



| IBM Software Group

## Discovering business application services, featuring IBM WebSphere Application Server ND V7.0

***Performance and Runtime provisioning***



@business on demand.

© 2008 IBM Corporation  
Updated September 28, 2009

# Overview

- V7.0 Performance
  - ▶ Shared class cache
  - ▶ Compressed References
  - ▶ Performance Figures
  
- V7.0 Runtime Provisioning
  - ▶ Benefits of Runtime Provisioning
  - ▶ How it works

# Section

## ***V7.0 Performance***

## WAS v7 Runtime Performance Achievement

- Optimized runtime performance that decreases footprint, saves energy and cuts costs
- IBM Java SE Version 6 enhancements add scalability and performance
- Runtime provisioning selects only the needed function for memory and space improvements
- Benchmark results lead the industry



# WAS v7 Runtime Performance Achievement

## Addresses a Fundamental Challenge for IT

**Increased Computing Demand**

**Changing Cost Dynamics**

**Data Center Lifecycle Mismatch**

- In the next decade, growth in server shipments will be 6x and 69x for storage – *IBM / Consultant studies*
- Data centers have doubled their energy use in the past five years.<sup>3</sup> - *Koomey, February 2007*
- Per square foot, annual data center energy costs are 10 to 30 times more than those of a typical office building.<sup>2</sup> - *William Tschudi, March 2006*
- US commercial electrical costs increased by 10 percent from 2005-06.<sup>4</sup> - *EPA Monthly Forecast, 2007*
- By 2012 for every \$1 spent on hardware, \$1 will be spent on power and cooling – *IDC, 2007*
- “Eighty-six percent of data centers were built before 2001”<sup>5</sup>
- “Twenty-nine percent of clients identified” data center capability affected server purchases ”- *Ziff Davis*

*Gartner: “The underlying consumption of energy in large data centers ... is likely to increase steadily during the next ten years.”*

# Goal: Leverage Benefits of Java SE 6 Support

## You want ...

- Continued application startup time improvements
- Continued memory footprint reduction
- 64-bit memory and performance equivalent to 32-bit

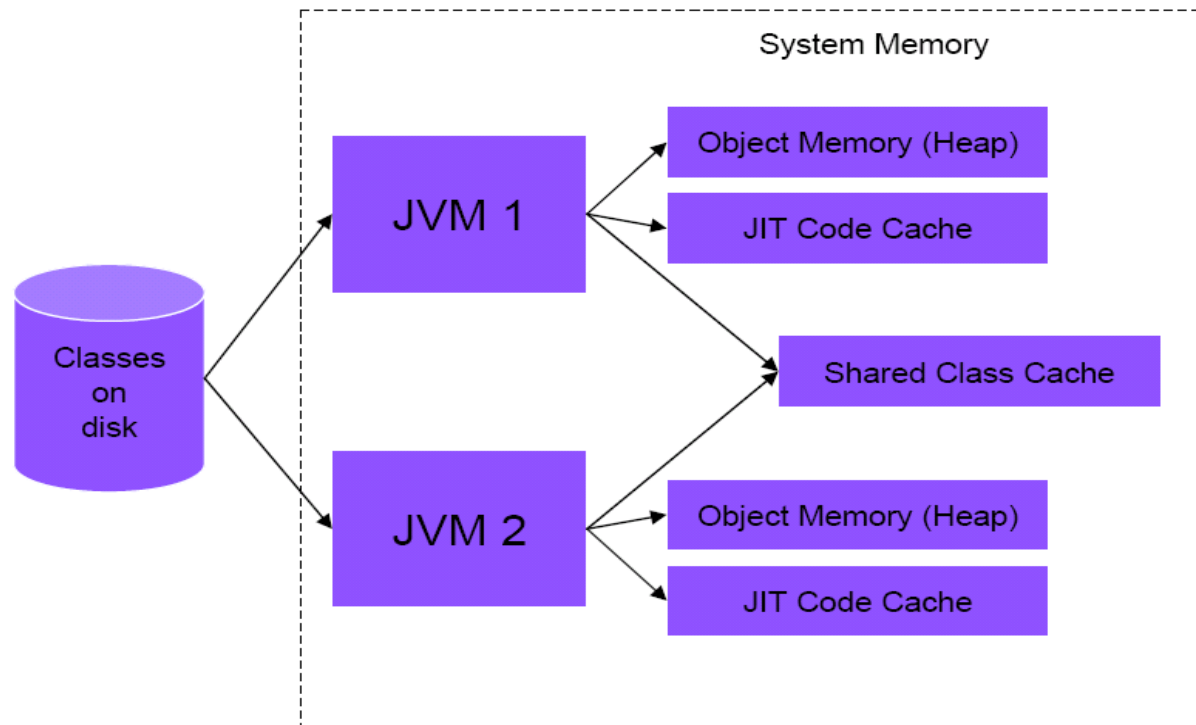
## IBM Solution

- Broad Java SE 6 platform coverage
- Java SE 6 leading implementation
  - ▶ Compressed references allow applications to access a larger heap space without using 64-bit objects
  - ▶ Shared class cache and Ahead-of-time compilation for improved startup time
  - ▶ Support for persistent shared class cache



# Leveraging Benefits of Java SE 6 Support

## WAS v7 Shared class Cache Provides New Performance Gains (continued)



## Leveraging Benefits of Java SE 6 Support

### WAS v7 Shared class Cache Provides New Performance Gains (*continued*)

- With WAS v7 implementation ...
  - ▶ Shared cache information can be written to a memory-mapped file
  - ▶ Initial JVM startup after a reboot will no longer take a performance hit
  - ▶ You can control persistence using command line options
    - Example (enabling persistence): `-Xshareclasses:persistent`
    - Example (disabling persistence): `-Xshareclasses:nonpersistent`
    - Option for cache persistence is enabled by default on all platforms except z/OS®

## Leveraging benefits of Java SE 6 support

### WAS v7 64-bit Provides New Performance Gains

- In 64-bit deployments, each Java memory address reference is 8 bytes, versus 4 bytes on 32-bit deployments.
  - ▶ This results in an increased memory footprint
  - ▶ 60-70% memory growth for the WebSphere Trade 6 application
  - ▶ 32-bit JVM is limited to ~3Gb heap sizes
  - ▶ Need for more Java heap space than available with 32-bit JVM, without full overhead of 64-bit JVM

## Leveraging benefits of Java SE 6 support

### WAS v7 64-bit Provides New Performance Gains (continued)

- IBM Java SE 6 introduces **compressed references** technology to dramatically improve memory footprint on 64-bit platforms
- Use 32-bit heap offsets to reference heap data
  - ▶ Java objects take up less space
  - ▶ Runtime internals translate between 32-bit references and 64-bit references
  - ▶ More Java heap space available without full overhead of 64-bit JVM
  - ▶ For Java heap sizes up to 25GB

## Leveraging Benefits of Java SE 6 Support

### WAS v7 64-bit Provides New Performance Gains (continued)

- Pointer size comparison:

Pointer size	Space	Max heap	Efficiency
31 bits	2 GB	1.3 GB (z/OS)	100%
32 bits	4 GB	1.7 GB (Win) 3.2 GB (AIX)	100%
64 bits compressed	4 to 25 GB	4 to 25 GB	~70-100%
64 bits	16 EB	16 EB	50-70%

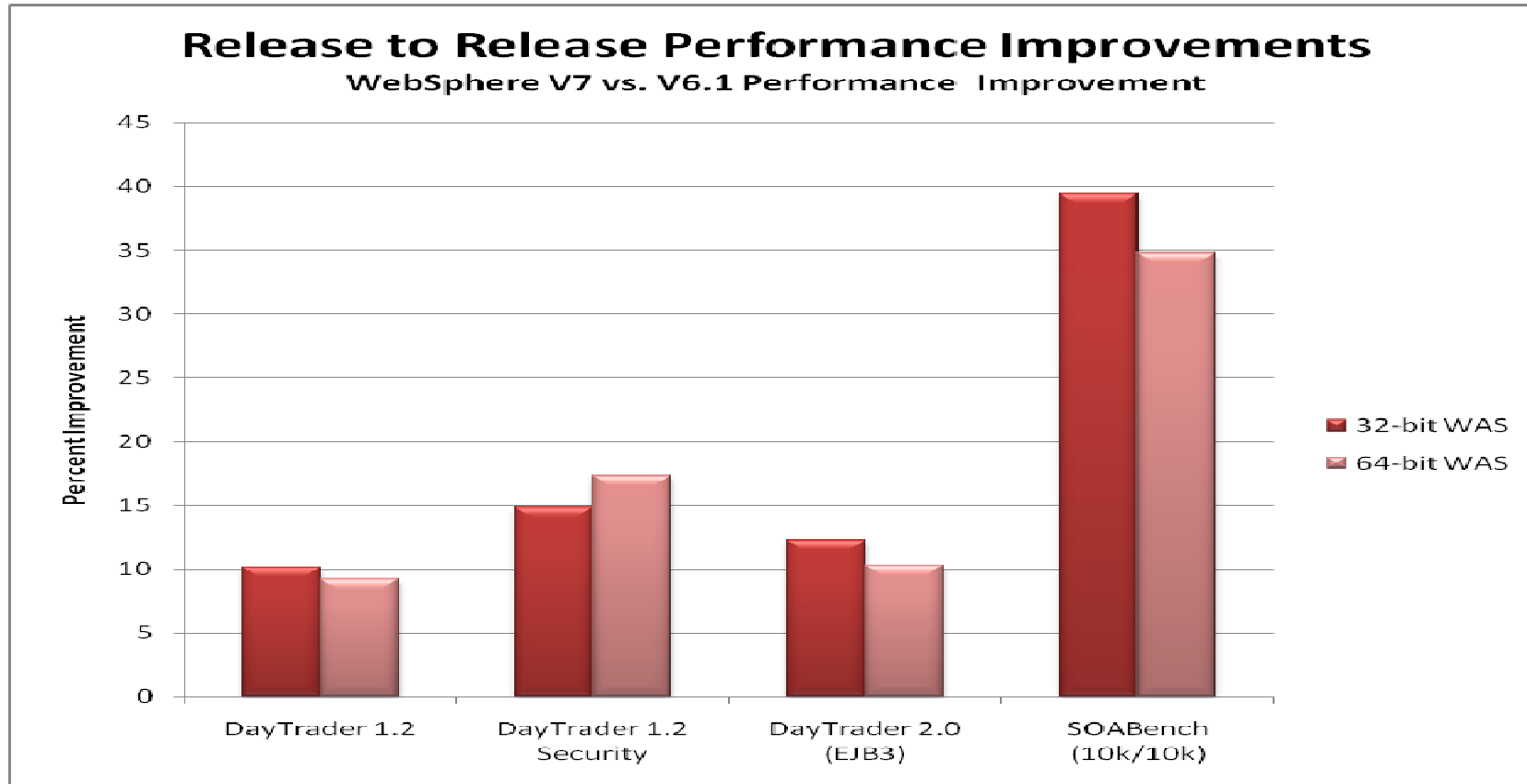
# Section

## ***v7 Performance Figures***

## Review of primary high level areas improved

- ▶ Improved performance with Java 6
- ▶ EJB improvements based on heavily optimized EJB3/JPA runtime
- ▶ Improved secure runtime performance
- ▶ Significantly improved Web Services performance
- ▶ Startup time and memory footprint improvements

## Example performance improvements from the WAS 7 performance report:



## WAS 7 performance improvements for other platforms over WAS 6.1 from internal performance report

Perf Test	DayTrader 1.2 (no security)	DayTrader 1.2 (with security)	DayTrader 2.0 EJB3	SOABench WebServices (Common scenario 10kin10kout)
AIX	10%	15%	>=10%	>=35%
Windows	10%	15%	>=10%	>=40%
Power Linux	10%	15%	>=10%	>=40%
Intel Linux	10%	15%	>=10%	>=40%
Solaris	10%	15%	>=35%	>=30%
Solaris x86 64 bit	10%	25%	>=15%	>=10%
iSeries 64 bit	20%	40%	>=10%	>=70%

# Section

## ***Runtime Provisioning***

## Benefits of runtime provisioning

- Improvements in the amount of time it takes to start an Application Server
- Reduction in the application server memory footprint
- Only a subset of components are started in a running server
  - ▶ Server start time is reduced
  - ▶ Memory footprint is reduced
- When combined with new runtime configuration component an overall memory footprint decrease is projected
- Estimated 10 to 15% improvement in startup time depending on the application set

# Section

## *How it works*

## Runtime provisioning

- Intelligent analysis of application set and server configuration determine the set of components to activate
  - ▶ Administrators and application deployers do not have to modify any processes to take advantage of provisioning
  - ▶ A check box on the server page in the administrative console activates provisioning on a per server basis
  - ▶ Disabled by default

# How to activate runtime provisioning

- In the administrative console
  - ▶ Application servers > serverName page > Start components as needed

The screenshot shows the IBM WebSphere Administrative Console interface. On the left is a navigation tree with categories like Welcome, Guided Activities, Servers, Applications, Services, Resources, Security, Environment, System administration, Users and Groups, Monitoring and Tuning, Troubleshooting, Service integration, and UDDI. The 'Servers' category is expanded to show 'Application servers', 'Web servers', and 'WebSphere MQ servers'. The main content area is titled 'Application servers > server1' and contains a description: 'Use this page to configure an application server. An application server is a server that provides services required to run enterprise applications.' Below this are two tabs: 'Runtime' and 'Configuration'. The 'Configuration' tab is active and shows two columns of settings. The 'General Properties' column includes fields for 'Name' (server1) and 'Node name' (N612Node01), and checkboxes for 'Run in development mode', 'Parallel start', and 'Start components as needed' (which is checked and circled in red). There is also a dropdown for 'Access to internal server classes' set to 'Allow'. The 'Container Settings' column lists various container settings like 'Session management', 'SIP Container Settings', 'Web Container Settings', 'Portlet Container Settings', 'EJB Container Settings', 'Container Services', and 'Business Process Services'. At the bottom, there is a section for 'Server-specific Application Settings' with a 'Classloader policy' dropdown.

## Scenarios

- Examples of scenarios where some components are not activated
- Web Application using Servlets and Java Data Base Connectivity (JDBC)
  - ▶ No EJB, no security, no naming directory components are activated saving start time and memory footprint
- Node Agent, Deployment Manager, z/OS Control Region server, Proxy Server, Administrative Agent
  - ▶ All servers start more components than they need
  - ▶ With Runtime Provisioning enabled only subsets of components are loaded

## Summary

- V7 WebSphere Performance improvements
  - ▶ Improved performance with Java 6
  - ▶ EJB improvements based on heavily optimized EJB3/JPA runtime
  - ▶ Improved secure runtime performance
  - ▶ Significantly improved Web Services performance
  
- V7 runtime provisioning provides performance benefits
  - ▶ Reduced start time of application servers