



Customers.
Performance.
Loyalty.



Accelerate Testing Cycles With Collaborative Performance Testing

Sachin Dhamdhare

Introduction

 Tools Don't Collaborate

 Typical vs. Collaborative Test Execution

 Some "Collaborative Planning" Ideas

 Extending to Collaborative Analysis & Tuning

 Summary

 Q&A

- Does your organization execute performance tests prior to deploying their Web applications?
- What Tools Do you Use?
- How many people and which groups are involved in the performance testing process?
- Would you describe your recent performance testing activities as successful? Why or why not?

- When Executed Properly, Performance Testing is Critical to Ensuring a Successful Deployment
- Most Often Performance Tests are Not Executed Properly (or Completely) and Must be Run Again
- Critical Application Team Members are not Involved in the Process
- A Collaborative Approach to Performance Testing Will Accelerate Testing Cycles and Yield Better Results

- ❖ Load Testing is a Team Sport ... System Test Requires the Attendance/Input of Experts

- ❖ Current Load Testing Events Lack Keys to Successful Collaboration
 - Leadership in QA
 - Tools that can Support the Effort
 - Egos vs. Evidence Leads to Communication Bottleneck

- ❖ You Can't Adjust Measurements After the Test is Over

➤ Introduction

➤ Tools Don't Collaborate

➤ Typical vs. Collaborative Test Execution

➤ Some "Collaborative Planning" Ideas

➤ Extending to Collaborative Analysis & Tuning

➤ Summary

➤ Q&A

- Load Testing Tools Apply Load and Make Measurements
- People (and their expertise & experience) Find Bottlenecks & Resolve Them
- Collaboration With Experts Resolves Issues Faster & Cheaper
- Key to Successful Collaboration = Access to People & Access to Measurements
- Tools Should Support Collaboration

What Kind of Tools Support Collaboration?

- What are Some of Your Ideas?
- Certainly WebSharing Tools (WebEx, GotoMeeting, etc)
- Web Based Load Testing Tools – Access From Browser

What are Some of the Obstacles You Face?

How Do You Work With Your Developers?



- Introduction
- Tools Don't Collaborate
- Typical vs. Collaborative Test Execution
- Some "Collaborative Planning" Ideas
- Extending to Collaborative Analysis & Tuning
- Summary
- Q&A

- ❖ Common Load Test Practice #1 (stop me if you've heard this one before)
 1. Plan your load test (define scenarios, pass/fail criteria, ...)
 2. Prepare to run tests (create scripts, set up server monitors, prepare test environment, ...)
 3. Run your tests (run specific load scenarios, ramp VUs, monitor throughput,...)
 4. Collect results and identify failures (scenario "x", fails at load level "y", under the conditions "z", ...)
 5. Gather data and send to architect/developers for deeper analysis
 6. Architect/developers dismiss or don't believe results
 7. Get asked to go run your test again
 8. Go back to Step 1

❖ Rule #1: People (developers, architects, DBAs, ...) who are not engaged in the test process will not be engaged or feel ownership for the results.

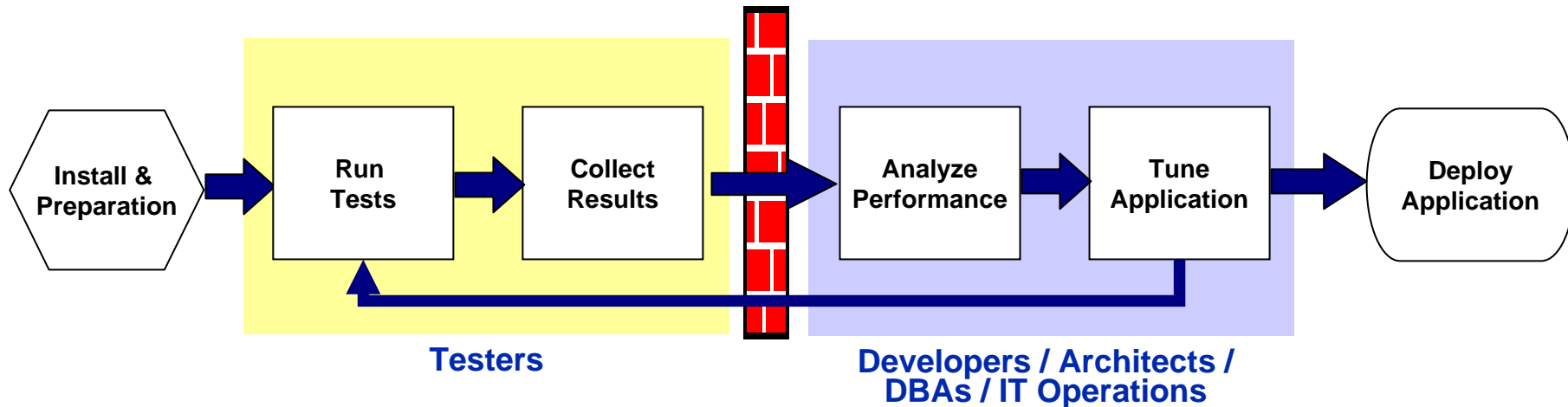
❖ Rule #2: Testing and tuning the performance of enterprise Web applications is a complex and iterative process.

Pay Attention, There is a Quiz on This Later

❖ Rule #3: “Breaking” the application and identifying failures is not the desired outcome; finding bottlenecks, tuning the application and maximizing performance is!

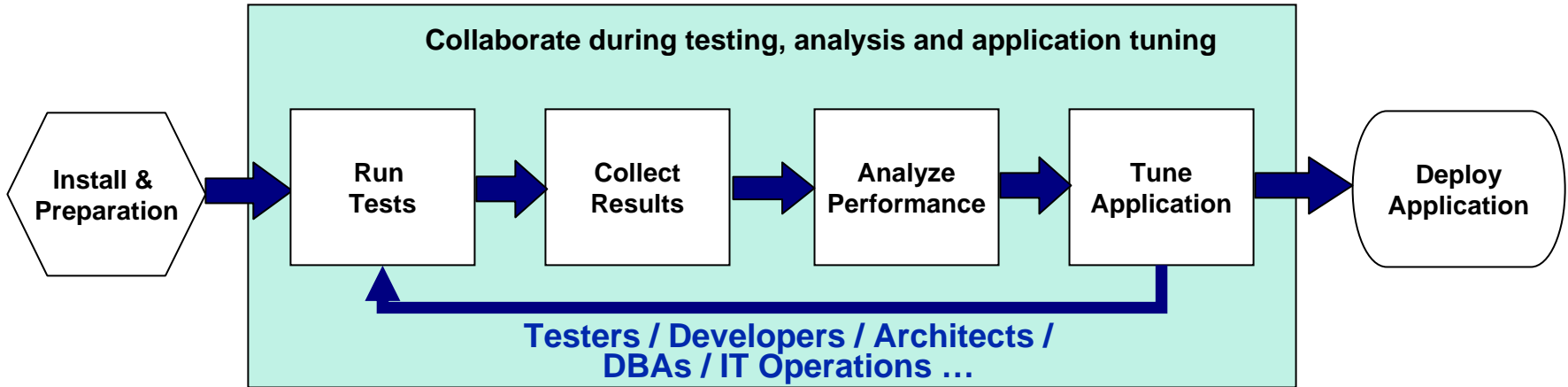
❖ Therefore: Tester, developers, architects and others need to work together to analyze load test results, tune the application and re-run tests to validate changes

Typical Load Test Scenario



Problems with this Approach:

- ❖ People running tests and gathering results are not engaged with those that may be needed to analyze data and identify root cause
- ❖ Analysis & tuning is complex and often requires real time data analysis
- ❖ Its easier to dismiss poor or failing results after the fact than when they are occurring



Advantages:

- Enables real-time performance analysis and tuning by entire team
- Reduces test cycle times and communication delays between test & development
- Yields higher performing applications

👉 If it's that easy, why isn't everyone doing it?

- Don't understand the objectives of the test
 - “What are we trying to prove here? Why is this considered a failure?”
- Lack of trust in the testing tools and results
 - “It can't be the application, it must be the testing tools.”
- Lack of trust in the testers
 - “This can't be right, you must have done something wrong.”
- Feel that it's unnecessary
 - “My Application design is bullet-proof.” –or- “We've added so much extra capacity on server side it doesn't matter.”
- No time to participate
 - “Go away.”

Keys to enabling a collaborative load test process

- Step 1: Establishing Credibility
 - Gain and then demonstrate a clear understanding of the application and its requirements (functional and performance). Know what you are talking about
- Step 2: Gaining Consensus on the Test Process
 - Clearly define the goals of the test, expected results and pass/fail criteria and make sure everyone is on board. Take the lead in getting this done!
- Step 3: Getting Buy-In on the Tools & Methods
 - Identify the tools you will be using to test, provide any relevant documentation, and if necessary do a demo in advance to show how things will work

Keys to enabling a collaborative load test process (continued)

- Step 4: Establishing the Value of the Test
 - Outline how you plan to target specific critical areas of the application and how you will identify application and server capacity issues combining both Virtual User load and server-side monitors
- Step 5: Making it Convenient
 - Provide real-time status of tests in progress, enable real-time viewing of results, and make it easy for people to participate (IM, Web share, Web enabled tools)

- Introduction
- Tools Don't Collaborate
- Typical vs. Collaborative Test Execution
- Some "Collaborative Planning" Ideas
- Extending to Collaborative Analysis & Tuning
- Summary
- Q&A

- Financial Company with a Critical Online Application & Limited Test Window
- Complex Application (Tax Returns) With Over 800 .asp Pages – Too many to test!
- Several Permutations of Tax Documents. Too many transactions!
- In the End, Only 11 “Loops” ... A couple of transactions

- Key questions that need to be answered before you test (the 4Ws and 1H of Test Planning)
 - What are we testing?
 - Why are we testing it?
 - How are we testing it?
 - When and Where will the tests be run?
 - Who needs to be involved?

- These questions cannot be answered in isolation – a collaborative team effort is needed
 - Determining the answers to these questions is critical to success

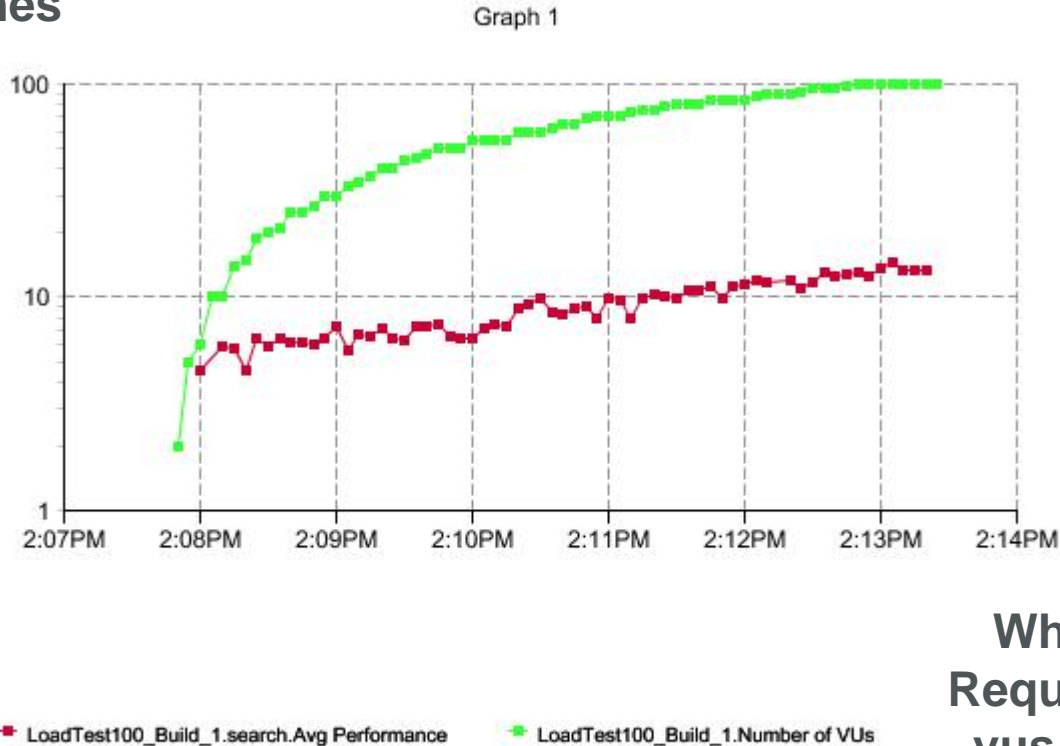
What are we testing?

Need	Who to Work With	Usage
Application requirements	Product Managers, Business Analysts, Development Manager, Customers	Determining what to test, expected results, pass/fail criteria
Common use cases, usage scenarios	Product Managers, Business Analysts	Script creation, defining test scenarios
Application architecture / design specification	Development Manager, Architects, Developers	Performance analysis, tuning, test setup

How About This Example? Is This Acceptable Performance?

What if I said the
Requirements are 100
vus and under 10s
Response Times

What Is Your Answer?



What if I said the
Requirements are 50
vus and under 10s
Response Times

Why are we testing it?

Need	Who to Work With	Usage
Company needs / objectives / goals	Product Manager, Application Owner	Understanding service level agreements, customer expectations
Test objectives / goals	Development Manager, QA Manager, Architects/ Developers	Creating test plan, defining test scenarios, performance analysis

How are we testing it?

Need	Who to Work With	Usage
Test plan	QA Manager, Performance Engineers, Other Testers	Guide the testing effort, prioritize testing activities
Test hardware	QA Team, IT/Operations	Load controller, load test agents to run VUs, performance monitors
Testable application	Development Manager, Developers	Production environment or representative staging environment to test against; stable!

When and Where will the tests be run?

Need	Who to Work With	Usage
Project schedule	Project Manager, Development Manager	Identify when testing will occur, understand overall project/dev milestones
Detailed test schedule	QA Manager, Project Manager	Further define test scenarios, outline specific timeframe for tuning / retests
Test Lab	QA Manager, IT/Operations	House the test hardware used to run test, host application under test
Contingency plans	Development Manager, Developers	Plan for (un)expected development delays, prioritize test efforts

Who needs to be involved?

Team Member	Role
Performance Testers / QA	Create test plan, design tests / scripts, execute tests, gather results
Architects & Developers	Troubleshoot application failures, analyze application performance / business logic
DBA(s)	Troubleshoot database issues, analyze DB performance
IT / Operations	Troubleshoot hardware and network issues, aid in capacity planning
Project Manager	Manage the overall development / test effort, create project schedule, manage milestones
QA Manager	Manage the test team, assign testers, enable collaboration with development
Development Manager	Manage the development team, allocate resources to test effort, help determine goals

👉 Taking the Lead Isn't Easy

👉 Outside Consultants & Tool Vendors Do This All the Time! Why?

- We Get Paid on Results
- We Know it Gets the Job Done
- Internal “Walls” Don't Bother Us
- You Can Do the Same Thing!

👉 Know Your Subject, Know Your Limits ... People Like to Help, Just ask them!

- Test to Tune Application not Break (Developers Resist Idea that it “Broke”)
- Talk at Entire System Level and Let the Test Point Out the Component

- Introduction
- Tools Don't Collaborate
- Typical vs. Collaborative Test Execution
- Some "Collaborative Planning" Ideas
- Extending to Collaborative Analysis & Tuning
- Summary
- Q&A

- Remember Rule #3 (I did mention the Quiz earlier)
 - Rule #3: “Breaking” the application and identifying failures is not the desired outcome; finding bottlenecks, tuning the application and maximizing performance is!
 - Not only the True Goal but it Helps Everyone to Think This Way
- So How Do We Do That?
 - Get the right people, working collaboratively, to get the job done (Collaborative Test Planning and Execution)
 - Give them access to the data they need to be successful (Collaborative Performance Analysis and Tuning)

Three Ways to Look at Performance Diagnostics for Analysis and Tuning

- View 1: Analyze Performance and Response Times from the End-User Perspective
- View 2: Analyze Performance of Your Back-end Infrastructure
- View 3: Analyze Performance and Response Times of Server-side Transactions
- Each “View” provides another layer of data that will be relevant to different team members. Each “View” Shows A Symptom Diagnosed by the Next.

➤ View 1: Analyze Performance and Response Times from the End-User Perspective

- **Pages Represent Business Step & Tier**
- **Provides standard load test metrics for script/page/object level response times, throughput, concurrency, etc.**
- **Easiest to relate to and reflect actual end-user experience**
- **Load scenarios and individual scripts can be designed to identify specific application bottlenecks**

➤ Who needs this data?

- **Testers: To report test results and relate them to application requirements and test objectives**
- **Development Team: To identify general performance limitations and start troubleshooting bottlenecks**
- **Project Managers / Others: To begin understanding how the application will behave once deployed to production.**

View 2: Analyze Performance of Your Back-end Infrastructure

- **Provides Key Performance Indicators (KPI) for Infrastructure**
- **Allows you to understand server-side resource utilization and limitations**
- **Lets you target individual servers or network components, generally without having to install anything (agent-less)**

Who needs this data?

- **Testers: To provide a complete view of application performance from both client and server side**
- **Development Team: To more easily trace performance issues to specific tiers / components in the application infrastructure**
- **IT Operations Team: To understand server and network capacity issues and size accurately for production deployment**

- View 3: Analyze Performance and Response Times of Server-side Transactions
 - Provides profiling-level data to view actual transaction response times of individual objects, methods and database queries
 - Allows you to understand where time is being spent within the individual server component
 - Lets you target actual business logic and identify problems in the code impacting performance
 - Use in conjunction with Level 1 and 2 metrics to correlate client and server side performance degradation to application business logic
- Who needs this data?
 - Architects: To identify issues in the design of the business logic that may be limiting performance
 - Developers: To identify issues in specific functions or methods that may be creating performance bottlenecks
 - DBAs: To identify queries or commands that may be optimized to reduce data access times and improve performance

- Providing data at different levels of granularity will aid team in quickly analyzing bottlenecks
- Giving individuals data specific to their areas of need or interest will also drive greater participation and buy-in
- Bringing the team together to collaboratively analyze data and make tuning recommendations will yield better performing applications in less time

- Introduction
 - Tools Don't Collaborate
 - Typical vs. Collaborative Test Execution
 - Some "Collaborative Planning" Ideas
 - Extending to Collaborative Analysis & Tuning
- Summary
 - Q&A

- Testing and tuning the performance of enterprise Web applications is a complex and iterative process
- Tester, developers, architects and others need to work together to collaboratively plan the testing effort, execute tests, and analyze results to tune performance
- Collaboration is critical to the success of the team and each member needs to be engaged in the process
- Collaborative load testing can ultimately yield better performing applications in less time



Customers.
Performance.
Loyalty.

Thank You!