



Performance for SOA DB2 Java Applications

SOA Performance Considerations

By David Beulke

DaveBeulke@cs.com
703 798-3283

Save \$Millions in CPU and I/O costs!

Dave Beulke

- IBM Gold Consultant
- Past President of IDUG
- Best speaker at CMG conference & former TDWI instructor

- Co-Author of certification tests
 - DB2 V8 & V7 DBA certification test
 - IBM Business Intelligence certification test

- Columnist for DB2 Magazine and former editor of the IDUG Solutions Journal
- Seminar Author
 - Data Warehouse Performance Seminar
 - How to do your own DB2 Performance Review
 - NEW! Performance for SOA Java DB2 Environments

- Extensive experience in performance and design of large databases and DW systems
 - Working with DB2 on z/OS since V1.2
 - Working with DB2 on LUW since OS/2 Extended Edition

- Author of Syspedia - Data element analytics - www.Syspedia.com



Agenda

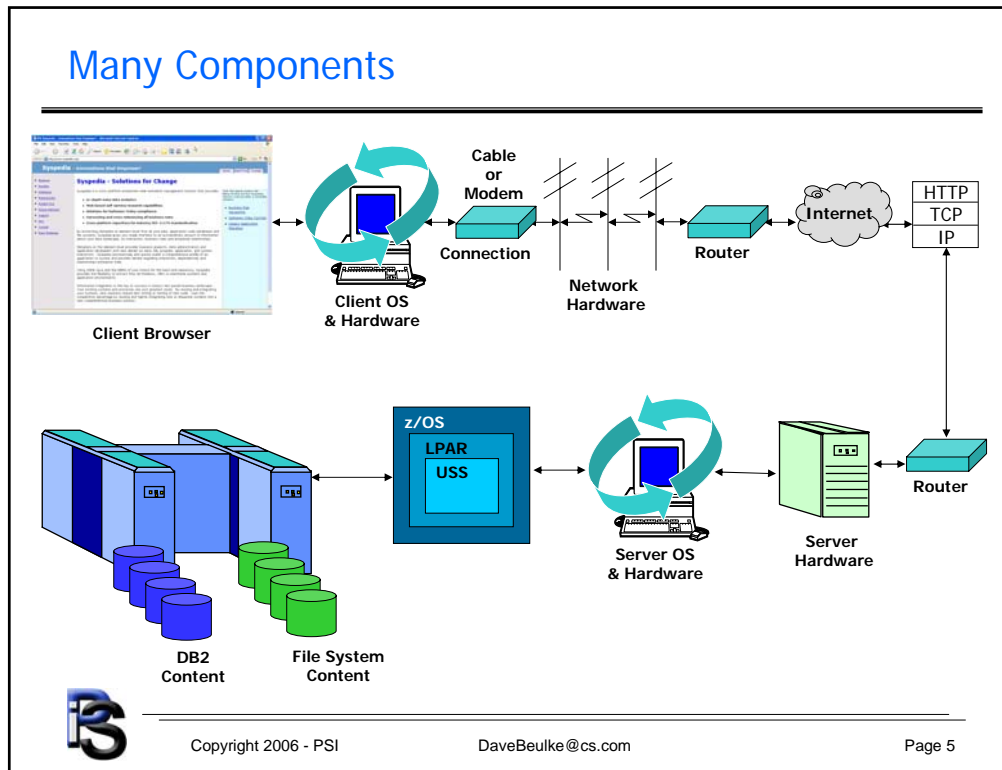
- Architectures for Performance
 - Trends, Server, Network
- Programming Performance Considerations
 - Servlets, JSPs and Beans
- Connections and SQL types
 - JDBC, Connection Pooling and Cursors
- Questions for System Administrator and Developers



Trends, Fades and Reality

- SOA – Early adoption phase
 - Beginning to see new applications leveraging SOA concepts
 - First phase of client/server of circa:2005
- Salesmen and Magazine Hype
 - More articles discussing the advantages
- Real world savings
 - **Cost:** Leveraging the value of existing systems, the functionality of which can be integrated into new applications with minimal investment
 - **Vision:** Understanding the benefits of code reusability, rapid integration, and real-time access to data and functionality throughout the enterprise





- ### Server Performance – doubles every two years
- Sizing of the server
 - Number of processors or cores
 - Processor speed
 - Memory capacity (Std/Max)
 - DASD storage capacity
 - Autonomic/RAS capabilities
 - LPAR error containment
 - Memory - bit steering
 - ChipKill technology
 - CPU fault detection/reallocation
 - Dynamic CPU redirection
 - Special Capabilities
 - Virtualization
 - CPUs, I/Os and load balancing
 - Micro partitioning
 - Shared CPU pool
 - Virtual I/O Server
 - Virtual LAN
 - HttpOps Server Capacity Formulas
 - 5M/Day=58s but 3x for peak
 - Allow 10KB/op=175KB peak
 - 30% network overhead = 2275KB
 - 18MBits/s network bandwidth
 - T1 connection is 1.5M/s
 - Ethernet is 10M/s
 - T3 connection of 45M/s
 - Simultaneous multithreading
 - Two threads per CPU 30% faster
 - Plan for the future!
- A logo 'B' is in the bottom left corner.
- Copyright 2006 - PSI DaveBeulke@cs.com Page 6

Network Performance

- Vendor benchmarks are perfect, your environment isn't
 - It will never go as fast every environment is different
 - System is more likely to be I/O or CPU performance constrained
 - Keep historical records of performance elements

- Synchronize monitoring **hardware, software, network & users**
 - Understand all the elements of the performance puzzle
 - Only monitor what you can change or manage
 - Too much monitoring overhead will become a problem

- Report start/end duration of activities
 - Use accurate tools - A fool with a tool is still a fool!
 - Use automated testing whenever possible



Programming pattern - Model View Controller

- **Pattern Languages of Program Design 5 (Software Patterns Series)** by Dragos Manolescu, Markus Voelter, James Noble

- From Sun JAVA Blueprints:
<http://java.sun.com/blueprints/patterns/MVC-detailed.html>

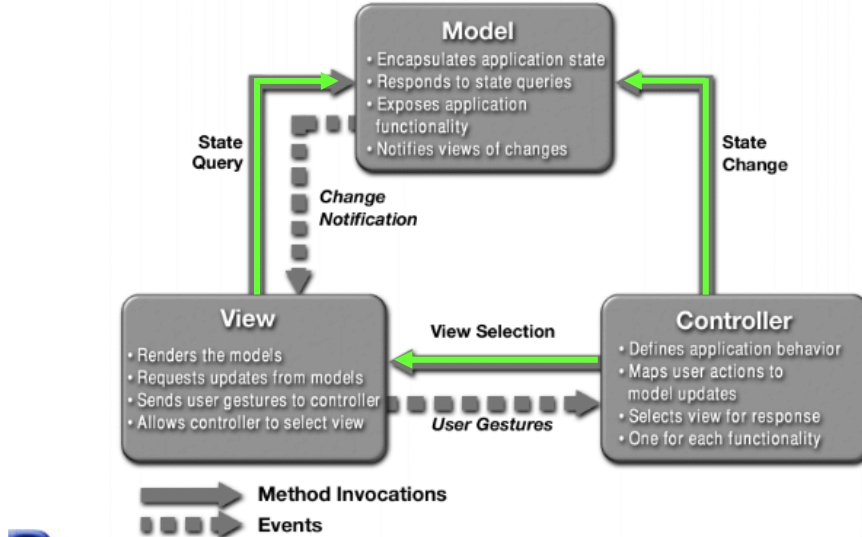
- MVC provides the architecture pattern for performance
 - Separate components to do what they do best
 - Complex applications require multiple output types

- MVC - Model
 - The model represents enterprise data and the business rules that govern access to and updates of this data. Often the model serves as a software approximation to a real-world process, so simple real-world modeling techniques apply when defining the model.

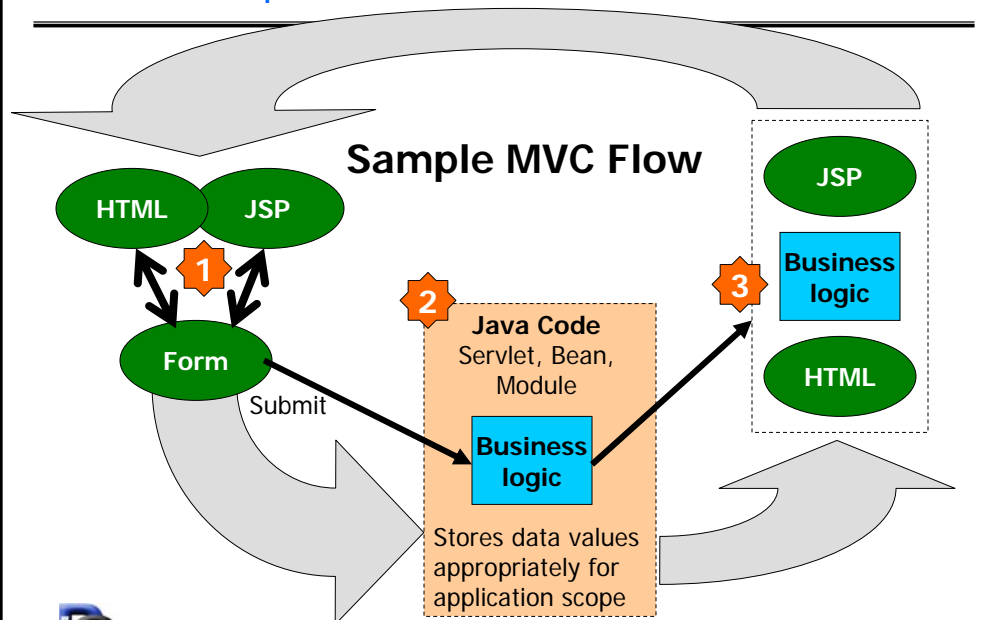


MVC - Flow

➤ As Sun details MVC:



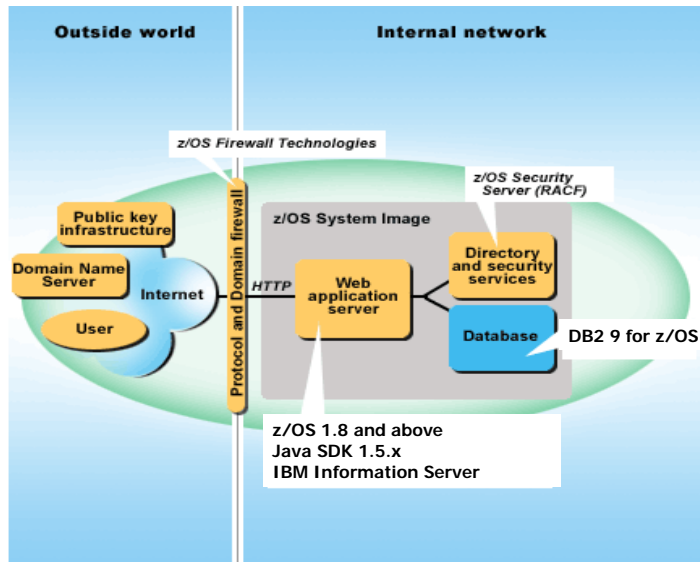
MVC – Sample MVC Flow



Common Components

➤ Web World

➤ z/OS



©Copyright IBM Corporation, 2004. All rights reserved.



Servlet Programming Pattern

➤ Do you have a "magic servlet"

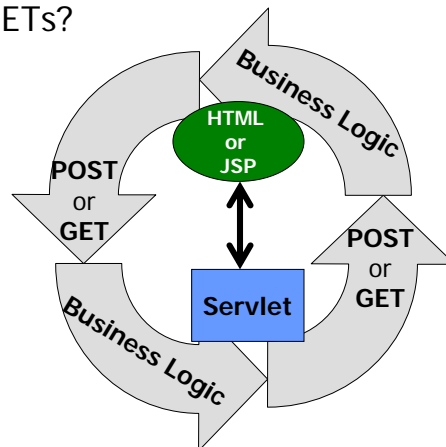
- Servlet that performs too many tasks

➤ Servlet have both POSTs & GETs?

- How many outputs?
- HTML, JSPs or other servlets

➤ How big is the servlet?

- 2k, 5k or 10k?



Servlet Performance Considerations

- Web applications are really batch oriented
 - Not interactive - nothing is waiting for a reply
 - Stateless internet environment
 - Remember – Init and Destroy (Final) routines for housekeeping
- Big servlets can result from web rewrites
 - Perl – scripting consolidated into a servlet
 - Cold Fusion
- Jakarta Struts, actions, implement the command style
 - Understand the servlet output paths and possibilities
 - Debug through log file, error pages and print statements
- Make the servlet appropriate size for simple functions
 - Break up big servlets into smaller modules



JSP Performance Considerations

- Marty Hall – Core Servlets & JSPs Book
- You have two options:
 1. Put 25 lines of Java code directly in the JSP page
 2. Put those 25 lines in a separate Java class and put 1 line in the JSP page that invokes it
- Why is the second option *much* better?
 - Development. You write the separate class in a Java environment (editor or IDE), not an HTML environment
 - Debugging. If you have syntax errors, you see them immediately at compile time. Simple print statements can be seen.
 - Testing. You can write a test routine with a loop that does 10,000 tests and recompile it after each change.
 - Reuse. You can use the same class from multiple pages.
- Validation of form values
 - Where is the form validated
 - Only validate what is necessary
 - No centralized validation
 - User Base has more CPU than your infrastructure
- Minimize code within the JSP
 - Single line instead of volumes of coding
 - How many dynamic links within the JSP page
 - How many is too many
- Minimize the number of different tag libraries
 - All Tags must be resolved
- One time code sections
 - INIT section – Executed when the
 - DESTROY section- Executed when the server deletes the servlet instance
- Remember web activities are **batch oriented**
 - POST or GET the appropriate level of data detail
 - Validate only what is needed for next page



Beans – Session, Entity & Messaging

- Is it State-ful or stateless?
 - Not a cookie
 - How much to remember
 - Depends on the service or task for the session bean
- Session Bean Services
 - Stateful lives forever on the server
 - Minimize memory of the stored items
 - Expire after a minimum amount of time
 - Entire session of one client – how many concurrent clients
 - Stateless – used to do something and then thrown away
 - Does not remember a thing
 - Big advantage for scalability
- **Entity Beans** provides the framework for mapping data
 - Durable data stores for caching
- EntityBeans provide the **getXXX** and **setXXX** capabilities
 - No arguments or empty constructor
 - No public variables
 - For boolean properties use **isXXX**
- **Java Messaging Services- JMS**
 - Stateless message service
 - Asynchronous messaging Does not wait for a reply
 - Provides a framework for messaging
- Framework for distributed messaging
 - Marshalling messages across networked server
 - Provides for recoverability and scalability
- Part of the EJB 2.0 specification
 - Uses Remote Method Invocation- RMI
 - Built on top of the CORBA and IIOP



Managing Bean Persistence – BMP vs. CMP

- The BMP Bean is managed by the developer
 - SQL against the database updates the bean properties/values
 - Older architectures execute the Bean for database access values
- CMP are managed through the EJB container
 - Container is managed through the web server vendor environment
 - Container lifecycle is maintained through vendor framework facilities
- Be a “Bean counter” - do the EJB memory math
 - Number of users
 - How many EJBs for each session
 - Caching 100, 200, 10000 entries?

Bean Managed Persistence	Container Managed Persistence
BMP way to handle data	Strategic way to cache data
Developer handles persistence	Mapping to the data through metadata
Hand coded optimized queries	Mapping/data optimized by vendor



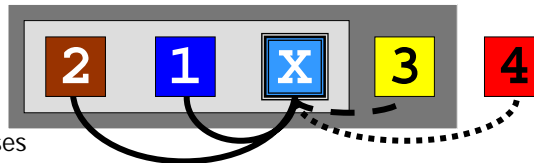
Java Virtual Machine

- JVM Heap Size
 - How big should your Heap be?
- Garbage Collection Frequency
 - How often do you take it out?
- Garbage Collection Time
 - How long does it take?
- Average JVM Heap Size After Garbage Collection
 - Does it make any difference
- Garbage Collection Frequency
 - Multiple GC at once
 - How long does GC take
 - How big performance impact is GC
- JVM garbage collector takes care of it
 - Clean up unreferenced memory items
- Testing Methodology
 - Concurrency test
 - How many users, application types and overlap duration
 - Long running test
 - Memory problem happen after application runs five hours
 - Repetitive test
 - Application memory issues
- JVM Settings
 - `-XX:MinHeapFreeRatio=%` , `-XX:MaxHeapFreeRatio=%`,
 - `-XX:NewSize=bytes` , `-XX:MaxNewSize=bytes`,
 - `-XX:NewRatio= value` , `-XX:SurvivorRatio= number`
 - `-XX:TargetSurvivorRatio= %` , `-XX:MaxPermSize=MB`
 - `-XX:-CleanPagesOnUncommit`
- Program compilation options
 - `-XX:-InlineUnreachableCalls`
 - Defaults are usually okay



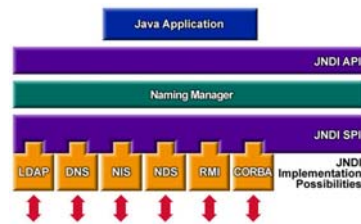
Many servers manage the Beans

- Distributed processing considerations
 - Make sure your outside methods are good performance partners
 - Have your performance critical path as local as possible
- Component should communicate with a limited number of other components
 - Components are local or remote or distance
 - Same server, another server in data center or remote data center
- Biggest memory users are always cleaned up - **java.lang.OutOfMemoryError**
 - ResultSet
 - Vectors & Arrays
 - HashTables
 - All within static classes
 - Event Listener



Connection Options

- JDBC is best performing
 - Driver Type - 1, 2, 3 or 4
 - Local or remote access considerations
- Several forms of error:
 - "TypeLoadException"
 - "No Suitable Driver"
 - "Error setting up the static cursor cache"
 - "Error establishing socket"
 - "Login has timed out"
 - Slow connection establishment
- JNDI – Java Naming and Directory Interface
 - <http://java.sun.com/products/jndi/tutorial/getStarted/overview/index.html>
- Connection Pooling
 - Reuse verification
 - API- Application Programming Interface
 - SPI - Service Provider Interface
- JNDI Settings
 - Local or Remote database connection
 - Security overhead
 - User authentication



Three different statement types

- Statement types
 - Dynamic, DAO or SQL

```
<%@ page import='java.util.ArrayList' %>
<%@ page import='java.util.*' %>
<%@ page import='java.sql.*' %>
<%@ page import='javax.naming.*' %>
<%@ page import='javax.sql.*' %>
```

1

```
<%
// =====Pool Connection=
Context initCtx = new InitialContext();
Context envCtx = (Context) initCtx.lookup("java:comp/env");
if(envCtx == null )
throw new Exception("Boom - No Environment Context");
// the following matches the resource name defined in dbv20.xml
DataSource ds = (DataSource) envCtx.lookup("jdbc/db001");
// =====Pool Connection=
// =====if=
if (ds != null) {
Connection con = ds.getConnection();

if(con != null) {
// =====if=
```

2

```
String query = "SELECT AMOD_MMBR_LIB, AMOD_MMBR, "
+ "AMOD_ENVR_ID, AMOD_APPL_ID, "
+ "AMOD_MMBR_TS, AMOD_EXTR_TS "
+ "from APPL_GUIDES "
+ "WHERE AMOD_ENVR_ID LIKE '" + envrname2 + "%' "
+ "AND AMOD_APPL_ID LIKE '" + applname2 + "%' "
+ "AND AMOD_MMBR LIKE '" + mmbrcode2 + "%' "
+ "orderbyOption2";

Statement stmt = con.createStatement();
log(getClass() + " : did the stmt ");

ResultSet rs = stmt.executeQuery(query);
log(getClass() + " : executed the query: " + query);

ResultSetMetaData md = rs.getMetaData();
log(getClass() + " : got the metadata array ");
```

3

```
while (rs.next()) {
for (int i=1; i<=colcount; i++) {
String applhdr = (rs.getString(i));
colnbr = colnbr +1
}
}
```

4

```
// important to cleanup
rs.close();
stmt.close();
con.close();
```

5



Statement Types

- Many advantages to SQLJ
- Developing static SQLJ SQL statements

- <http://www-128.ibm.com/developerworks/db2/library/techarticle/0302tsui/0302tsui.html>

- Example

Table 1. Comparing SQLJ and JDBC

	SQLJ	JDBC
Standards	ISO/ANSI (not part of J2EE)	Sun (part of J2EE)
SQL specification level	SQL-1999	N/A (must support at least Entry Level SQL-92)
Security	Strong	Average
Performance	Faster (static access plan created during development)	Slower (dynamic access plan created during applications program execution)
Syntax	High (compact)	Low (cumbersome)
SQLJ and JDBC interoperability	Yes	N/A
Type and schema checking	Strong (Performed during development)	Weak (Performed during runtime)



Use Scrollable cursors

- Page-able data

- Only get from the database what you are going to use
 - Google page Results 1 - 10 of about 118,000 for.....

- Scrollability

- <http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp?topic=/com.ibm.db2.udb.apdv.cli.doc/doc/c0007645.htm>

- ScrollForward only cursor

- Perform better
 - Less Overhead

- Many parameters for cursors

- Make sure scrolling is within centralized methods

- Default is auto-commit mode

- Each individual SQL statement is treated as a transaction
 - Will be committed as soon as its execution finishes

- `con.setAutoCommit(false);`

Cursor type	Cursor sensitivity	Cursor updatable	Cursor concurrency	Cursor scrollable
forward-only(1)	unspecified	non-updatable	read-only concurrency	non-scrollable
static	insensitive	non-updatable	read-only concurrency	scrollable
keyset-driven	sensitive	updatable	values concurrency	scrollable
Dynamic(2)	sensitive	updatable	values concurrency	scrollable

1. Forward-only is the default behavior for a scrollable cursor without the FOR UPDATE clause. Specifying FOR UPDATE on a forward-only cursor creates an updatable, lock concurrency, non-scrollable cursor.
 2. Values concurrency is the default behavior, however, DB2 on LUW also support lock concurrency, which will result with pessimistic locking.



User types and transaction types

- Transaction type effects – CPU, Memory and I/O
 - Sheppard, Farmer, Designer, Weaver, Buyer, Shipper, or Seller
 - **What are your user categories?**
 - **How much does each cost?**
- Number of occurrences
 - Concurrent, idle and ending badly
- Error Handling should be built into all modules
 - How many levels of method calls?
 - Have a thorough test plan
- Performance is within the project plan!



Questions for your system administrator

- How many SOA transactions are we configured for?
 - We are expecting 5M transactions doing 1M updates, 2M inserts with 15M-20M page views per day
- What are the hardware components of the server/LPAR?
 - Number of CPUs – Cores and the speed of the CPUs
 - Memory allocation for the LPAR
 - I/O connection speed for the Network
- What monitoring facilities are being used?
 - What CPU load statistics are available?
 - Current network traffic utilization is ????
- Web server platform software
 - Operating system, server level and patch level(s)



Questions for application development

- What framework are you using? For persistence?
 - How much persistence per user, session, transaction or idle thread?
- What automated testing tools are going to be used?
 - What are the performance expectations of the web services?
- Is the web content new or where does it exist now?
 - The services are 75% dynamic returning 500 rows of data each
- What are a list of the ^{Database}_{Web server}Application server_{Operating system} error conditions that are produced?
 - How can I help with the testing of the database conditions?
- What retry logic is within the web services?
 - What methods perform the retry logic? How many times?
 - How are the methods insuring transaction rollback/commit integrity?

