements are very different from Online

SLA Guidelines

Batch : SLAs by total batch duration or completion times

Considerations:

- Consider who uses batch output? When do they need it?
- Is there a resource contention possible between online and batch work?
- Batch frequency (daily, weekly)
- Split SLA by specific batch function (interfaces processing)
- Can every batch peak period be processed by the beginning of the next business day?

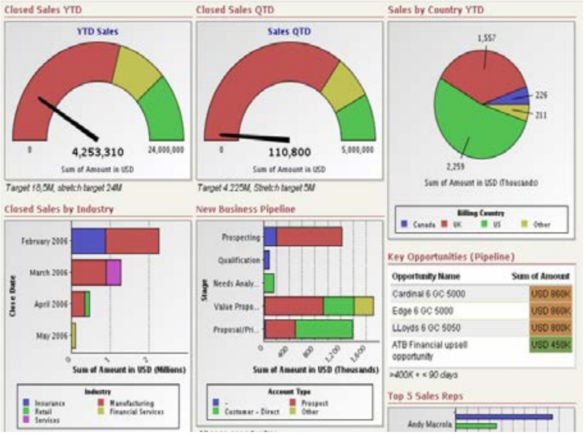


PERFORMANCE REPORTING REQUIREMENTS

Identify issues early, respond effectively and proportionately, diagnose accurately

Performance issue identification and Analysis Requirements

- Generate timely alerts (before end user experiences issues)
- Ability to monitor current performance at Application, server and Infrastructure level
- Ability to collate and present summary/dashboard
- Ability to dynamically collect trace/detailed information for issue analysis
- Must have low overhead (in quantifiable terms, say <1%)



Key Opportunities (Pipeline)

Opportunity Name	Sum of Amount
Cardinal 6 GC 5000	USD 860K
Edge 6 GC 5000	USD 960K
LLoyds 6 GC 5050	USD 800K
ATB Financial upsell opportunity	USD 450K

>400K + < 90 days

PERFORMANCE TESTING TOOLSETS AND TOOLSET CHALLENGES

More and More choices in tools, but more challenges in building tests

Tool	Description
Hyperformix	In depth view of mission-critical business applications and supporting platform.
NeoLoad	Java based performance testing tool capable of capturing platform, hardware, network and application performance statistics.
LoadRunner	Robust performance testing tool capable of capturing platform, hardware, network and application performance statistics.
Appvance	End-to-End platform targeted enterprise performance testing application.
Loadster	Desktop based advanced HTTP load testing tool - Local bandwidth based virtual users.
QEngine	Easy-to-Use Remote testing, functionality testing, compatibility testing, stress testing , load testing and regression testing.
Loadstorm	Cost effective and easy to use cloud based performance and load testing tool.
CloudTest	Cloud based load and stress testing for cloud based computers and mobile applications.
LoadUI	Flexible and interactive web testing tool with soapUI functionality.
WebLOAD	Performance test tool used to identify bottlenecks of the website and capacity assessment.

Cost	Small	Large
High	- Multi Protocols Support - Robust - Multi User - Multi Site	- Multiple Protocols Support - Robust - In-depth Analysis - Supports Testing tools
Low	- Single User - Limited Protocol Support	- Limited Protocol Support - Remote Test - Multi User

Functionality

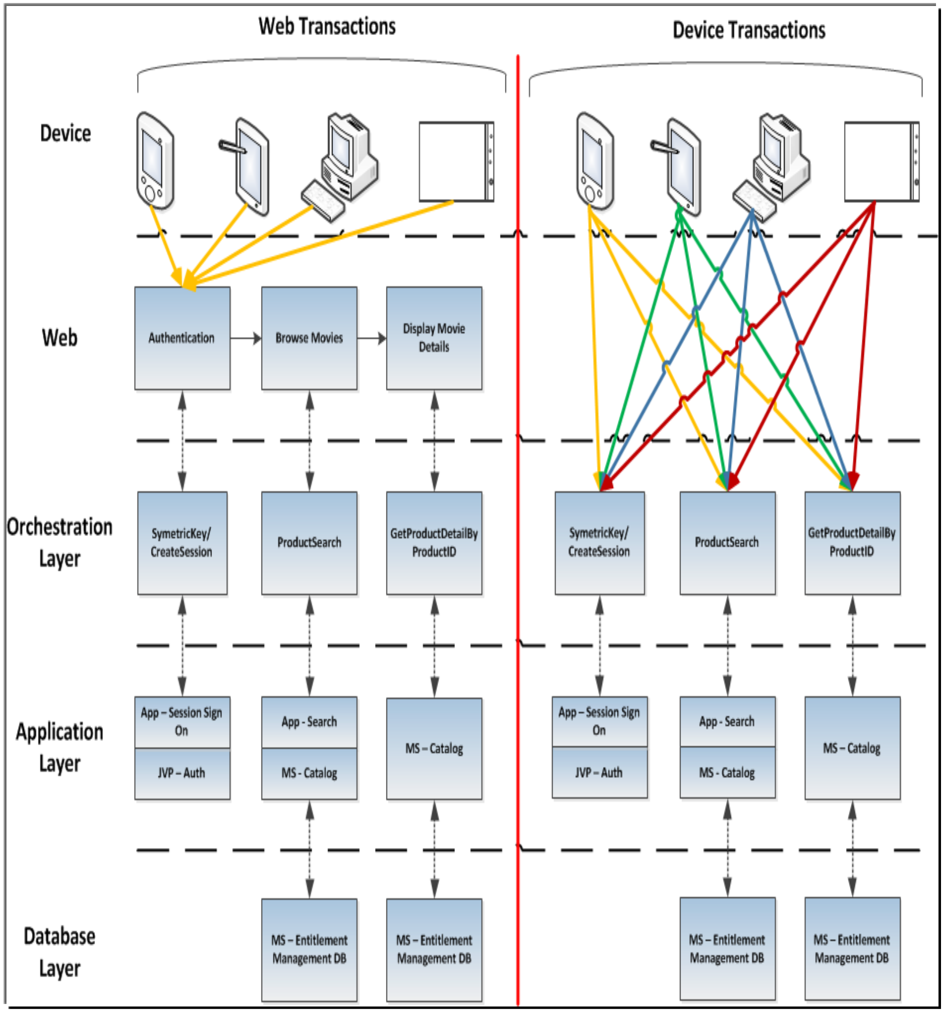
Even with the most robust performance test tools, creating a comprehensive performance-testing environment is challenging

- Constraints inherent in smart systems and applications (E.g. business and database rules)
- Disparities with various protocols, services, and integration systems
- Challenges of pin pointing bottlenecks in highly complex systems

PERFORMANCE TESTING CHALLENGES

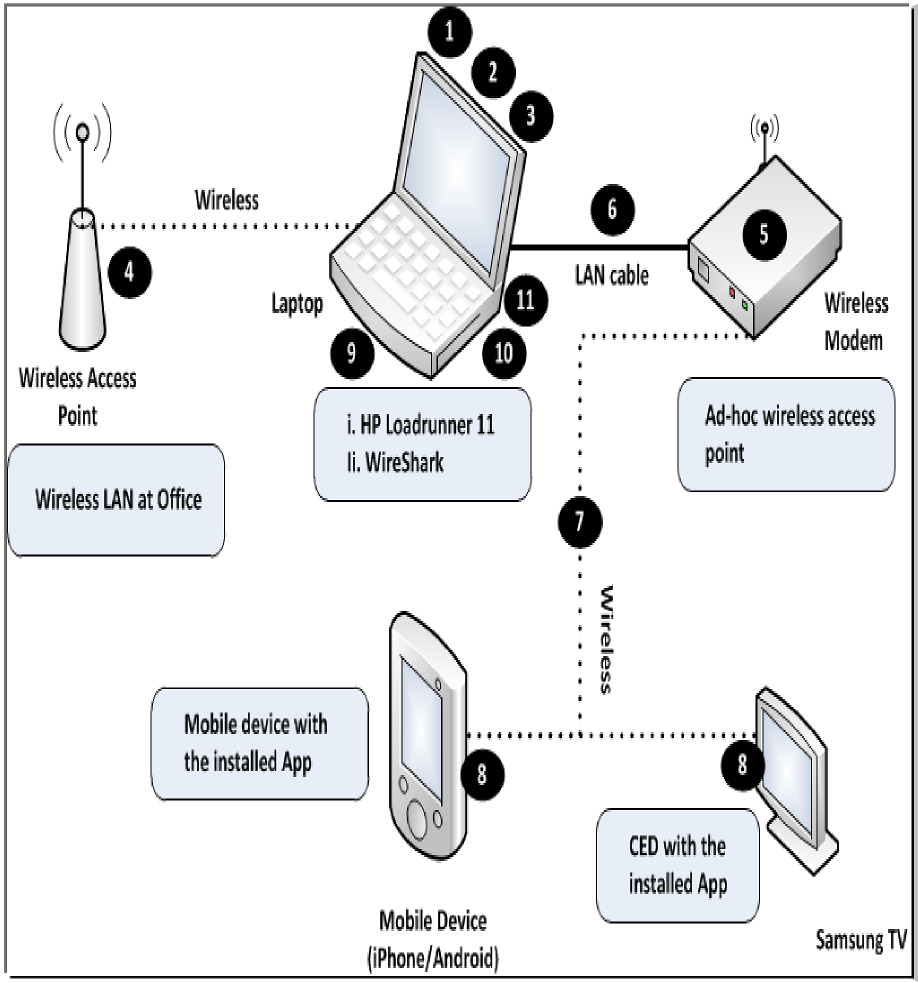
Load Testing Across Multi-Media Access Points

Some Challenges of Developing A Performance Test Approach
Not many performance tools provide a utility for capturing traffic data (at least no utility that is very interactive and intuitive)
Unstable emulators crashed during scenario capture
Scripting and conversion requires a high number of manual adjustements subject to human error
No proven approach for newer protocols, technologies, and development methodologies
The Challenges of Developing An Approach For Capturing Multi Access Point Network Traffic Data



UNIQUE LOAD SCRIPT CAPTURING AND TUNING METHODS

Key to a multi-channel load testing solution is using innovative methods for capturing traffic with Wireshark tool and converting http(s) traffic into LoadRunner scripts.



High Level Setup Steps

1. Install HP LoadRunner 11 Patch 4
2. Install latest WireShark 1.6.7
3. Add project applicable security certificates to SSL protocol in WireShark (Note: Necessary only if using https traffic)
4. Connect laptop to wireless access point
5. Acquire wireless modem
6. Connect wireless modem with laptop via LAN cable
7. Connect mobile device with the ad-hoc wireless access ponit
8. Execute scenario on mobile device and monitor traffic in WireShark
9. Save captured traffic as a *.pcap file
10. Import *.pcap file in LoadRunner using ‘Network Analyzer’ tool
11. Follow necessary steps duing import and a LoadRunner script is generated based on captured traffic

PERFORMANCE TESTING TOOLSETS AND TOOLSET CHALLENGES

Customized Load Script Capturing and Tuning Methods

Challenges	Mitigation Approach
Unstable Code: • Scripts fail due to unstable functional issues • Results varied from run to run	• Setup and run small volume test to validate functionality before any load
Frequent Builds: • Frequent build releases caused LoadRunner re-scripting issues and schedule	• Limit the scenarios for performance testing and re-testing • Update scripts during deployments based on the latest release notes from development
Data : • Data generation limited due to system ability • Frequent environment issues caused data creation script to fail	• Generate data throughout the day and overnight with small stable load.
Schedule: • Dynamic changes to scripting and execution schedules caused by code instability and functional defects	• Maintain close communication with all impacted teams to adjust plan accordingly.
Environment: • No dedicated load test environment • No configuration management plan to maintain environment integrity.	• Coordinate load test activities with functional team • Maintain environment spreadsheet to track downtime, current/future changes, and any open issues.
Scripting: • With so many new features in the application, numerous and different scripting methods finding the correct tool which will correctly encrypt scripts is challenging.	• Setup proxy with hub to allow device traffic to flow through the scripter’s laptop to capture http traffic while executing the test case on target system device.

HOW DOES THIS RELATE TO QUALITY ASSURANCE?

Performance Testing: A critical part of the testing process:

Needs to be part of the SDLC

Needs to be planned, budgeted, and executed

- ➢ Evaluate Performance Testing and Analysis just like you evaluate the rest of your Testing Functions

Do the right things, and do them right

- ➢ Don’t execute and review performance testing just to “jump through a hoop”
- ➢ Instead, execute and review performance testing to confirm the quality of the application
- ➢ Try to stress and strain the application before it goes into production!

AVOIDING COMMON PERFORMANCE TEST PLANNING PITFALLS

Performance Testing PITFALLS:

Categories:

- ➢ Understanding your Workloads ✓
- ➢ Application Releases and Test Data
- ➢ Environments
- ➢ Interfaces

APPLICATION RELEASES & TEST DATA:

A fully functioning application, along with production-like test data, may not always be deployable in the performance testing environment

Is the latest application build stable?

- The latest build with the latest functionality may be the least stable build

Does the latest application build contain all key functionality?

- If not, additional performance testing will need to be conducted

We need to avoid status reports like the following:

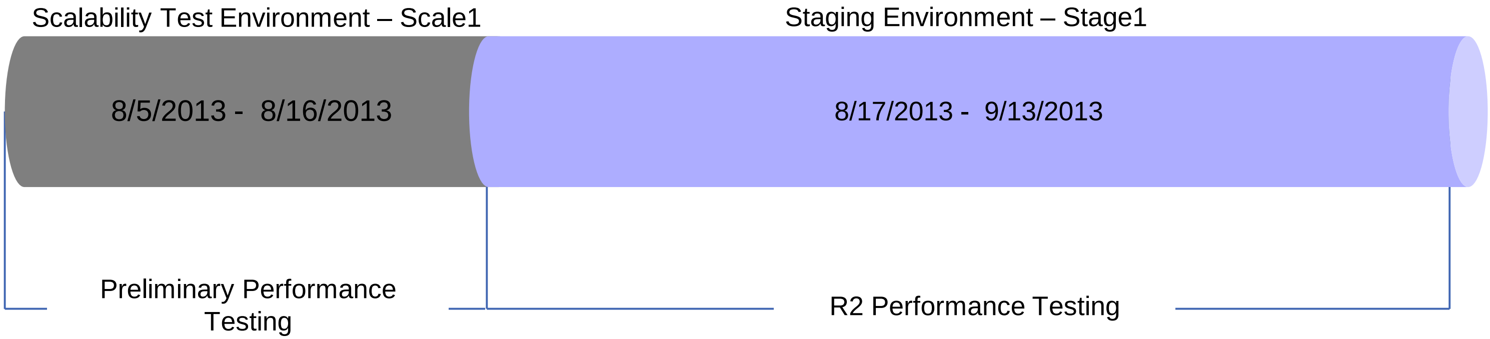
> "……Across the entire *NewProd* solution, the project lacks an effective, ongoing performance tuning process. Timely and long-lasting resolution of application subsystem, infrastructure, environment, and data problems has been difficult to achieve. This has negatively impacted the stability of the *NewProd* system under test and the stability of the environment in which the *NewProd* system runs….."

APPLICATION RELEASES & TEST DATA:

A fully functioning application, along with production-like test data, may not always be deployable in the performance testing environment

Examples of Scheduling Issues and Processes:

Performance Testing – Promoting Builds for Performance Testing


Scalability Test Environment – Scale1
Staging Environment – Stage1
8/5/2013 - 8/16/2013
8/17/2013 - 9/13/2013
Preliminary Performance Testing
R2 Performance Testing

8/5/2013 – 8/8/2013

1. First round of performance tests on Build#20
2. Performance defects will be logged for Build#20
3. Scripts will be updated for Build#21 in SIT and prepared for SCT

8/12/2013 – 8/16/2013

1. Second round of tests will happen on Build#21
2. Performance defects will be logged for Build#21
3. Scripts will updated for Build#22 in SIT and prepared for Staging

8/7/2013 – 8/13/2013

1. Shakeout the environment
2. Load test data and enrollment client records in preparation for testing (target 1.5 million)

8/17/2013 – 8/30/2013

1. First round of performance tests on Build#22
2. Performance defects will be logged for Build#22
3. Scripts will be updated for Build#23 in SIT and tested in SCT

9/3//2013 – 9/13/2013

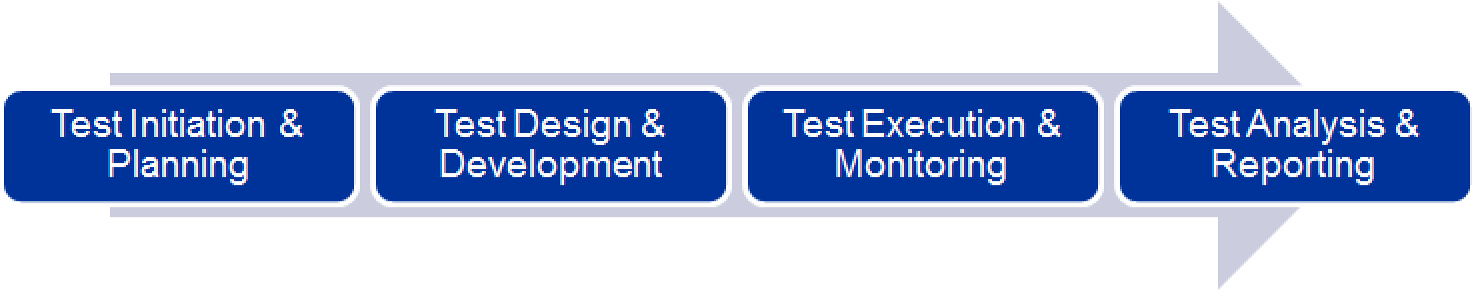
1. Second round of performance tests on Build#XX
2. Validate performance for Build#XX
3. Take additional defects depending on critical performance defects fixed

Additional Performance Testing post Go-live is TBD

Testing Large-scale Distributed Systems

SIMPLE CASE:

➢ Standalone Performance Testing Lifecycle – All the stages of testing go through the below phases sequentially



Test Initiation & Planning

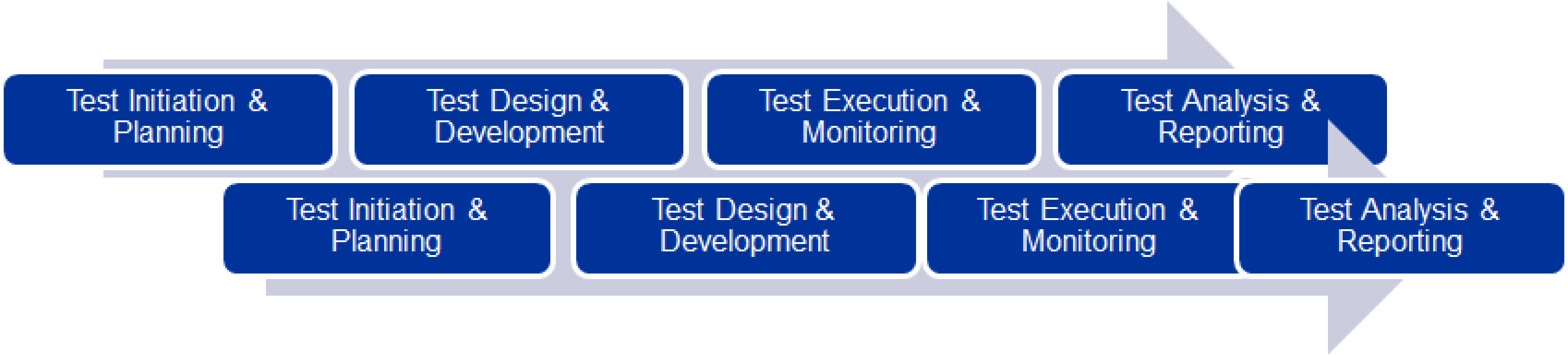
Test Design & Development

Test Execution & Monitoring

Test Analysis & Reporting

➢ Why doesn't it always work well for Large Scale Distributed Systems?

- Agile releases - Developers may be working continuously on other releases
- Multiple phases & stages of testing need to run in parallel
- One external agency may be ready to start testing sooner than the other



Test Initiation & Planning

Test Design & Development

Test Execution & Monitoring

Test Analysis & Reporting

Test Initiation & Planning

Test Design & Development

Test Execution & Monitoring

Test Analysis & Reporting

APPLICATION RELEASES & TEST DATA:

Production-like test data may not always be deployed in the performance testing environment

Performance Test Data Issues

Data Issues – Basics:

Performance testing generally requires at least three types of data - 1) data required by the scripts for execution, 2) the data required in various databases and files used by the application, which constitutes the data volume, and 3) User-Specific Data.

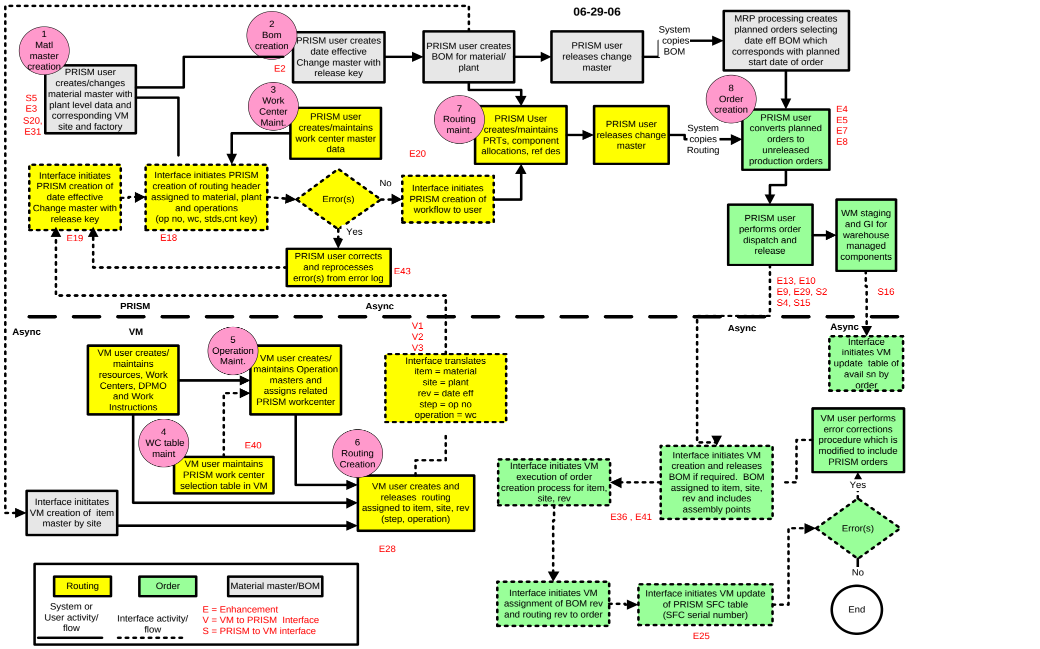
Constraints need to be identified and the performance team needs to prepare the data accordingly.

Data Issues – Data Creation:

Some application data is completely valid only if the data is created by the application itself – there can be downsides with artificially manufacturing data (RI issues)

PRISM/VM Production - Create and Release

06-29-06


1
Matl master creation
PRISM user creates/changes material master with plant level data and corresponding VM site and factory
S5
E3
S20,
E31
2
Bom creation
PRISM user creates date effective Change master with release key
E2
PRISM user creates BOM for material/ plant
PRISM user releases change master
System copies BOM
MRP processing creates planned orders selecting date eff BOM which corresponds with planned start date of order
3
Work Center Maint.
PRISM user creates/maintains work center master data
7
Routing maint.
PRISM User creates/maintains PRTs, component allocations, ref des
E20
PRISM user releases change master
System copies Routing
8
Order creation
PRISM user converts planned orders to unreleased production orders
E4
E5
E7
E8
Interface initiates PRISM creation of date effective Change master with release key
E19
Interface initiates PRISM creation of routing header assigned to material, plant and operations (op no, wc, stds,cnt key)
E18
Error(s)
No
Yes
Interface initiates PRISM creation of workflow to user
PRISM user corrects and reprocesses error(s) from error log
E43
PRISM user performs order dispatch and release
WM staging and GI for warehouse managed components
E13, E10
E9, E29, S2
S4, S15
S16
PRISM
Async
Async
VM
V1
V2
V3
Async
Async
VM user creates/ maintains resources, Work Centers, DPMO and Work Instructions
5
Operation Maint.
VM user creates/ maintains Operation masters and assigns related PRISM workcenter
Interface translates
item = material
site = plant
rev = date eff
step = op no
operation = wc
Interface initiates VM update table of avail sn by order
4
WC table maint
VM user maintains PRISM work center selection table in VM
E40
6
Routing Creation
VM user creates and releases routing assigned to item, site, rev (step, operation)
E28
Interface inititates VM creation of item master by site
Interface initiates VM execution of order creation process for item, site, rev
E36 , E41
Interface initiates VM creation and releases BOM if required. BOM assigned to item, site, rev and includes assembly points
VM user performs error corrections procedure which is modified to include PRISM orders
Yes
Error(s)
No
End
Interface initiates VM assignment of BOM rev and routing rev to order
Interface initiates VM update of PRISM SFC table (SFC serial number)
E25
Routing
Order
Material master/BOM
System or User activity/ flow
Interface activity/ flow
E = Enhancement
V = VM to PRISM Interface
S = PRISM to VM interface

APPLICATION RELEASES & TEST DATA:

All Necessary Production-like test data may not always be available in the performance testing environment

User Data, Valid User ID's:

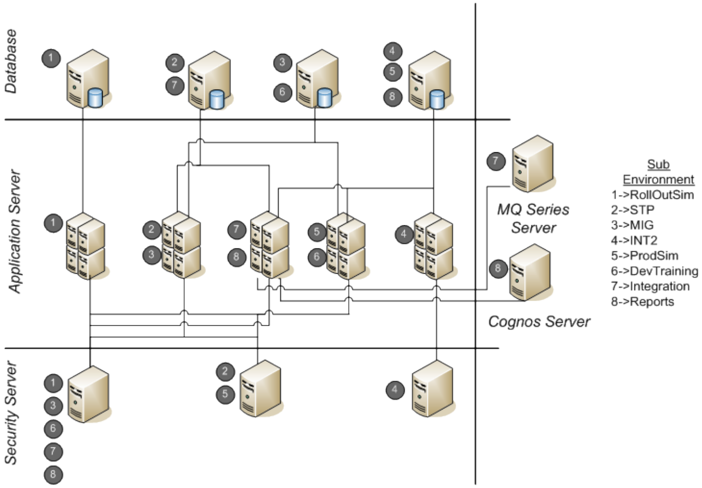
- Don't underestimate the importance of valid UserIDs and passwords

Consumption of Data / Refresh of Data – must involve processes such as:

Activity	Owner(s)
Identify test data needs	Performance Testing Team
Load production like reference data in Performance Environment	Data Conversion Team
Load conversion data in Performance Environment	Data Conversion Team
Define conversion transaction data in Performance Environment	Data Conversion Team/Performance Test Team
Restore transaction data before commencing each performance test execution	Database Administrator
Load conversion data in production environment	Data Conversion Team
Create mock transaction data for a sustainable level to emulate a business day volume in production environment	Data Team
The database state has to be restored to the original state before executing the next set of tests, utilizing a 'Flashback Restore Point' data restore which should insure that the database is restored to original state before commencing the test. This involves: • Taking a snapshot of current database (flashpoint for Oracle) • Restoring the database to its actual state after the test run	Database Administrator

ENVIRONMENTS:

A representative, complete, and dedicated performance testing environment may not always be available when needed:



Hardware

Production Like Hardware

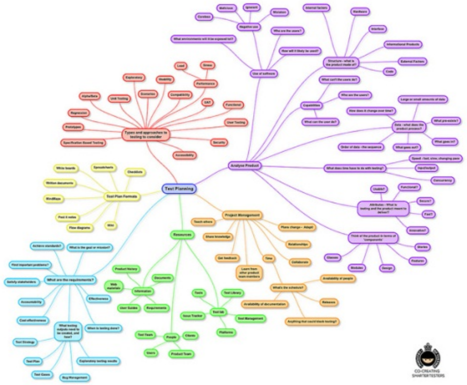
Production Sized Hardware (ideal)

Same models, same versions

Systems Software

Production Equivalent Systems Software (OS, Middleware, Utilities)

Same versions, same releases



Product Licenses

Same versions, same releases

Same constraints and settings

+ **how to overcome challenges**: Ongoing, automated review of CMDB details

ENVIRONMENTS

<u>A representative, complete, and dedicated performance testing environment may not always be available when needed:</u>

Interfaces

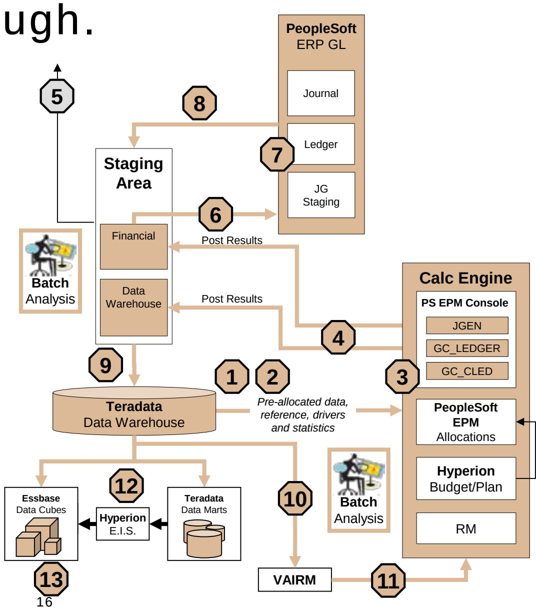
Which interfaces are to be included in performance testing?

- ➢ **<u>Interfaces are complex applications in themselves</u>**: They have internal resources that can become saturated. For example:
  - ➢ <u>Synchronous Interfaces</u>: Processors can become overloaded when trying to satisfy increased volume.
  - ➢ <u>Asynchronous Interfaces</u>: “Holding tanks” for queues (buffers, tables, etc.) can become full if not cleared out quickly enough.

Stubbing needed?

Service Virtualization options?

- ➢ <u>Vendor Offerings</u>:
  - (HP SV, CA-Lisa, IBM Rational Test Virtualization)



NEW PERFORMANCE TESTING CHALLENGES AND HOW THEY RELATE TO QUALITY ASSURANCE

System Performance on Cloud Applications:

No guarantee that these applications will always perform well – NOT EVEN IN AN "AS-A-SERVICE" ENVIRONMENT

Generally, cloud infrastructure and application solutions offer MUCH more cost-effective computing, but are less under our control

Hypotheses:

- Along with the numerous successes and positive momentum with virtualization and the Cloud, the fact remains that there are still unknowns relative to the performance of applications running in these new technology platform environments
- For applications running in virtualized environments, including private/public Clouds, good performance may not always be guaranteed.
- Resources can still become constrained, and some hardware and software cloud providers may have scalability limitations
- Poorly written code will have performance issues regardless of the platform on which it runs

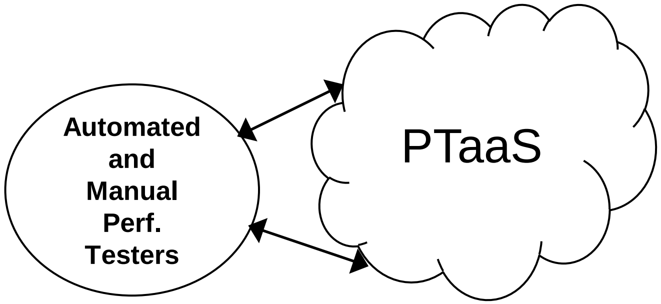
CLOUD ENVIRONMENTS:

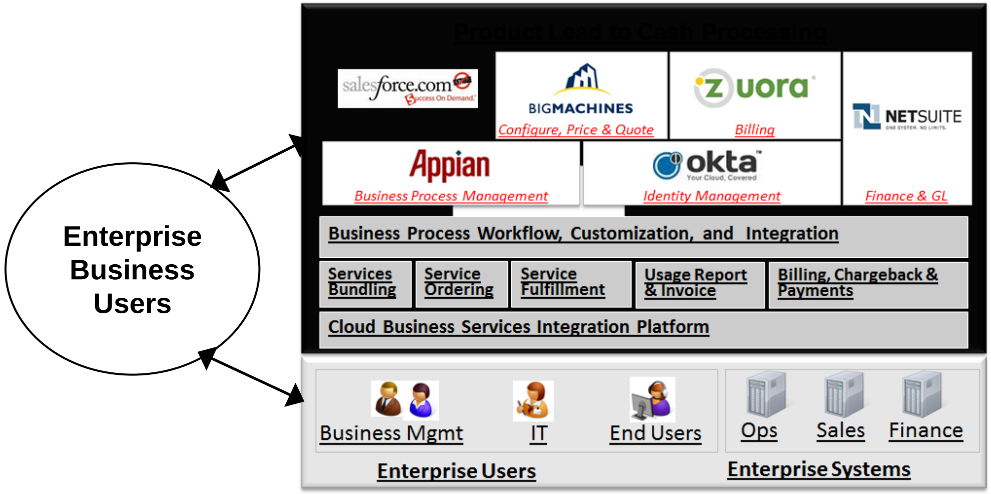
A representative, complete, and dedicated performance testing environment may not always be available when needed:

Performance Testing in Cloud vs. Running Production in Cloud:

- Need to mimic the same capabilities and constraints as production will have
- No guarantee that resources for performance test are comparable to virtual cloud resources for production

Performance Testing Service vs. Commercial Cloud Application





PERFORMANCE TESTING IN CLOUD ENVIRONMENTS:

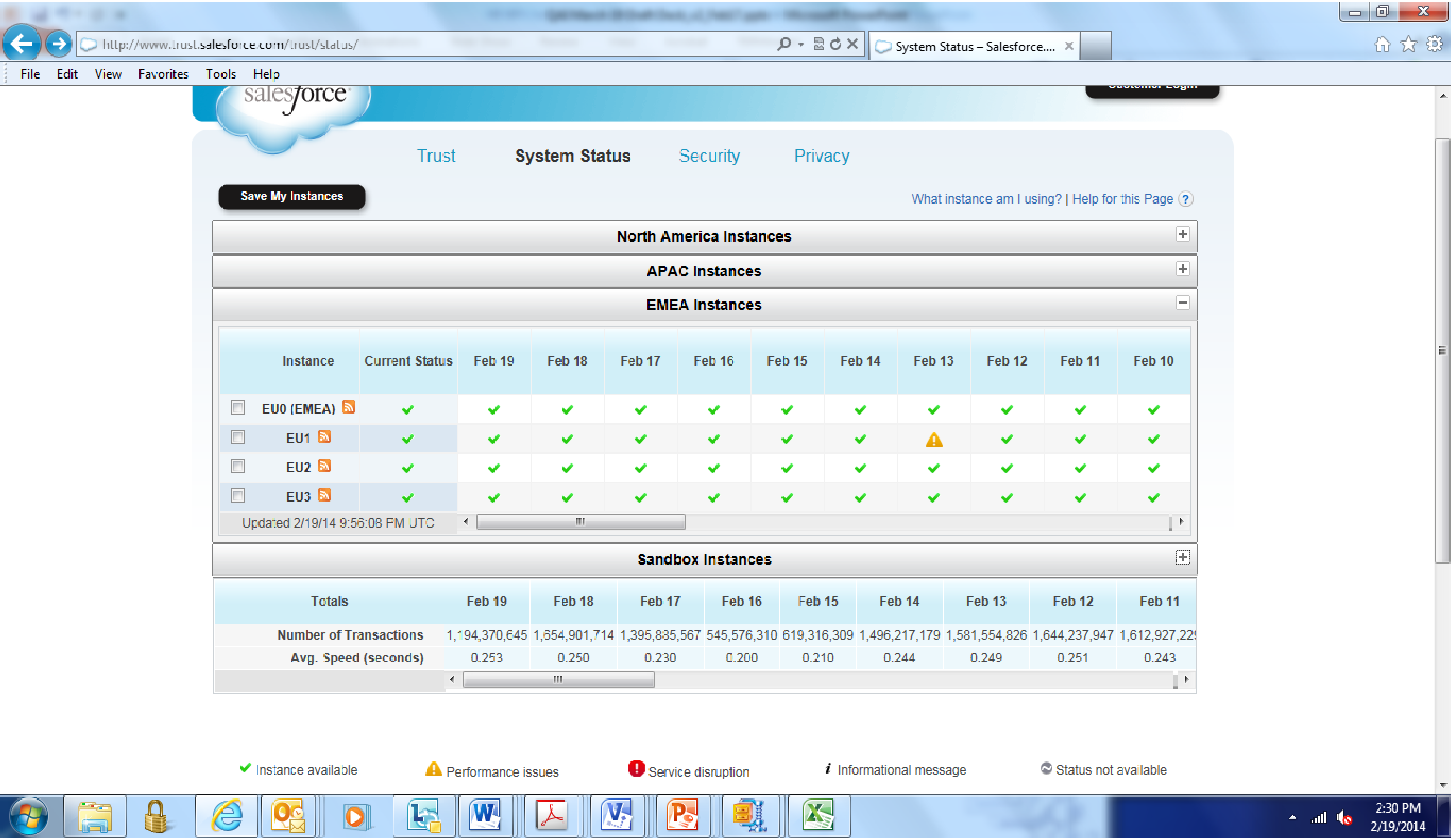
Performance Testing Cloud Applications:

We have observed a variety of performance testing issues and production performance issues for applications hosted in the cloud

- Outsourced HR system
  - Inconsistent versions of HP LoadRunner in different test environments
  - Scheduled performance tests were not matched up will with schedules of required support staff
  - Vendor was performance testing 8X the previous highest volume supported
- Commercially hosted licensing system
  - Stress testing to a failure point was not in the plan
  - Stability / Endurance test was not in the plan
  - Over 10 new 3rd party business partners needed to integrate their functionality into the solution (concurrent with the cloud-hosted release)
- Sales force automation solution
  - Experienced faults in storage tier
  - Experienced application instance outages
  - Had periods of email service degradation
  - But issues were resolved; AND the vendor was transparent with users (see following slide)

Example: Desired Service Level Reporting Provided to Customers

(Source: www.trust.salesforce.com/trust/status) Copyright @ Salesforce.com



	Instance	Current Status	Feb 19	Feb 18	Feb 17	Feb 16	Feb 15	Feb 14	Feb 13	Feb 12	Feb 11	Feb 10
□	EU0 (EMEA)	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
□	EU1	✓	✓	✓	✓	✓	✓	✓	⚠	✓	✓	✓
□	EU2	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
□	EU3	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Updated 2/19/14 9:56:08 PM UTC

Totals	Feb 19	Feb 18	Feb 17	Feb 16	Feb 15	Feb 14	Feb 13	Feb 12	Feb 11
Number of Transactions	1,194,370,645	1,654,901,714	1,395,885,567	545,576,310	619,316,309	1,496,217,179	1,581,554,826	1,644,237,947	1,612,927,22
Avg. Speed (seconds)	0.253	0.250	0.230	0.200	0.210	0.244	0.249	0.251	0.243

✓ Instance available ⚠ Performance issues ❗ Service disruption *i* Informational message ○ Status not available

Copyright © 2014 Deloitte Consulting LLP. All rights reserved.

HAS THERE EVER BEEN A SILVER BULLET?

Excluding the correction of basic mistakes,

I’ve seen one silver bullet in the last five years:

Under Degraded Performance Status, and after doing hundreds of diagnostic activities:

Looked at ULIMIT in LINUX (ulimit open file and user processes in Linux):

- ➢ Discovered a default setting on a database OS:
- ➢ Max user processes setting on a DB Linux OS was at default of 2048
- ➢ Easily changed to 64K (=65535)

```
max user processes   (-u)   65535
```

- ➢ And Everything ran fine!

SUMMARY

Key takeaways from this presentation:

1) Understand your business workloads, their service level targets, and how they make demands on technical resources

2) Understand your peak workload conditions

3) Get your monitoring and performance testing tools in place:

   a) Even with the most robust performance test tools, creating a comprehensive performance testing environment is challenging
   b) Constraints inherent in smart systems and applications challenge the use of performance test tools available
   c) Despite the multitude of performance tools available on the market, disparities in protocols, services, and integration systems challenge our capability to pinpoint bottlenecks in highly complex systems

4) All hands must be on deck for critical performance, load, and stress tests, including infrastructure/development teams on standby for remediation

5) Run through the final gauntlet of stress tests, endurance tests, and confirmation of performance in production before you claim victory

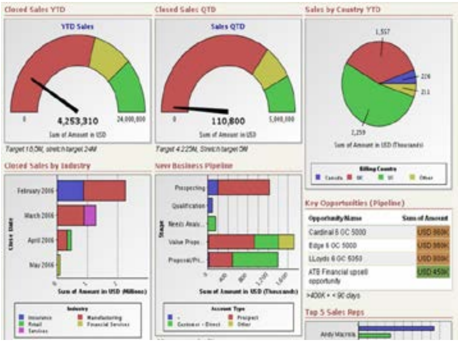
... all in the *modern* context of changing development methodologies, rapidly changing applications, new hardware technologies, increased choices in toolsets, “Anything”-as-a-service, and more pervasive cloud computing options

FINAL “ALIGNMENT” OF CAPABILITIES

JUMPING OVER THE MOST IMPORTANT HURDLES:

Understand your business workloads and their service level targets

Understand your peak workload conditions



Get your monitoring and performance testing tools in place

All hands on deck for critical performance, load, and stress tests, including infrastructure, development teams on standby for remediation

The final gauntlet of Stress Tests, Endurance Tests, and confirming performance in production