A Formal Performance Tuning Methodology: Wait-Based Tuning

Steven Haines
Quest Software

Agenda

- State of the Market
- Performance Testing Process
- Performance Tuning Process
- Load Testing Methodology
- Wait-Based Tuning
  - ✓ Identifying Wait-Points
  - ✓ Tune Backwards

State of the Market

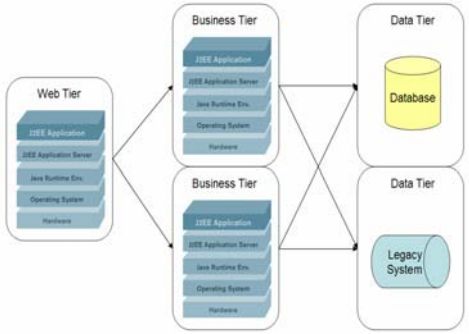
- *Forrester reported that among companies with revenue of more than $1 billion, nearly 85% reported experiencing incidents of significant application performance degradation. Respondents identified the application architecture and deployment as being of primary importance to the root cause of application performance problems.*

The Cost of Failure

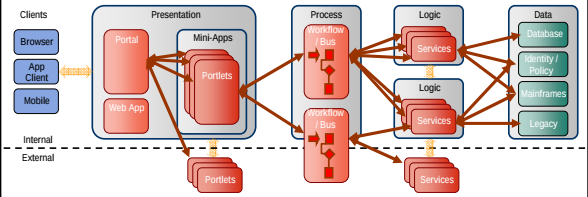
- Business to Consumer
  - ✓ Site abandonment = lost revenue
- Business to Business
  - ✓ Damaged business relationships = lost opportunity
- Internal
  - ✓ Loss of organizational efficiency
  - ✓ Slower time-to-market
  - ✓ Loss of competitive edge

Java EE Layered Execution Model

Distributed Java EE Layered Execution Model



Complexity is on the rise



This IT Objective…	Has This Impact…	Creating These IT Challenges
- Build a real-time, adaptive infrastructure	- Application environments are moving along an architectural continuum: **Client-Server → 3-Tier → SOA**	- Integrating existing apps - Changes in existing apps - Performance issues - Application consolidation - New application deployment

7

Performance Testing Process

- The most effective way to ensure the performance of your application is to adopt a Performance Testing Methodology that spans the entire development lifecycle

8

Performance Tuning Process

- Load test
- Tune container
- Identify application bottlenecks
- Iterate

9

Load Testing Methodology

- Your preproduction tuning efforts are only valuable if the load represents real end-user behavior
  - ✓ Referred to as *balanced and representative service requests*
- Different process for new and existing applications

10

Load Testing an Existing Application

- Learn what your users are doing
  - ✓ Access Logs
  - ✓ End User Experience Monitor
- Construct load tests to reproduce the top 80% of user actions

11

Load Testing a New Application

- Estimate
  - ✓ Well-defined use cases are essential
  - ✓ Establish balance between application technical and business owners
- Validate
  - ✓ Validate usage patterns against expectations
- Reflect
  - ✓ Post-mortem analysis of estimations
  - ✓ Learn more about your users

12

Load Testing Process

- Ideally mirror production
  - ✓ Problem = $$
- Scale down strategies
  - ✓ Scale down number of machines, but same class
  - ✓ Scale down the class of machines
  - ✓ Scale down both the number and class of machines

13

Wait-Based Tuning

- Tuning against performance ratios and percentages can be a laborious and unfruitful task
  - ✓ Difficult to assign priority to tuning parameters
  - ✓ Are you really helping your users?
- Instead ask, where are my requests waiting?

14

Wait-Based Tuning Evolution

- Oracle 9 database tuning theory
  - ✓ Where are queries waiting?
- IBM WebSphere tuning theory
  - ✓ Four areas
    - Web Server
    - Web Container
    - EJB Container
    - Database connection pools

15

Wait-Points

- A *wait-point* represents any place in your application that a request can wait

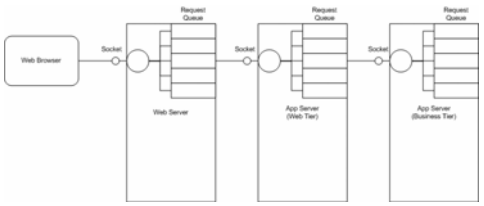
16

Wait-Point Architectural Analysis

- Wait-points need to be identified in the context of *your* application architecture

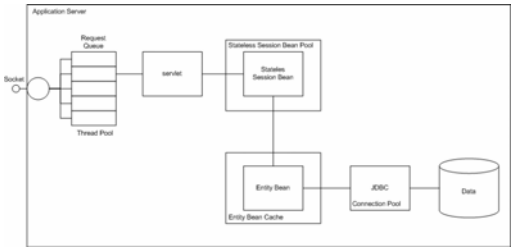
17

Tier Wait-Points



18

Technology Wait-Points



Common Wait-Points

- Web server thread pools
- Application server or tier thread pools
- Stateless Session Bean and component pools
- Caching infrastructure
- Persistent storage or external dependency pools
- Messaging infrastructure
- Garbage collection

Tune Backwards

- It is better to queue requests in a business logic-lite tier to minimize the impact on the business tier
  - ✓ If a request has a Web server thread and it is not ready for processing, why obtain an application server thread and database connection?
  - ✓ Instead, the request should wait at the Web server

Process

- Open all wait-points and load test until a wait-point resource saturates
- Scale down the limiting wait-point until it no longer saturates
  - ✓ This identifies the capacity of the wait-point's resource
- Tune other wait-points to only feed enough load to the limiting wait-points

For Example

- If an application server instance can only service 50 simultaneous database requests, then you want to send through only enough requests to generate at most 50 database requests
- Any additional requests will simply queue up at the database

Bringing It All Together

- Analyze architecture and identify wait-points
- Open all wait-points
- Generate balanced and representative load
- Identify limiting wait-point's saturation point
- Tighten wait-points to facilitate only the maximum load of the limiting wait-point
- Force pending requests to the Web server
- If load is too high, setup cutoff point and redirect to a "Try again later" page

Summary

- Applications are not meeting their performance criteria in production
- The solution is to
  - ✓ Implement performance testing across the development lifecycle
  - ✓ Tune your container according to the Wait-Based Tuning Methodology
- Wait-Based Tuning
  - ✓ Design proper load tests
  - ✓ Identify wait-points
  - ✓ Tune backwards

25