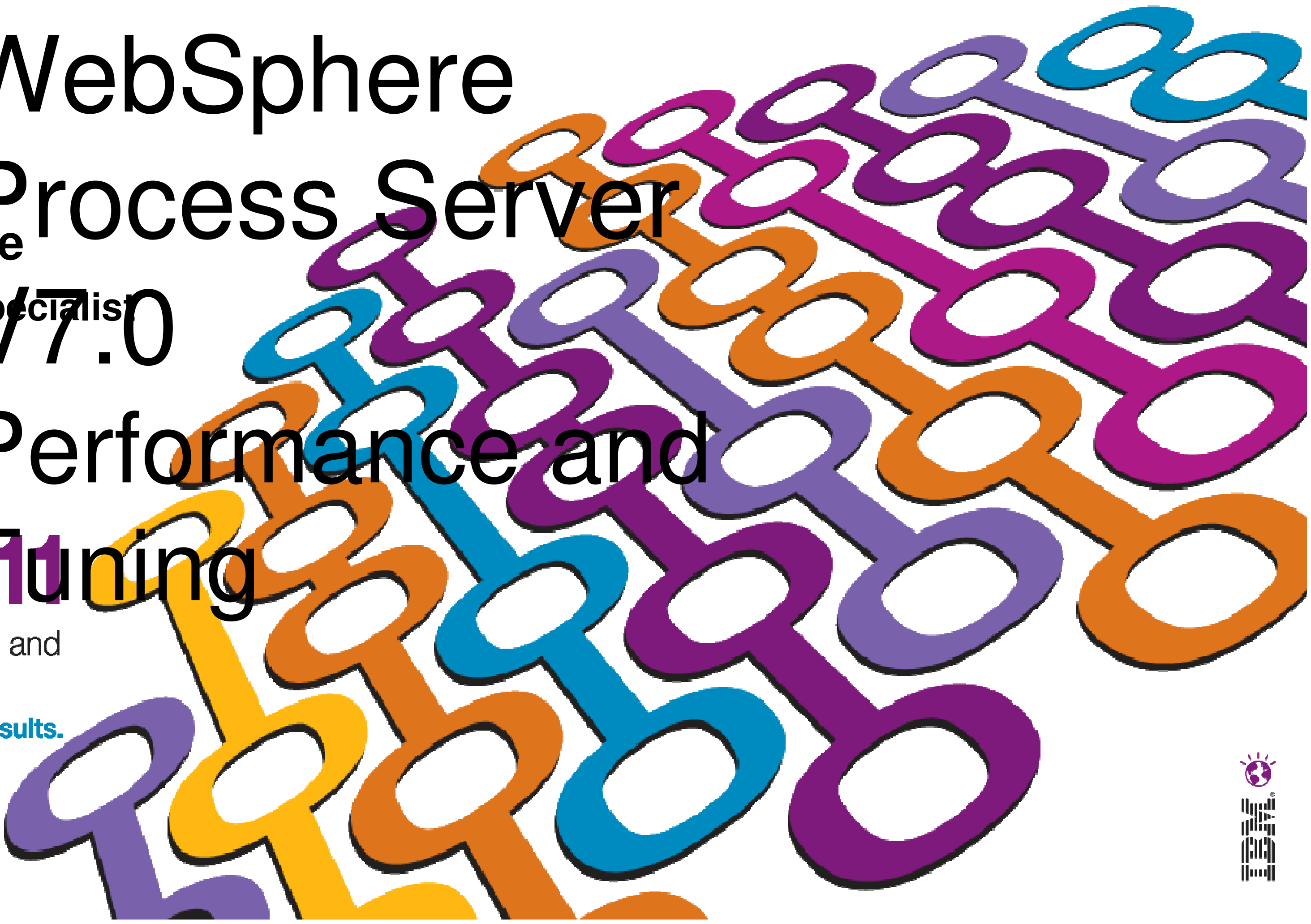
IBM Software
AVP
Accelerated Value Program
ibm.com/software/support/acceleratedvalue/

WebSphere Process Server V7.0 Performance and Tuning

Thomas Reinecke
Accelerated Value Specialist

IBM Software
Impact2011
Changing the Way Business and IT Leaders Work
Optimize for Growth. Deliver Results.



IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal at IBM's sole discretion. Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.

The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

Agenda

1. Workshop objectives
2. The cost of poor performance
3. Tunable components in a WPS environment
4. Tune application design and implementation
5. Introduction to WPS Performance Tester sample
6. Tune WebSphere Application Server configuration
7. Tune Java Virtual Machine
8. Summary : WPS specific tuning guidelines

IBM Software AVP

1 Workshop objectives

- In this workshop, you will learn the advanced theoretical skills needed to tune the runtime performance of IBM WebSphere Process Server with focus on version 7.0.

- After attending this workshop you will be able to explain the most important configurable items (Thread Pools, Activation Specifications, JMS and the Java Heap) at a WAS infrastructure level and its impact on WPS performance.

- You will also learn the functionality and characteristics of the asynchronous communication flavor of the SCA and messaging binding types.

After the workshop, there is an accompanying lab which will allow you to observe the differences in performance between the component Binding types (SCA, WS, HTTP, Messaging) and flavors (synchronous, asynchronous).

What is meant by Performance ? What are the goals ?

Do you need...

Throughput ? Response ?

(might be competing goals)

Agenda

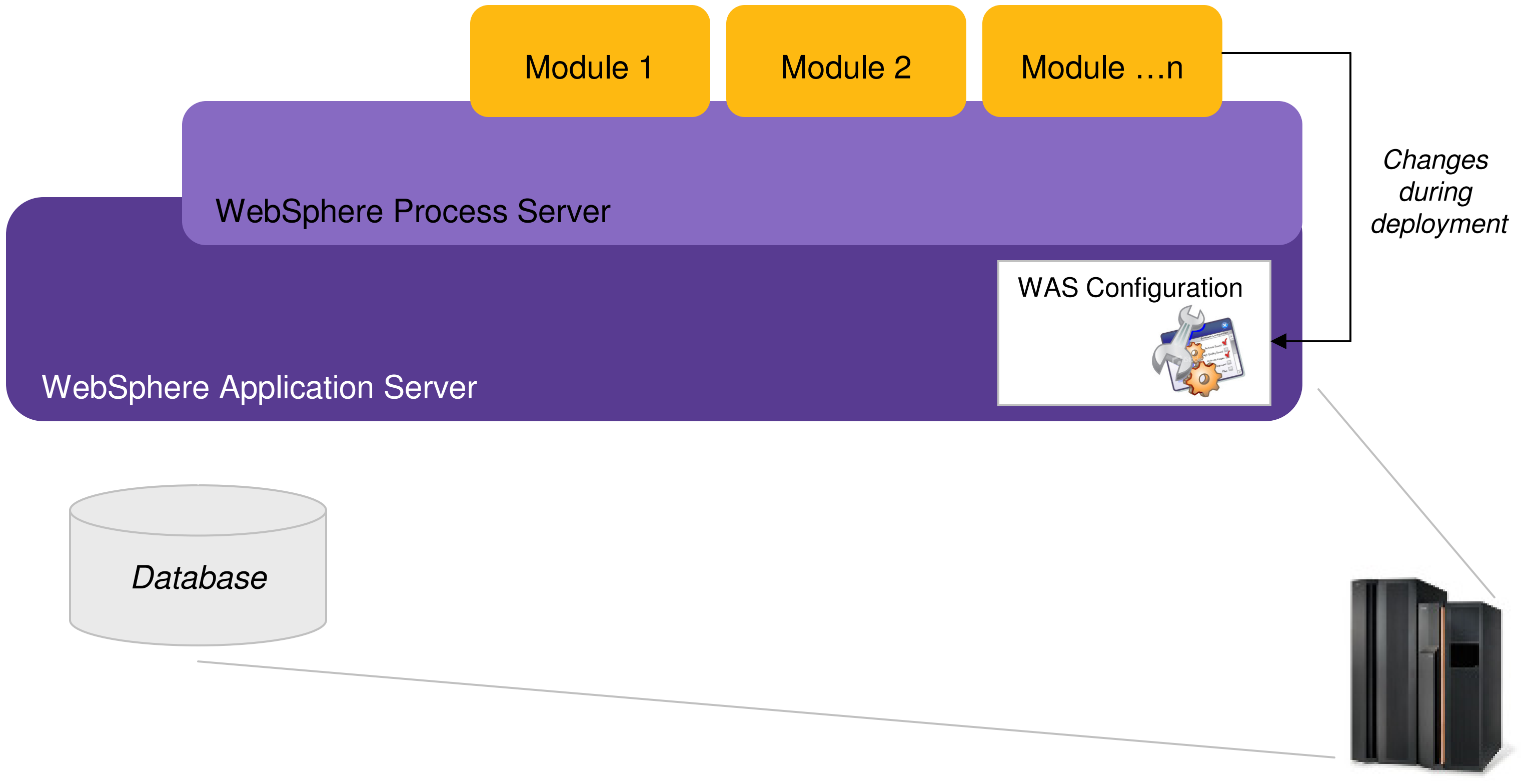
1. Workshop objectives
2. The cost of poor performance
3. Tunable components in a WPS environment
4. Tune application design and implementation
5. Introduction to WPS Performance Tester sample
6. Tune WebSphere Application Server configuration
7. Tune Java Virtual Machine
8. Summary : WPS specific tuning guidelines

2 The cost of poor performance

- Escalating application support cost
  - More resources needed to support application in production (Hardware, Human Resources)
  - Poor performance may affect other (upstream) components and applications
- Loss of customer confidence
  - Generally slow or reduced response times are not acceptable for the users
- Loss of credibility
  - Level of Performance promised in service level agreements are not met
  - May mean end of life for your application
- Loss of revenue
  - The inevitable effect of losing customers

Agenda

3 Tunable Components in a WPS environment



3 Tunable Components in a WPS environment

The remaining topics are not a subject of this workshop

- Hardware
- Operating System
- Database
- ✓ Application Design and Implementation

80 %

- ✓ WAS Configuration - Threadpools / Activations Specs
- ✓ Java Virtual Machine

20 %

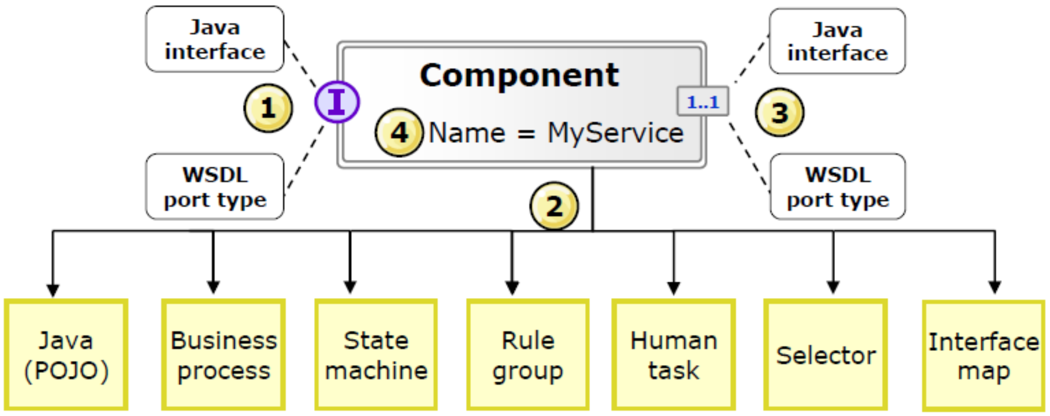
Agenda

4 Tune Application Design and Implementation

- SCA review
- Comparison of the binding types
- SCA binding performance details
- How does the asynchronous SCA binding work ?
- Why is an asynchronous SCA invocation performing slower ?

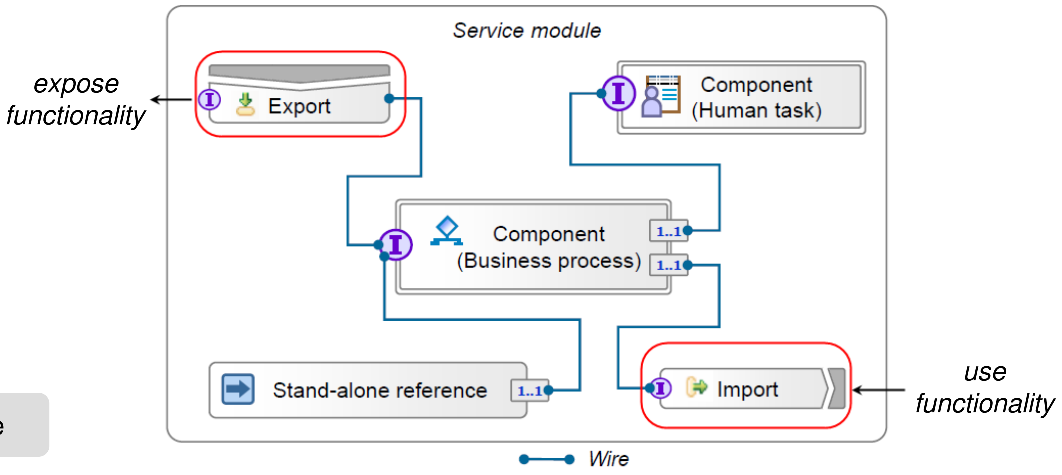
SCA Review

What is a SCA component ?

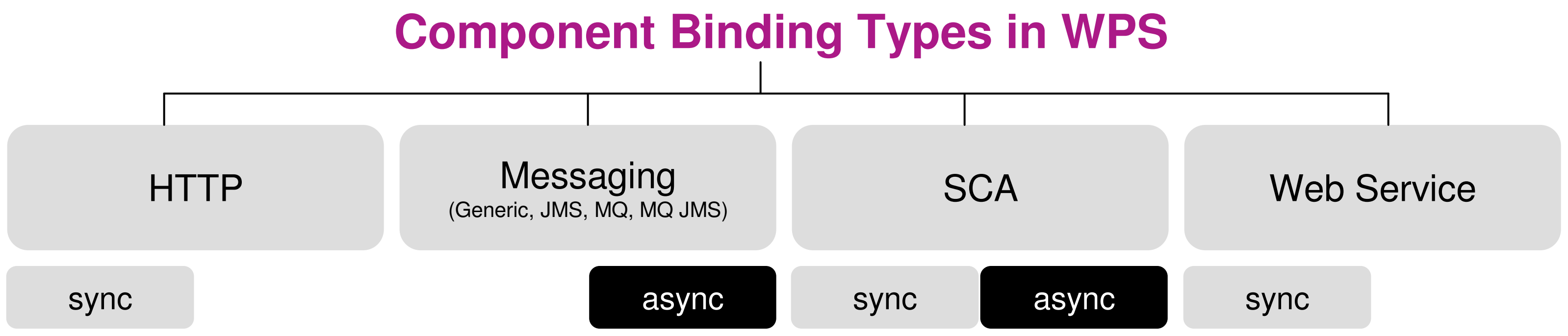


How do SCA components interact with each other ?

HTTP | Messaging | SCA | Web Service



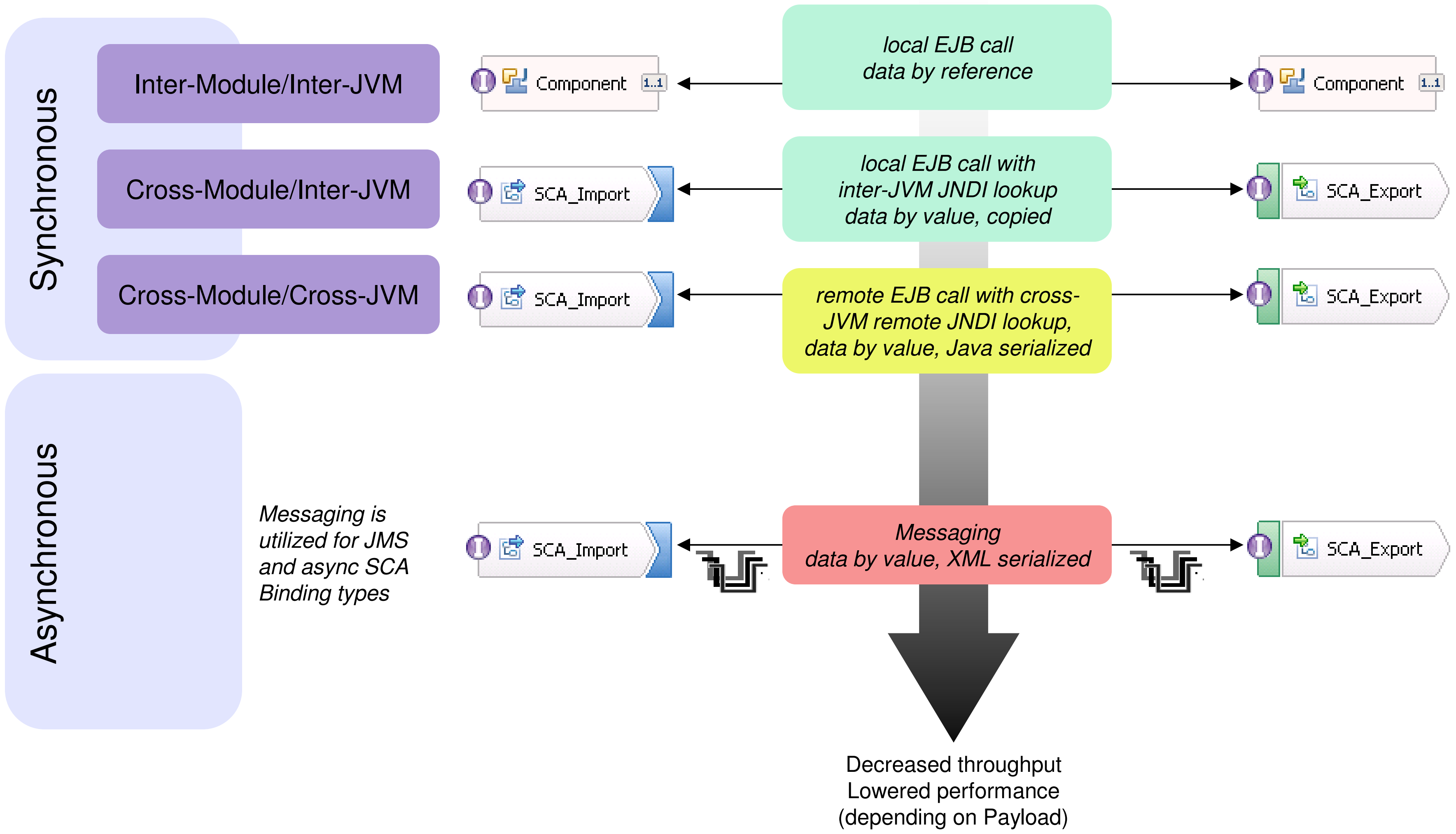
Comparison of the Binding Types



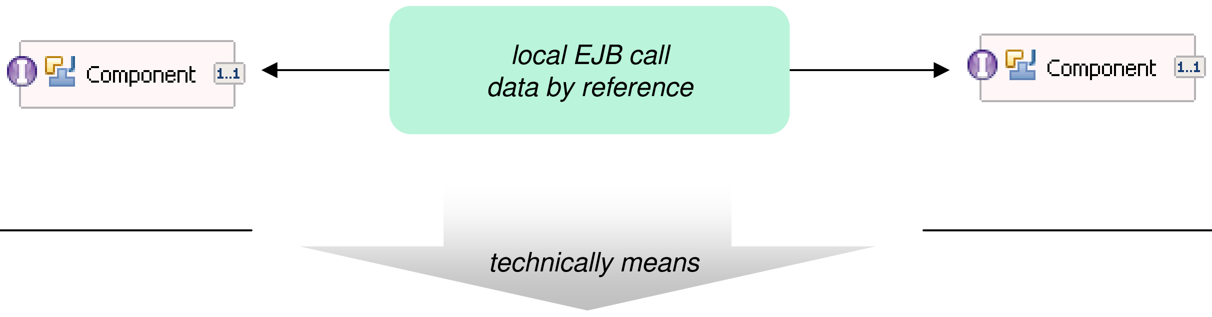
Export binding type	Operation Type	Performance attributes > Interaction Style	Invocation style
SCA	one-way	Asynchronous	Asynchronous (default)
	one-way	Synchronous	Synchronous
	request-response	any	Synchronous
HTTP	any	n/a	Synchronous
MQ MQ JMS JMS	one-way	n/a	Asynchronous
	request-response	n/a	Asynchronous (with callback)
Web services (soap/http or soap/jms)	any	n/a	Synchronous

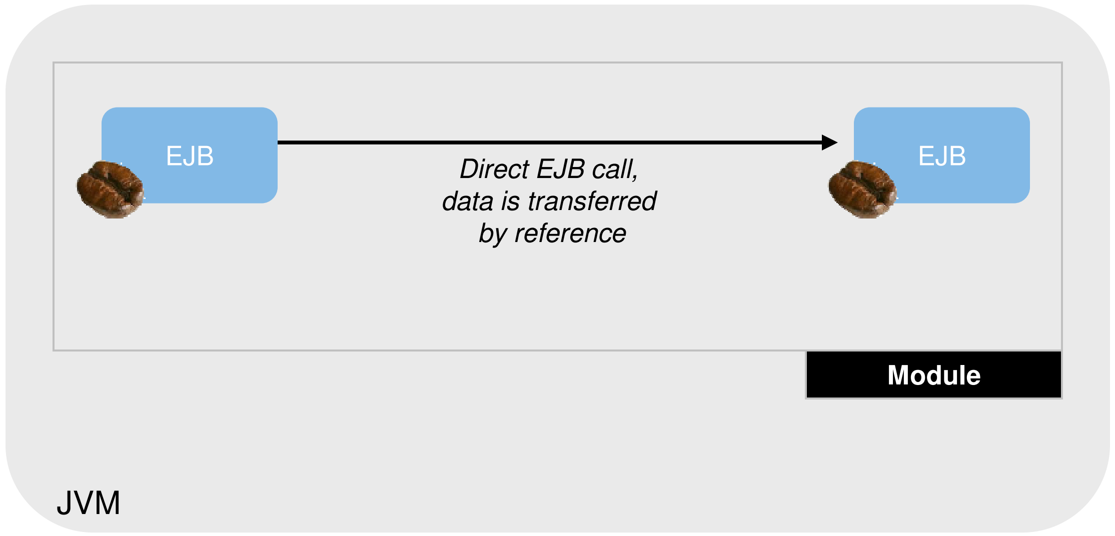
SCA Binding Performance Details

Communication types of SCA Components

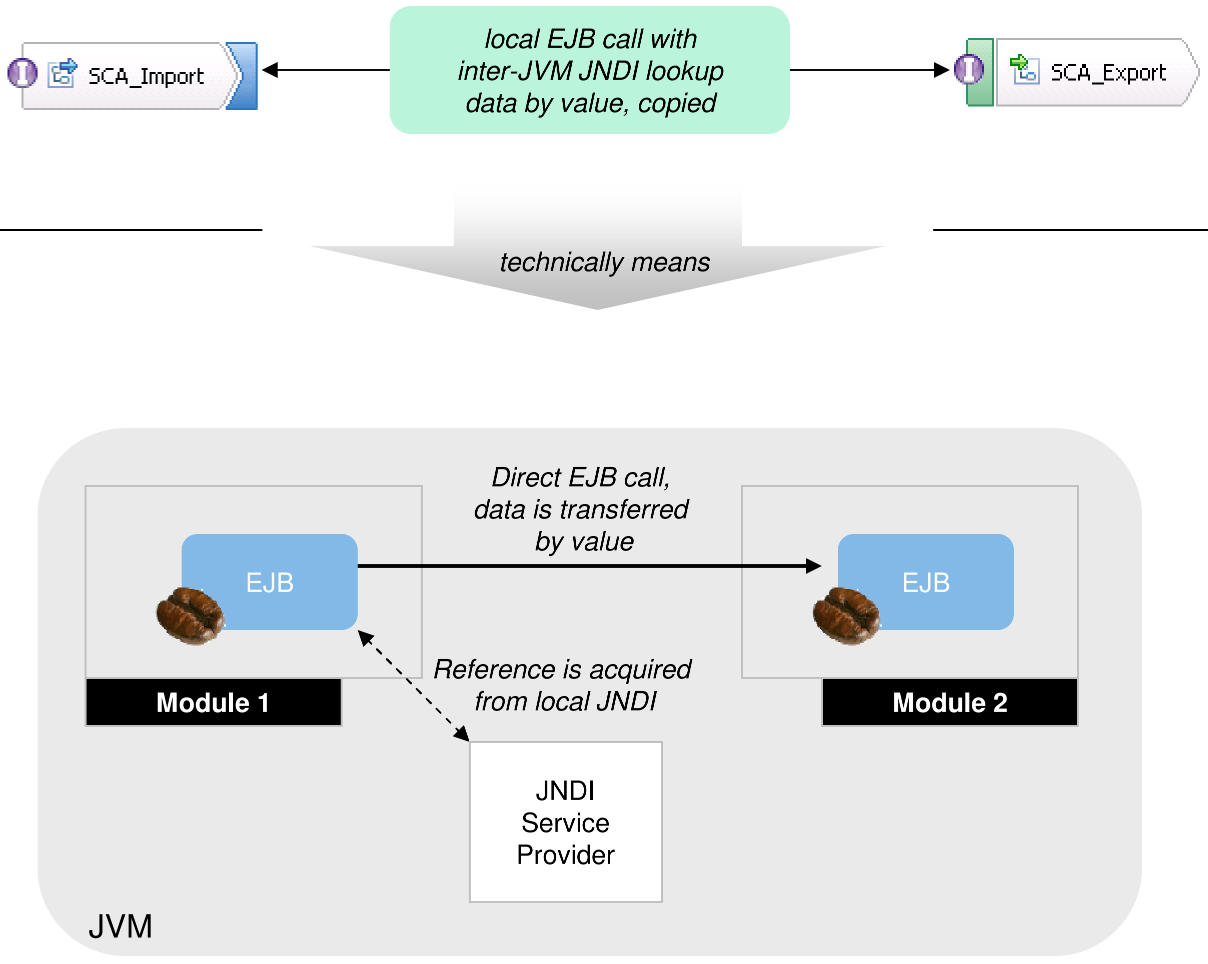


Synchronous Inter-Module / Inter-JVM (Technical Details)

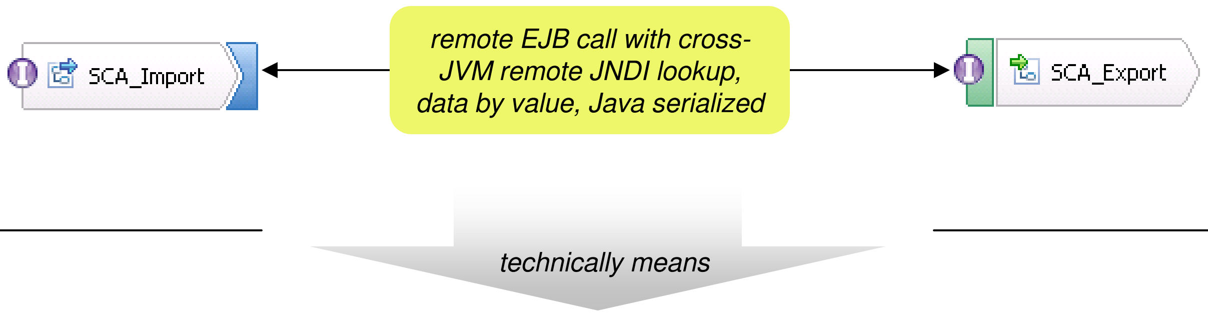


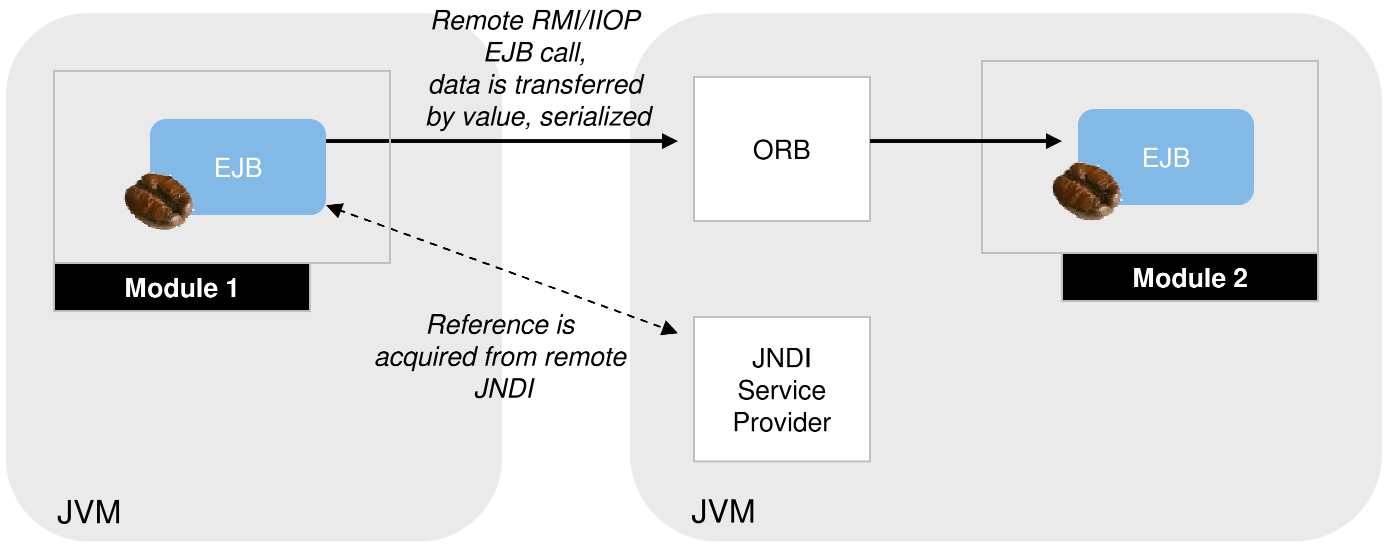


Synchronous Cross-Module / Inter-JVM (Technical Details)

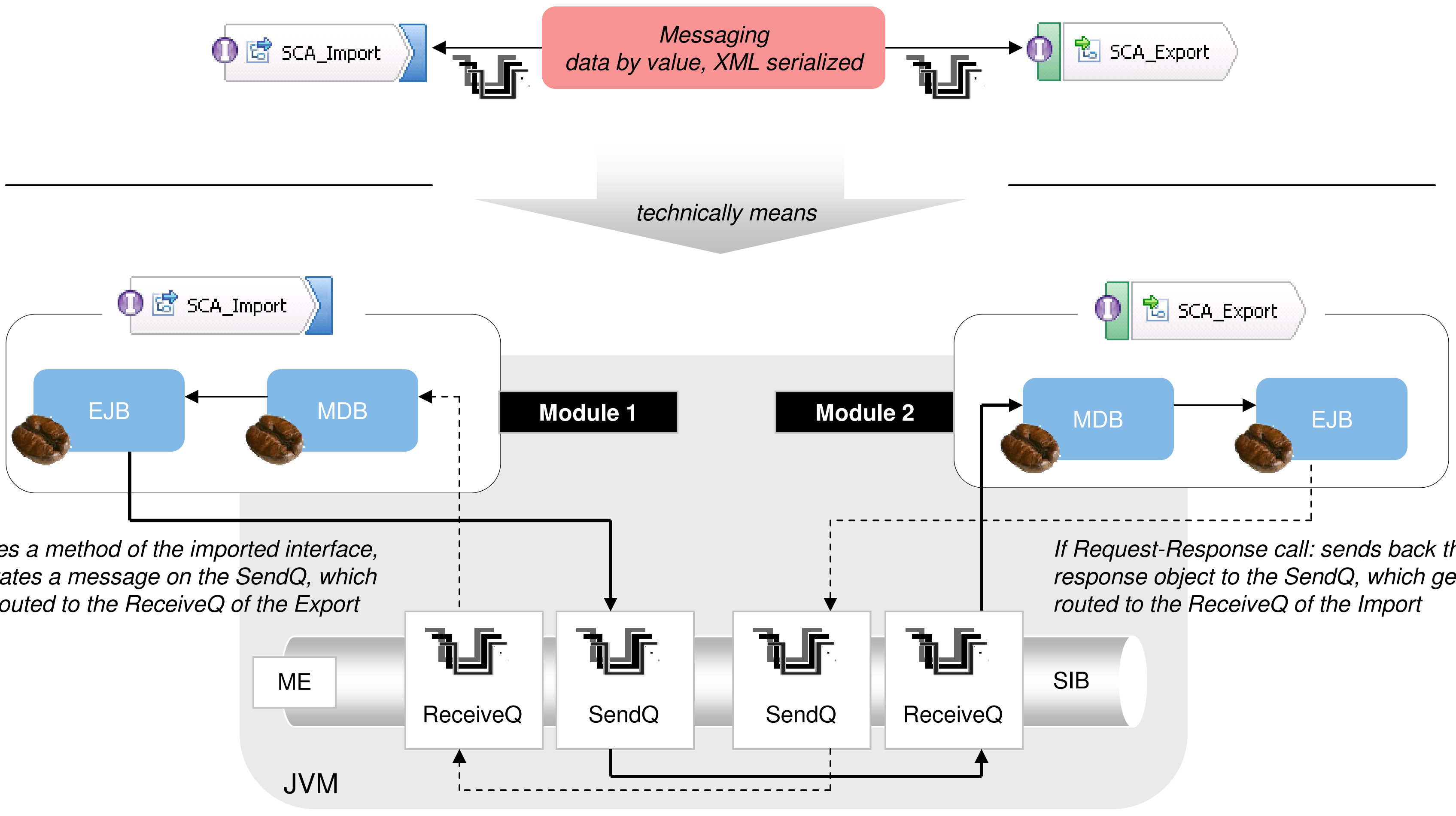


Synchronous Cross-Module/Cross-JVM (Technical Details)


SCA_Import
remote EJB call with cross-JVM remote JNDI lookup, data by value, Java serialized
SCA_Export
technically means


Remote RMI/IIOP EJB call, data is transferred by value, serialized
EJB
Module 1
ORB
EJB
Module 2
Reference is acquired from remote JNDI
JNDI Service Provider
JVM
JVM

How does the Asynchronous SCA Binding work ?



Invokes a method of the imported interface, generates a message on the SendQ, which gets routed to the ReceiveQ of the Export

If Request-Response call: sends back the response object to the SendQ, which gets routed to the ReceiveQ of the Import

Why is an asynchronous SCA invocation performing slower ?

Asynchronous SCA Invocation is slower (in most cases, but not always) compared to synchronous SCA Invocation because the following additional functions have to be performed:

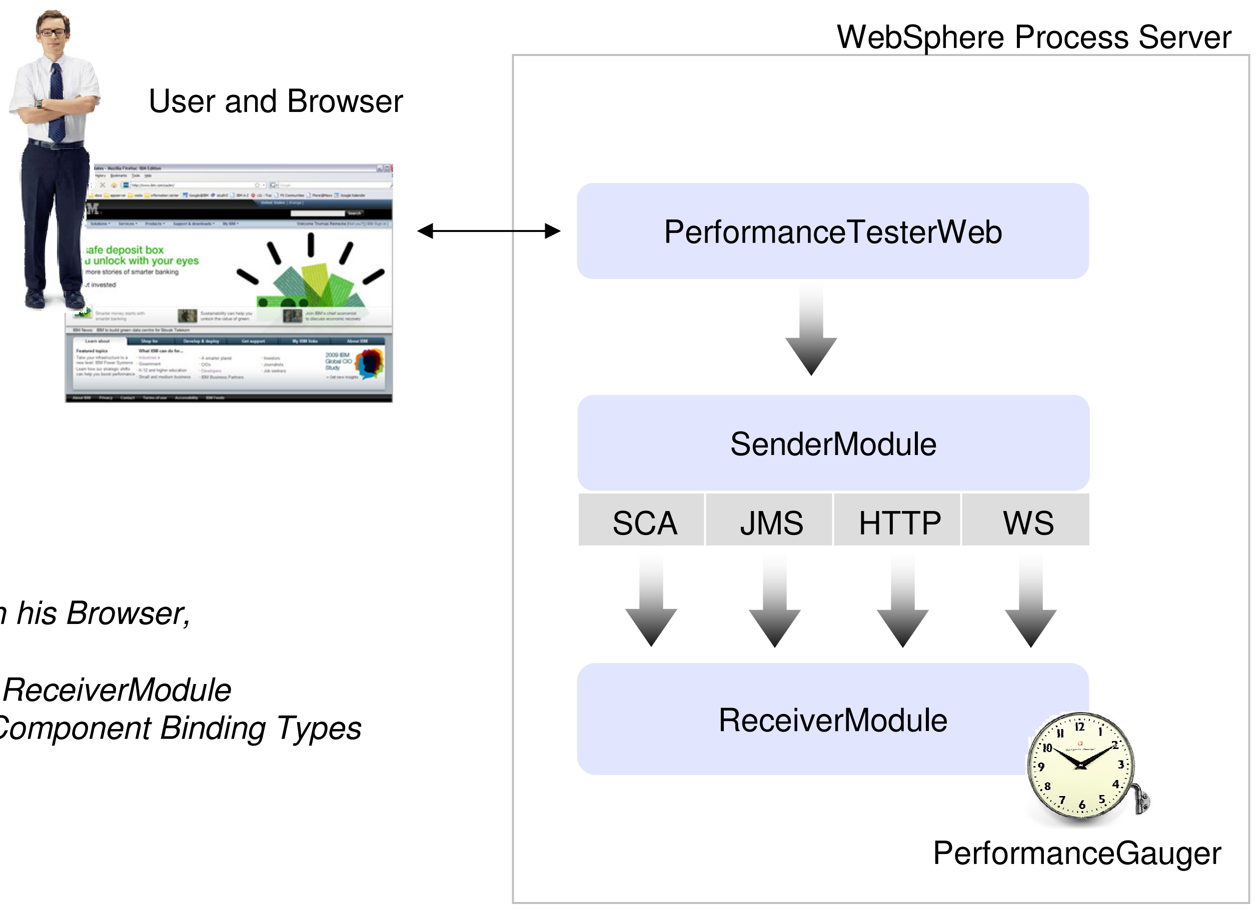
- XML-Serialize the outgoing business object
- Write the serialized BO to a queue
- Add message header and create a correlation/callback ticket
- Trigger an MDB on the target site
- MDB needs to parse message header and correlation information
- Read the serialized BO from the queue
- Correlate the BO and operation to a particular component
- On Sender Side, spawn a second thread to handle response messages (depending on async invocation style, e.g. callback, deferred-response)

Best practice in application design

- Avoid overusing cross-module bindings
  (put components in the same SCA module if possible, avoid fine grained modules)
- Use application specific BO's
  BO's should carry the payload which is necessary for the next module, only
- Run Modules in the same JVM if possible
- Utilize SCA synchronous bindings and utilize multi-threaded clients
- Utilize Microflows instead of Long-Running processes wherever possible
- Use process design patterns where appropriate
- Use scopes to bind variables to certain process areas in a macroflow
- Configure transactional boundaries in macroflows
  (add many activities to a single transaction scope)
- Avoid Common Base event emitting
  (and if you need to emit events, do it asynchronously in a new transaction)

Agenda

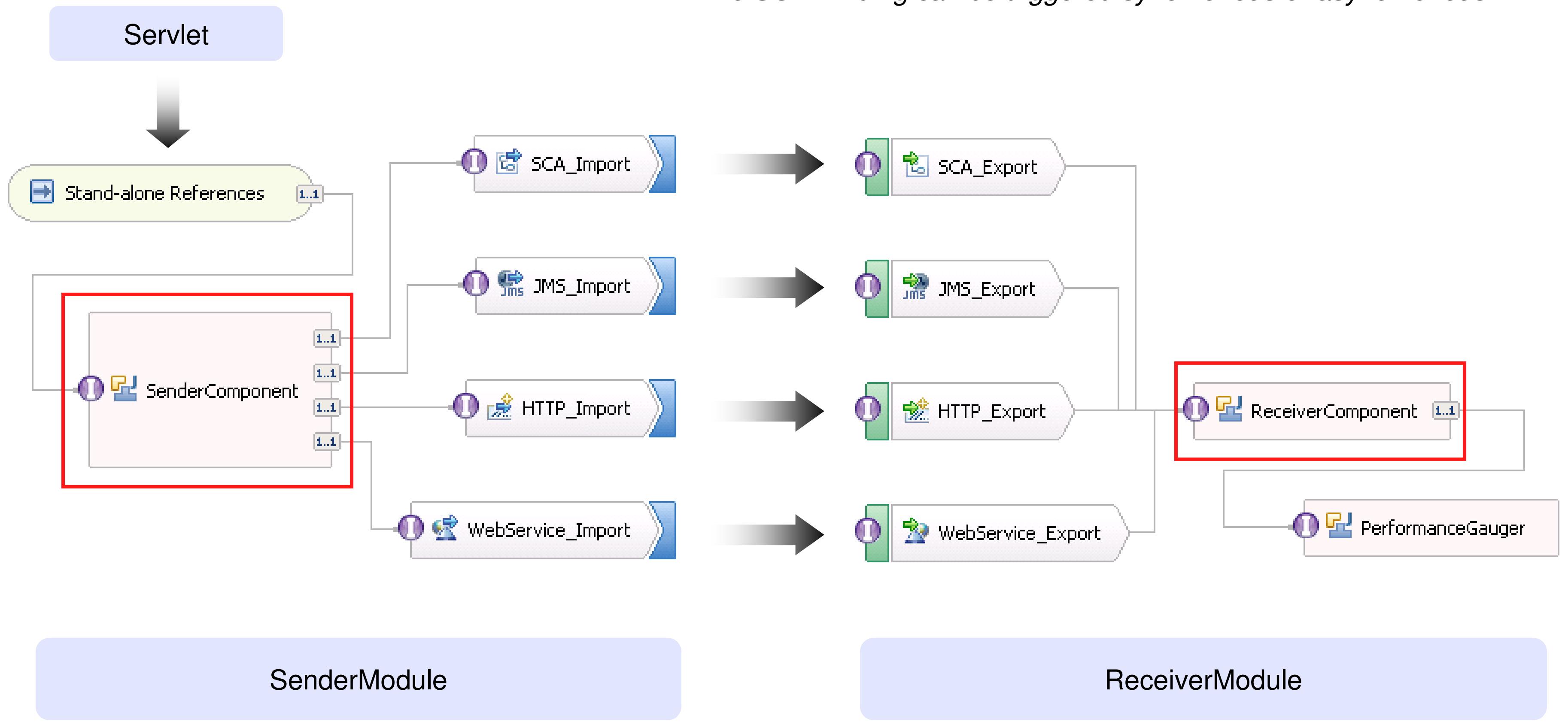
5 Introduction to WPS Performance Tester Sample



The user calls a Servlet in his Browser, which triggers the SenderModule to call the ReceiverModule based on different WPS Component Binding Types

Technically this means…

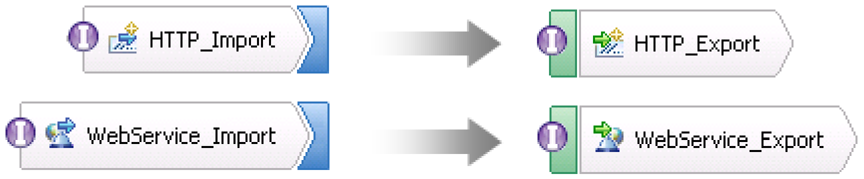
The SCA Binding can be triggered synchronous or asynchronous.



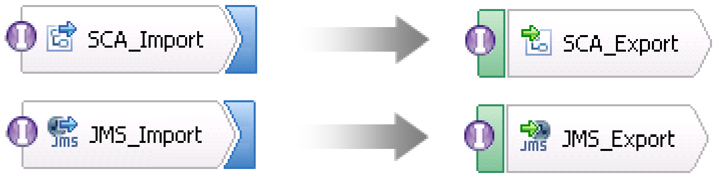
Agenda

1. Workshop objectives
2. The cost of poor performance
3. Tunable components in a WPS environment
4. Tune application design and implementation
5. Introduction to WPS Performance Tester sample
6. Tune WebSphere Application Server configuration
7. Tune Java Virtual Machine
8. Summary : WPS specific tuning guidelines

6 Tune WebSphere Application Server Configuration



Tune WebContainer Thread Pool
Tune IBM HTTP Server

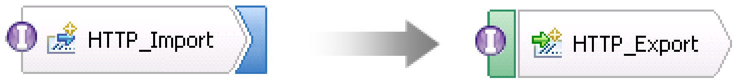


Tune Messaging Resource Adapters
Tune Activation Specs and Thread pools
Tune ORB for Inter-JVM Communication

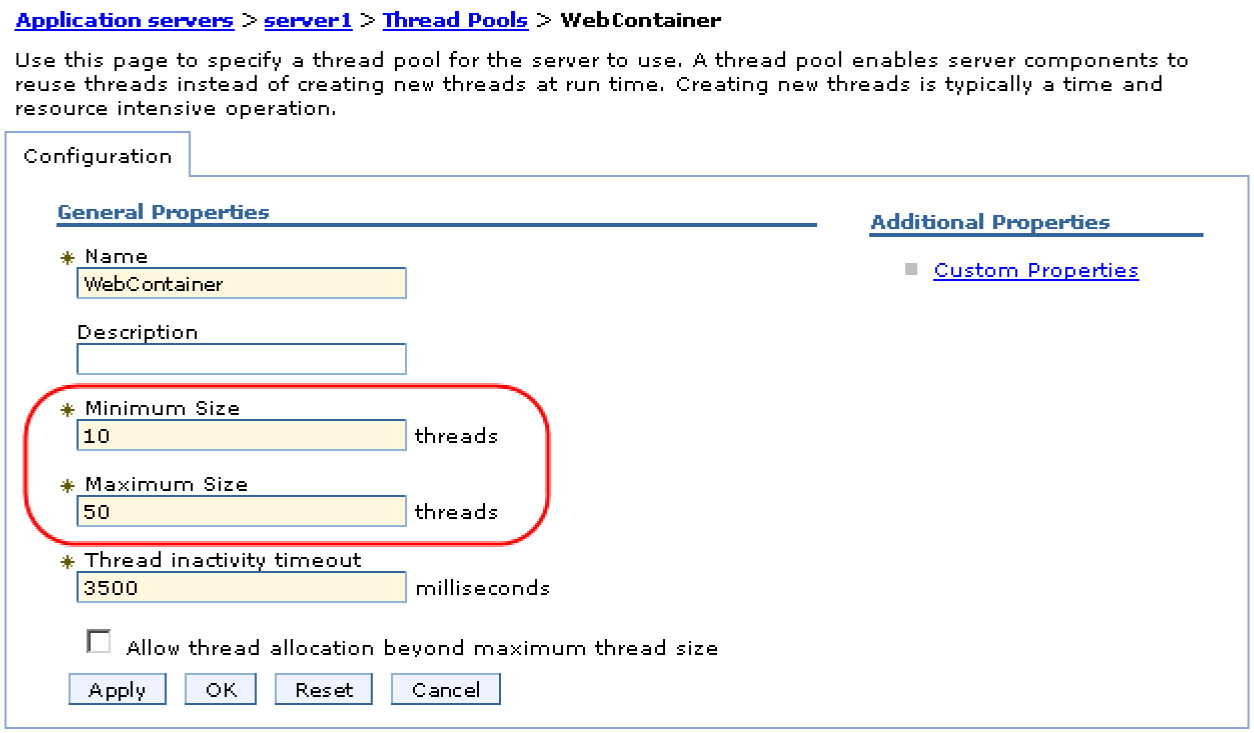
Always: Tune the Java Heap, apply clustering !

IBM Software AVP

HTTP/WS sync - Tune WebContainer Thread Pool



WAS Admin Console > Servers > Application Servers > (server name) > Additional Properties > Thread Pools > WebContainer



Application servers > server1 > Thread Pools > WebContainer

Use this page to specify a thread pool for the server to use. A thread pool enables server components to reuse threads instead of creating new threads at run time. Creating new threads is typically a time and resource intensive operation.

Configuration

General Properties

* Name: WebContainer

Description:

* Minimum Size: 10 threads

* Maximum Size: 50 threads

* Thread inactivity timeout: 3500 milliseconds

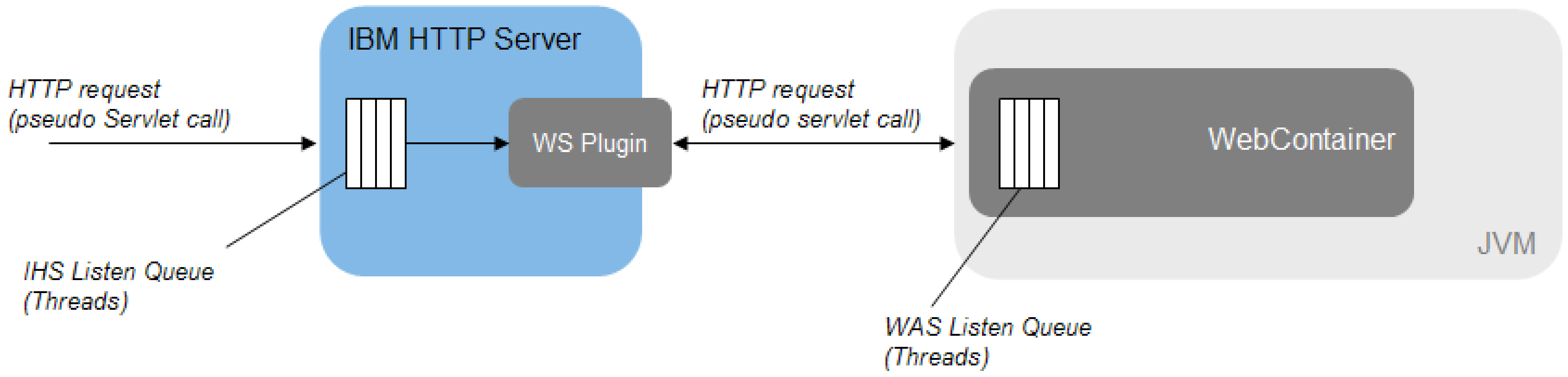
☐ Allow thread allocation beyond maximum thread size

Additional Properties

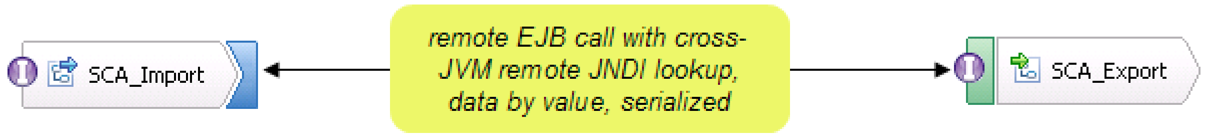
- Custom Properties

HTTP/WS sync - Tune IBM HTTP Server

- Concurrency level is determined by configuration of the IBM HTTP Server (IHS) and the configuration of the Web Container
- Use standard WebSphere Application Server configuration best practices for HTTP configuration (servlet calls)
  - Tune number of threads per IHS processes and number of IHS processes
  - Tune number of Web container threads
  - Tune HTTP timeout as required


IBM HTTP Server
HTTP request (pseudo Servlet call)
WS Plugin
HTTP request (pseudo servlet call)
WebContainer
JVM
IHS Listen Queue (Threads)
WAS Listen Queue (Threads)

SCA sync/async - Tune ORB for Cross-JVM Communication


SCA_Import
remote EJB call with cross-JVM remote JNDI lookup, data by value, serialized
SCA_Export

- Concurrency level determined by client
  - First module will run on caller's thread
- Downstream concurrency level determined by ORB thread pool size
  - Optimal downstream concurrency will be less or equal than to the upstream concurrency level (thread funneling)
- ORB thread pool size will effect level of concurrency
  - Set ORB maximum thread pool size to a level that will support the optimal concurrency level for the downstream module
  - From the WAS admin console:

ORB thread pool size: set "Maximum Size" (Default is 50)

```
Servers > Application servers > (server name) > Container Services:
ORB Service > Thread Pool
```

JMS/SCA async - Tune Messaging Resource Adapters

- Separate SPI and Default Messaging Provider Thread Pool
- Per default configuration, the
  Platform Messaging Component SPI Resource Adapter and the
  SIB JMS Resource Adapter
  share the same Thread Pool 'Default'.
- Create two dedicated Threadpools for the **SPI** and **JMS** Resource Adapters and assign them to the adapters. This separates the WAS internal Thread management for SCA async and JMS Bindings.

```
Servers > Application servers > (server name) > Thread Pools

Resources > Resource Adapters > Platform Messaging Component SPI
Resource Adapter
Resources > Resource Adapters > SIB JMS Resource Adapter
```

SCA async - Tune Activation Spec Concurrency

- Concurrency level determined by configuration of SIB for both modules
  - Set MDB thread pool size to achieve the optimal concurrency level for each module
  - MDB thread pool is usually taken from the "Default" EJB thread pool or from the newly created SPI Pool
  - Increase the size of the Thread pool to handle any increase in configuration of the SIB MDB thread pool

  - From the WAS admin console:
    SIB MDB: set "maxConcurrency" (Default is 10) – maximum Number of MDBs

```
Resources > Resource Adapters > Platform Messaging Component SPI
Resource Adapter > J2C Activation specifications > (module name)_AS >
J2C activation specification custom properties
```

JMS - Tune Concurrency

- Concurrency level determined by configuration of JMS (in the SIB) for the downstream module
  - Set JMS MDB thread pool size to achieve the optimal concurrency level
  - JMS MDB thread pool is taken from the “Default” EJB thread pool
    Increase the size of the “Default” EJB thread pool to handle any increase in configuration of the JMS MDB thread pool
  - From the WAS admin console:
    JMS MDB: set “Maximum concurrent endpoints” (default is 10)

```
Resources > JMS Providers > Default messaging > JMS activation specification  >
(export name)_AS > J2C activation specification custom properties
```

Agenda

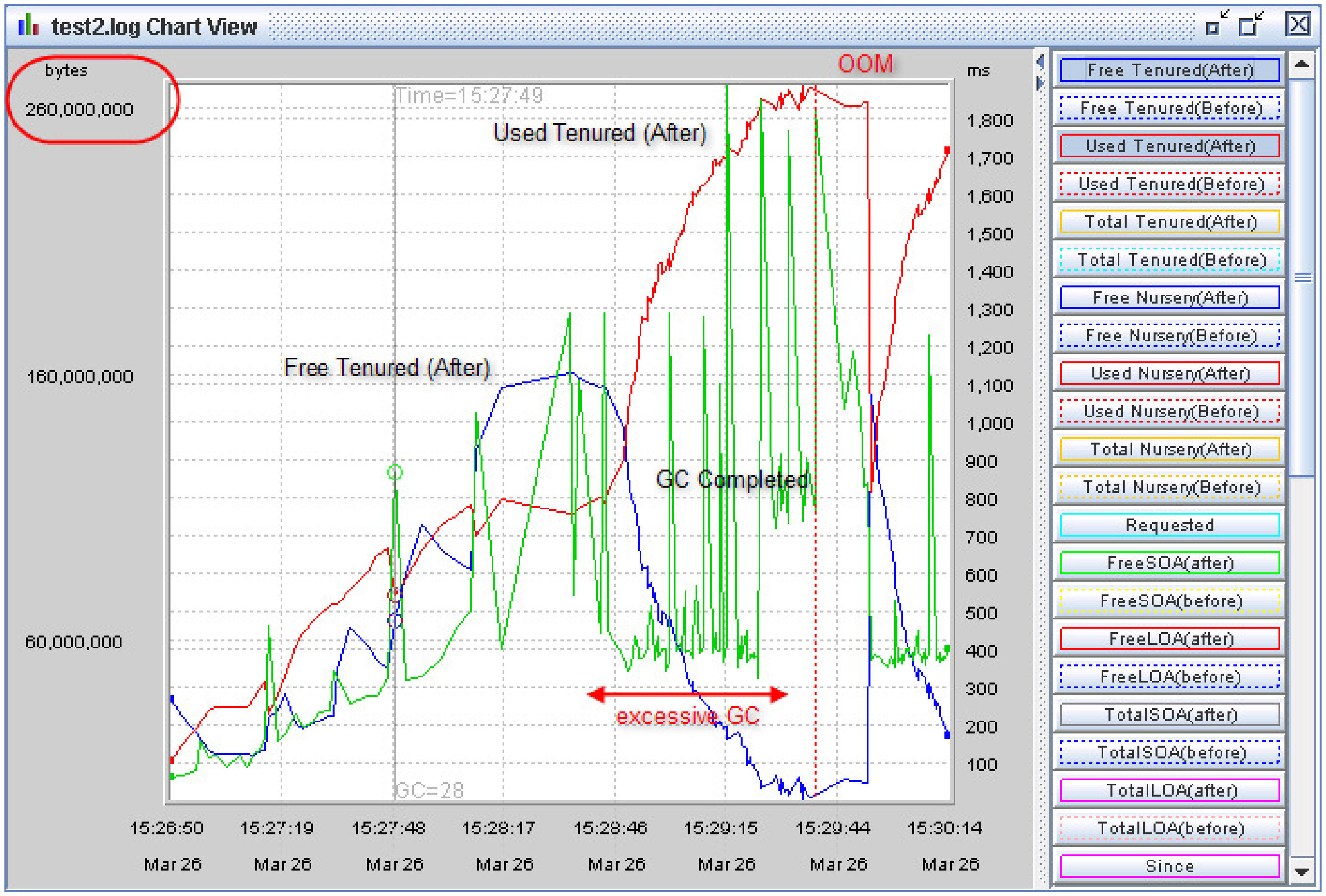
JVM Tuning Best practice

- Activate verbose Garbage collection
  (just basic GC, approx. 0.5-1.5% overhead)
- Monitor the JVM Heap allocation details
  - How many big objects get allocated ?
  - Does the virtual machine run into excessive Garbage Collection ?
  - How much time does the Garbage Collection consume compared to normal JVM operation ?
  - Do you run out of memory ? (Heap or native memory) ?
  - How many Threads and other Shared Resources have to be managed by the JVM ?
  - How much memory is actually addressable by the JVM and OS ?
  - Which Garbage Collection Policy fits to my requirements ?

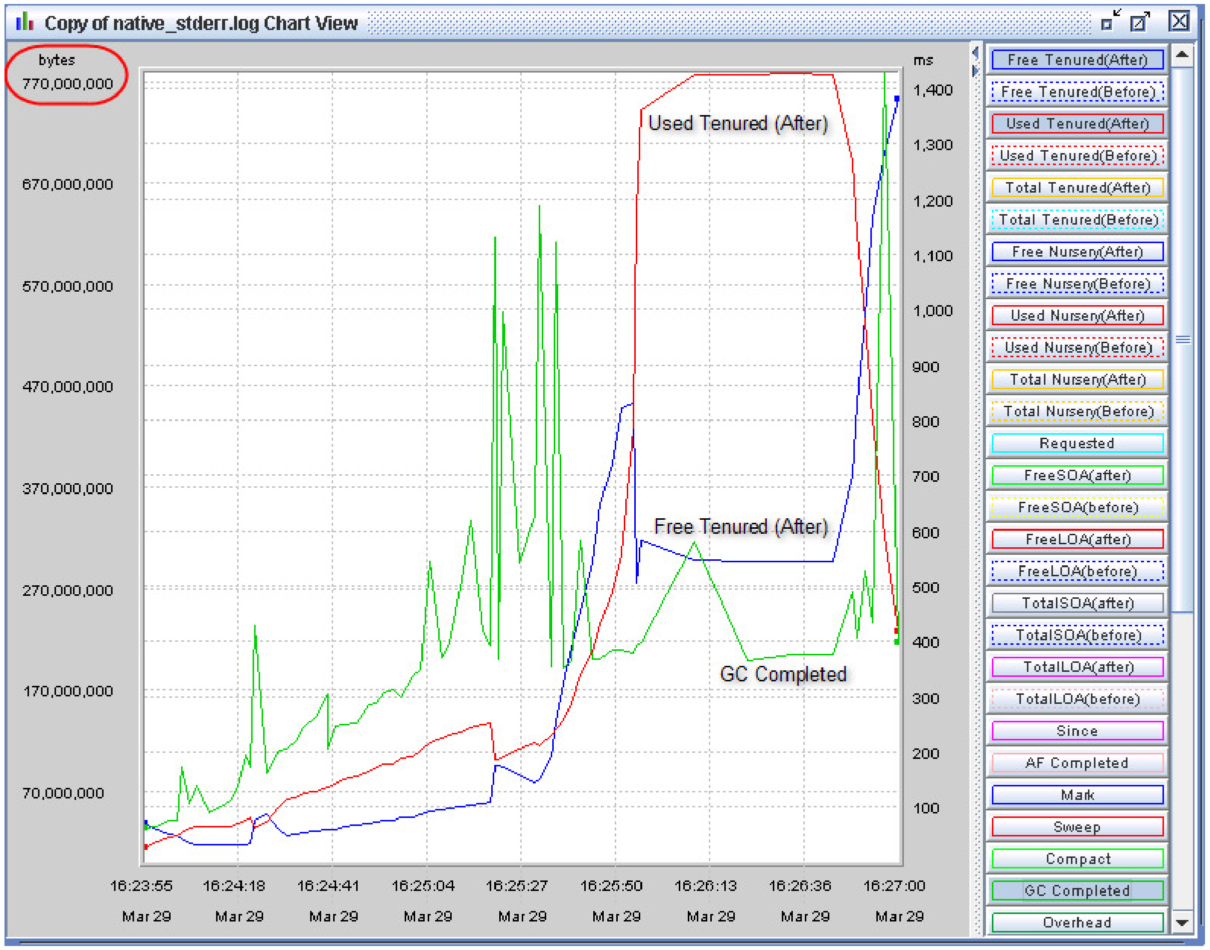
Shared Resources…

- Physical Memory
  - Avoid over-committing physical memory and swapping
  - Monitor aggregated memory statistics and per-process memory statistics
- Swap space
  - Rule of thumb: maximum swap space should be larger than amount of physical memory (try not to use the swap)
- Threads
  - Monitor number of threads in each JVM
  - Monitor per thread native memory utilization (per thread native stack configured by “Xss”)
  - Monitor threads in a JVM by “category”
    - Containers (Servlets, JSPs, EJBs…)
    - ORB
- Database Connections
  - Consider users of database connections in aggregate (all applications servers in a cluster and all components running in each application server)

Example of Excessive Garbage Collection and OutOfMemory



Example of a well tuned Heap for WPS under Heavy load



Agenda

1. Workshop objectives
2. The cost of poor performance
3. Tunable components in a WPS environment
4. Tune application design and implementation
5. Introduction to WPS Performance Tester sample
6. Tune WebSphere Application Server configuration
7. Tune Java Virtual Machine
8. Summary : WPS specific tuning guidelines

8 Summary: WPS specific tuning guidelines

- Move database off the default Derby to a high performance database management system (Derby is not supported for Production environments)
- Do not enable security where not needed and avoid fine-grained J2EE security
- Do not run a production server in development mode or with development profile
- Wherever possible, utilize microflows instead of long-running processes
- If possible create a separate process that contains the microflow components to keep the long-running process compact (the larger the long-running process the more data needs to be persisted in the BPEDB)
- Use work-manager based navigation to improve throughput for long-running processes
- Avoid unnecessary usages of asynchronous invocations
- Verify the preferred interaction style set on each SCA component
- Configure the message-driven bean (MDB) activation specification
- Configure WebSphere Process Server for clustering where applicable
- Configure thread and connection pools for sufficient concurrency and size
- Disable validation in the Common Event Infrastructure (CEI) emitter

8 Summary: WPS specific tuning guidelines

- Use the appropriate hardware configuration for performance measurement.
  (For example, laptops and desktops are not appropriate for realistic performance evaluation)
  - Have a test environment available that reflects the structure (HW, OS, SW, DB etc.) of the Production environment
- Disable tracing and monitoring when possible (except verbose:gc)

Database

» Place database table spaces and logs on a fast disk subsystem (SAN drive)

» Size the database cross-referencing tables correctly

» Place logs on a separate device from table spaces

DB2 Specific

» Maintain current indexes on tables and views

» Update the catalog statistics, run ongoing reorganizations

» Set the buffer pool size correctly

Java

» Set the heap size to delivery optimal throughput and response time

» Keep Verbose GC enabled at all time to monitor the JVM

» Adjust specific settings when appropriate or benefitial

Links & References

General WebSphere Application Server performance URL
http://www.ibm.com/software/webservers/appserv/was/performance.html

Tuning Performance section of the WebSphere Application Server V6.1 information center
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.nd.doc/info/ae/ae/welc6toptuning.html

Tuning the JDBC data source of a messaging engine
http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=/com.ibm.websphere.pmc.doc/tasks/tjm0230_.html

DB2 performance (V9.5)
ftp://ftp.software.ibm.com/ps/products/db2/info/vr95/pdf/en_US/db2d3e950.pdf

Would you like to dive even deeper ?

- IBM Course Code WB724
  WebSphere Process Server V7.0 Performance and Tuning

WB724 – Classroom

ttps://www-304.ibm.com/jct03001c/services/learning/ites.wss/us/en?pageType=course_description&courseCode=WB724

We value your feedback

- Please complete the survey for this session
- Hands-On Lab WPS Performance and Tuning

  (right next after this workshop and repeated on Thursday morning)

Thank you…

IBM Software AVP