PREMIER ISSUE!

WebSphere® DEVELOPER'S JOURNAL

The World's Leading Independent WebSphere Developer Resource

WebSphereDevelopersJournal.com

PREMIER VOLUME 1 ISSUE 1

EXCLUSIVE NEXT MONTH...
An Interview with the Father of WebSphere

MANAGING JDBC CONNECTIONS

Programming for Performance and Scalability

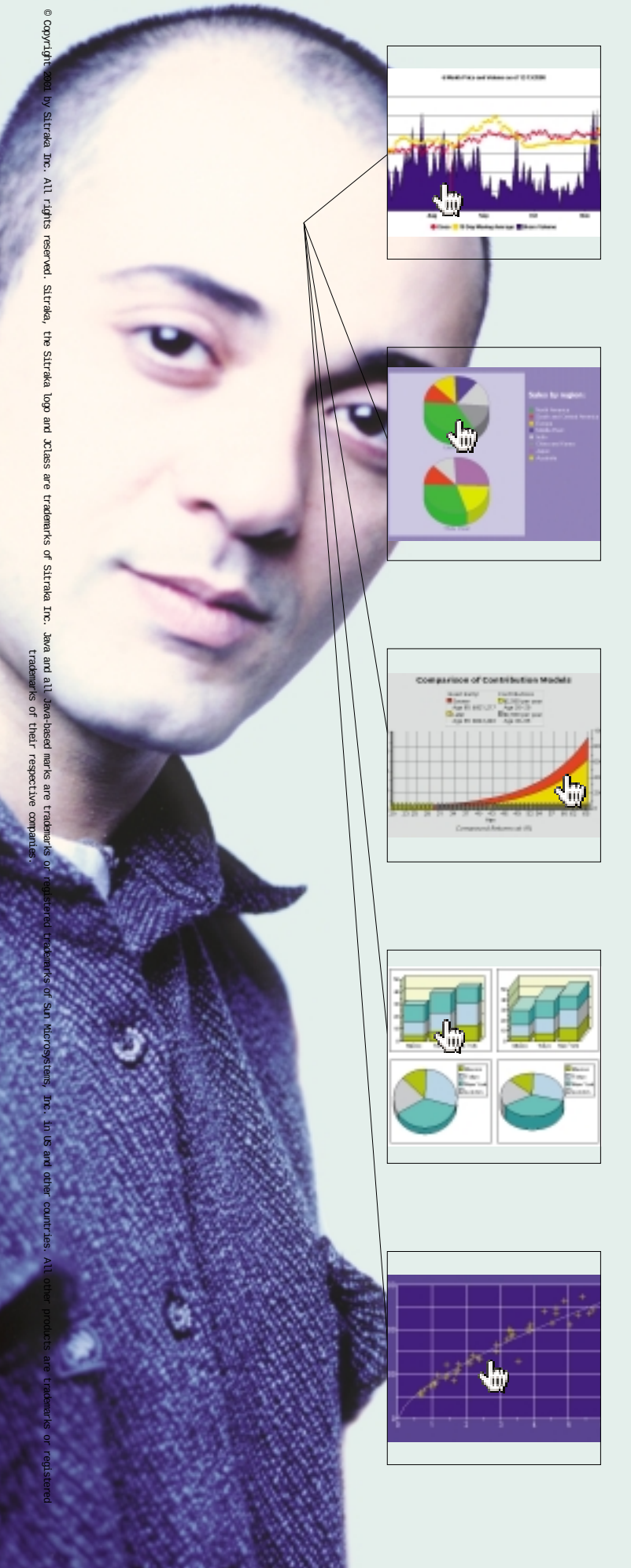
BY ANDY McCRIGHT PAGE 42

WEBSPHERE CONNECTION MANAGER...

SYS-CON MEDIA

WILY TECHNOLOGIES

WWW.WILYTECH.COM



sitraka

JClass™ ServerCha

Interactive, server-side charting

Sitraka JClass ServerChart: the first truly powerful solution for thin-client data visualization. With a range of chart types and built-in image map tag generation, JClass ServerChart brings interactive charts from your J2EE application server to the browsers of your entire network of users.

Includes multiple business and scien-tific
chart types

Integrates with all major application servers and IDEs

Works in headless environments when
used with eTeks' PJA toolkit

sitraka

the Java™ advantage

Download a free eval now

www.sitraka.com/jclassSS/wdj

1 888-361-3264 | Europe +31(0)20 510 67 00

WebSphere® DEVELOPER'S JOURNAL

SUBSCRIPTIONS
For subscriptions and requests for Bulk Orders, please send your letters to Subscription Department.
SUBSCRIBE@SYS-CON.COM
Cover Price: $15/Issue Domestic: $49/YR (12 Issues)
Canada/Mexico: $169/YR Overseas: $179/YR
(U.S. Banks or Money Orders)

EDITORIAL OFFICES
SYS-CON Publications, Inc.
135 Chestnut Ridge Road, Montvale, NJ 07645
Telephone: 201 802-3000 Fax: 201 782-9637
SUBSCRIBE@SYS-CON.COM
WebSphere Developer's Journal (ISSN# 1535-6914) is published monthly (12 times a year)
Postmaster send address changes to:
WebSphere Developer's Journal, SYS-CON Publications, Inc.
135 Chestnut Ridge Road, Montvale, NJ 07645

SYS-CON MEDIA

It's You, Not the Software

BY JACK MARTIN

The IBM WebSphere family of products can be a lot of different things to different people, and – just like any family – each member has its own strengths and weaknesses.

Some are fully developed and mature; others are just starting out. Each member has its own way of doing things, and of working with other family members. Some members are very cooperative and eager to work with outsiders; others stay close to home.

The most important part of the family, though, is not the software itself, it's the people who design, build, customize, deploy, and administer the software. Without all of you...there would be no WebSphere.

The professional life of a talented group of men and women at IBM is entirely focused on WebSphere, and literally thousands of companies have joined the family to help deploy, customize, administer, and create their own applications that run with it. There are currently over 20,000 installations worldwide that have chosen WebSphere for their company's mission-critical applications. These customers are people too.

The "family" is, of course, in reality a suite of software products for developing, testing, and deploying e-business applications. Built on industry standards such as Java and XML, and able to run on all major operating systems, such as Linux, NT, AIX, HPUX, Solaris, AS/400, S/390, NetWare, OS/2, and Windows 2000, WebSphere allows the integration of e-business apps with data on 35 different platforms and a multitude of third-party applications.

The WebSphere family brings together many of IBM's technologies with additional products and technologies from business partners. WebSphere contains a deep, J2EE-compliant implementation of leading open standards technology (including the latest Web services specifications) across a wide range of platforms, and complies with existing standards such as XML and UDDI, as well as with open technology platforms like Linux.

WebSphere Developer's Journal will bring you insights and examples each month from the industry's leading experts from both inside and outside IBM to help you get everything to work, connect, and scale in the environment you're working in.

In the coming months we'll focus on the foundation upon which everything is built, the WebSphere Application Servers and MQSeries products. These help provide the core functions of handling transactions and extend to back-end business applications. They also integrate business processes and applications and deliver them to the Web using Java-based application environments for building, deploying, and managing Web applications.

We'll also focus on the foundation extensions, including tools like VisualAge for Java, the IDE and toolset used to build business logic and connect to back-end data without having to write middleware code; on WebSphere Studio, the toolset for Web site developers building Web apps in a team environment; and on the WebSphere EveryPlace Suite, which helps connect Web and enterprise data to a wide range of non-PC devices, including wireless handsets, PDAs, and other Internet appliances.

We'll examine the WebSphere family's capabilities for security and systems management, and we'll visit the development, presentation, and deployment extensions and the application accelerators. Most importantly, we'll look regularly to the people behind the projects – the designers, the developers, and the administrators of WebSphere – to share their success stories. ***WSDJ*** wants to showcase just how good some of you are at what you do!

Our overall focus will be "WebSphere at Work" – that is to say, at work in real installations, from the initial business challenge through the production system in use today. With over 20,000 WebSphere installations worldwide we're expecting to hear from quite a few of you nominating your own interoperable e-business applications, from simple Web serving to the most sophisticated transaction-based solutions.

WSDJ's publisher, SYS-CON Media, is also the publisher of giants of the Internet technology world, like ***Java Developer's Journal*** and ***XML-Journal***, the world's leading Java and XML publications. As the newest member of the flourishing SYS-CON *i*-technology stable, we at ***WebSphere Developer's Journal*** have some very big footsteps to follow.

We fully expect ***WSDJ*** to become the leading WebSphere publication in the world. So whether you are a developer, an administrator, or an enterprise manager, we will strive in the coming months to make ***WebSphere Developer's Journal*** a part of your WebSphere family.

ABOUT THE AUTHOR... Jack Martin, editor-in-chief of ***WebSphere Developer's Journal***, is cofounder and CEO of Simplex Knowledge Company, an Internet software boutique specializing in WebSphere development. Simplex developed the first remote video transmission system designed specifically for childcare centers, which received worldwide media attention, and the world's first diagnostic-quality ultrasound broadcast system. E-MAIL... jack@sys-con.com

Promoting a clear separation between development and administration

Best Practices in WebSphere Application Packaging and Deployment

BY PRASAD THAMMINENI

Most of the time spent in an application development project is in developing and testing the application. Less time is actually spent designing and creating a repeatable and reproducible packaging and deployment model. A well-designed "build, package, and deploy" model has numerous benefits, including improved developer productivity, reduced turnaround time for builds and code fixes, better consistency in application code, and reinforcement of development policies.

ABOUT THE AUTHOR

Prasad Thammineni is the president of jPeople, a leading WebSphere and J2EE consulting services firm that offers architecture, mentoring, design, development, training, performance assessment, and tuning services. He and his team have designed numerous J2EE and WebSphere solutions since WebSphere 1.0 days.

E-MAIL

prasad@jpeople.com

In this article I present the best practices to use in packaging and deploying WebSphere applications. The best practices presented here have been applied to small and large enterprise projects with equal success. Although the focus of this article is WebSphere Application Server 3.5, it can be adopted to WebSphere 4.0 as well. I have created a fictitious application called PetStore to illustrate these practices.

What Are Packaging and Deploying?

Packaging is grouping logically-related classes into JAR files and staging these JAR files and related application runtime resources such as HTML and JSP pages, images, and property files into appropriate directories.

Deploying is installing the code on the destination WebSphere Application Server and configuring the application runtime components such as Web Applications, EJBs, servlets, and JSPs in WebSphere.

Characteristics of an Enterprise Application

A typical enterprise application has the following characteristics:

- It is implemented using servlets, JSPs, and EJBs.
- It is integrated with one or more enterprise datasources and applications.
- The source is version managed using version management software such as Rational ClearCase.
- There are multiple releases of the application. With every release there are new features, enhancements, and bug fixes.
- It is promoted through multiple stages – development, integration, test, and production.
- There are frequent (daily, weekly, and monthly) builds and installs within each stage.
- Developers, not administrators, know how the application components should be configured in WebSphere.
- Administrators, not developers, deploy the application in WebSphere.

Objectives of an Effective Packaging and Deploying Model

To satisfy the above listed characteristics of an enterprise application and to simplify the tasks in deploying such an application, a good packaging and deployment model should meet the following objectives:

- The same package should be deployable across multiple stages – development through production.
- The package should be deployable across multiple operating systems – such as NT and Unix – without any change.
- The configuration of the application in WebSphere should be scripted and the install should either be automated or be executed by administrators.
- Application environment–specific properties such as datasource locations and authentication informa-

tion should be configurable during deployment.
- The deployment model should promote a clear separation between development and administration. That is, developers should define and maintain how the application should be configured in WebSphere and administrators should deploy the application.
- If multiple Web applications are to be deployed on the same application server, each application's WebSphere configuration should be independent. This way each application can be maintained independently.

Best Practices

For the rest of the article I will discuss best practices which, when implemented, will satisfy the characteristics and objectives outlined in the previous sections.

Planning

It's always tempting to jump ahead and start building and deploying the application. However, I strongly recommend that you plan all the tasks needed to package and deploy your application before performing them. When planning for an application install in WebSphere you should identify:

- All application components such as classes, third-party libraries, HTML and JSP pages, images, and property files
- All enterprise datasources and the connectivity details the application would use during runtime
- The types of JAR files and what classes make up these JARs
- The deployment directory structure where the application components will be installed

JAR Packaging

Understanding the purpose and behavior of the different classpaths in WebSphere will help you identify the different types of JAR files and what classes go into these. Table 1 lists the WebSphere classpaths and summarizes the types of classes specified on each of these classpaths.

After looking at the different WebSphere classpaths, it is recommended that you package your classes into one or more JAR files, which fall into the following categories:

1. Servlets
2. EJBs
3. EJB client classes
4. EJB dependent classes
5. Classes common to EJBs and Servlets
6. Application classes that do not fall into any of these categories

CLASSPATH TYPE	DESCRIPTION
Node-dependent classpath	Classes referenced by EJBs in their home and remote interfaces and directories containing the property files that are loaded by these classes
Application server classpath	Nonreloadable classes such as: • Classes referenced from servlets whose objects are added to sessions • Classes that call Java Native Interface methods • EJB access beans and EJB Client classes • Classes that do not fall into any of the other categories and directories containing property files that are loaded by these classes
EJB classpath	EJBs and classes within the EJB JAR files. This class path is dynamically constructed by WebSphere when EJBs are deployed in the EJB container
Web application classpath	Servlets and reloadable classes referenced by servlets and the directories containing property files that are loaded by these classes

TABLE 1: Classpaths in WebSphere

Table 2 lists all the JAR files and the corresponding WebSphere classpaths for the PetStore application.

Deployment Directory Structure

Designing a meaningful directory structure for where your application components will be placed will simplify application management and maintenance. Table 3 lists a manageable directory structure for the PetStore application.

Application Environment Configuration

Datasource connectivity details change as the application is promoted from development through production. Externalizing these details into property files makes your code portable across stages. If the application uses WebSphere connection pools, maintaining the datasource JNDI names in property files will decouple the references made in application classes from the configuration.

For each of the stages, development through production, maintain separate sets of environment property files. During deployment, these property files can either be renamed or copied to the location where the application components look for these files.

Building Applications

As the number of classes, EJBs, and JAR files increases in your application, it is essential that you have a defined build process. This will ensure that your application is built in exactly the same manner each time a build is executed. You should identify, document, and automate all the steps to building your application as much as possible.

An automated build process will speed up the promotion of the application from one stage to another. It also removes any issues related to compilation errors and classpath issues that can cause wasted time and money.

Use build tools such as Ant to create build scripts, which not only

CLASSPATH TYPE	TYPE OF CLASSES	CLASSPATH
petstoreclient.jar	Servlets, access beans and other servlet referenced classes	Web application classpath
petstoreejbs.jar	EJBs	Deploy into the EJB container
petstoreejbclient.jar	EJB client classes and access beans	Application server classpath
petstoredependent.jar	Classes referenced by EJBs in their home and remote interfaces	Node-dependent classpath
petstorecommon.jar	Classes referenced by both Servlets and EJBs	Application server classpath
Petstoreclasses.jar	Classes that do not fall into any of the above categories	Application server classpath

TABLE 2: Table 2: PetStore application JARs

DIRECTORY	DESCRIPTION
petstore	Root directory containing all PetStore application components
petstore/ejbs	EJB jars
petstore/servlets	Servlet jars
petstore/lib	EJB Client JARs, dependent JAR files, common JARs, and third-party JAR files
petstore/properties	Application property files
petstore/web	HTML, JSP pages and image files
petstore/xmlconfig	XMLConfig scripts that will be used to deploy the application in WebSphere

TABLE 3: PetStore directory structure

compile your classes but also package them into JAR files and create distributable software. Ant is an open source Java-based build tool that lets you construct your build scripts in much the same fashion as the "make" tool in C or C++. You can use a large number of built-in tasks in Ant to create your build scripts. The build scripts, unlike make files, are XML documents. For the majority of applications, the built-in tasks are sufficient. If they're not, you can create your own custom tasks in Java.

In addition to creating build scripts it is recommended that you run these build scripts against the source located in your software configuration management tool. This ensures that your builds are using the latest version of the checked-in source.

Configuring Applications in WebSphere

You can configure an application in WebSphere either interactively, using the WebSphere Administrative Console, or from the command line using the XMLConfig tool. Creating scripts to configure applications offers these benefits:

- The same XMLConfig scripts can be used to configure the application across multiple stages.
- Scripts eliminate potential errors that are caused if applications are deployed otherwise.
- If you have Windows NT in development and Unix in the rest of the stages, the same scripts may be used to configure the application across operating systems.
- Developers who have the most knowledge of how the application components should be configured can update the XMLConfig scripts and administrators can execute them.
- If you need to restore a new server in cases of server failure, these scripts can speed up the process dramatically.

In this article I will not address WebSphere 3.5 security configurations as most applications do not use them and it isn't easy to script the definitions.

Creating XMLConfig Scripts

When creating reusable XMLConfig scripts I recommend the following procedure:

1. Use the administrative console the first time you are configuring the application in WebSphere.
2. Once you have the application tested and you are satisfied with the configuration, create a model from the application server. Models allow you to create configuration that is portable across stages and domains.
3. Delete all the clones, including the application server from which the model was created.
4. Export the administrative configuration into an XMLConfig file.

Once you have exported the XMLConfig script file, I recommend that you refactor the script into multiple scripts based on the following guidelines:

1. A node configuration script containing node-specific configurations such as the node-dependent classpath

SCRIPT NAME	ADMINISTRATIVE OBJECTS
node.xml	Node-dependent classpath and other node-specific configuration
resources.xml	Data sources and JDBC driver definitions
vhosts.xml	Virtual host definitions
model.xml	Application server model, EJB container, Servlet Engine, and session manager definitions
petstore.xml	PetStore Web application, EJBs, and Servlets
installjdbc.xml	JDBC installation definitions for a node

TABLE 4: Portable XMLConfig scripts for the PetStore application

2. A datasource configuration script containing datasource and JDBC driver definitions
3. A script for virtual host definitions.
4. An application server model script containing the application server, EJB container, EJBs, servlet engine, Web applications, and servlet definitions
5. If you have multiple Web applications you can create one script for each application that contains the application-specific Web application, EJBs, and servlet definitions
6. A script that is used to install JDBC drivers on each node

Based on these guidelines, you would refactor the XMLConfig script for the PetStore application into multiple scripts as shown in Table 4.

As a best practice add these scripts to your software configuration management repository and promote them through the various application stages. Also, while packaging the application include these scripts in the package. This way when the package is installed on the destination server, the scripts needed to deploy the application are available on the server.

Making XMLConfig Scripts Reusable Across Stages

Once you have created the multiple XMLConfig scripts the next step is to make these scripts reusable across multiple stages. The following elements tie the scripts to a particular stage or environment:

1. IP addresses and database aliases used in data source definitions of the resources.xml script
2. Authentication details specified on the EJB container, CMP EJBs and Session Manager of the model.xml and petstore.xml scripts
3. Host aliases specified as attributes for the virtual host in the vhosts.xml script

Replace these configuration elements with XMLConfig substitution variables. For example, replace the occurrence of a host IP address with "$HOST_IP$". When executing these scripts, specify the values to substitute for these variables at the command line.

Making XMLConfig Scripts Portable Across Operating Systems

There are three major differences between Windows NT and Unix that affect our XMLConfig scripts:

1. ***The path separator:*** On NT the class path separator is semi-colon ";" and on Unix it is colon ":".
2. ***The directory separator:*** On NT the directory separator is backslash "\" and on Unix it is the forward slash "/".
3. ***The root drive:*** On NT you have a root drive letter called C: and on Unix you have none.

To make the scripts portable across platforms:

1. Replace all occurrences of the path separator in the classpaths with the XMLConfig tool's predefined substitution variable $psep$.
2. Replace all occurrences of the NT-specific directory separator "\" with "/". The Java virtual machine treats the forward slash as it would treat a backslash on Windows NT.
3. If the application is installed on the same drive as Windows NT, you could safely omit the drive letter from the classpath.

Deploying the Application

Deploying the application is a two-step process. First, you have to copy the application software to the destination server. Second, you have to import the XMLConfig scripts into WebSphere.

When importing the scripts, you need to specify the values to substitute for the variables that you used in the scripts. As a best practice, you can create convenient shell scripts that can be used to import the scripts with the appropriate parameters. If you want to run these shell scripts on NT, you could install software such as CygWin, which provides Unix-like utilities for Windows.

Summary

If you follow the best practices discussed in this article you can create a clear separation between the responsibilities of developers and administrators, and simplify building, distributing, and configuring WebSphere applications across multiple stages and platforms.

Resources

- Gissel, Thomas R. "Understanding WebSphere Classloaders." http://www7.software.ibm.com/vadd-bin/ftpdl?1/vadc/wsdd/pdf/gisell/UnderstandingWebSphereClassLoaders.pdf. Describes the different classpaths and provides tips on how to package applications.
- Ant – a Java build tool from Apache: http://jakarta.apache.org/ant/index.html
- Cygwin – the popular GNU development tools and utilities for Windows: http://sources.redhat.com/cygwin

with

"Son, I tried once, and I failed. The moral of the story is: never try." – Homer Simpson

Iterative development is a process of growing software into existence. While the thought of software growing may sound a little strange, that's exactly what happens. As you iterate through the development process, more and more behavior is added to your application. With each new addition, your application gets closer and closer to a finished product. One of the really neat things about building an application iteratively is that it is almost always in a state where it can run. When iterative development is done right, you are never more than 15 minutes away from a demo. That's really powerful.

In the reality of today's software development requirements, iterative development is the only choice. Where development cycles were once measured in years, they are now measured in months. Where users were once measured in tens or hundreds, they are now measured in thousands and millions. By employing iterative development techniques you can bring software to market quickly (albeit incrementally). Additional functionality can be either scheduled or evolved to accomodate changes due to unanticipated requirements.

Iterative development is more than just an "analyze, design, implement, test" cycle repeated feature by feature. The stages of development are visited and revisited throughout the cycle. While implementing a single feature, you may have to revisit and modify a single method a number of times to adapt to evolving understanding. Iterative development requires discipline. Managed poorly, it is little more than hacking; the net result of which is very often a tangled mess of code that is going to cause endless grief and cost a lot of money over its lifetime. Iterative development involves change. As change is introduced into your application, things will break. By maintaining a comprehensive set of tests, you can quickly find and fix what's broken.

Iterative Development from the Top Down

A development iteration starts with a problem description; usually a short textual description. To demonstrate this process, we'll implement a simple object broker to handle storage and retrieval of BankAccount objects to and from a relational database.

ABOUT THE AUTHOR

Wayne Beaton is a senior software consultant for Software Services for WebSphere, part of the IBM Software Group. His diverse role involves him in lots of interesting stuff, from the WebSphere Skills Transfer and Migration programs to general consulting. Wayne likes to spend his free time convincing people that Extreme Programming, Refactoring, and Unit Testing actually work.

E-MAIL

wbeaton@ca.ibm.com

Iterative Development VisualAge for Java

Visiting and revisiting your code throughout the cycle

BY WAYNE BEATON

An Object Broker is responsible for handling the life cycle of an object (an Enterprise JavaBean Home is an example of an object broker). In its simplest form, an object broker takes care of translating objects into rows in a relational database and rows from a relational database into objects. That is, the object broker abstracts the implementation of our persistence mechanism. Rather than dealing directly with Java Database Connectivity (JDBC) code, our application code interacts with JDBC indirectly through the object broker (see Figure 1).

Some understanding of what the object broker needs to do is required before starting the construction. Let's start with an example:

```
BankAccountBroker broker = new
BankAccountBroker();
broker.insert(new BankAccount(42));

BankAccount account =
broker.findByNumber(42);
account.credit(1000.0);
broker.update(account);
```

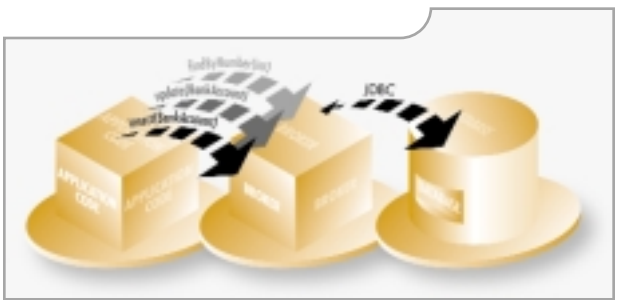


FIG. 1: Object brokers provide a mechanism for application code to indirectly make use of JDBC.

In this example, there are three things to test: inserting, finding, and updating bank accounts. With three things to test, at least three test cases are needed. More tests are probably required, but they can be addressed after the initial implementation. For example, the case where an attempt is made to insert a BankAccount that already exists should be considered.

Before starting to write code, some Java packages (and a VisualAge for Java Project) must be defined. The object broker is going to involve domain model objects (BankAccounts), object brokers, and tests for the object brokers. Appropriate packages are defined accordingly:

```
_ com.wtb.models;
_ com.wtb.brokers;
_ com.wtb.brokers.tests;
```

As a matter of practice, test code should be in a separate package from the code being tested. Classes have special access to the protected members (fields or methods marked as protected) of classes defined in the same package. If test classes were to be defined in the same package as the classes being tested, then the accessibility of the classes might not be effectively tested. More pragmatically, tests are useful for development, but are not so useful for deployment. By keeping the tests in a completely different package, deployment becomes a simple matter of excluding the test package.

JUnit (www.junit.org) is a useful tool for developing tests. In JUnit, the test cases are written in Java and are subject to the same version control as the code being tested. By keeping the tests in the same VisualAge project, you are virtually guaranteed that everywhere the code goes the tests will follow. At the same time, if the tests are maintained along with the code, the VisualAge version control can guarantee that the code and the tests are always synchronized.

The basic building block in JUnit is the TestCase class. All tests are derived from this class, including the BankAccountTests class, which contains code to test the BankAccountBroker:

```
package com.wtb.brokers.tests;

import junit.framework.*;
import com.wtb.brokers.*;

public class BankAccountBrokerTests extends
TestCase {
  public BankAccountBrokerTests(String name)
{
        super(name);
  }
  public void testFind() {}
  public void testInsert() {}
  public void testUpdate() {}
}
```

By default, in a JUnit TestCase all methods prefixed by "test" are considered individual test cases and will be run automatically by the test runner. The test methods are not invoked in any particular order and are not expected to depend on other tests (that is, you should not generally define tests that depend on the results of other tests; each test is expected to clean up after itself). As they currently exist, these tests are not particularly interesting, the actual test code has not been filled in yet. In fact, the code being tested hasn't even been defined yet. The remaining steps are an exercise in filling in these methods and making them run. Once these methods have been created, and the tests run successfully, this part of the iterative development process is complete.

"WHEN ITERATIVE DEVELOPMENT IS DONE RIGHT, YOU ARE NEVER MORE THAN 15 MINUTES AWAY FROM A DEMO. THAT'S REALLY POWERFUL."

The easiest way to test to see if an insert works is to see if a row has been added to the database. For that, a little JDBC code will be required in the test method:

```
public void testInsert() {
  // set up the objects we need for the test
BankAccount account = new
BankAccount(42);account.credit(1000.0);
  // actually do the insert
  broker.insert(account);

   // Evaluate the results. If the insert
   // works, then we should find a row
   // in the BANKACCT table with the
   // corresponding values.
   ResultSet results = statement.executeQuery
   ("Select ID, BAL from BANKACCT where
     ID=42");
  if (!results.next()) fail("Account number
                             42 not found!");
  assertEquals(42, results.getInt(1));
  assertEquals(1000.0, results.getDouble(2),
0.0);
}
```

The last chunk of code in this method verifies the result of the test. When the testInsert() method runs successfully, a row with values corresponding to the data in the inserted BankAccount should exist in the database. If no such row is found, then the test simply fails and a message is reported. If a row is found, its contents are inspected to ensure that they contain the expected values.

The testInsert() method makes use of a number of classes and methods that do not exist yet, which will cause the compiler to warn of a lot of problems (see Figure 2). Some of these problems can be worked out immediately. For example, the use of the ResultSet class will be flagged as an error that can be resolved by importing the java.sql package. Most of the other problems can only be solved by writing more code. As it is, this method can be saved, but will appear with a red "X" beside its name to indicate that a problem exists (see Figure 3).

Many developers see the red X as something to be avoided at any cost (after all, it represents a problem in the code). This can lead to a bottom-up approach when writing software. It is better to think of the Xs as placeholders indicating what has to be done next. There is a lot of benefit to top-down development (as is demonstrated here); in particular, you build what you actually need, not what you think you need.

A nice feature of VisualAge for Java is the "All Problems" page (see Figure 4) in the Workbench. As might be expected, this page puts all code problems in one place for easy identification and editing. This page shows both errors and warnings (use of deprecated methods, for example), which makes finding and solving problems relatively easy.

VisualAge for Java's browser-centric paradigm is very helpful for iterative development. The testInsert() method can be opened in its own window and left open while the necessary pieces are built (keeping it handy for reference). In fact, any number of windows can be opened on arbitrary parts of the code. We might, for example, have one window open on the testInsert() method, and another

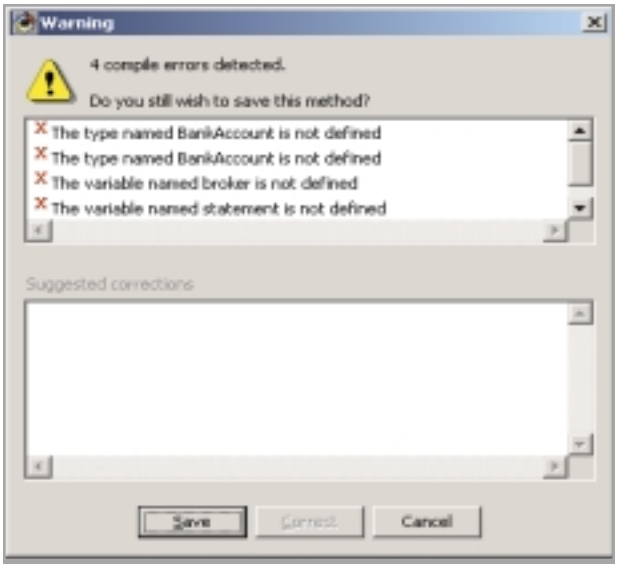


FIG. 2: VisualAge reports problems that prevent your method from compiling. You can save a method with problems.

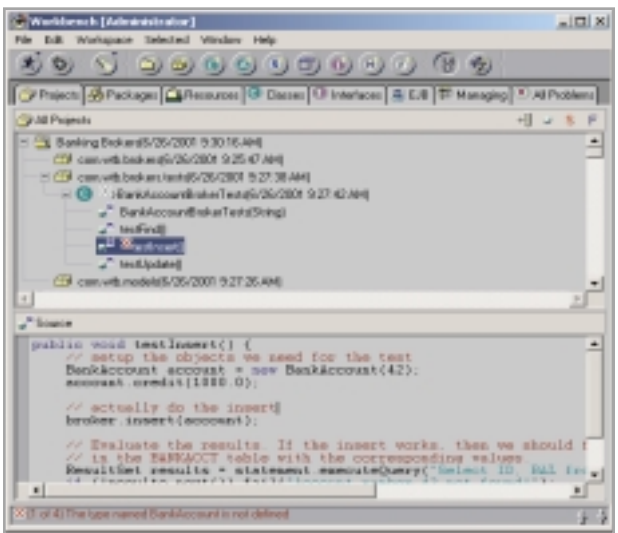


FIG. 3: VisualAge marks methods with problems with a big red "X". The gray "X" beside the class indicates that one of the class's members has a problem. The status pane describes the errors for the selected element.

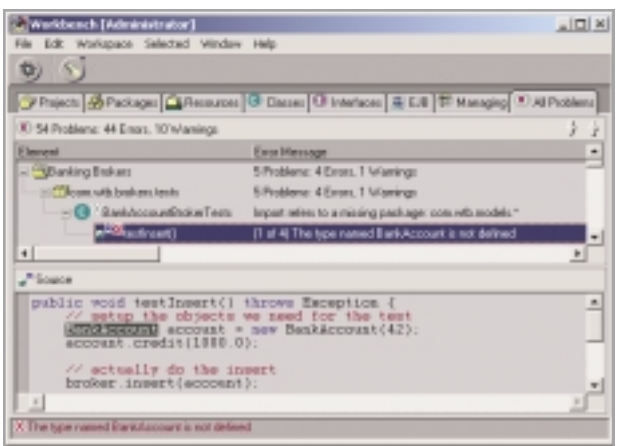


FIG. 4: The "All Problems" page in the Workbench localizes all existing problems (both errors and warnings) in all code currently loaded in the workspace. You can edit code directly in this page.

"VISUALAGE FOR JAVA'S BROWSER-CENTRIC PARADIGM IS VERY HELPFUL FOR ITERATIVE DEVELOPMENT"

open on the entire BankAccountBrokerTests class. Each window will update automatically to reflect changes in the other.

The immediate task is to make the testInsert() method work. Starting from the top of the method, it is clear that the com.wtb.models.BankAccount class with a one-argument constructor and a credit(double) method is needed. This is left as an exercise for the reader (don't forget to build tests).

Next comes the broker field. The BankAccountBrokerTest class is modified to include a field named "broker" with type com.wtb.brokers.BankAccountBroker. The field is initialized using the setUp() method:

```
public void setUp() {
  broker = new BankAccountBroker();
}
```

The setUp() method is called before each test method is run. A corresponding tearDown() method is invoked after each test. With the broker field populated, attention is focused on the insert(BankAccount) method for the BankAccountBroker class. Of course, the BankAccountBroker class does not exist yet and must be defined:

```
public class BankAccountBroker {
}
```

The insert(BankAccount) method is required by the BankAccountBroker class to convert the object into something that can be stored in the database.

```
public void insert(BankAccount account)
throws SQLException {
  Connection connection = null;
  PreparedStatement statement = null;
  try {
        connection = getConnection();
        statement = connection.prepare
        Statement("insert into BANKACCT (ID,
        BAL) values (?, ?)");
        statement.setInt(1, account.get
                         Number());
       statement.setDouble(2, account.get
                         Balance());
       statement.execute();
  } catch (SQLException e) {
       throw e;
  } finally {
       close(statement);
       close(connection);
  }
}
```

This method uses a standard JDBC PreparedStatement to insert values into the BANKACCT table. As this method was being constructed, it became clear that the SQLException, which is thrown by every JDBC method, must be handled. This method's answer to an SQLException is to clean up the resources it used and pass the exception on to the caller. At a later time, alternative exception handling might be explored, but for now, passing the SQLException gets the job done.

The insert(BankAccount) method calls a few methods that don't exist. The getConnection() method, as the name implies, gets a database connection:

```
private Connection getConnection() throws
SQLException {
  DriverManager.registerDriver(new
jdbc.idbDriver());
  return
DriverManager.getConnection("jdbc:idb:c:/idb
/test/ns.prp");
}
```

The getConnection() method really should use a DataSource to get a connection (at the very least, the JDBC driver and URL should not be hard-coded). DataSources provide connection pooling, but also require quite a lot of additional coding to make it work. Using the DriverManager from classic JDBC makes the test work without adding a lot of complexity. When this type of compromise is made, it is important to make a note to come back to this method and fix it.

With the getConnection() method in place, attention turns to the two "close" methods at the end of the insert(BankAccount) method. These methods are very simple in purpose and design. When closing a statement or connection, SQLException must be handled. Aesthetically speaking, handling these exceptions directly will make the code bulkier and harder to read; besides, exposing the exception handling code will not add any value. The close(Statement) method is shown below (the corresponding close(Connection) method is virtually identical):

```
private void close(Statement statement) {
  if (statement == null) return;
  try {
       statement.close();
  } catch (SQLException e) {
       // Ignore and move on.
  }
}
```

Note that the ResultSet is not explicitly closed; according to the JDBC specification (www.javasoft.com/jdbc), ResultSets automatically close when the statement is closed.

FIG. 5: With the testInsert() method highlighted, VisualAge displays the problems with the method in the status bar at the bottom of the window

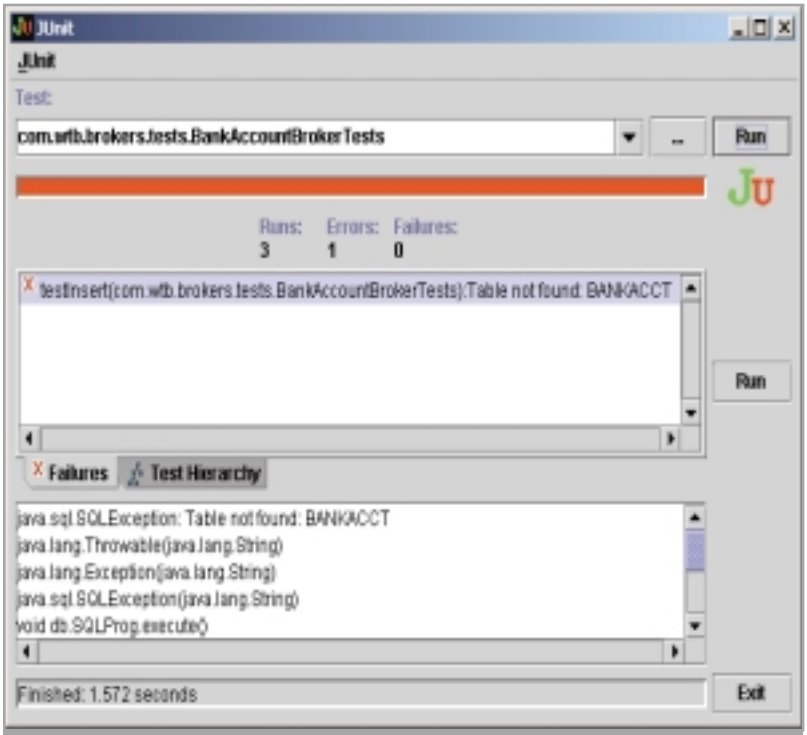


FIG. 6: The status bar at the bottom of the VisualAge code browser lists the errors in the selected method.

At this point, the BankAccountBroker class appears to be complete (though we won't know for certain until it passes our tests). However, the testInsert() method in the BankAccountTests class still has errors that must be resolved. This is how this method currently appears:

```
public void testInsert() {
  // setup the objects we need for the test
  BankAccount account = new BankAccount(42);
  account.credit(1000.0);

  // actually do the insert
  broker.insert(account);

  // Evaluate the results. If the insert
  // works, then we should find a row
  // in the BANKACCT table with the
  // corresponding values.
  ResultSet results = statement.executeQuery
  ("Select ID, BAL from BANKACCT where
    ID=42");
  if (!results.next()) fail("Account number
    42 not found!");
  assertEquals(42, results.getInt(1));
  assertEquals(1000.0, results.getDouble(2),
0.0);
}
```

With the testInsert() method highlighted, VisualAge displays the problems with the method in the status bar at the bottom of the window (see Figure 5). The status bar indicates that the statement field has not been declared. Remember that the purpose of the last chunk of code in this method is to ensure that the insert actually worked by checking the database to see if a corresponding row has been added. Given this understanding, the last chunk of code requires a statement from a connection to the same database employed by the BankAccountBroker class.

The statement and connection fields are added to the test case as fields of type java.sql.Statement and java.sql.Connection respectively. The setUp() and tearDown() methods are defined accordingly:

```
public void setUp() throws SQLException {
  broker = new BankAccountBroker();

  DriverManager.registerDriver(new
                          jdbc.idbDriver());
  connection = DriverManager.getConnection
   ("jdbc:idb:c:/idb/test/ns.prp");
  statement = connection.createStatement();
}

public void tearDown() throws SQLException {
  statement.close();
  connection.close();
}
```

With the new fields defined, you might expect that the job is done. However, VisualAge insists on continuing to mark the testInsert() method with an X. With a little help from the status bar, it seems that the testInsert() method needs to handle a number of exceptions. This is an easy fix: Exception is included in the throws clause for the method. JUnit will catch the exception and report it if it is thrown.

The JUnit user interface (see Figure 6) is opened from the Scrapbook by executing junit.swingui.TestRunner.main(new String[] {}). To run the tests, the full name of the test class (com.wtb.brokers.tests.BankAccountBrokerTests) is entered into the entry field at the top of the window and the "Run" button is pressed (note that the "..." button, which browses the file system for Java classes, does not work in VisualAge).

The red bar that stretches across the windows indicates that at least one test failed. The numbers (just below the bar) indicate that three tests have run and one error occurred. The test method containing the error is indicated in the list below. Recall that three test methods were built (testInsert(), testUpdate() and testFind()), but only one of these test methods actually does anything.

At the bottom of the JUnit window, the stack trace for the highlighted method is shown. According to the stack trace, the table has not been defined in the database. The table is defined and the tests are re-run. This time, all the tests pass. However, when we run the tests a second time we have a problem: the second time the testInsert() method is run, the row already exists (it was inserted by the first test). For an automated test to be truly useful, it must be repeatable. To be repeatable, the testInsert() method must undo the insert after it is complete.

Unit testing always gets messy when external entities are involved. In this case, the object broker depends on a

particular database manager being installed on our system. Further, that database manager must be set up with an appropriate database and table for the tests to work. Unit testing purists might suggest that these are not unit tests at all due to the dependence on manually-configured external entities – and they might be right.

Cleanup can be accomplished by recreating the database for each test. This is a bit of a sledgehammer approach, but it guarantees that the database is in a pristine state for each test. Alternatively, each test can simply delete the rows it creates at the end of the test. For now, the tests and tables are so simple that the simplest thing that could possibly work is to recreate the tables with each test. When this becomes too difficult, alternatives will be explored. To do this, the setUp() method in BankAccountBrokerTests is updated to drop and then re-create the table:

```
public void setUp() throws SQLException {
  broker = new BankAccountBroker();

  DriverManager.registerDriver(new jdbc.idbDriver());
  connection = DriverManager.getConnection("jdbc:idb:c:/
               idb/test/ns.prp");
  statement = connection.createStatement();

  statement.execute("drop table BANKACCT");
  statement.execute("create table BANKACCT (ID integer
                     primary key, BAL numeric(7,2))");
}
```

Conclusion

The process demonstrated here takes about 10 minutes out of the day of a reasonably productive VisualAge for Java developer. The code is not perfect. In fact, it is flawed in a number of ways. However, it does provide the functionality that is required. Further, that functionality has been implemented in the least complex way possible. As the requirements are better understood, the code will be evolved. Over time, the code will grow in sophistication and complexity. With care, the complexity can be managed.

VisualAge for Java provides unparalleled support for iterative development. VisualAge's browser-based code-editing paradigm permits code to be viewed and edited as projects, packages, types, or methods. With multiple windows, a developer can effectively edit many parts of the application simultaneously, an absolute necessity for iterative development. The "All Problems" page in the Workbench can be used to find and resolve code problems quickly.

The iterative development process is more than just coding a feature and testing it. Each iteration in the process is itself composed of many fine-grained iterations. By making these iterations, tests are run more frequently so small problems are found before they become big problems and the process runs more efficiently. Very often tests are the first things to get cut when time gets tight and this is a big mistake. Without effective tests in place, iterative development can quickly degenerate into a coding nightmare.

Iterative development works well from the top down. By writing the test case first and then the code, you set the stage in advance for what needs to be built. When building from the bottom up, there is a tendency to build too much, or to build the wrong things. Top-down development, with a little help from JUnit and the little red Xs, builds exactly what you need.

This month, I've tried to provide a glimpse into the iterative development process. In future articles, we'll evolve the object broker (and figure out where it fits in the architecture) with a little help from the VisualAge debugger.

References

1. Fowler, M.(1999). *Refactoring: Improving the Design of Existing Code.* Addison-Wesley.
2. Davis, M. "Incremental Development with Ant and JUnit, VisualAge Developer Domain," www.106.ibm.com/developerworks/library/j-ant/?n-j-11160

Faster, Cheaper, Smarter

BY BOB GAULT

We've all seen the headlines and understand the recent downturn of the economy and of business in general, but what we in the technology community continue to struggle with is hearing the omnipresent mantra of "faster, better, and cheaper." Demonstrating increased value within these constraints becomes one of the key drivers for all types of technology projects – from day-to-day software maintenance activities to delivery of that one Web-enabled function that will placate corporate business owners until the next "must have" wave of technology emerges.

Within the retail or business-to-consumer community, new business models incorporating the Internet have evolved so quickly that most organizations have worked endlessly and invested large sums of money to keep up with the demands of business owners and consumers. Every business had to have a Web presence just to stay competitive. In reaction to this environment, executive management was forced to review numerous business plans that determined the Internet strategy of the company and approve funding for Web-based technology projects where the break-even point was often five years or longer. In some cases, there was no break-even; however, companies were required to invest in these projects to maintain both brand recognition and, ultimately, market share.

These new Internet business models have also had an intriguing effect on many technical staffs. Not only did the skills required expand but the very look and feel of Information Technology (IT) departments changed. We find ourselves doing battle daily to retain the highly skilled associates in whom we invested heavily. We also had to spend time and money attracting associates who had skill sets based around the new economy that was developing. This new economy did not require one or two new skills; it required a level of knowledge and sophistication unseen in the technology community. Suddenly, we were faced with the complexities of the Web layer, application layer, data layer, virtual private networks, 24/7 operations, worldwide audiences, security, and those outsiders who wanted nothing more than to hack their way into your systems. The challenges have been difficult, costly, time-consuming, and sometimes not fully understood, yet they continue to evolve at breakneck speed.

These statements are not necessarily news to anyone at this point, but the downturned economy and current market conditions have changed the way we think about our businesses. We must reflect on how to stay competitive, how to retain our highly skilled associates, how to increase market share, and how to maintain brand recognition within the confines of stunted revenue growth and extreme cost pressures.

We are now seeing investments in technology projects being more closely scrutinized; departmental head counts are remaining static or being reduced, and the costs of materials continue to rise. As a result, each dollar spent carries a greater weight than it did three to five years ago. We see companies slowing down the pace of technology changes by stepping back and evaluating each change for its viability and value to the organization based on its likely return on investment. While there has been a slowdown in overall spending for technology projects, funding has not stopped – it is just being analyzed and reviewed in a more diligent manner.

Evidence of this smarter spending is demonstrated by the manner in which companies are aligning themselves with selected technology partners. A recent article from the Reuters News Service, on a study conducted by International Data Corporation (IDC), states that there are early signs of a trend in which application software buyers stay loyal to firms who have supplied them with other technologies. The article stresses that this is especially true in the application server market, where people who previously purchased from IBM are electing to use WebSphere.

In the retail industry, WebSphere has been implemented by many national and international chains, and not just for the business-to-consumer segment of the business. WebSphere Application Server offers options that allow businesses to maximize control over their infrastructure by enabling them to choose how they respond to the changing marketplace. The specific demands of the retail industry require scalability to meet the changing workload and market requirements while also handling the large number of product categories, combinations, and options offered. Exceeding these demands is key in driving the retail business. Whether the client is a large international book and music retailer or a nationally recognized major department store, WebSphere Application Server provides the power and flexibility needed.

As I said earlier, other challenges faced by Information Technology departments include ways to improve efficiency and to obtain faster turnaround on critical technology projects. WebSphere provides a flexible software infrastructure that enables the development of applications on open industry standards within multivendor environments. WebSphere provides high performance, scalability, and security to maximize application integrity so customers can access a firm's Web site and complete transactions quickly and securely – this is why they return over and over again.

The bottom line is that as companies continue to make smarter technology decisions and align themselves with strategic technology partners, software selection and vendor support are of paramount importance. Price protection coupled with evolving and robust functionality is expected. This continues to be the sweet spot for IBM and WebSphere in the model of "faster, better, and cheaper."

ABOUT THE AUTHOR... Bob Gault is vice president, Distribution Industry, for IBM Global Services in the Americas. He is part of a team dedicated to the needs of customers in retail and wholesale distribution, consumer packaged goods, and computer services industries. His primary focus is strategic outsourcing. A graduate of Cornell University with a degree in industrial engineering and computer science, Bob is also a former U.S. Naval Officer.

E-MAIL
gault@us.ibm.com

CORDA TECHNOLOGIES

WWW.CORDA.COM

BUILDING AN APPLICATION

ESTABLISHING A SOLID FOUNDATION FOR YOUR NEW HOME

ABOUT THE AUTHOR

Joe Farsetta is an engineer with more than 20 years of industry experience in telecommunications, networking, operations, business process architecture, applications, and support. An entrepreneur and inventor, Joe's past engagements have included Unilever, NJ Transit, and a regional directorship at Bell Atlantic Network Integration. He is currently working as an independent consultant in the New York metropolitan area.

E-MAIL

jjf10965@yahoo.com

Your latest assignment includes building those applications and application servers that will propel your firm's business to the next level. Where to physically locate these new servers is the question, though. So, it's up to you to help design or evaluate new or existing data center space. Not sure of how to do this? It's not an insurmountable task if you know the basic building blocks which go into every comprehensive data center. This article will steer you through the many options available, and the factors that affect a data center's design.

BY JOE FARSETTA

SERVER DATA CENTER

Data centers no longer only include the cavernous mega-centers of the past. With processing power fitting into a smaller physical footprint, and smaller companies requiring less processing equipment than the much larger enterprise-class company, data centers now range in size from huge facilities to spaces no larger than a janitor's closet. Basic data center characteristics remain constant, however. The key is to build the right data center for your application servers. With this comes the realization that serious up-front planning is required prior to deploying your application servers, whether in a new data center or in an existing one; whether large-scale or small. Again, the formula remains constant.

Your firm has decided to outsource its data center? The same questions should be asked of your outsourcing partner. How robust is their data center, really? Remember that beauty is only skin deep. The same is true for data centers. After you've peeled that onion skin back a bit, you may be pleasantly surprised, or shocked. So, be prepared for both.

Core Elements

No one consciously builds a house on a shaky foundation. The same logic should naturally follow for any data center. One would think that was the case, but you would be surprised to know how

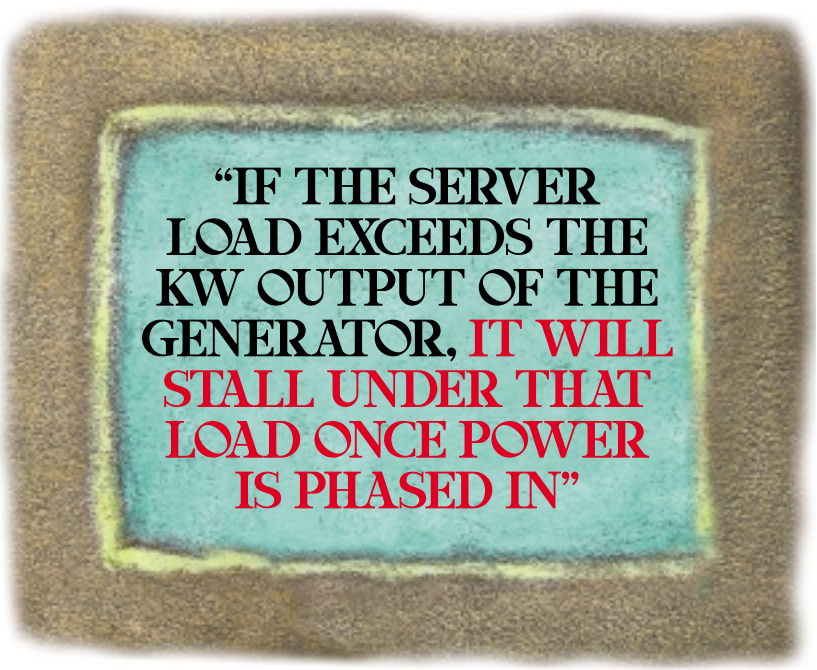
many servers are truly exposed out there. With this in mind, it's best to establish some Service Level Objectives (not to be confused with a Service Level Agreement) prior to digging into your data center design. Service Level Objectives are just that: the minimum level of service your firm has reasonably established in terms of server reliability, performance, maintenance windows, up-time, etc. Some of these concepts bridge all three elements of application-server data center design. I call them core elements. They are: *physical characteristics*, *logical characteristics*, and *criteria of operations*.

PHYSICAL CHARACTERISTICS

We'll start with the most basic element: physical characteristics. Components include space, power, environmental conditions, physical security, fire suppression, lighting, and workspace.

Space

This is a basic concept, but one that can sometimes be underestimated on the importance scale. When planning the application server data center, many factors play into the space question. For instance, are your servers rack-mountable, shelf-mountable, or freestanding? Is the only space available near any windows? How is conditioned air pumped in? Does it flow down from the ceiling, up from a raised floor, or does it flow through the open air? What is the planned density of servers per square foot of floor space? How much aisle space will be allowed for physical connectivity, maintenance activities, and daily care and feeding of these servers? How about access into the data center? Not only from the human side (you wouldn't want to use space you can only access through the men's room), but from the Telco side (how far away is Telco's D-Mark?). Likewise, locating your servers under roof drains, condensate drains, or any other drain is bad enough, but locating them under any pressurized water lines is outrageous. Raised floor systems are nice, but not absolutely necessary. Where raised floors are unavailable, overhead cable-trays or ladder racks will do just fine. As you can see, these questions are all critical and require real thought.



Power

Easy, right? A few receptacles here and there, some power strips or surge suppressors, and you're in business. Well, not exactly. Power, conditioned power that is, has been the savior of many a server. However, incorrectly sized or improperly grounded power systems can quickly become a nightmare. This ranges from things as common as improperly sized circuit breakers to under-powered sub-panels. There are many options available to provide conditioned power, including such things as motor generators and isolating transformers.

Tagged to the notion of conditioned power is uninterruptible power. Uninterruptible power systems are just that: sources of "virtually" uninterruptible power. The "virtual" piece is somewhat subjective. In order for power to truly be uninterruptible, it must be supplied via a series of fault-tolerant and redundant subsystems, and be backed by commercial and private power sources, which include batteries and generators. The generators also need fuel to run continuously, so this is also a consideration when designing the logistical portion of an uninterrubtible power system. These systems are available in a variety of sizes, from small standalone systems to huge multi-tiered installations. Sizing the unit is a basic prerequisite. The load of a large server-farm will crush an undersized UPS. This holds true for any power solution you may be planning, including uninterrupted power backed by generators. If the server load exceeds the kW output of the generator, it will stall under that load once power is phased in.

I've seen it happen. Believe me, it's not a pretty sight. An in-depth discussion regarding uninterruptible power is especially timely if you are doing anything pertaining to a data center at this time. I say this to remind you of the current energy crisis in the U.S., and the rolling blackout conditions which occur almost daily in several parts of the country. If your firm has already located, or is planning to locate, any mission-critical applications in these affected areas, now is a good time to truly evaluate the power situation in your existing, future, or outsourced data center. Grounding is another critical factor in the power equation. Improper grounding of main and branch circuits can create ground loops, inviting potentially disastrous situations to all grades of computing equipment. I can't stress how important proper grounding is, especially if your data center is located in a geographic area prone to lightning strikes.

Environmental Conditions

Servers generate heat. Several colocated servers generate lots of heat. Not a problem, you say? You've got these servers isolated and in their own big room with plenty of AC ducts overhead. Problem solved! Well, maybe not. Is the conditioned air supplied to the data center part of the office air conditioning? If it is, then chances are that that same system will supply cool air in warm months and warm air in cooler months. The last thing you want is a heated data center in the winter. So, this piece requires some thought. Don't forget about windows. Windows to the outside world (especially ones with a southern exposure) can potentially bring in lots of heat on sunny days, so plan on tinted windows, blinds, or shades.

Large-scale data centers designed for that purpose typically come equipped with large air conditioning units installed within the data center itself. These systems provide conditioned air, properly cooled and humidity controlled. Too little humidity promotes static electricity; too much promotes sweat on or near heat-producing equipment. Smaller, standalone AC systems are available for small-scale data centers.

The last piece of the AC puzzle pertains to how the conditioned air is distributed. Properly directing the cool air is critical to its effect on application server farms. Physical placement of the servers feeds into this as well. For instance, the density of installed servers per square-foot of floor space determines the combined BTU output. If there is no available space on the side or back of the servers, airflow into the units is severely limited. Likewise, if the servers are positioned front-to-back, you are almost guaranteeing that having many servers in a high-density server installation will result in overheating. Back-to-back server installations are smarter, providing that you take some precautions, which we'll discuss later.

Physical Security

Think this isn't important? Think again. Do you really want your application lifeline hidden under John the Programmer's desk? No. So, what do you do? This is one of the reasons you want a data center in the first place – controlling access to the boxes and protecting them from physical harm.

Those windows discussed in the previous section may also pose a unique security concern. It's easier to break through glass than brick and mortar. So once again, planning is key.

Physical access to an office data center can be controlled, quite literally, via lock and key. When more sophistication is needed, a card scanner may be employed to track who is in the data center and when. A log entry describing the purpose of a visit is also a good idea. As the sensitivity of the nature of the application servers intensifies, so does the security needed to protect them. In some cases, high-level security measures, such as the use of security guards, hand or retina scanners, and mantraps, are warranted. This rule holds true for data centers of all sizes and configurations. Security measures are definitely something to think about.

Fire Suppression

Often overlooked and sometimes misunderstood, fire suppression is a key component to any data center. Depending on how you view the importance of your application servers, several options are available for fire suppression systems. The simplest system is the basic sprinkler head placed above the protected servers. Effective? Most definitely, except that 10,000 gallons of water dumped on those servers doesn't lend itself to server survivability. Let's face it, sprinklers are meant to protect the building from fire, not to protect the contents of your data center. So, how about FM200? This gas is the successor to Halon, and is environmentally friendly. Unlike a CO_2 system, which removes the oxygen from within the data center, FM200 suppresses fire by discharging as a gas onto the surface of the combusting materials. Large amounts of heat energy are absorbed from the surface of the burning material, lowering its temperature below the ignition point. Not deadly to the human occupants of the data center, the FM200 system can deploy and, except for being totally scared when the system is activated, the workers can exit the data center calmly and without worrying about injuries.

One other thing... . Remember the conditioned air? Be sure to have the AC units in the data center automatically shut down whenever the FM200 is released or else it will be sucked out with the warm air. Don't forget to seal the data center. FM200 is only effective if it stays at a certain density within the air, so it can't leak out. Also, don't plan to put any servers or equipment directly under the FM200 dispersion-head. The force is great enough to severely damage anything directly below it.

Be aware of how conditioned air is distributed and returned. Air plenums have specific fire code restrictions on the types of cables that pass through them. Raised floor systems and dropped ceilings are typically utilized as supply or return plenums. If you plan to run your communication cables through these spaces, be sure to use materials rated for those specific environments.

Last but not least, your fire insurance underwriter may insist on installing a sprinkler system inside the data center anyway. Just be sure you can install high-temperature heads in the system. Doing so may save you some headaches later.

Lighting

The proper amount of lighting is a key factor in any working environment. Ask any commercial construction company or competent electrician how easy it is to get a building or electrical permit for a commercial endeavor without a reflected ceiling or lighting plan. Beyond code requirements, it's a good idea. How are you supposed to be productive if you can't see what you're working on? Simple but true. So, don't skimp on the number of fixtures. Mood lighting may work well on a date, but has no place in a data center environment.

Work Space

Again, a relatively minor issue that is continually overlooked. Let's face it, there will be the occasional all-nighter spent inside the data center during the course of application server support. So, where do you plan on working? Think about it. The PC balancing act gets kind of old at 2 a.m. Is this any way to properly support your server farm? Work space is a reasonable and justified component of any data center, especially if you are planning to document what you were doing for those five hours. We're not talking about granite work surfaces and mahogany desks here. A small desk or table with a small file cabinet would be nice. A telephone is also essential, especially if you ever need to reach tech support. A chair to sit in would be helpful, as would a shelf for technical documentation and manuals. Forget the coffee maker, though. Liquids really don't belong alongside those servers.

LOGICAL CHARACTERISTICS

The second core element pertains to the logical characteristics of the data center. Logical characteristics play into the specifics surrounding what you are actually installing. This is where *the physical characteristics of the data center are essential to properly supporting the logical characteristics*. For instance, the square feet of data center floor space in relation to server densities, power requirements, BTU output, etc.; remember how we touched on all of these? Let's see how this is all intertwined.

There are some basics to think about. Start with those servers. How large are they? Where do you plan to install them; on furniture, in cabinets or racks, as standalones? What about network connectivity? Where is the Telco D-Mark? Is it a switched network, routed network, or hybrid? How about network security? Are multiple DMZs warranted? If your Service Level Objectives call for the high availability of your servers, is the network bulletproof? You can't access the servers without a network. How about server clustering? What kind of clustering? NT? Veritas? Sun? How about dual-NICs? Multi-pathing? Storage-Area-Networks? All of these concepts are important, but just how important? Let's dig a bit.

Server Density

First, you must decide on server densities. Small-scale server farms are no problem. Large-scale farms are a different story. For instance, you could comfortably fit six Compaq 6400R servers in a standard rack. Let's say you need to install 60 servers. Ten racks should do it. But what about power? If your Service Level Objective is high, you may want to consider dual power supplies in those servers. If you put an amp-probe across the lines of an active server, you'll notice that the total load is "X", with both power supplies pulling equal amounts of power. But, when one power supply fails, the remaining supply now pulls the amperage equal to the entire load ("X"). What does this mean in terms of design? It means that you must be *very* careful to balance those power supplies across multiple electrical circuits, and to ensure that each individual circuit can carry the whole server load by itself. It may even be advisable to feed these circuits from separate sub-panels or separate UPS systems, if possible. Again, planning is key, and all roads lead back to the Service Level Objectives you have set as goals for your data center and the application servers it houses.

Servers and Network Equipment

How will they be connected, and how will those physical connections be properly managed? Well, that depends upon server densities and those pesky Service Level Objectives. Whether the densities are large or small, interconnectivity should be accomplished via some manner of BICCSI-approved structured cable plan. Correctly specified patching panels and/or cable schemes are key to high data throughput. However, installations with high server densities bring their own set of problems; namely, effectively managing those server-to-network cross connections. For instance, a 60-server installation with dual NICs equates to

120 ports of server patching. You should place a patch panel at the top of each server rack or cabinet. These panels tie to corresponding "server" panels in a central network equipment rack (or row), or to distributed network equipment racks for installations with even higher densities.

The network gear should be connected to its own set of "network" patching panels. This methodology allows for clean and manageable network and server demarcations within a single rack or cabinet. Considering the densities of servers and network ports, you could wind up with 240 total ports being patched. This is the maximum number of ports I recommend patching in a single cabinet or rack. More than this can quickly become your worst nightmare, even for the most experienced cable installer. Color-coding of primary and secondary NIC connections is also a good idea, as it lends itself to the fast visual identification of different network-to-server connections.

BTU Output

This includes all active gear (application servers, network devices, UPS [if installed in close proximity], etc.) A single row of 60 servers is fairly easy to cool, providing your HVAC unit can handle the BTUs. Two rows of servers is a bit more challenging. Let's assume that you've followed OSHA and NFPA (fire code) regulations for clear aisle space and access surrounding this equipment. So, how do you place all those servers in those racks if space is limited, but legal? You could have the rows face front to back, but that would mean that the heated exhaust air from the servers in Row A would hit the faces of the servers in Row B. Not too smart. No problem, though. Have them facing back-to-back. Much smarter! Okay, now for the challenge. Where do you direct the conditioned air? To the rear of each server, right? After all, that's where all the heat is coming from. Wrong! Direct the air at the face of each server. The cool air gets drawn into the unit and cools the internal components. The resultant exhaust air is also much cooler. So, whether the air comes up from the raised floor, or down from ducts above, strategically place these to direct the air at the front of the servers. This holds true for anything your server is mounted or placed in, except for enclosed cabinets. In those instances, have the conditioned air pumped in from directly above or below the cabinet. Be sure to have those cabinets equipped with fans to draw the conditioned air through and out.

Connectivity

How about connectivity to the outside world? If your servers face, or are tied to anything Web-facing, it is in your best interest to consider more than one way to the Internet. This concept bridges both physical and logical mapping. If your data center is in-house, your options may be somewhat limited, unless you happen to have SONET equipment or are fed via alternate serving wire centers. If you have outsourced your data center, be sure to closely examine your provider's peering arrangement with other carriers. It's really important.

Backup and Restoral

Other things to think about include backup and restoral. This particular requirement bridges physical, logical, and operational criteria. So, what about backup and restoral? Is it okay to store all backups on site, or must you transport tapes to a secure facility? Will the RAID arrays in your servers do the trick by themselves? Will you have someone perform the backups manually? Will you utilize SAN technology, or might you want a tape robot? Backup and restoral is a critical component to think about, especially when high Service Level Objectives have been set.

Logical Access

Make sure you plan for easy logical access to each server. The last thing you want to do is drag keyboards, monitors, and mice into and out of the data center. This is especially true when you are upgrading OS, middleware, or applications for multiple servers in the data center. Instead, be sure to map out a strategy for this, via either a LAN or a traditional KVM-style access. Remote VPN access is nice whenever remote support is required.

These are some basic examples of the types of logical issues that must be properly planned.

CRITERIA OF OPERATIONS

Now, on to the third and final core element: criteria of operations. These are the operational realities and plans that must be followed in order to meet the Service Level Objectives that have driven the data center design to this point

The criteria of operations is the all-encompassing blueprint for the daily activities, which will affect the data center and the servers it houses. It maps out the strategy for everything from change management to application release methodologies and control. It is the very heart of every successfully run data center. It covers all applications, servers, network components and connections. It addresses provisioning, capacity planning, sparing levels for immediate repairs, and what to do in a crisis mode. It covers routine access and escalation procedures. It is the bible of your data center. It should contain the Run-book for every application and system that needs to be supported.

The criteria of operations can range from a simple document covering a small data center housing a single application server, to an online interactive series of procedures and documents that support multiple, geographically dispersed data centers, applications, and servers. Planning and building your criteria of operations requires a great deal of thought, and is the glue that ties everything back to the Service Level Objectives you started with.

Conclusion

If your head is spinning after reading this, don't worry. Building a robust data center can be relatively pain-free if you simply stop and think before charging forward. Understanding the components and concepts presented in this article will ultimately provide you with much more than the basic building blocks needed for an endeavor of this nature. It will provide you with a roadmap to performing the type of due-diligence needed to ensure success, whether building your own data center, helping to evaluate an existing center, or choosing/evaluating an outsource partner's data center. No kidding.

Even if a facilities professional has been chartered with building or evaluating a data center, don't be afraid to open your mouth if you observe or hear something that seems off-kilter. It's easy for anyone to mistakenly overlook an item. In the end, the server you save may be your own.

web services EDGE conference&expo™

INTERNATIONAL WEB SERVICES CONFERENCE & EXPO

JDJ EDGE conference&expo™

INTERNATIONAL JAVA DEVELOPER CONFERENCE & EXPO

XML EDGE conference&expo™

INTERNATIONAL XML CONFERENCE & EXPO

FUNDAMENTALLY IMPROVING THE SPEED, COST AND FLEXIBILITY OF BUSINESS INTEGRATION

Sponsored by:

IONA | END 2 ANYWHERE™ · TogetherSoft · ADDS · bea · SilverStream · MERANT · Federal Computer Week · SD Times · Charles F. Goldfarb's XML TIMES.com · jCert · wrox · Webservices.org · Java Skyline · WebServices JOURNAL · JAVA DEVELOPER'S JOURNAL · XML JOURNAL · wireless BUSINESS&TECHNOLOGY · BEA WebLogic DEVELOPER'S JOURNAL · ColdFusion Developer's Journal · WebSphere DEVELOPER'S JOURNAL · CF Advisor

Sponsored by: SYS-CON MEDIA

Produced by: SYS-CON EVENTS

Web Services – Skills, Strategy, and Vision

For the developer. the latest tools and techniques...

For the architect, the latest designs....

For the VP/CIO, technology management issues

- Invaluable information in the form of discussions, presentations, tutorials, and case studies
- Unmatched Keynotes and Faculty - gurus in the Java, XML, .NET, and Web Services world
- The largest independent Web Services, Java, and XML Expos
- An unparalleled opportunity to network with over 5,000 *i*-technology professionals

New in 2002!

Each track will feature "Hot Breaking" Sessions to keep attendees up-to-the-minute on Emerging Technologies and Strategies!

Featuring 2 Keynote Panels and Industry Perspectives from the Visionaries shaping Next Generation Technologies and Business Strategies

2002

ONLINE
EARLY BIRD REGISTRATION NOW OPEN!
SAVE $400

JUNE 24–27
JACOB JAVITS CONVENTION CENTER
NEW YORK, NY

OCTOBER 1–3
SAN JOSE CONVENTION CENTER
SAN JOSE, CA

FOR MORE INFORMATION
SYS-CON EVENTS, INC
135 CHESTNUT RIDGE RD.
MONTVALE, NJ 07645

201-802-3069
WWW.SYS-CON.COM

Who Should Attend…
- Developers, Programmers, Engineers
- *i*-Technology Professionals
- Senior Business Management
- Senior IT/IS Management
- Analysts, Consultants

Who Should Exhibit…
Java, XML, Web Services, and .NET Technology vendors, staking their claim to this fast-evolving marketplace

Web Services Edge Conference Committee

Alan Williamson
Java Track Chair
Editor-in-Chief
Java Developer's Journal

Java Track
- Java 1.4: What's New?
- Building Truly Portable J2EE Applications
- J2ME: MIDlets for the masses
- WebScripting Technologies: JSP/CFML/Velocity
- Where is Swing in this New Web Services World?

Ajit Sagar
XML Track Chair
Editor-in-Chief
XML-Journal

XML Track
- XML Web Services: Standards Update
- Using XML for Rapid Application Development and Deployment with Web Services
- Unlocking the Promise of XML: From Hype to How-To
- The Use of XML Technologies to Enhance Security
- Content Management with XML

Sean Rhody
Conference Tech Chair
Web Services Track Chair
Editor-in-Chief
Web Services Journal

Web Services Track
- Starting Out in Web Services: Fundamentals of Web Services (WSDL, UDDI, SOAP)
- Exploring .NET myServices Initiative
- Standards Watch: Reviews and Discussions of the Interactions of All the Relevant Standards
- Guarding the Castle: Security and Web Services
- The Microsoft Way: .NET, Passport, and other MS Technologies for Web Services
- Laying Down the Rules: Web Services and Business Process Engines
- Practical Experiences with Web Services and J2EE
- The Odd Couple: Making .NET and J2EE Work Together

Andy Astor
IT Strategy Track Chair
International Advisory Board
Web Services Journal

IT Strategy Track
- Key Architectural Issues in a World of Web Services
- .NET vs J2EE – From Religious Wars to Fact-Based Decision Making
- Getting Started with Web Services
- Selecting a Framework: Toolkit, Platform, or Roll Your Own?
- Infrastructure Vendors–A Map of the World
- Extreme Programming and Web Services Projects: Defining ROI
- Standards You Need to Know About
- Best Practices You Need to Insist On
- Introduction to Business Rules and Rules Engines
- Panel Discussion- Web Service Business/ Economic Models

Jim Milbery
Vendor Track Chair
Product Review Editor
Java Developer's Journal

Vendor Track
- .NET, J2EE, JMS and other Messaging
- Case and Development Tools
- The Use of Testing Tools
- How-to Demonstrations

An approach for success

WebSphere Migration

BY ROBERT COSTELLO

The popularity of distributed computing has gained tremendous momentum over the last three years. Much can be attributed to the maturing of two intricate parts of the solution: the Internet as network solution architecture, and application server technology. One of the more popular application server technologies in the marketplace is IBM's WebSphere application server solution.

Based on my experience, this article discusses some technological and organizational issues that arise when migrating to WebSphere application servers. Whether you're migrating from another application server or getting into application server architectures for the first time, there are a few basic areas that must be examined, and solidified, to ensure success.

Too often companies take a client/server approach to application server development. They allow these servers to proliferate across the enterprise based on division or unit requirements. Application servers differ from a client/server environment in a number of ways:

- In a client/server environment, the majority of applications are developed by development professionals. Individuals need a certain level of sophistication in order to use the tools. With Web-based development, there can be many levels of development, from end-user in HTML to sophisticated Java development.
- Because these applications are designed for Web-based architecture, your application could be accessed by a large audience, perhaps in the millions, whereas in a client/server architecture the number of users could be controlled and managed by either the DBMS or the application itself. So although a Web-based architecture allows certain freedoms, it has the potential to create a performance bottleneck if not architected correctly.
- For the most part, client/server applications dealt with the presentation of data (i.e., data queries, etc.) to a predesigned GUI interface. The technology limited the potential for multiple "data types" within the application. Web applications tend to be very content-rich. Because of this architecting for performance in a browser-based solution, using application servers becomes the number one priority. Content presentation as well as management and network caching must be taken into consideration.

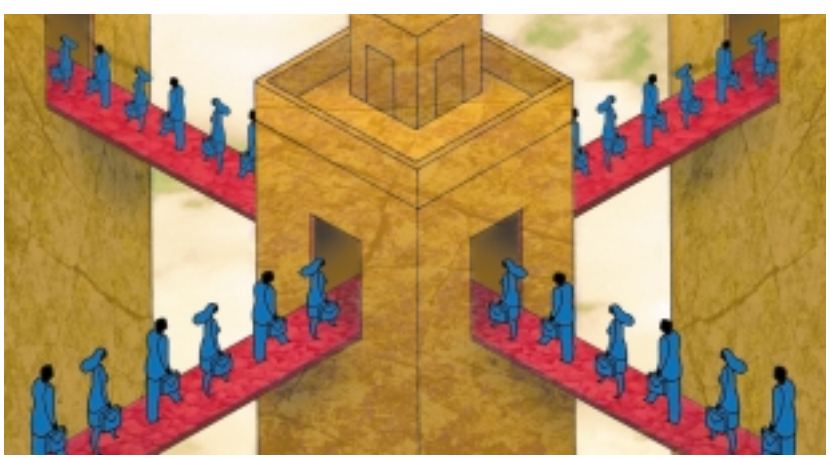
Most transaction-oriented Web applications are what I term virtual applications: multiple instances of that application may be running across multiple application servers. How you manage applications changes, updates, and performance become strategic to the success of the application. This is quite a bit different from a client/server architecture where the focus is on managing database access. Applications are not the concern. The database takes the priority.

When beginning this migration process, here is how I've seen this introduced successfully. The computing enterprise should be looked at in terms of:

- Operations architecture
- Technical architecture
- Development architecture
- Application architecture

Operations Architecture

Operations architecture is a framework describing the collection of hardware and network components,



ABOUT THE AUTHOR

Bob Costello is an application architect with more than 15 years of experience in application development.

EMAIL

Rcostel@home.com

operating system-level software, and operations architecture processes/services that comprise the production application workload. It encompasses security management, storage management, disaster recovery, configuration management, and performance management.

Why do you need operations architecture? As you move to an application server/netcentric model, you'll rely more on your enterprise servers and mainframes. Since you're moving to a netcentric model, I'll assume you'll have proliferation mobile devices, including wireless, and are maintaining service levels to your customers while managing change. As we discuss the migration process in more detail, you'll see the importance of this architecture in managing the application server.

Technical Architecture

Technical architecture defines the common runtime technical services upon which an application executes in a production environment. Technical architecture is concerned with:

- Increasing developer productivity: Developers avoid reinventing the wheel when technical architecture defines a common development platform and a common source for technical services.
- ***Maintaining consistent application quality:*** Inconsistent adoption of common technical services leads to inconsistent application behavior and quality, resulting in applications that are more difficult to build, use, and maintain. Technical architecture seeks to resolve this problem.
- ***Reducing time-to-market:*** Technical architecture can reduce time-to-market of new business capabilities by using standard architecture services, adhering to industry standards, and making consistent tools and implementation techniques available.
- ***Controlling impact of product changes:*** Because no one vendor has a complete solution for a netcentric architecture, multiple vendors and applications need to be integrated. Changes from a vendor can severely affect your systems.
- ***Increasing complexity of business systems:*** Technical architecture can help manage complexity by providing common technical services developed once by skilled personnel who can understand the current state of technology and anticipate its future.
- ***Integrating systems within and between businesses:*** It fosters solutions with a higher degree of integration by helping tie together disparate software, platforms, and protocols into one comprehensive framework.

Development Architecture

Development architecture defines the approach to developing business systems. When methodology, tools, and environment are defined, there is a reduction in the complexity of developing in a netcentric environment. Development architecture focuses on these issues:

- ***Consistent developer tools:*** In the Web market, the multitudes of development tools and their inability to work together can have a severe impact on version control, as well as testing and Q/A.
- ***Ease of training:*** How complex or easy is it to learn and implement the development tools? Are there tools for the end user as well as the developer? What is the learning curve on the tools set?

Application Architecture

Application architecture is a framework describing how the user channel, business processes, and data layers will be organized to support business capabilities. Application architecture provides a common framework and vocabulary for analyzing application systems, and the business implications of such systems, in order to standardize them and minimize duplication throughout the enterprise. The interrelationships of these layers in a fictional company are shown in Figure 1.

- ***User layer:*** The user layer provides the channels through which internal users (the crew) or external users (the clients) may access the company's services. A variety of channels, such as Web access, Voice Response Units (VRU), workstations, and paper mail enable users to do business with the company.

MIGRATION ISSUES

- ***Business process layer:*** The business process layer provides an enterprise-wide view of the processes required to perform the major business functions/capabilities, and how such processes interact with users and external parties. The business process layer ensures that applications support specific business capabilities. This layer is an essential component that allows solutions to be aligned with business needs.
- ***Data architecture layer:*** The data architecture layer defines the business entities and relationships that support the business requirements, and represents the hierarchical breakdown of data from the enterprise level to the application level.

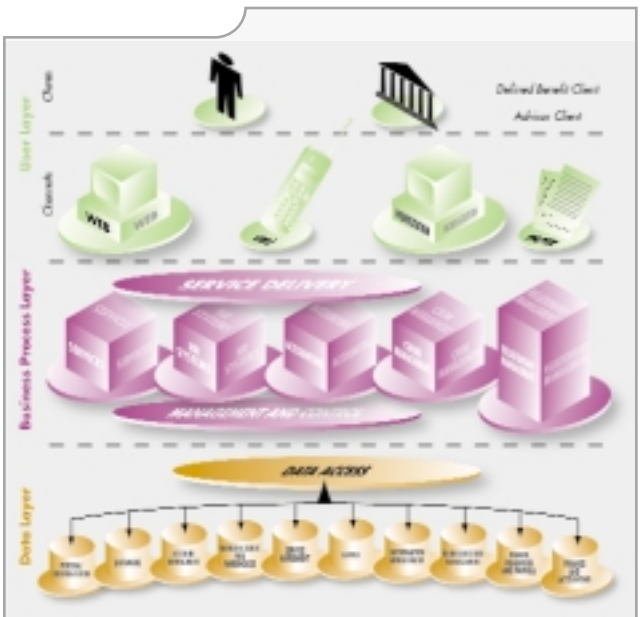


FIG. 1: Enterprise-wide application architecture framework

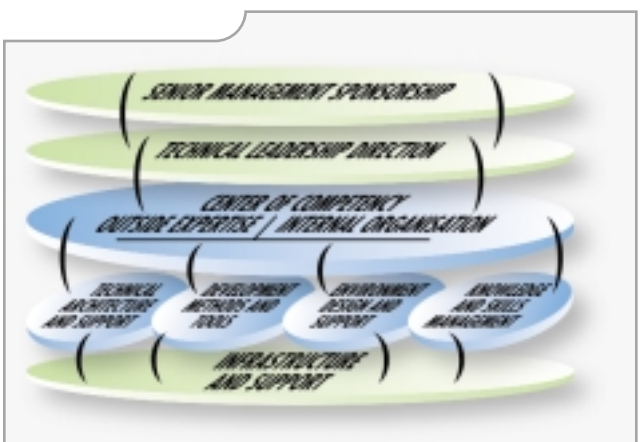


FIG. 2: Center of competency model

In essence, the data architecture layer describes what information is necessary to complete a process. It supports the application architecture by providing an integrated enterprise-wide framework to allow for shared corporate data. This layer is the overall blueprint for properly organizing and managing the data and information.

The application architecture outlines a framework that allows the business to communicate more clearly with business units and shareholders, and to act more efficiently on their behalf. By addressing the IT challenges listed above, the Application architecture can provide the following benefits:

- Provide a common business definition and communication framework for IT and the business units.
- Assist in determining the impact analysis for new business capabilities.
- Identify strengths and weaknesses in applications based on business requirements.
- Promote sharing common processes across business units and working in unison to develop a migration plan to the future.

Plan to Implement Architecture Tasks

When planning a migration, each of these considerations should be reviewed relative to the WebSphere architecture. Below, I've listed some tasks required for each architectural area.

OPERATIONS ARCHITECTURE

1. Review, analyze, and document prototype architecture for intranet and extranet applications.
2. Understand the WAS configuration management, product and application upgrades, capacity management, security management, event management, disaster recovery, and performance turning.
3. Examine the design and testing of an automated installation of a WAS on to your production platforms.
4. Begin defining an operations production environment and the associated processes that support your intended platforms: Unix, NT, etc.
5. Begin testing the automated start, stop, and restart of the WAS and its dependent processes (NES, Configuration Database) in an orderly fashion.
6. Test fault and event management integration into the existing network system management environment. This includes an identification of fault/event meanings and the appropriate actions.
7. Test WAS Network and route diversity and reachability.
8. Test or create processes for automated replication and load balancing.
9. Design associated with business continuity (disaster recovery, or DR) and high availability.
10. Design and evaluate performance and capacity measurement and management.
11. Begin assessing and making recommendations on the capacity forecasts and processes used as part of storage management.
12. Begin assessment of the impact on the workload job scheduling process to determine if a review and rework is needed.

DEVELOPMENT ARCHITECTURE

1. Evaluate, design and implement software configuration management processes such as software build, compile, link, and register, as well as application promotion processes and procedures for WebSphere from development to production elevation.
2. Design, implement, and document software version control prototype processes.
3. Design, implement, and document prototype services that manage and control configuration, global and properties settings.
4. Design and implement prototype services to manage the porting of security LDAP rules.
5. Evaluate, design and implement performance-tuned compilers for WebSphere, i.e. Java HotSpot compiler.
6. Design, implement, and document best practice processes and procedures for runtime environment provisioning and retirement.
7. Develop a working model/prototype in support of technical engineering services and application

programs identified for initial deployment.
8. Evaluate and deliver tools and training materials in support of WebSphere datastore registry LDAP, Oracle, DB2, etc.) load processes.

TECHNICAL ARCHITECTURE

1. Identify and define test-case scenarios, including system configuration, input data, and results analysis, as well as component-level performance testing of applications developed to run on WebSphere's latest release.
2. Upon release of WAS Service Pack releases, identify and define test case scenarios including system configuration, input data, and results analysis , and then define test execution of component level performance of applications developed to run on the latest release.
3. Define the integration and develop implementation procedures for deployment of applications to run under the WebSphere architecture.

APPLICATION ARCHITECTURE

1. If migrating from another application server, then begin investigation into application migration, including code reviews, best practices for WebSphere application development, and education.
2. Begin evaluation and investigation into content management processes for WebSphere and personalization technology.
3. Identify, recommend, and document strategies for configuring and deploying WebSphere-based applications into multiple environments.

First Things First

The following are items I lbelieve are priorities. If you can do nothing else, at least focus on these items:

- Develop Configuration/Release management procedures to support the netcentric integration/promotion model.
- Create standard procedures for source code control within the WebSphere domain.
- Evaluate common environments to host intranet and Internet applications.
- Evaluate HW, SW, and network configurations to support test environments.
- Evaluate network configurations to support development, test, QA, and production environments as well as high availability, load balancing, data recovery, and WebSphere coexistence with competitor application servers, if required.
- Ensure that system management is configured to support process monitoring, shared logging, and Tivoli integration.
- Evaluate data architecture for WebSphere datastore requirements (Oracle, DB2, Sybase and LDAP).

Establishing a Center of Competency

Of course, none of this works unless your organization is supportive. I recommend assembling a center of competency for WebSphere implementation. The COC could follow the model shown in Figure 2.

The purpose of a center of competency is twofold: it creates an area of expertise to deal with problems as they arise, and it can be the basis for a strong mentoring program. I'd recommend a staffing model similar to this:

- Center manager
- Full-time e-design integrator
- Full-time WebSphere architect/mentor
- Part-time WebSphere architect/mentor
- WebSphere developer/mentor
- Part-time education consultant
- Part-time product technical specialist
- Release management specialist
- Part-time release management specialist

You may have to go to IBM consulting or to a business partner to get the necessary expertise to assemble this center. Keep in mind that WebSphere is a powerful but sophisticated solution that was built as a way to run your business. You must treat it as such, and roll it out slowly with an extreme emphasis on testing and quality assurance. Take your time and be cautious, and WebSphere will give you the ROI you expect.

WEBSPHERE USER GROUPS...

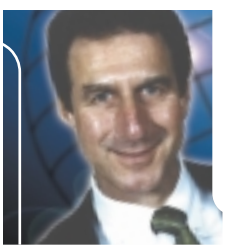
www.websphere.org offers Web space to user groups worldwide for posting contact and meeting information. The site's LISTSERV offers peer-to-peer e-mail Q & A and has hundreds of subscribed members from all over the world. The site's current instant poll ("Share Your Views") enables members to vote for appropriate meeting topics. For site sponsorship and general information contact info@websphere.org.

Sharing, learning, and networking

WebSphere UGs

BY MICHAEL MANDE

At WebSphere 2001 I participated in an "instant poll" held during a general session. The results showed that participants – ostensibly "developers" – expected to spend more than 20% of their time on learning, research, and education in order to keep up with current IT issues and solutions. Developers are constantly searching and surfing for information as part of their daily jobs, and management should be aware of this, not that CNN or Yahoo screens are regarded as significant sources of such information!



ABOUT THE AUTHOR

Michael Mande, a consultant for Group Intelligence, based in New York, founded and runs the Manhattan WebSphere User Group. In June 2000 he initiated the WebSphere user group site www.websphere.org. Previously, he was COO of Tangent International Consulting, and of Planetworks, an integration middleware product vendor prior to its acquisition by IBM. Michael has worked extensively in IT and business development in the U.S., UK, Europe, and Africa.

E-MAIL

mmande@websphere.org

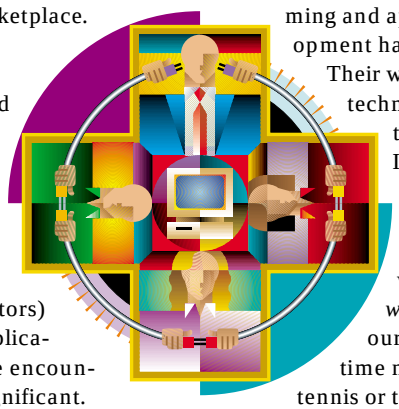
One component of the information tool kit is the user group, and the growth of WebSphere user groups worldwide is certainly a measure of the widespread eagerness to participate in sharing and learning while also connecting and networking in the nontechnical sense.

The goal of a WebSphere user group is to provide a forum for valuable sharing of information related to WebSphere – now a broad IBM platform – and the branding of many IBM products and solutions. Each user group may express this goal differently but the expectations are the same: to focus on common issues and experiences, and new developments around the WebSphere family. The aim is to foster knowledge and understanding of the IBM WebSphere family of products.

In order to have a successful user group, it is essential to strike the proper balance between education and commerce through discussion and presentations. The following guidelines should be followed for any group:

1. ***Build a network*** of people interested in WebSphere, enable an exchange of information among the members, and encourage member suggestions and participation.
2. ***Promote knowledge*** and education of WebSphere-related technologies, best practices, issues, products, and solutions.
3. ***Have a mix of programs*** consisting of people presenting real business experiences, IBM WebSphere product knowledge, and vendor-related products and solutions.
4. ***Accept open participation,*** yet make members aware that this is not a forum for solicitation or recruitment. The group will include many levels of developers, consultants, managers, vendors, recruiters, and curious parties, all part of the growing WebSphere marketplace.

Over the past 18 months, I've organized 10 meetings of the WUG in Manhattan. The common experiences that developers (as well as architects, designers, managers, and administrators) in the WebSphere/application server space have encountered are strikingly significant.



Our enthusiasm to participate in user group meetings comes from anticipated knowledge, learning, and networking. The best meetings are when members hear other people's real-life experiences, and especially their proposed or proven solutions. The WebSphere User Groups really do provide the opportunity for this sharing and learning, and I would encourage everyone to participate in local meetings.

What I find significant today is that developers are also savvy users in the traditional IT sense of the word. In the past, developing was achieved with a perforated wall between the "users" and the "builders." The written specs often failed to keep current with the changing user requests, and the lack of user experience was often cited as the Achilles' heel of IT projects. But today developers are familiar with what works, and what doesn't, through their own experiences as e-business users: the frustration of awkward logon procedures, passwords, recovery, and access to integrated data versus the attraction of real-time data from your bank (not last night's batch update). All these translate into meaningful experiences for the developer in the workplace. The challenges of integration, scalability, navigation, design, response time, and content management have a significance for the development team because of experiences in the team members' personal lives.

The current environment is even further integrated into our lives. Our families and friends now have a good idea of what our work in programming and application development has been about. Their window into our technical world comes through the Internet, which has become truly pervasive. Additionally, the words *server* and *wireless* are heard in our living rooms, this time not associated with tennis or trapeze. As for WebSphere, when this word that my teenagers have been overhearing for the last two years pops up on TV between breaks in *The Weakest Link*, it's time to say *hull-oh-oh*, not *goodbye*. Who would've thought it?

ABOUT THE AUTHOR

Rod Bicknell is an ex-newspaper journalist who entered the technology field on Wall Street in the early '80s. He is also the editor of CFAdvisor.com

E-MAIL

rbicknell@ulster.net

Big Blue's Basics on Pervasive Computing

REVIEWED BY ROD BICKNELL

IBM is renowned for their generosity in providing extensive documentation with their technology and products. If Big Blue is to be faulted in this area, it's in the excessive wording of much of the documentation.

IBM presents its WebSphere Everyplace Suite Version 1.1 as an "integrated end-to-end software solution for mobile e-business," or a method to "enable Web and enterprise application access from pervasive computing devices."

Pervasive computing, according to IBM, gives users a convenient means of accessing information whenever they may need it, regardless of where they are located. It uses a series of technologies that allow tasks to be accomplished, typically using a new class of portable intelligent devices.

The WebSphere Everyplace Suite is an integrated suite of existing products that work together to provide a deployable solution for pervasive devices. The six functional areas addressed by the suite are: connectivity, content handling, security, optimization, subscriber and device management, and core service.

Once a method of navigating the style has been successfully developed, readers will find that the 264 pages of IBM Redbook's *An Introduction to IBM WebSphere Everyplace Suite Version 1.1* describe "implemented architectures and technologies" for the "planning and successful implementation of solutions" designed to "assist businesses in accessing Web and enterprise applications from desktop browsers and the newer classes of client devices including WAP phones, PalmPilots, and WorkPads." The book isn't a coding resource; it provides descriptions, along with schematics, of how WebSphere Everyplace Suite uses its components to construct solutions. While this is a necessity in constructing technology blueprints, the developer seeking deeper and coded examples should rummage through the Everyplace Suite online documentation and other related resources.

The $35 volume provides no CD-ROM, but it's available on CD-ROM, PDF or HTML at: <http://publibb.boulder.ibm.com/-Redbooks.nsf/RedbookAbstracts/sg245995.html?Open>. While it's good that the book is searchable online, it would have been better if the book had provided a searchable format without having to be downloaded from the Web. The index, while not extensive, is reasonably complete.

The book is divided into three sections containing 127 graphics. The Introduction includes e-business evolution, product overview, architecture, performance and scalability, and security; the Services section covers authentication, wireless device support, transcoding, managing subscribers/devices, and pervasive messaging and queuing; the third section, Optimization, includes proxy caching and load balancing.

There are also two appendices and a three-page glossary. The book assumes the reader has "a basic knowledge of HTTP and WAP protocols as well as some understanding of Web and Java technologies (XML, HTML, WML, servlets, and JSPs) and the terminology used in Web and enterprise applications."

The three key components to the Everyplace Suite are the Everyplace Wireless Gateway, Everyplace Authentication Server and the Everyplace Transcoding Publisher.

Obviously, it's a daunting task to configure so many products into a pervasive solution. The book provides the basics that are required to understand the functionality of each product and demonstrates that, in many situations, Everyplace Suite product subsets can provide valuable and reasonably complete solutions.

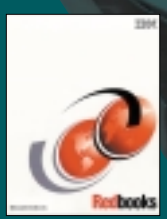
Overall, the book is well organized. Each chapter first provides a basic overview and then moves into more detail. For example, in Chapter 9, subscriber and device management is outlined as follows:

9.1	**Tivoli Personalized Services Manager overview**	9.3.3	Customer care
		9.3.4	Director
		9.3.5	Reporting
9.1.1	Logical overview	**9.4**	**Pervasive device management**
9.1.2	Functional overview		
9.1.3	Architecture	9.4.1	Device manager overview
9.1.4	Deployment		
9.1.5	TPSM in the Everyplace Suite	9.4.2	Pervasive devices supported
9.2	**Business and data model**	9.4.3	Initial enrollment/ setup
9.3	**Subscriber and system management**	9.4.4	Software distribution
		9.4.5	Deployment within the Everyplace Suite
9.3.1	Enrollment		
9.3.2	Self care		

While this covers the topics extensively, as do the other sections, the developer without deep pockets may want to consider whether the $35 required to add this book to the reference shelves is a worthy investment, especially since it's available online.

In the "Evolution of E-business" chapter in the Introductory section, lead writer Juan R. Rodriguez, senior software engineer at the IBM OTSO Center, Raleigh, noting what the book doesn't cover, says: "Development now continues apace to continue to integrate networking support into many of the new and existing devices that contain embedded computers, using new technologies and open standards such as the Open Systems Gateway initiative (OSGi), Bluetooth, fast IR, Wireless Ethernet and more. This is resulting in new classes of devices such as the Internet appliance – an Internet-capable device that has all the usability features of a normal household appliance (such as a television or washing machine) and yet communicates via open standards with e-business solutions to provide added value to its users."

Despite starting its voyage in a turgid swamp, this book provides the required background basics to provide pervasive computing. Those using WebSphere would be advised to have access to its contents, either from the printed book or online.



TITLE:
AN INTRODUCTION TO IBM WEBSPHERE EVERYPLACE SUITE VERSION 1.1 ACCESSING WEB AND ENTERPRISE APPLICATIONS
AUTHORS:
JUAN RODRIGUEZ WITH RICHARD APPLEBY, BERNT BISGAARD, HAO WANG, ADRIENNE McGRORY, ABDULAMIR MRYHIJ, AMY PATTON, AND MUHAMMED OMARJEE.
264 PAGES
ISBN: *0738419346*
PUBLISHER:
IBM REDBOOKS/FATBRAIN
PUB. DATE: 10/11/2000
IBM FORM NUMBER: SG24-5995-00

IBM WebSphere Studio 3.5 Advanced

REVIEWED BY
PATTI MARTIN

ABOUT THE AUTHOR

Patti Martin is cofounder of Simplex Knowledge Company, where she is vice president of Creative Services. She manages the company's Web servers and oversees Web content and creation. Patti received her education at the New School in New York City and has taken continuing education classes at NYU and the School of Visual Arts.

E-MAIL
pat@skc.com

When first presented with the idea of moving into a WebSphere production environment, I looked for the exit door. I had been designing in a Web environment since it went graphical in 1994.

Through the years I used a relatively small collection of software tools to accomplish my tasks and happily hand-coded all HTML using a text editor. At that point you were really cooking if you could gather information into a database using CGI. Then my neat, comfortable world changed. I began to work with "others."

I welcomed this shared responsibility at first. My strength was in layout, design, and communication of the message. The new functionality promised by more elaborate programming would make my sites sing. The honeymoon was over when I had to make the commitment to version control.

That Was Then; This Is Now!

With WebSphere Studio, IBM has delivered a complete design environment. I think of it as moving into a pristine apartment: it has all the amenities anyone could wish for, but like a new apartment, there is room to bring along some of my old favorites. WebSphere Studio has a workbench, built-in editors, and wizards. It also comes with companion Web development products that make it an out-of-the-box, end-to-end tool.

There are two versions of WebSphere Studio 3.5 Fix Pack 2 (V3.5.2), the Professional Edition and the Advanced Edition. The latter is like the same apartment, only with a balcony. There are obvious advantages with the Advanced Edition.

Studio Workbench

The Workbench helps manage and maintain Web site applications and files. Using a familiar folder hierarchy, it allows you to group and organize files and lets you perform all functions from this central location. Create, open, edit, and copy files, move them around, delete and publish them. You decide what organization makes sense and arrange your files accordingly. Perform all these tasks using the Workbench features and your preferred Web development software.

Maintain your files individually or in a shared version control system (VCS). Store files locally (on your own workstation) or on any other system in your network accessible by the net use command. With the Advanced Edition, you can also maintain files in a full-function VCS.

Regardless of where you store your files, Studio always checks out the file while someone edits it. When more than one person has access to the same files, this keeps them from overwriting each other's work. While one person is working on a file, it is locked so that others cannot change it.

By default, local project files are stored in the "MyDocuments" directory, as defined by your Windows operating system, and checked-out files are stored in the same directory where Studio resides – but you have full control over their placement. You can specify the location of each project when you create it. You can change the location of your check-out directory using the Preferences dialog from the Tools menu. You can work in a collaborative manner and have guaranteed file integrity.

You can edit and update files with your preferred tools and choose which editing programs to use for each file type. When opening a Studio file, you can launch your first choice (default) tool or select one of your alternates. And, with Studio's global search and replace function that works across all the files in a project, file maintenance is easier.

The most common problem in active and expanding Web sites is broken or dead links. With Studio you can assess file relationships to find broken links. The Relationship view provides a visual representation of how files link to each other. You can quickly see how many other files link to a particular file, which files have broken links, and which files have no links at all. When you find a broken link, you can correct it immediately. By following the links from file to file, you can navigate your Web site and verify its integrity.

I am visually driven and usually work with two browsers open to view pages as they are developed. In Studio I can publish the Web site at any stage of development on my Test WebSphere Application Server and quickly test the results of my work. When the Web site is ready for prime time, it's easy to switch to the final stage and publish the files to the production servers. In the Advanced Edition, any personalization rules are published along with the rest of the site, another great reason for going Advanced.

I've played around with text editors in the past, but I don't think I ever saved a file in one. Studio is bundled with Page Designer, an advanced-function HTML editor that allows the construction of complex Web pages. Dynamic element support enables you to include form elements, Java applets, embedded scripts, dynamic controls, and JavaServer Page (JSP) tags in your pages. You can toggle between visual and textual mode to preview your work. IBM has included WebArt Designer and AnimatedGif Designer, but I can't abandon my Adobe products. There is clip art stashed in the default installation even though most users will require custom designs.

Here's the part that I find so attractive. I can, with basic skills and the Studio wizards, add real interactivity to my pages. The wizards help me retrieve and update information from common databases and use server-side JavaBeans. With them, I can do everything from a simple table lookup to complex self-service applications. And I don't have to be an expert at SQL syntax or Java programming. The wizards walk me through step-by-step and then generate the servlet code.

Applet Designer

Applet Designer is included with WebSphere Studio, but you must choose to install it. Conveniently, it may be installed separately if you forget. Applet Designer's applet wizard gives you a fast path for creating a new applet. It launches automatically the first time you open Applet Designer, but you can turn this option off. By clicking through the wizard and answering simple questions, you can quickly build an applet. The wizard lets you insert animations, audio, images, rollover buttons, ticker tape, and links to databases.

Studio Applet Designer includes a JAR wizard to help you rapidly convert a Java class into a complete JavaBean and save it in a Java archive (JAR) file. You can convert a Java class you created yourself or one you got from another source. You can also use the wizard to modify classes that are already in JAR files.

Import Wizard

Studio includes an Import wizard that automates the transfer of a complete Web site from its publishing server into a Studio project. This wizard migrates existing sites and therefore jump-starts your Web development. It can copy files from any Web server accessible by FTP or HTTP, including those behind firewalls and secure gateways. The Import wizard prompts you for the information it needs to find the Web site and navigate it. You do have options for limiting the scope of the import and eliminating pages you don't want.

As you create new projects for your imported Web sites, be aware that Studio 3.0 projects have a property setting to specify the version of the WebSphere Application Server you will be using for all the files in the project. Based on your selection, Studio varies how it handles processes such as code generation, compiling Java files, and publishing.

Personalization Wizards

This feature is very cool and in great demand. Personalization in WebSphere Studio Advanced Edition enables you to customize Web pages. You can dynamically vary the content on your pages according to predefined conditions or circumstances by creating English-like statements called rules.

You can customize your Web pages based on:

- Any information you have about the person visiting the Web site
- What you know about the date and time the page request is made
- What you know about the page request and user session during a visitor's session
- What you know about the content that you might display
- What you know about the objects created during a visitor's session

Rules-based personalization involves users, content, and the rules that map users to specific content. Put this feature to work in the real world by offering discounts to frequent buyers, different navigation features to first-time visitors, or weather reports based on access location.

What you know about your visitors must be permanently stored and accessible by your Web site. This collection of resources might consist of some or all of the data known about the users of your site. These resources will be analyzed by the rules you set to determine which personalized content to deliver.

Studio's Advanced Edition includes text-based editors for Personalization rules. There is a separate editor for Classifiers, Actions, and Rules. Personalization is an add-on feature to the WebSphere Application Server.

VoiceXML Editor

Studio has a built-in editor for VoiceXML files. It includes a "code assist" feature to guide you as you add or change code and tags. This feature helps ensure that your VoiceXML files will be syntactically correct.

This VoiceXML editor is the default editor for files with the .vxml and .jsv extensions. Whether you add a new VoiceXML file to your project, import or copy an existing one, or create one with the Studio wizards, you can edit it with the default VoiceXML editor. You can also edit VoiceXML files with any other editor you have registered for files with these extensions.

Companion Products

If this isn't enough, Studio comes with an integrated set of companion products to help build, manage, and publish dynamic Web sites.

- ***VisualAge for Java, Professional Edition:*** VisualAge is used for building and testing Java applets, servlets, and Enterprise JavaBeans components. It supports the development and management of Java programs that can scale from Windows NT to OS/390 application servers, and includes advanced functions, such as incremental compilation and the ability to invoke methods while debugging.
- ***WebSphere Application Server:*** For your convenience WebSphere Studio comes with a complimentary copy and a developer's license for the WebSphere Application Server.
- ***WebSphere Personalization for Multiplatforms:*** Included with the Advanced Edition, it is installed on your WebSphere Application Server to achieve the full development and runtime environment for WebSphere Personalization resources and rules.

Conclusion

It's nice to know that in one software suite you get all you need to create a vibrant, interactive Web presence. You still need a good working knowledge of HTML, TCP/IP, Web browsers, and Web deployment. Other valuable skills include the ability to code in Java, an applied knowledge of Object-Oriented Programming concepts and some experience in relational databases (SQL).

IBM maintains a WebSphere Studio Web site at www.ibm.com/software/websphere/studio. You may download an evaluation version of Studio and track current news and software releases there as well.

IBM WEBSPHERE STUDIO

COMPANY INFO:
IBM Corporation
New Orchard Road
Armonk, NY 10504
1-800-IBM-4YOU

URL:
www-4.ibm.com/software/websphere/studio

TEST ENVIRONMENT:
IBM 300GL / PII 300 / 256 MB Memory / 4G HD
Windows NT, Fix Pack 6a

PRICING:
North American English Program Pack, Advanced Edition: $1899
North American English Program Pack, Professional Edition: $570

MANAGING JDBC CONNECTIONS

PROGRAMMING FOR PERFORMANCE AND SCALABILITY

If application servers like WebSphere are the heart of e-business, then the database is surely the lifeblood. Whether your firm is managing thousands of airline flights a day or simply exposing a library's card catalog to the Web, you must deal with database connectivity.

Java developers have a simple way to manage database connectivity at their disposal: JDBC. It provides a convenient interface to access a variety of vendors' databases via JDBC drivers. Simply plug in a new driver and your application can talk to DB2, Oracle, Sybase, Informix, and others.

The JDBC version 2.0 extensions introduced a feature called connection pooling, providing better performance and scalability. IBM implemented a JDBC 2.0–compliant connection pool in WebSphere 3.5, and continues to enhance it in version 4.0.

In addition to connection pooling, the WebSphere connection manager offers many additional qualities to enhance an application's performance and scalability for the savvy Web developer. This article will highlight those features and show how they can be used effectively for building better Web applications. In addition, this article will describe many best practices and potential pitfalls for developing with WebSphere's connection manager.

While the following information will provide an improvement to applications, note that this is meant to aid in application design, not replace it. No amount of "tweaking" can make up for a poor design, but a well-designed application with poor implementation of database connectivity logic can be improved. This article will address the latter case.

Background

The WebSphere connection manager has many facets. At the core, however, it's simply a connection pool. Object pools are nothing new.

The basic reason for a pool is that creation of certain objects is expensive in performance terms (time, memory allocation, etc.). Rather than creating these objects over and over again, it's more efficient to create a pool of already initialized objects that can be allocated to application components, and then returned to the pool when no longer needed. Thus, the hit to performance is only incurred once, when the object is actually created.

Anyone doing performance-tuning of JDBC applications knows that creating database connections is an expensive operation. Connection pooling is a natural fit for Web applications that utilize servlets or EJBs that require many simultaneous threads to connect to databases. Figure 1 provides a more visual example. In this diagram, three application components (a servlet, JSP, and session bean) have each received a Connection object from the pool (whose current size is six).

In addition to pooling connections, the WebSphere connection manager monitors the health of a database connection. If a connection has been allocated from the pool but hasn't been used in a specified amount of time (orphan timeout), the connection manager will mark the connection as "stale" and will return it to the pool (preempt the connection).

In the event of a connectivity failure (database error, network connection dropped, etc.), the connection manager will throw a StaleConnectionException, a subclass of SQLExcep-tion, on the next JDBC operation. This

ABOUT THE AUTHOR

Andy McCright is a software engineer for IBM software group. His responsibilities include development of the WebSphere Application Server and customer engagement. Recently, he has been researching Web services and how they will play into the next generation of application servers.

E-MAIL

andymc@us.ibm.com

allows application developers to handle failure conditions more flexibly, such as allowing the application to retry the database operation. Further, the connection manager removes all Connection objects from the pool.

WebSphere's connection manager supports a variety of databases, including DB2, Oracle, Sybase, Microsoft SQL Server, and Informix (only available in WebSphere 4.0). SQL Server and Oracle (for two-phase commit operations) require the use of the Merant SequeLink 5.1 JDBC Driver, which ships with WebSphere Application Server, Advanced Edition.

Configuration of the connection manager is simple and painless (see Figures 2 and 3). Each data source represents a connection pool to a specified database, and is linked to a single JDBC driver.

Accessing a data source programmatically requires a JNDI lookup such as:

```
Context ctx = new
InitialContext();
DataSource ds =
(DataSource)

ctx.lookup("jdbc/
MyDataSource");
```

The string, "jdbc/MyDataSource", represents the JNDI name of the datasource. In version 4.0, you may use a resource reference such as "java:comp/env/jdbc/MyDataSource", as per the J2EE specification. The DataSource object contains the methods getConnection() and getConnection(String username, String password) to allocate a Connection object from the pool.

Simply close the Connection object to return it to the pool. For example:

```
Connection con =
ds.getConnection(
username, password);
...
con.close();
```

Now that you understand the basics of the WebSphere connection manager, let's take a look at some of the best practices.

BY ANDY MCCRIGHT

Keep the Connection Life Span Minimized

Ideally, an application should never hold on to a Connection object beyond a single method. Nor should a single thread acquire more than one Connection. The longer one part of an application holds onto a Connection, the longer other parts cannot use that same Connection.

An application designed to acquire more than one Connection per thread and hold them for very long periods of time could easily come to a screeching halt when multiple threads collectively request more Connection objects than the maximum pool size.

The temptation here is to simply increase the size of the pool, but that will impact performance. Each Connection, whether it's in use or not, can cost over two megabytes of JVM memory. Furthermore, a higher pool size means creating more Connection objects which, as previously mentioned, is expensive. Efficient usage of fewer Connection objects is a far better design.

Additionally, when an orphaned Connection times out, the connection manager will reclaim the Connection for the pool, throwing a StaleConnectionException in the offending client on the next JDBC operation. While this is costly for the single thread, it does at least allow the Connection object to be freed. However, inside the bounds of a transaction (e.g., inside an EJB or when using UserTransaction objects), a Connection will never be orphaned. Thus, long-running transactions could keep Connection objects out of commission for a long time, and could lead to deadlock-like conditions.

While keeping connectivity life span minimized, a developer should be careful not to compromise functionality. There are times when holding a ResultSet open for an extended operation is necessary to preserve transaction isolation (row locks). Closing a ResultSet (or its creating Connection or Statement object) too early might result in your application making decisions that assumed the state of the data in the database was X when another thread has modified it to be Y.

From an application development standpoint, the best alternative is to use fewer connections in your pool, but to use them wisely. Once a method has finished its database utilization, it should free the Connection object by calling its close() method.

Cache the Datasource, Not the Connection

To take full advantage of the connection pool and make limited calls to expensive operations, an application developer should cache a datasource object when database connectivity may be utilized often. Caching the Connection has the negative effects described above and defeats the purpose of pooling.

Also, be aware that declaring Connection objects as static class variables can have unexpected and undesirable effects. For example:

```
public class MyJDBCClass {
      static Connection con;
      ...
}
```

When a Connection is declared static, it's shared by all instances of the declaring class. Therefore, it's possible that two threads might attempt to utilize the same Connection object simultaneously. The results could include unexpected table locking, incorrect ResultSets, and serious data integrity problems.

Also, keep in mind that all global variables of a servlet are static by default (even if you don't declare them static). If you decide to cache a Connection, make sure not to declare it static.

Caching the datasource means creating a global instance variable in one of your classes. That class will maintain the ref-

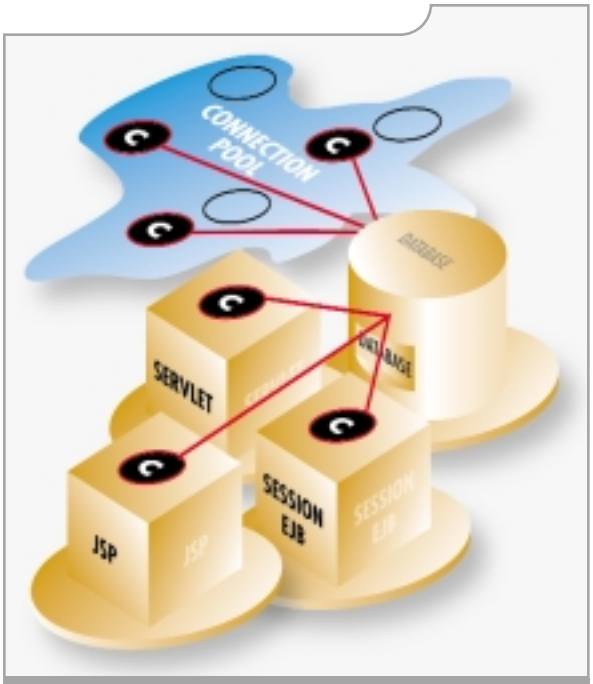


FIG. 1: Connection Pooling

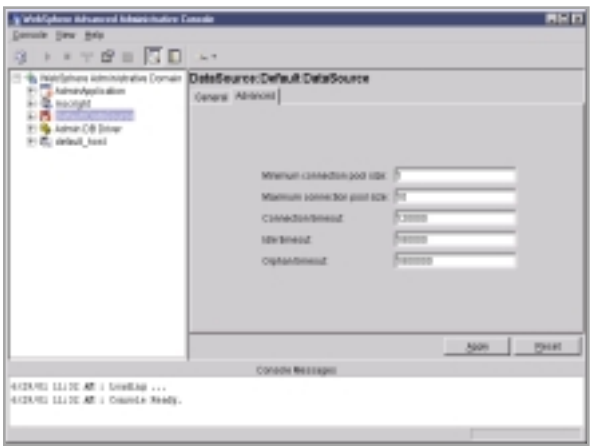


FIG. 2: Configuring a DataSource in WebSphere 3.5

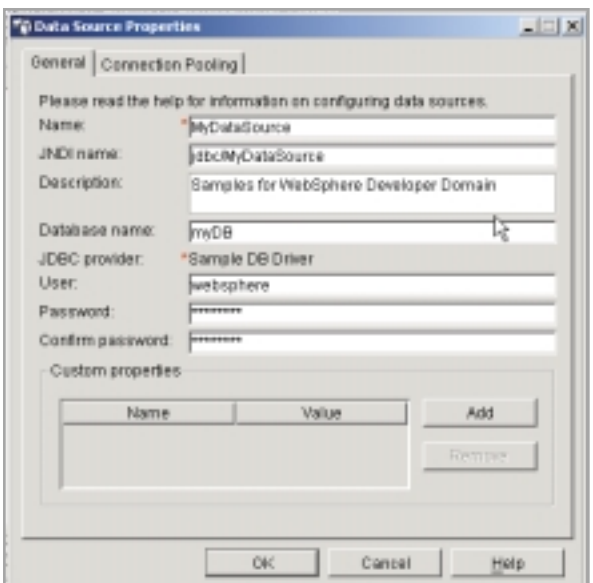


FIG. 3: Configuring a DataSource in WebSphere 4.0

erence to the datasource object throughout its life. When a member method needs a Connection, it creates one by calling the datasource's getConnection() method.

The advantage of caching the datasource is that you prevent multiple JNDI lookups, another expensive operation. WebSphere provides additional caching in this area, but caching it locally is advantageous.

Make Sure to Clean Up!

The WebSphere connection manager does a good job of caring for your connections, but nothing makes up for careless coding. Make sure to clean up the open database resource objects properly.

The rule of thumb when cleaning up JDBC objects is to close them in the reverse order they were created. As an example, most database read operations encompass a Connection, which creates or prepares a Statement (or PreparedStatement), which produces a ResultSet. In this case, close the ResultSet first, the Statement second, the Connection last. The order of the cleanup is important to prevent database-specific annoyances, like running out of cursors.

The best way to ensure proper cleanup is to place all cleanup logic in a finally clause of a try-catch-finally block. The code in a finally clause is guaranteed to run in all cases short of a JVM crash, so a Connection.close() in a finally clause is sure to return the Connection to the pool. Example:

```
try {
      con = ds.getConnection();
      ps = con.prepareStatement("select *
      from mytable where mycolumn = ?");
      ps.setString(1, someString);
      rs = ps.executeQuery();
      // do something with the ResultSet
} catch (SQLException ex) {
      // handle the SQLException
} finally {
      try { rs.close(); } catch(SQLException ex) {rs = null;}
      try { ps.close(); } catch (SQLException ex) {ps = null;}
      try { con.close(); } catch (SQLException ex) {con = null;}
}
```

One alternative to closing Connection objects in finally clauses is to close them in a finalize() method of an object. This approach is *not* recommended, as the finalize() method is only called prior to garbage collection, an asynchronous operation. Also, the only reason to postpone closing a Connection in this manner would imply that you're caching it in the object, another poor practice. Cleaning up JDBC objects correctly can prevent many unnecessary headaches.

When Good Connections Go Bad

Occasionally database connectivity breaks unexpectedly. Network cables get pulled, electricity goes out, or sometimes a database server just gets pegged. Although nothing can be done until the database is brought back online, these sorts of outages don't have to permanently bring down your application's functionality.

The WebSphere connection manager provides a convenient manner of handling temporary connectivity outages. Proper handling of the StaleConnectionException allows developers to create more fault-tolerant applications.

The StaleConnectionException is an exception that extends SQLException. This is necessary in order to follow the JDBC specification (most JDBC methods may only throw a SQLException). This is important to note because if an application is to catch a StaleConnectionException, it must be caught before catching the SQLException. The actual package name for the StaleConnectionException in version 3.5 is com.ibm.ejs.cm.portability; in 4.0 it's com.ibm.websphere.ce.cm (the 3.5 package name will work in 4.0 but is deprecated).

Just for clarification, when a Connection is marked "stale," that means that on its next operation it will throw a StaleConnectionException.

A Connection can "go bad" in any number of situations:

1. The application attempts to acquire a Connection during a database failure or when the database has not yet been started.

2. All of the Connection objects in the pool have gone bad due to a database failure. In this case, any new calls to getConnection() will return a stale Connection object, and all Connection objects in use would throw this exception when the application attempts to use them.
3. The application using the Connection has previously called its close() method, but later tries to use the connection again.
4. The application acquired a Connection but didn't use it within the orphan timeout period. In this case, the connection manager orphans (preempts) the Connection and the next time the application attempts to use it (or a JDBC resource that it created; e.g. Statement, ResultSet, etc.) a StaleConnectionException will be thrown.

Handling a StaleConnection-Exception is entirely up to the application designer. For more fault tolerance, one recommended approach is to catch the StaleConnection-Exception and retry the database operation up to a predefined number of tries. Here's an example:

```
boolean retry = false;
 do {
  try {
        retry = false;
        con = ds.getConnection();
        stmt = con.createStatement();
        rowsUpdated = stmt.execute("update myTable set myColumn =
                                    5000");
  } catch(com.ibm.ejs.cm.portability.StaleConnectionExce-
          ption sce) {
        if (retryAttempts++ < MAX_RETRIES)retry = true;
       } catch(SQLException sqle) {
              /* handle a more severe fail-
    ure and exit loop */
  } finally {/*Close stmt and con */cleanupJDBCResources();
                    }
} while (retry);
```

Normal execution of this code will only execute the logic in the loop once; only in error conditions will the thread retry the operation. The connection manager could throw a StaleConnectionException on any of the operations on lines four through six. In this event, the thread will check to see if it's reached the maximum number of retries (set prior to execution), and if so retry the entire operation, starting by obtaining a new Connection object.

Handling the StaleConnectionException in this manner will allow your Web application to recover from a number of temporary database outages that would normally return an error page to your Internet customer.

Wrapping Up

The practices outlined here are recommended as best practices, and in general are good advice for building WebSphere applications. Nevertheless, none of these tips will replace a poor design; they are meant to complement a well-designed application. They're also meant to provide insight into how the connection manager functions. This information should help in future development projects as well as in maintaining your current applications.

Managing JDBC connections can be tricky, but with a good application design and by following these recommendations, you should be well on your way to delivering high-performing and scalable enterprise solutions.

Resources

Cuomo, G., *A Methodology for Production Performance Tuning*
Lauzon, S., and Schommer, P., *Handling Database Failures*
Erickson, D., *Connection Manager Whitepaper*
WebSphere 3.5 Handbook Redbook

These articles and whitepapers can be found online at:
www.ibm.com/software/webservers/appserv/library.html
www7b.boulder.ibm.com/wsdd and www.redbooks.ibm.com

wireless EDGE conference&expo

MAY 7–9, 2002

Santa Clara Convention Center
Santa Clara, California

STRATEGIES & SOLUTIONS FOR UNWIRING THE ENTERPRISE

International Wireless Business & Technology Conference & Expo

Wireless Edge will provide the depth and breadth of education and product resources to allow companies to shape and implement their wireless strategy. Developers, *i*-technology professionals and IT/IS management will eagerly attend.

Plan to Exhibit

Provide the Resources To Implement Wireless Strategy

The conference will motivate and educate. The expo is where attendees will want to turn ideas into reality. Be present to offer your solutions.

Who Should Attend

Mobile & Wireless Application Professionals
who are driving their enterprises' wireless initiatives:

- Program Developers
- Development Managers
- Project Managers
- Project Leaders
- Network Managers
- Senior IT and Business Executives

CONFERENCE TRACKS

TRACK 1: DEVELOPMENT

- WAP
- i-Mode
- Bluetooth / 802.11
- Short Messaging
- Interactive Gaming
- GPS / Location-Based
- Wireless Java
- XML & Wireless technologies

TRACK 2: CONNECTIVITY

- Smart Cards
- Wireless LANs incl. Bluetooth
- UMTS/3G Networks
- Satellite Broadband

TRACK 3: WIRELESS APPS

- Education
- Healthcare
- Entertainment
- Transport
- Financial Services
- Supply Chain Management

TRACK 4: HARDWARE

- Cell Phones/WorldPhones
- PDAs
- Headphones / Keyboards / Peripherals
- Transmitters / Base stations
- Tablets

TRACK 5: MANAGEMENT

- Wireless in Vertical Industries
- The WWWW
- Unwired Management
- From 3W to 4W: Issues and Trends
- "Always-On" Management
- Exploiting the Bandwidth Edge
- Unplugged Valueware
- Wireless Sales & Marketing

FOR EXHIBIT & SPONSORSHIP INFORMATION PLEASE CALL
201 802-3004

Arie Mazur
President, Chief Executive Officer, and Chairman, Slangsoft

Chris Bennett
Architect, Freedom Technologies

Daniel Elliott
Senior Vice President, Mobile Business, CompuCom

Douglas Lamont
Professor of Marketing, DePaul University

Kenneth Leung
Business Development Manager, Retail and Emerging Technologies, IBM

Keith McIntyre
Vice President and Chief Technologist, Stellcom

Steve Milroy
Wireless Technologist, Immedient

Tony Wasserman
Director, Mobile Middleware Lab, Hewlett-Packard Company

WWW.SYS-CON.COM/WIRELESSEDGE2002

Exclusive Sponsorship Available

Rise above the noise. Establish your company as a market leader. Deliver your message with the marketing support of

SYS-CON MEDIA

ONLINE REGISTRATION NOW OPEN

WWW.SYS-CON.COM/WIRELESSEDGE2002

Or, To Register By Phone
Call: (201) 802-3069

	Development	Connectivity	Wireless Applications	Hardware	Management
MAY 8, 2002					
8:30-9:45	Embedded Java (Bill Ray, Network 23 Limited)	A Real Time Model of (3G) UMTS Access Stratum (Niloy Mukherjee, MIT Media Laboratory)	Using Mobility to Streamline Traditional Business Processes (Dan Elliott, CompuCom)	Java Software Performance On Wireless Devices: Myths and Realities (Ron Stein, Nazomi)	LMDS - The Last Mile Enabler of Class 5 Revenues with VoIP (Ed Peters, Ensemble Communications, Inc.)
10:00-11:15	KEYNOTE				
11:30 A.M.	EXPO FLOOR OPEN				
	LUNCH BREAK				
1:30-2:30	XML & Wireless Technologies (Karl Best,OASIS)	Securing Wireless Data Via Smart Card (Joseph Smith, New Dominion Software)	Collaboration for Wireless Warriors (Timothy Butler, SiteScape, Inc)	User Interactivity for Information Appliances (Arie Mazur, Slangsoft)	Leveraging Wireless In Customer Acquisition and Retention (Kenneth Leung, IBM)
2:45-3:45	KEYNOTE				
3:45-4:45	Developing Mobile Web Applications (Tony Wasserman, Hewlett-Packard Company)	In-building Wireless, the Next Frontier (Mary Jesse, RadioFrame Networks)	Turbocharge Mobile Applications with J2EE (Dr. Jeff Capone, Aligo, Inc.)	Cell Phones/ WorldPhones (Speaker TBA)	Mobilize Your Enterprise (Chris Bennett, Freedom Technologies)
5:00-6:00	VoiceXML Workshop (Bryan Michael, BeVocal)	Satellite Broadband (Speaker TBA)	Mobile Portals - The First Step Towards the Mobile Enterprise (Tony Wasserman, Hewlett-Packard Company)	PDAs (Speaker TBA)	Wireless Solution Return On Investment (ROI) In the Real World (Steve Milroy, Immedient)
MAY 9, 2002					
8:30-9:45	Brew, I-Mode, WAP, and J2ME: How the Battlefield Is Shaping Up at the Start of the Mobile War (Reza B'Far, eBuilt, Inc.)	The Future of IP Mobility (Antti Eravaara, NetSeal, Inc.)	Multimodality: Revolutionizing Our Wireless Lifestyles (Arvind Rao, OnMobile Systems)	Tablets (Speaker TBA)	Marketing Value in Automotive Telematics Through Mobile Communications (Douglas Lamont, DePaul University)
10:00-11:15	KEYNOTE				
11:30 A.M.	Guide to Developing Applications with Micro Java (Kurt Baker, Kada Systems)	EXPO FLOOR OPEN			
	LUNCH BREAK				
1:45-3:00	Building Secure Mobile Solutions (Keith McIntyre, Stellcom)	Smart Card Communications (Bill Ray, Network 23 Limited)	Developing New Applications Using VoiceXML (Jonathan Taylor, Voxeo)	PDAs (Speaker TBA)	Policies and Profiles: The Key to Mobile Data Services (Doug Somers, Bridgewater Systems)
3:15-4:30	Bluetooth™ Wireless Technology and Java ™ Technology (Michael Portwood, Exuberance, LLC)	UMTS/3G Networks (Speaker TBA)	Instant Messaging and Wireless Computing Collide (JP Morgenthal, IKimbo)	Transmitters / Base Stations (Speaker TBA)	Total Business Solutions B2E (Speaker TBA)
4:45-6:00	Creating Carrier Optimized Wireless Internet Applications (David Young & Victor Brilon, Lutris)	Wireless LANs/Bluetooth (Speaker TBA)	Rapid Development of Successful Wireless Applications (Rod Montrose, AVIDWireless)	Headphones / Keyboards / Peripherals (Speaker TBA)	Wireless Sales & Marketing (Speaker TBA)

REGISTER BEFORE March 1 And Save Up To $400

Plan to Attend the 3-DAY Conference

PRODUCED BY
SYS-CON EVENTS

May 7 will feature full-day tutorials. Consult www.sys-con.com/wirelessedge2002 for up to-the-minute conference news.

Traditional catalog retailer transitions to e-commerce model

Lillian Vernon and E-commerce: Categorically Best of Breed

BY PAUL GOODMAN

The fastest-growing and most exciting part of Lilian Vernon's multi-channel business is the company's new Web site, www.lillianvernon.com. We've shown our confidence in both the future of the Web and our overall e-business strategy by making a major investment in this new Web site.

Lillian Vernon receives 4.4 million orders a year; during our peak holiday week last year, we processed more than 289,000 orders. Our challenge had been keeping the information on our Web site current and in line with promotions, pricing, and a constantly evolving product line.

Rethinking Our Strategy

In recent years, the Lillian Vernon Web site staff needed help in overcoming the production bottlenecks we encountered getting all this information online during this busiest time of year. The site clearly needed to decentralize, streamline, and automate the creation, management, and delivery of content. This challenge, along with increasing e-commerce opportunities during the holidays and the demands of our shoppers, caused us to rethink our Web strategy.

We agreed that our goal was to deliver an attractive, accurate, easy-to-navigate, high-performing site that pleased our customers and resulted in additional profits for Lillian Vernon. We needed an exceptional technology infrastructure to support that goal. We believed that choosing the right blend of technology would empower our Web staff as well as content contributors, significantly boosting the processing of holiday sales and helping us provide world-class customer service.

The Makeover Begins

With these goals in mind, 12 months ago Lillian Vernon initiated a project to revamp the design and production of our online catalog. We made two essential technology choices: IBM's WebSphere Commerce Suite and Content Server Enterprise Edition (CSEE) from divine. [See *Editor's Note* at the end of this article.]

Lillian Vernon turned to IBM to provide the e-commerce technology that would enable shoppers to browse and place orders easily. The company also wanted a site that would both have a fresh new look and be easier to update, so we looked to divine's technology for new, more powerful content management capabilities.

Our goal was to empower not just the Web team, but the entire staff that contributes content to our online catalog and to speed up the content creation, editing, approval, and publishing process. We knew that creating an exceptional open standards–based technology Web infrastructure would create efficiencies, save money, and generate additional sales. Equally important was the launching of a compelling site that would encourage customer loyalty by providing a superior shopping experience.

IMPLEMENTING CSEE

With divine's CSEE, Lillian Vernon executives, marketing staff, and other nontechnical personnel can now easily update site content. The distribution of content contribution across a larger group of employees and a new, more efficient workflow significantly reduces the time it takes for new information to appear on-site. Prior to implementing divine's technology, the company had no content management and delivery solution. Web site editors manually gathered, edited, and approved content, giving the IT department the job of posting it for our online catalog – a time-consuming and inefficient process, to say the least. The new system reduces the impact on our IT resources and more efficiently leverages the content and retail expertise of the rest of our staff.

LOOKING TO IBM'S WEBSPHERE COMMERCE SUITE

IBM's WebSphere Commerce Suite allows Lillian Vernon to quickly, easily, and securely sell our catalog items through either our Web site or our customer's wireless devices while integrating all our internal business processes. Commerce Suite also includes personalization capabilities and other features to market products to customers online. IBM's commerce solution helped Lillian Vernon create a Web site that's flexible, customized, and tailored for different groups of products that can be viewed by shoppers in multiple categories. Commerce Suite also handles order processing, pricing, partial orders, and multiple shipping addresses. But besides the critical issues around quick and accurate processing of orders and transactions, we had other requirements for our new site.

Expanding Web Content Management

Recognizing that content management and delivery is the foundation of

ABOUT THE AUTHOR

Paul Goodman is vice president, Lillian Vernon Online, a division of Lillian Vernon Corporation.

successful e-commerce, we needed a system powerful enough to publish rapidly changing information. divine's CSEE made it easy to post new information on our Web site quickly, giving our shoppers the most current information available, increasing customer satisfaction and helping to enhance site profitability. With its complete integration with Commerce Suite and the IBM Websphere application server, CSEE met two critical Lillian Vernon requirements – to build on and maximize the investment in our existing WebSphere Commerce Suite system. And, their close integration with WebSphere application server provides us with scalability as we grow and the site reliability our customers demand.

TECHNOLOGY BUILT FOR THE WAY PEOPLE WORK

CSEE's foundation Content Server uses Java components and XML-based services, so our developers can rapidly create and deploy new applications with reuseable components. In addition, the time-saving Java tools that are a part of Content Server are a clear enhancement to their productivity. Another CSEE product, Content Center, streamlines the steps in site production by tying together content access, creation, and custom workflow management. Lillian Vernon's site is a hybrid of static content and dynamic pricing, ordering, and inventory applications. This makes it a perfect candidate for Content Center, which is designed to combine static and dynamic content delivery into one smooth and seamless process. In doing so, Content Center helps our sales, marketing, and fulfillment departments easily update information and manage the look and feel of the site themselves.

BUILDING ON OPEN STANDARDS

The combination of WebSphere Commerce Suite and divine's CSEE is a testament to the benefits of building on open standards, like Java 2 Enterprise Edition (J2EE). Its real advantage derives from how fully it exploits the scalability, flexibility, reliability, and power of the J2EE WebSphere application server. The combined technologies accommodate peaks in site visit and buying demand by drawing on the scalability inherent in J2EE. It's easy to add other J2EE-based components as Web site traffic grows. It delivers content faster, with a lower cost for hardware, software, operations, and services than other systems because the number of CPUs required by CSEE is approximately 30% less than those of competitive systems. This gives Lillian Vernon the option of dedicating spare servers to the handling of overflow traffic during high volume buying periods.

GETTING WHAT WE NEEDED FOR THE HOLIDAYS

When assessing new content management systems, we also were impressed that CSEE let users preview HTML pages containing new content before publishing them on the site. We also appreciate CSEE's workflow tracking tool, which assigns security levels and specific tasks to editors, graphic artists, and marketing executives as well as our Webmaster – who, tracks work progress and approvals.

Our writers now input content into the system themselves, preview it, and send it along a customized workflow path for editing and approval. In fact, staff from all departments can contribute to the Web site when appropriate and content editors can better manage online content and promotions. This has cut down the number of revisions and approval cycles, speeding production considerably. Now that our IT staff has been freed from handling content, they can focus on the Web site's technical operation and performance.

In the catalog retailing business, time is always of the essence, but especially during the holiday shopping season. divine and IBM are eliminating previous bottlenecks, allowing us to streamline our online production processes, content delivery, order management, and transactions. This means more time for our customers and for our business.

The bottom line is: happy customers equal profits. We can't have a first generation Web site and a second class e-business. Our ability to excel in a crowded marketplace depends on implementing the best Web technology infrastructure on the market.

Developing a Brand Name

More than 45 million Americans know the Lillian Vernon name, according to a recent Opinion Research Poll. During the past five years, the name has appeared on 877 million catalogs, 28 million shipping boxes, and 93 million products. In 2001, the company mailed 169 million catalogs in 35 editions, generating revenues of more than $287 million. The company's database comprises more than 23 million customers in 50 states.

COMPANY HISTORY

Lillian Vernon founded her company on her kitchen table in 1951. Born in Leipzig, Germany, Lillian's family fled from the Nazis before World War II, first to Amsterdam, and then to New York City, where she grew up. In 1951, a housewife expecting her first child and living in Mount Vernon, New York, Lillian used $2,000 of her wedding gift money to launch her company. Using her hometown's name, she called the company Vernon Specialties. She designed a purse and belt, and placed a $495 ad in *Seventeen Magazine* offering a unique service – personalization with her customer's initials free of charge. The ad brought in $32,000 in orders and her company was launched.

At first she sold only matching bags and belts through small magazine ads, but then Lillian began to branch out. She was a pioneer in offering personalized merchandise by mail. In 1954, she moved into a storefront that served as her warehouse. In 1956, she mailed her first 16-page catalog.

With one of the largest, state-of-the-art personalization departments in the country, Lillian Vernon has eliminated the weeks of delay usually associated with ordering personalized merchandise. It can personalize any name on practically any item – from luggage to stationery to children's toys. Being able to offer such a unique service on a huge scale is what distinguishes the Lillian Vernon brand.

Lillian Vernon has grown over the past 51 years. The company headquarters are in Rye, New York. Its National Distribution Center sits on 62 acres in Virginia Beach, Virginia. The center, which fulfills those 4.4 million orders, encompasses a million square feet of warehouse space. The company also has a business-to-business divison and 15 outlet stores in Virginia, New York, South Carolina, Tennessee, and Delaware. In 1995, Lillian Vernon started marketing its products over the Web. The mail order retailer also operates a seasonal call center in New Rochelle, New York. The peak season is from mid-September through December.

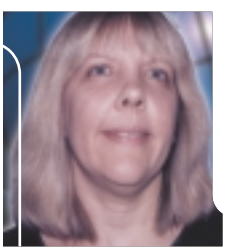
Putting IBM's WebSphere Commerce Suite and divine's CSEE to work, Lillian Vernon has significantly improved its Web site production and the experience we provide for our online shoppers. With easier ordering for online shoppers and lower costs of operating our Web site, we plan on rapidly accelerating our online sales in the years ahead.

Editor's Note: Open Market, Inc., was acquired by Chicago-based divine, Inc., on Oct. 19, 2001. See www.divine.com for more information.

Understanding the bean life cycle

Configuring Entity Beans In WebSphere Application Server 4.0

BY LAUREL NEUSTADTER



EJB application servers speed up software development by eliminating the need to build costly infrastructure. In particular, developers don't need to worry about implementing code that handles concurrency. This does not mean, however, that developers can totally ignore concurrency issues: Configuring an application server for optimal performance requires a thorough understanding of concurrency issues.

ABOUT THE AUTHOR

Laurel Neustadter is a senior consultant at Valtech Technologies, Inc., where she is a trainer specializing in J2EE technologies. Prior to becoming a trainer, she led Java server-side software development projects. Laurel has a Ph.D. in Operations Research from UCLA.

EMAIL

lneustadter@yahoo.com

This article looks at how entity beans synchronize with the underlying database, and the different ways in which entity beans can be configured in WebSphere Application Server 4.0 (WAS 4.0). Entity bean configuration settings impact performance. Determining the optimal settings requires understanding the entity bean life cycle, transactions, and WAS 4.0 configuration options, as well as understanding the application's requirements.

Entity Bean Life Cycle

Consider the scenario where an entity exists in the database, and a client wishes to access that entity via an entity bean. Figure 1 shows the valid states for an entity bean, and how an entity bean can transition from one state to another.

When the entity bean instance is first created, the EJB container invokes setEntityContext() on the instance. Once setEntityContext() completes, the instance is associated with an entity context object and is in the Pooled state. When an instance is in the Pooled state, it is not associated with any entity (i.e., with any primary key). Instances in the Pooled state can be used to satisfy finder requests.

When the container needs the bean instance to service a client, the container invokes ejbActivate() on the instance. When that completes, the instance is in the Ready state – the bean is associated with an entity (i.e., with a primary key).

While the instance is in the Ready state, the following methods can be invoked:

- Business methods
- ejbLoad()
- ejbStore()

Any of these methods can be invoked multiple times. The methods ejbLoad() and ejbStore() synchronize the bean instance's state with that of the database. The ejbLoad() method reads from the database, and the ejbStore() method writes to the database.

When the bean instance is no longer needed, the container invokes ejbPassivate() on the instance. Once that completes, the instance is again in the Pooled state, and the entity bean instance is no longer associated with an entity (i.e., with a primary key).

An entity bean instance could move between the Pooled and Ready states many times in its lifetime, and could be associated with different primary keys at different points. At any *given* point in time, however, an entity bean instance is associated with at most one primary key – it is associated with no primary key if it is in the Pooled state, and is associated with one primary key if it is in the Ready state.

Finally, just before the entity bean instance is destroyed, the EJB container invokes unsetEntityContext() on the instance.

There are WAS 4.0–specific assembly properties that impact the pattern of ejbActivate(), ejbPassivate(), ejbLoad(), and ejbStore() calls, and hence impact performance.

The methods ejbCreate() and ejbPostCreate() are called when a new entity is inserted into the database; ejbRemove() is called when an entity is removed from the database. The assembly properties discussed in this paper aren't relevant to the creation and removal scenarios.

Transactions and the Entity Bean Life Cycle

A call to ejbLoad() typically results in a SQL SELECT being performed, and a call to ejbStore() results in a SQL UPDATE being performed. When

does the container invoke ejbLoad() and ejbStore()? The method ejbLoad() is typically invoked when a transaction begins, and ejbStore() is typically invoked when a transaction commits.

For example, in Figure 2 Client 1 and Client 2 are simultaneously accessing Account 1. In an EJB implementation, clients access Account data via entity beans. Each client has its own instance of an Account entity bean, and each Account entity bean instance holds a copy of the account data for ID 1.

Figure 3 shows how transaction boundaries drive calls to ejbLoad() and ejbStore(). Assume deposit() has the transaction attribute RequiresNew, and the transaction isolation level is serializable. Account 1 starts out with $100. When Client 1 invokes deposit(100) on instance1, a transaction involving instance1 starts, so the container invokes ejbLoad() on instance1. This results in instance1 loading the amount $100. The method deposit(100) is then invoked, which changes instance1.amount to $200. The transaction commits and the container invokes ejbStore() on instance1. This saves the amount $200 to the database. The same general process is repeated for instance2. (Since the isolation level is serializable, Client 2 must wait for Client 1's transaction to complete before loading the Account 1 data.) This time, instance2 loads $200 from the database, and writes $300 back to it.

We've seen that when transaction boundaries drive calls to ejbLoad() and ejbStore(), data integrity is preserved. There's one drawback, however, to invoking ejbLoad() and ejbStore() on each transaction: reading from and writing to the database is expensive. What if we know that the underlying data in the database has not changed since the last ejbLoad()? Can we avoid paying the price of the unnecessary ejbLoad()? What if clients don't actually update the entity during a transaction? Can we avoid paying the price of the unnecessary ejbStore()? WAS 4.0 allows you to configure entity beans to avoid unnecessary ejbLoad() and ejbStore() calls.

Entity Bean Commit Options

In the previous section, I said that when a transaction begins, the container typically invokes ejbLoad(), and when a transaction commits, the container typically invokes ejbStore(). Actually, the EJB 1.1 specification discusses three possible options for synchronizing the state of a given bean instance with the data in the database.

OPTION A: CACHED BEAN

Option A requires that an entity bean instance have exclusive access to its data. With this option, when a transaction commits, the entity bean instance's data is written to the database (i.e., ejbStore() is called), and the instance remains in the Ready state. Since the instance has exclusive access to the entity's data, the instance's data remains valid, and the next time a transaction begins, ejbLoad() is not invoked. If there are concurrent requests for the same entity, the EJB container queues up the requests (see Figure 4)

The assumption that entity beans have exclusive access to their data is a difficult one. In particular, this assumption precludes clustering and the idea that other applications may access the entity data.

If you use Option A, make sure that your entity beans truly have exclusive access to their data! Note that Option A does reduce concurrency.

OPTION B: STALE BEAN

Option B assumes that an entity bean instance does not have exclusive access to its data. With this option, when a transaction commits, the entity bean instance's data is written to the database, and the instance remains in the Ready state. Since the instance does not have exclusive access to its data, it is assumed the instance's data does not remain valid. As a result, the next time a transaction begins, ejbLoad() is invoked. If there are concurrent requests for the same entity, the EJB container assigns a different entity bean instance to each request. For example, if there are two concurrent requests for Account 1, there will be at least two Account entity bean instances in memory at the same time, both of which

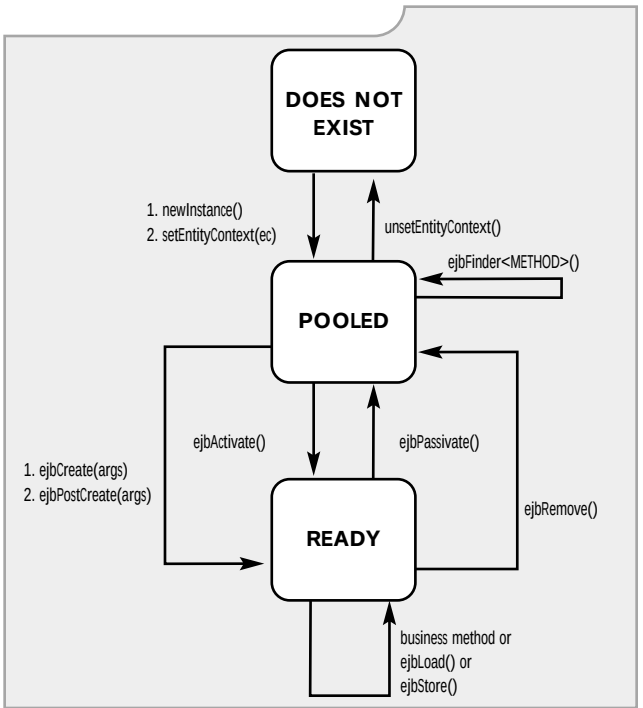


FIG 1: Entity Bean state model

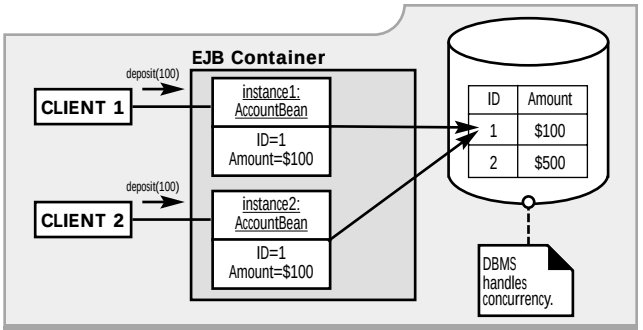


FIG 2: Two clients concurrently access Account 1

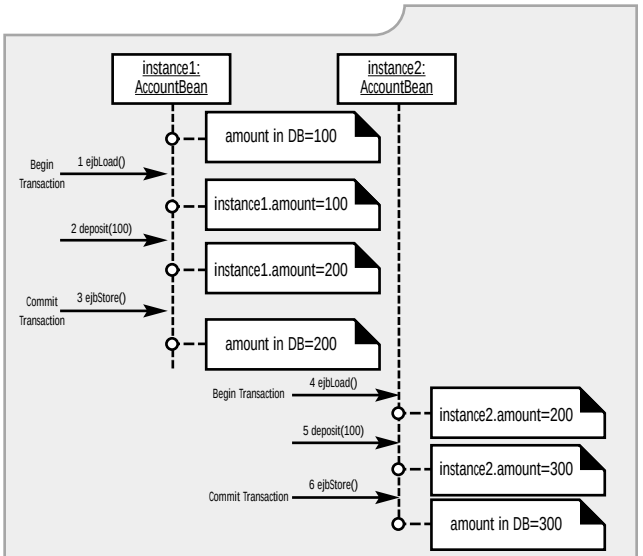


FIG 3: Concurrency with Transactions

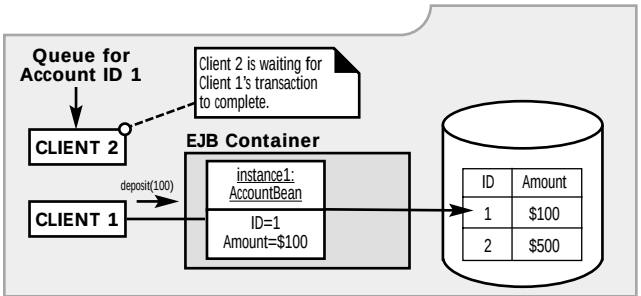


FIG. 4: Commit Option A and Pessimistic Server-Side Locking

	ONCE	TRANSACTION
ACTIVATION	OPTION A ("CACHED BEANS")	OPTION C ("POOLED BEANS")
TRANSACTION	OPTION B ("STALE BEANS")	OPTION C ("POOLED BEANS")

TABLE 1: Mapping Bean Cache Assembly Properties to Commit Options

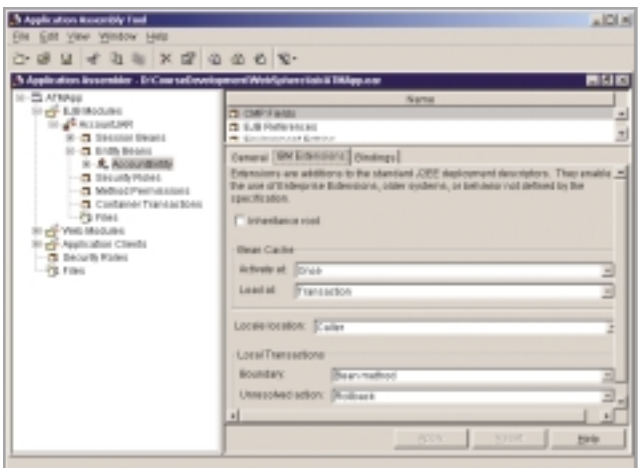


FIG 5: Specifying Bean Cache Properties

have primary key 1. Note that Option B provides better performance with respect to concurrency than Option A, but it results in additional ejbLoad() calls and more memory being used. The earlier discussion of transactions implicitly assumed Option B.

OPTION C: POOLED BEAN

Like Option B, Option C assumes that an entity bean instance does not have exclusive access to its data. And like Options A and B, when a transaction commits, the entity bean instance's data is written to the database. However, unlike those options, the instance does not stay in the Ready state. Instead, it is returned to the pool, and it is no longer associated with any entity (i.e., with any primary key). The difference between Options B and C is:

- With Option B, ejbLoad() is called at the beginning of each transaction, and ejbStore() is called at the end of each transaction.
- With Option C, ejbActivate() and ejbLoad() are called at the beginning of each transaction, and ejbStore() and ejbPassivate() are called at the end of each transaction.

If ejbActivate() and ejbPassivate() are expensive calls, Option B may offer a performance advantage. If ejbActivate() and ejbPassivate() are not expensive calls (e.g., they are no-ops), Option B probably doesn't offer any performance advantage with respect to callback processing time.

Option C has the potential to use less memory than Options A and B, since beans are not kept in the Ready state. Look again at Figure 2. With Option B, once Client 1's and Client 2's transactions end, instance1 and instance2 will remain in memory in the Ready state, and both instances remain associated with Account 1. If a client then makes a request for an Account entity bean for Account 2, a new Account entity bean will be created, and will be associated with primary key 2.

In contrast, with Option C, once client 1's and client 2's transactions end, instance1 and instance2 are no longer associated with Account 1, and both instances are returned to the pool. If a client then makes a request for an Account entity bean for Account number 2, instance1 or instance2 can be used to hold the Account 2 data. A similar analysis holds for Option A versus Option C.

Specifying Commit Options

In WAS 4.0, you specify Commit Option A, B, or C via the assembly properties "Bean Cache—Activate At" and "Bean Cache—Load At". The property "Bean Cache—Activate At", which specifies at what point in time a bean should be activated, can take the values "Once" or "Transaction". The property "Bean Cache—Load At", which specifies when ejbLoad() should be called, can take the values "Activation" or "Transaction". Table 1 shows how these values map to the Commit Options discussed earlier. For example, if you want an entity bean to use Option A, set "Bean Cache—Activate At" equal to "Once", and set "Bean Cache—Load At" equal to "Activation".

You can use the Application Assembly Tool (AAT) to set these properties. Figure 5 shows the screen in which the Bean Cache properties are specified. It shows that the Account entity bean will use Commit Option B.

Locking: Where is it Done?

Entity locking can occur in two places: in the WebSphere Application Server, or in the DBMS.

WEBSPHERE APPLICATION SERVER

In Commit Option A, WAS performs locking – commonly referred to here

"CONFIGURING AN APPLICATION SERVER FOR OPTIMAL PERFORMANCE REQUIRES A THOROUGH UNDERSTANDING OF CONCURRENCY ISSUES."

as pessimistic locking. With pessimistic locking, the application server serializes all transactions accessing a bean instance. Figure 4 illustrates pessimistic locking in the application server.

In Commit Options B and C, WAS performs no locking – sometimes (erroneously) referred to as optimistic locking in the application server. With optimistic locking, the application server doesn't serialize transactions. Instead, each transaction creates its own copy of a bean instance. All locking is handled by the underlying DBMS. Figure 2 illustrates so-called optimistic locking in the application server.

DBMS

With Commit Options B and C, all locking is delegated to the underlying DBMS. The type of locking done depends on the transaction isolation level. You can specify the desired transaction isolation level for each entity and session bean method as:

1. Read Uncommitted
2. Read Committed
3. Repeatable Read
4. Serializable

The higher the number, the more isolated the transactions are from one another. There is a trade-off between isolating transactions and performance. Always use the lowest isolation level possible.

The exact meaning of each isolation level is beyond the scope of this article The WAS 4.0 document Enterprise Beans has a good discussion of the different isolation levels. The exact behavior of a given isolation level is DBMS-dependent, so be sure to consult your DBMS documentation.

In WAS 4.0, you specify the isolation level for a method via the assembly property "Isolation Level Attributes/Isolation Level". You can use the Application Assembly Tool (AAT) to specify a method's isolation level. Figure 6 shows the screen where isolation levels are specified. In that screen shot, deposit() has been specified to have the isolation level "Serializable".

In a transactional context, the isolation level associated with the first method invocation becomes the required isolation level for all methods invoked within that transaction. If a method is invoked with a different isolation level from that of the first method, the java.rmi.RemoteException is thrown.

Specify Read-Only Methods

We have seen that different Commit Options result in different entity bean callback patterns. All Commit Options, however, result in an ejbStore() being invoked. What if an entity bean instance's data isn't modified during a transaction? In this case, calling ejbStore() and doing a SQL UPDATE is unnecessary. In WAS 4.0, you specify that an entity bean method is read-only via the assembly property "Access Intent—Intent Type". Within a transaction, if only read-only methods are invoked on an entity bean instance, the EJB container will not invoke ejbStore() on that instance.

You can use the Application Assembly Tool (AAT) to specify that a method is read-only. Figure 7 shows the screen where read-only methods are specified.

In Figure 8, getBalance() is the only method invoked on instance1. Since getBalance() is read-only, the call to ejbStore() is unnecessary. If you've specified getBalance() as read-only, ejbStore() won't be called. If you haven't specified that getBalance() is read-only, ejbStore() is called.

Conclusion

You've learned how several WAS 4.0 assembly properties – Bean Cache—Activate At, Bean Cache, Load At, Isolation Level Attributes/Isolation Level, and Access Intent–Intent Type – impact concurrency, memory usage, and performance. Now that you understand the implications of these assembly properties, you can choose the best settings for your particular application.

References

- Enterprise Beans, PDF document, IBM WebSphere InfoCenter. See section 6.6.5: Administering EJB modules.

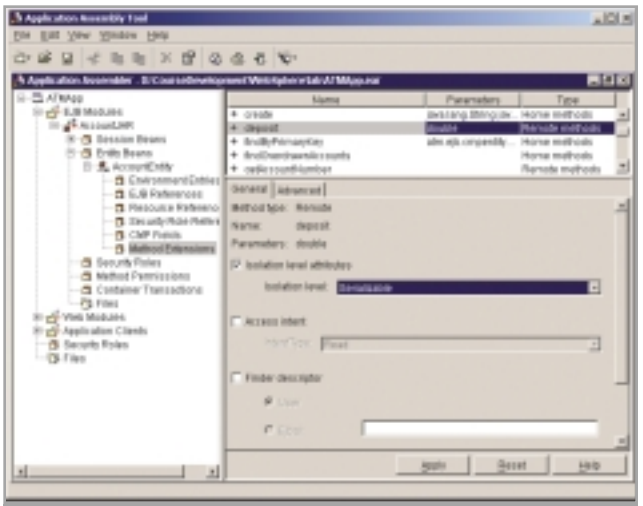


FIG 6: Specifying Transaction Isolation Levels

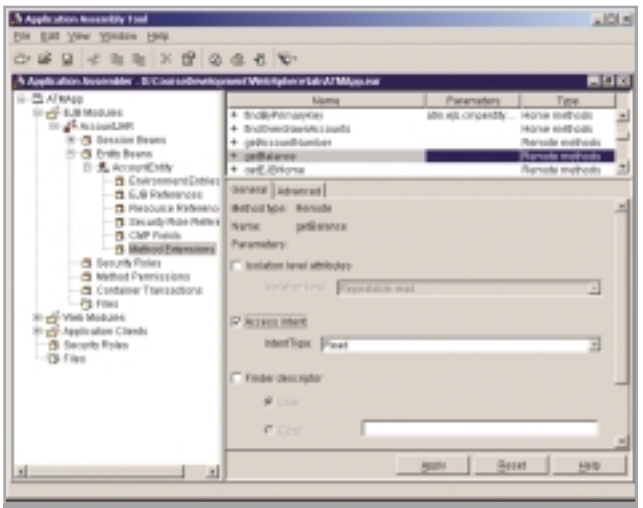


FIG 7: Specifying Read-Only methods

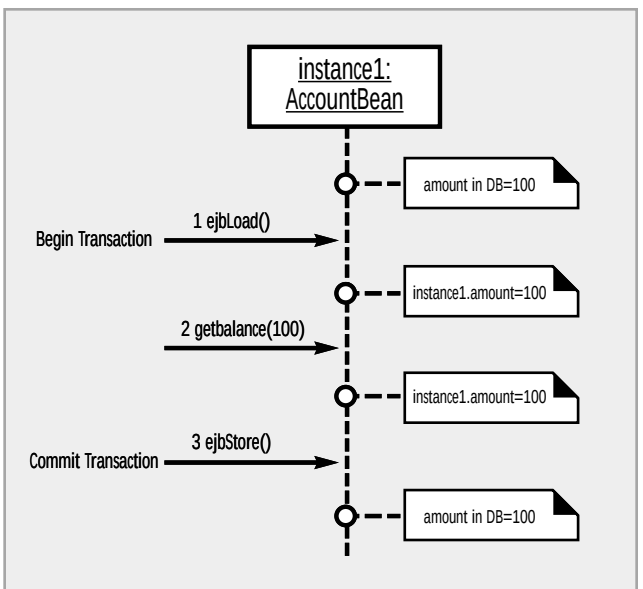


FIG 8: Why invoke ejbStore()?

WEBSPHERE NEWS

Compoze Software to Support Websphere's Application Server

(Philadelphia) – Compoze Software, Inc., a provider of Java software components for adding collaborative functionality to Web-based applications and portals, has announced full support for WebSphere Application Server in its Harmony Component Suite. It will enable customers to quickly build and deploy highly scalable and reliable e-business and portal applications, resulting in reduced time-to-market and development cost savings.

Harmony includes components for calendaring, Web mail, to do, contacts, and discussion; and integrates with legacy systems such as Microsoft Exchange and Lotus Domino. Harmony has been certified on WebSphere with support for all leading operating systems and databases. www.compoze.com/download.html

Independent Test Certifies WebSphere's Performance Leadership

(Somers, NY) – IBM announced that WebSphere has achieved the best overall performance as compared to another major application server vendor running the industry-standard ECperf benchmark test. The results were certified and validated by the ECperf Expert Group, an independent, third-party panel representing the Java community. This benchmark measures scalability, performance, and total cost of ownership. The results show that WebSphere delivers the highest performance software to deploy e-business applications while enabling customers to maintain the lowest total cost of ownership. For more information about WebSphere's ECPerf benchmark results, as well as the full disclosure report, visit http://ecperf.theserverside.com/ecperf/

Healthcare Insurer Selects WebSphere for Infrastructure

(Somers, NY) – BlueCross/BlueShield of Tennessee has chosen WebSphere's e-business infrastructure software, MQ Integrator, to standardize and route claims and manage administrative costs. Officials hope to reduce costs and achieve ROI within two years. In addition, the WebSphere solution will provide rich application functionality and integration across the enterprise, helping to eliminate redundancies.

SEAGULL Announces Support for WebSphere MQSeries

(Atlanta) – SEAGULL, a provider of legacy application enhancement software technology, announced the incorporation of extensive support for MQSeries in its Transidiom software solution, which generates XML, SOAP, Java, and COM interfaces to mainframe and iSeries legacy application functions.

The new MQSeries Communication Manager feature of SEAGULL's Transidiom application integration server provides nonproprietary adapters to mainframe and iSeries applications, connecting legacy systems to the MQSeries infrastructure while requiring no modification to those systems. The combination of MQSeries messaging and Transidiom legacy integration reduces the time, cost, and risk associated with host application integration initiatives. www.seagull-sw.com/index.html

divine Releases New Portlets for WebSphere

(Chicago) – divine, inc., an integrated solutions provider, announced the release of eight content and content-management portal applications, or portlets, for IBM WebSphere Portal Server.

The combination of divine's Enterprise Content solutions and WebSphere Portal Server gives companies the tools to integrate, manage, and deliver both internal and external information to their employees, customers, and partners. For more information about divine portlets for WebSphere Portal Server, visit divine's Web site, www.divine.com. The portlets are available for download at www.divine.com/ibmportlets.

WebSphere Voice Server Brings Transcription to the Enterprise

(SOMERS, N.Y.) – IBM has announced new speech recognition technology for companies to implement advanced, enterprise-wide dictation and transcription. Based on IBM's ViaVoice technology, WebSphere Voice Server for Transcription allows developers to build and customize solutions for specific industries.

The new software has a vocabulary of more than 160,000 words and is designed for multiple users – whether in a small law firm or a large corporation. Also available are additional, specialized vocabularies for medical and legal fields. www.ibm.com/software/speech

UPDATE

eBay and WebSphere: A Successful Match

IBM and eBay announced in September a forged alliance on three fronts. First, IBM landed a very public software coup when eBay selected WebSphere as its next-generation trading platform. Then IBM expanded its presence on eBay – making the trading site a new sales channel. Finally, both companies will explore marketing opportunities in online and offline media.

IBM and eBay will profit from the alliance in several ways. There will be a lot more IBM merchandise to buy/sell on eBay, which will drive even more traffic to one of the busiest sites on the Web. The deal is good for IBM because they've found a place to market their products, which are experiencing explosive growth. They will now tap vast numbers of small business/consumer shoppers through eBay.

A visit the IBM site on eBay, just prior to our deadline, found at the top of the list an AS/400 reburb 9406/650 Server with an asking price of $214,500, wow! There were a dozen lesser-priced servers for the corporate audience and many consumer priced desktops and ThinkPads as well.

It certainly appears that WebSphere's integrated aspects – development, personalization and transcoding, lend themselves to eBay's business model.

In the words of Lou D'Ambrosio, VP Marketing and Sales, IBM Software Group, "eBay is the proof point of success on the Internet." He says eBay is a tremendous vehicle to showcase WebSphere.

JAVAONE

HTTP://JAVA.SUN.COM/JAVAONE/

Fast solutions to WebSphere dilemmas

Breaking into WebSphere

BY JACK MARTIN

ABOUT THE AUTHOR

Jack Martin, editor-in-chief of ***WebSphere Developer's Journal***, is cofounder and CEO of Simplex Knowledge Company, an Internet software boutique specializing in WebSphere development. Simplex developed the first remote video transmission system designed specifically for childcare centers, which received worldwide media attention, and the world's first diagnostic quality ultrasound broadcast system.

E-MAIL

jack@sys-con.com

Q: How do I change the classpath for WebSphere?

A: You can add to the classpath for WebSphere Application Server through its property sheet. To replace a classpath, click the environment field on the general tab to bring up the property editor environment editor. In this editor, change the classpath or other environment variables. To add a classpath, enter the classpath in the command line arguments field.

Q: Do I need to reconfigure server groups if a machine becomes unavailable in the cluster?

A: If a machine becomes unavailable, you don't need to reconfigure server groups to reflect unavailable servers on that machine. However, if the machine is unavailable for an extended period, you can reconfigure the model to optimize performance.

Q: How do I secure an HTML page (assuming security is already configured)?

A: Follow these steps:

1. Create an enterprise application.
2. Assign permissions to the enterprise application (what groups have what access).
3. Create a resource that is the URI for the HTML file.
4. Add the resource to the enterprise application.
5. Specify security mapping for this resource.

Q: How do I install the IBM HTTP Server?

A: It's simplest to install the IBM HTTP Server (IHS) component during the WAS installation. To do so, find the HTTPServer stanza in the WAS response file, and set Result=1. Follow the instructions in the comments of the template custom response file or the InfoCenter documentation to set the installation directory to a location other than the default.

If you want to install the IHS separately from the application server, you can run the IHS installation program directly. The IHS installer executable is on the product CD at X:\NT\HTTPD\setup.exe. Note that the command line syntax to invoke the IHS installer in silent mode is slightly different from the syntax for the WAS installer.

To run the IBM HTTP Server installer in silent mode, specify the following parameters on the command line:

```
X:\NT\HTTPD\setup -
f1<response files> -f2<log
file> -s
```

The following is an example of running the IBM HTTP Server installer in silent mode:

```
D:\NT\HTTPD\setup -
f1C:\Temp\ihs.iss -
f2C:\temp\ihssetup.log -s
```

Just as with WAS, you need to provide a custom response file to run the InstallShield program in silent mode. And, just as with WAS, a template response file is included on the product CD. You can find the template response file at X:\NT\HTTPD\Typical.iss. Copy this file to create a basis for your own response file. Be sure to update the szAskDestPath-0 stanza so the szDir property points to the target installation directory. Also, edit the UserID-0 stanza and ensure that the properties szEdit1 and szEdit2 are respectively set to a valid user ID and password on the target system.

Q: How do I activate the X Keyboard Extensions?

A: To enable the X Keyboard Extensions on your AIX system, your system administrator must modify the /usr/lpp/X11/defaults/xserverrc file, which is the file that starts the X server.

This example xserverrc file entry invokes the following options:

```
+kb     Enables keyboard
extensions
+accessx      Enables acces-
sibility extensions
#--------------------------
# Turn X11R6 keyboard exten-
sion on.
#--------------------------
EXTENSIONS="$EXTENSIONS +kb
+accessx"
```

Q: What does it mean that WebSphere Application Server is Tivoli Ready?

A: IBM WebSphere Application Server Version 3.0 is Tivoli Ready, meaning it can be managed through Tivoli Enterprise Console (TEC) or Tivoli Global Enterprise Manager (GEM).

To enable Tivoli Ready support, you must download, install, and configure the Tivoli Ready enablement package. That support enables you to:

- View the status of WebSphere Application Server Version 3.0 through Tivoli Global Enterprise Manager 2.2 or Tivoli Enterprise Console (TEC) 3.1 or higher.

- Inventory WebSphere Application Server Version 3.0 using Tivoli Inventory Version 3.2 or higher.

Q: What are the steps involved in deploying an applet on WebSphere Application Server?

A: Applets are hosted by WAS as part of a Web application. Therefore, applets are part of the document root structure of the Web application hosted by WAS. To deploy an applet on WAS, follow the same steps as you would to deploy an HTML file.

Q: When I deploy a clone on different nodes, who handles the class files?

A: A clone knows about the configuration of your application, servers, and so on. You still need to physically make the files available on different nodes.

Q: Where should I specify classes that are loaded only once when the server starts?

A: The application server classpath is automatically set when you install the product. The default setting for the classpath contains all the application server APIs (the JAR files in the AS_install_root\lib directory). When the application server starts, the system classloader automatically loads the classes in the application server classpath. Those classes are not reloadable. You can also pass in the classpath to the application server's Java Virtual Machine (JVM) by specifying it in the command line arguments on the general page of the application server.

Q: If I make a change to my application, do I have to make changes to each clone?

A: To make changes to an application, you should make changes to the model instead of the clones. Changes made to a model are automatically propagated to all clones associated with that model. For example, to add or remove a bean from an application server, you add or remove the bean in the server model. With one action, you're able to add or remove the bean from all clones of the application server.

Q: How does IBM component broker fit into the IBM WebSphere family?

A: Component broker is an enterprise solution for distributed object computing that includes an operational environment and toolset. It is part of the WebSphere Enterprise Edition. For more information, visit the IBM component broker Web site at www.ibm.com/software/ad/cb.

Q: My WebSphere application server doesn't start because my DB2 user ID and password are different from what I specified during WebSphere installation. How do I change my user ID and password without reinstalling?

A: You can change the database user ID and password in the WebSphere application server by changing the following lines in the admin.config file:

```
com.ibm.ejs.sm.adminServer.d
bUser=db2userid

com.ibm.ejs.sm.adminServer.d
bPassword=db2password
```

Or, you can create another user in the database with the user ID and password supplied to WebSphere.

Q: How can I make the Web application pick up my new class files that are not servlets or Enterprise JavaBeans (EJBs)?

A: When you configure a Web application, you specify the application classpath containing servlets and their nonservlet Java components. The classloader monitors the application classpath and reloads all of the Java components in that application classpath whenever it detects that a loaded servlet has been updated. In the WebSphere 3 administration console using the topology tab, navigate to your Web application, for example, the default_app. On the advanced tab, specify the classpath (reloadable classpath). The classes should physically reside in the path specified here.

WebSphere Developer's Journal Advertiser Index...

WebSphere Developer's Journal Coming Next Month...

— ***An Interview with Don Ferguson***
Talking with the Father of WebSphere

— ***Creating Performance-Optimized JDBC Applications for DB2***
Thoughtful design and implementation improve performance

— ***Automating Business Logic for WebSphere Using Transaction Rules***
Freeing developers from routine infrastructure issues

— ***Why They Buy:***
The How and Why of Selling WebSphere

— ***Automating Business Logic for WebSphere Using Transaction Rules***
Integrating the automated system with external applications

Networks

Moving to a new level of performance

BY JIM MARTIN

Within the next few years we will all be connected to networks for almost every purpose in our lives. Our TVs, our cars, our refrigerators, and maybe even our pets, are all going to be plugged into some network that will make our lives better. Better? Is that simpler or more complex? Either way it won't matter, it's going to be more interesting. Let's accept for a moment that soon everything will be plugged into the great network and we will all achieve digital Nirvana – who can say what the great network is? Is it the Internet? Is it some global, self-personalizing extranet? Is it some wireless broadband technology? The fact is that it is all of them and then some.

The IP paradigm has enabled datastores to be opened up to the masses and the data to be used in many new and creative ways. The Internet has made many of the recent technology strides possible because of its broad audience, and along with it, the scale to realize profits from creating things for the Internet environment. This newfound method of expression helped to finance the accelerated build-out of big bandwidth Internet backbone capacity. However, big hulking frame relays, ATM, private IP, and wireless networks have leveraged the same technology strides to exploit business applications that have increased productivity manyfold. As network providers continue to build out bandwidth of every variety at an ever-quickening pace, the large-scale network technology decisions take on a greater importance.

With the cost of building out networks dropping (I'm waiting to see dark fiber and optical switches at the Price Club), carriers and Internet companies are carefully pondering which technology to embrace. By no means is building out a business-class network a small-change affair. Price wars have driven margins to next to nothing on bandwidth so even with a reduced cost of entry there is high risk in a network build-out and the core technology decision can only heighten the risk. What if your target market doesn't embrace the technology? What if the transmission gear becomes too difficult or too costly to acquire? Each core network technology has inherent risks and rewards: ATM is great for video and quality of service, but can be pricey; Frame Relay is inexpensive and easy to use but not as efficient; IP is the common vernacular, and wireless will unleash the next generation of computing apps. It's going to be a tough call even for the smart guys. Today, on their own, none of these technologies are a cost-effective or reliable means of delivering any type of data between users. Major network providers are still investing in all of these technologies and their availability keeps the enterprise computing along certain ways of thinking.

Businesses will continue to outsource applications, and the marketplace for outsourcing these services will continue to grow at a brisk pace; however, the landscape of service providers is changing just as quickly. Network providers have made substantial investments in hosting centers, and hosting centers have made fairly substantial investments in networks. Neither has realized tremendous returns because neither is in control of the piece that lies between their disciplines, the application environment, and the ability to manage that environment as a straight-through process. These providers will need to evolve from being just facilities or network providers to being infrastructure providers delivering managed services at the network and application level backed by mission-critical SLAs.

As the enterprise opens further, inviting in partners, customers, and suppliers, networks and applications will need to work together seamlessly to deliver the same enterprise experience to all users. Each new participant will already have a network architecture that is the underpinning of its business. The success of an open enterprise will ride on new participants' experiences after becoming part of the enterprise environment. Next generation applications will be engineered with the understanding that application performance is dependent upon network performance. In order to properly manage one you must be able to manage the other. Managed network services will enable the bottlenecks to be reduced throughout the straight-through process and applications will be able to perform well over any network architecture. Real-world decisions are made on a cost/risk basis. Can you deliver an institutional financial trading application over the Internet? Sure, but you will have concerns such as security, performance, and scalability. Maybe an extranet running over ATM with a high level of guaranteed quality of service is a better answer, but it could cost more. Will this application be able to go wireless? The right answer really depends on the tools that will be used to build and deliver the application and what the user population's existing computing environment is.

All of the common communication technologies will have their place in the new world order of ubiquitous wired devices. What will make each technology successful will be its reliability and ability to interface to the other technologies in use today with a minimal amount of cost and re-engineering of the current business enterprise. The next generations of messaging and network services will make it possible for all of the popular network technologies to survive and prosper. Digital Darwinism will decide what each technology will be used for and at what cost. It will be the confluence of the technologies that will be the challenge. This is where scalable application platforms and customer-centric network services will rule the day. This will be a must for network and hosting service providers; they will need to have a scalable and flexible application platform at the core of the service model.

As we approach yet another crossroads of technology we will all do well to take a well-timed moment to decide how best our data should travel and where it will have to go to become information.

ABOUT THE AUTHOR Jim Martin has worked in the system integration and communications industry for the past 15 years. Working on design and implementation teams, he has been instrumental in deploying Web-based, mission-critical systems for the financial industry. Jim currently works for Nova Group in New York City.

E-MAIL
jimmartin@pagenetips.com